Network Working Group
Request for Comments: 2801
Category: Informational

D. Burdett
Commerce One

April 2000

Internet Open Trading Protocol - IOTP
Version 1.0

Status of this Memo

Copyright Notice

Abstract

The Internet Open Trading Protocol (IOTP) provides an interoperable framework for Internet commerce. It is payment system independent and encapsulates payment systems such as SET, Secure Channel Credit/Debit, Mondex, CyberCoin, GeldKarte, etc. IOTP is able to handle cases where such merchant roles as the shopping site, the Payment Handler, the Delivery Handler of goods or services, and the provider of customer support are performed by different parties or by one party.

Table of Contents

Table of Figures

1. Background

The Internet Open Trading Protocol (IOTP) provides an interoperable framework for Internet commerce. It is payment system independent and encapsulates payment systems such as SET, Mondex, CyberCash, DigiCash, GeldKarte, etc. IOTP is able to handle cases where such merchant roles as the shopping site, the Payment Handler, the Delivery Handler of goods or services, and the provider of customer support are performed by different parties or by one party.

The developers of IOTP seek to provide a virtual capability that safely replicates the real world, the paper based, traditional, understood, accepted methods of trading, buying, selling, value exchanging that has existed for many hundreds of years. The negotiation of who will be the parties to the trade, how it will be conducted, the presentment of an offer, the method of payment, the provision of a payment receipt, the delivery of goods and the receipt of goods. These are events that are taken for granted in the course of real world trade. IOTP has been produced to provide the same for the virtual world, and to prepare and provide for the introduction of new models of trading made possible by the expanding presence of the virtual world.

The other fundamental ideal of the IOTP effort is to produce a definition of these trading events in such a way that no matter where produced, two unfamiliar parties using electronic commerce capabilities to buy and sell that conform to the IOTP specifications will be able to complete the business safely and successfully.

In summary, IOTP supports:

o Familiar trading models

o New trading models

o Global interoperability

The remainder of this section provides background to why IOTP was developed. The specification itself starts in the next chapter.

1.1 Commerce on the Internet, a Different Model

The growth of the Internet and the advent of electronic commerce are bringing about enormous changes around the world in society, politics and government, and in business. The ways in which trading partners communicate, conduct commerce, are governed have been enriched and changed forever.

One of the very fundamental changes about which IOTP is concerned is taking place in the way consumers and merchants trade. Characteristics of trading that have changed markedly include:

- Presence: Face-to-face transactions become the exception, not the rule. Already with the rise of mail order and telephone order placement this change has been felt in western commerce. Electronic commerce over the Internet will further expand the scope and volume of transactions conducted without ever seeing the people who are a part of the enterprise with whom one does business.

- Authentication: An important part of personal presence is the ability of the parties to use familiar objects and dialogue to confirm they are who they claim to be. The seller displays one or several well known financial logos that declaim his ability to accept widely used credit and debit instruments in the payment part of a purchase. The buyer brings government or financial institution identification that assures the seller she will be paid. People use intangibles such as personal appearance and conduct, location of the store, apparent quality and familiarity with brands of merchandise, and a good clear look in the eye to reinforce formal means of authentication.

- Payment Instruments: Despite the enormous size of bank card financial payments associations and their members, most of the world’s trade still takes place using the coin of the realm or barter. The present infrastructure of the payments business cannot economically support low value transactions and could not survive under the consequent volumes of transactions if it did accept low value transactions.

- Transaction Values: New meaning for low value transactions arises in the Internet where sellers may wish to offer for example, pages of information for fractions of currency that do not exist in the real world.

- Delivery: New modes of delivery must be accommodated such as direct electronic delivery. The means by which receipt is confirmed and the execution of payment change dramatically where the goods or services have extremely low delivery cost but may in fact have very high value. Or, maybe the value is not high, but once delivery occurs the value is irretrievably delivered so payment must be final and non-refundable but delivery nonetheless must still be confirmed before payment. Incremental delivery such as listening or viewing time or playing time are other models that operate somewhat differently in the virtual world.

1.2 Benefits of IOTP

ELECTRONIC COMMERCE SOFTWARE VENDORS

Electronic Commerce Software Vendors will be able to develop e-commerce products which are more attractive as they will inter-operate with any other vendors’ software. However, since IOTP focuses on how these solutions communicate, there is still plenty of opportunity for product differentiation.

PAYMENT BRANDS

IOTP provides a standard framework for encapsulating payment protocols. This means that it is easier for payment products to be incorporated into IOTP solutions. As a result the payment brands will be more widely distributed and available on a wider variety of platforms.

MERCHANTS

There are several benefits for Merchants:

- they will be able to offer a wider variety of payment brands,

- they can be more certain that the customer will have the software needed to complete the purchase

- through receiving payment and delivery receipts from their customers, they will be able to provide customer care knowing that they are dealing with the individual or organisation with which they originally traded

- new merchants will be able to enter this new (Internet) market-place with new products and services, using the new trading opportunities which IOTP presents

BANKS AND FINANCIAL INSTITUTIONS

There are also several benefits for Banks and Financial Institutions:

- they will be able to provide IOTP support for merchants

- they will find new opportunities for IOTP related services:

  - providing customer care for merchants
  - fees from processing new payments and deposits

- they have an opportunity to build relationships with new types of merchants

CUSTOMERS

For Customers there are several benefits:

- they will have a larger selection of merchants with whom they can trade

- there is a more consistent interface when making the purchase

- there are ways in which they can get their problems fixed through the merchant (rather than the bank!)

- there is a record of their transaction which can be used, for example, to feed into accounting systems or, potentially, to present to the tax authorities

1.3 Baseline IOTP

This specification is Baseline IOTP. It is a Baseline in that it contains ways of doing trades on the Internet which are the most common, for example purchases and refunds.

The group that has worked on the IOTP see an extended version being developed over time but feel a need to focus on a limited function but completely usable specification in order that implementers can develop solutions that work now.

During this period it is anticipated that there will be no changes to the scope of this specification with the only changes made being limited to corrections where problems are found. Software solutions have been developed based on earlier versions of this specification (for example version 0.9 published in early 1998 and earlier revisions of version 1.0 published during 1999) which prove that the IOTP works.

1.4 Objectives of Document

The objectives of this document are to provide a specification of version 1.0 of the Internet Open Trading Protocols which can be used to design and implement systems which support electronic trading on the Internet using the Internet Open Trading Protocols.

The purpose of the document is:

- o to allow potential developers of products based on the protocol to develop software/hardware solutions which use the protocol

- o to allow the financial services industry to understand a developing electronic commerce trading protocol that encapsulates (without modification) any of the current or developing payment schemes now being used or considered by their merchant customer base

1.5 Scope of Document

The protocol describes the content, format and sequences of messages that pass among the participants in an electronic trade - consumers, merchants and banks or other financial institutions, and customer care providers. These are required to support the electronic commerce transactions outlined in the objectives above.

The protocol is designed to be applicable to any electronic payment scheme since it targets the complete purchase process where the movement of electronic value from the payer to the payee is only one, but important, step of many that may be involved to complete the trade.

Payment Scheme which IOTP could support include MasterCard Credit, Visa Credit, Mondex Cash, Visa Cash, GeldKarte, eCash, CyberCoin, Millicent, Proton, etc.

Each payment scheme contains some message flows which are specific to that scheme. These scheme-specific parts of the protocol are contained in a set of payment scheme supplements to this specification.

The document does not prescribe the software and processes that will need to be implemented by each participant. It does describe the framework necessary for trading to take place.

This document also does not address any legal or regulatory issues surrounding the implementation of the protocol or the information systems which use them.

1.6 Document Structure

The document consists of the following sections:

- o Section 1 - Background: This section gives a brief background on electronic commerce and the benefits IOTP offers.

- Section 2 - Introduction: This section describes the various Trading Exchanges and shows how these trading exchanges are used to construct the IOTP Transactions. This section also explains various Trading Roles that would participate in electronic trade.

- Section 3 - Protocol Structure: This section summarises how various IOTP transactions are constructed using the Trading Blocks and Trading Components that are the fundamental building blocks for IOTP transactions. All IOTP transaction messages are well formed XML documents.

- Section 4 - IOTP Error Handling: This section describes how to process exceptions and errors during the protocol message exchange and trading exchange processing. This section provides a generic overview of the exception handling. This section should be read carefully.

- Section 5 - Security Considerations: This section considers from an IETF perspective, how IOTP addresses security. It includes: how to determine whether to use digital signatures with IOTP, how IOTP address data privacy, and how security built into payment protocols relate to IOTP security.

- Section 6 - Digital Signatures and IOTP: This section provides an overview of how IOTP uses digital signatures; how to check a signature is correctly calculated and how the various Trading Roles that participate in trade should check signatures when required.

- Section 7 - Trading Components: This section defines the XML elements required by Trading Components.

- Section 8 - Trading Blocks: This section describes how Trading Blocks are constructed from Trading Components.

- Section 9 - Internet Open Trading Protocol Transactions: This section describes all the IOTP Baseline transactions. It refers to Trading Blocks and Trading Components and Signatures. This section doesn’t directly link error handling during the protocol exchanges, the reader is advised to understand Error Handling as defined in section before reading this section.

- Section 10 - Retrieving Logos: This section describes how IOTP specific logos can be retrieved.

- o Section 11 - Brands: This section provides: an overview of Brand Definitions and Brand Selection which describe how a Consumer can select a Brand from a list provided by the Merchant; as well as some examples of Brand Lists.
- o Section 12 - IANA Considerations: This section describes how new values for codes used by IOTP are co-ordinated.
- o Section 13 - Internet Open Trading Protocol Data Type Definition: This section contains the XML Data Type Definitions for IOTP.
- o Section 14 - Glossary. This describes all the major terminology used by IOTP.
- o Section 15 - A list of the other documents referenced by the IOTP specification.
- o Section 16 - The Author’s Address
- o Section 17 - Full Copyright Statement

1.7 Intended Readership

Software and hardware developers; development analysts; business and technical planners; industry analysts; merchants; bank and other payment handlers; owners, custodians, and users of payment protocols.

1.7.1 Reading Guidelines

This IOTP specification is structured primarily in a sequence targeted at people who want to understand the principles of IOTP. However from practical implementation experience by implementers of earlier of versions of the protocol new readers who plan to implement IOTP may prefer to read the document in a different sequence as described below.

Review the transport independent parts of the specification. This covers:

- o Section 14 - Glossary
- o Section 1 - Background
- o Section 2 - Introduction
- o Section 3 - Protocol Structure
- o Section 4 - IOTP Error Handling

- o Section 5 - Security Considerations
- o Section 9 - Internet Open Trading Protocol Transactions
- o Section 11 - Brands
- o Section 12 - IANA Considerations
- o Section 10 - Retrieving Logos

Review the detailed XML definitions:

- o Section 8 - Trading Blocks
- o Section 7 - Trading Components
- o Section 6 - Digital Signatures and IOTP

2. Introduction

The Internet Open Trading Protocols (IOTP) define a number of different types of IOTP Transactions:

- o Purchase. This supports a purchase involving an offer, a payment and optionally a delivery
- o Refund. This supports the refund of a payment as a result of, typically, an earlier purchase
- o Value Exchange. This involves two payments which result in the exchange of value from one combination of currency and payment method to another
- o Authentication. This supports one organisation or individual to check that another organisation or individual are who they appear to be.
- o Withdrawal. This supports the withdrawal of electronic cash from a financial institution
- o Deposit. This supports the deposit of electronic cash at a financial institution
- o Inquiry. This supports inquiries on the status of an IOTP transaction which is either in progress or is complete

- o Ping. This supports a simple query which enables one IOTP aware application to determine whether another IOTP application running elsewhere is working or not.

These IOTP Transactions are "Baseline" transactions since they have been identified as a minimum useful set of transactions. Later versions of IOTP may include additional types of transactions.

Each of the IOTP Transactions above involve:

- o a number of organisations playing a Trading Role, and
- o a set of Trading Exchanges. Each Trading Exchange involves the exchange of data, between Trading Roles, in the form of a set of Trading Components.

Trading Roles, Trading Exchanges and Trading Components are described below.

2.1 Trading Roles

The Trading Roles identify the different parts which organisations can take in a trade. The five Trading Roles used within IOTP are illustrated in the diagram below.

```
   *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*

              Merchant Customer Care Provider resolves   ----------
         ---------------------------------------------->| Merchant |
        |          Consumer disputes and problems       |Cust.Care.|
        |                                               | Provider |
        |                                                ----------
        |
                   Payment Handler accepts or makes     ----------
        |    ------------------------------------------>| Payment  |
        |   |             Payment for Merchant          | Handler  |
        |   |                                            ----------
        v   v
    ----------    Consumer makes purchases or obtains    ----------
   | Consumer |<--------------------------------------->| Merchant |
    ----------             refund from Merchant          ----------
        ^
        |         Delivery Handler supplies goods or     ----------
        |---------------------------------------------->|Deliverer |
                       services for Merchant            | Handler  |
                                                         ----------

   *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 1 IOTP Trading Roles

The roles are:

- o Consumer. The person or organisation which is to receive and pay for the goods or services
- o Merchant. The person or organisation from whom the purchase is being made and who is legally responsible for providing the goods or services and receives the benefit of the payment made
- o Payment Handler. The entity that physically receives the payment from the Consumer on behalf of the Merchant
- o Delivery Handler. The entity that physically delivers the goods or services to the Consumer on behalf of the Merchant.
- o Merchant Customer Care Provider. The entity that is involved with customer dispute negotiation and resolution on behalf of the Merchant

Roles may be carried out by the same organisation or different organisations. For example:

- o in the simplest case one physical organisation (e.g., a merchant) could handle the purchase, accept the payment, deliver the goods and provide merchant customer care
- o at the other extreme, a merchant could handle the purchase but instruct the consumer to pay a bank or financial institution, request that delivery be made by an overnight courier firm and to contact an organisation which provides 24x7 service if problems arise.

Note that in this specification, unless stated to the contrary, when the words Consumer, Merchant, Payment Handler, Delivery Handler or Customer Care Provider are used, they refer to the Trading Role rather than an actual organisation.

An individual organisation may take multiple roles. For example a company which is selling goods and services on the Internet could take the role of Merchant when selling goods or services and the role of Consumer when the company is buying goods or services itself.

As roles occur in different places there is a need for the organisations involved in the trade to exchange data, i.e. to carry out Trading Exchanges, so that the trade can be completed.

2.2 Trading Exchanges

The Internet Open Trading Protocols identify four Trading Exchanges which involve the exchange of data between the Trading Roles. The Trading Exchanges are:

- o Offer. The Offer Exchange results in the Merchant providing the Consumer with the reason why the trade is taking place. It is called an Offer since the Consumer must accept the Offer if a trade is to continue

- o Payment. The Payment Exchange results in a payment of some kind between the Consumer and the Payment Handler. This may occur in either direction

- o Delivery. The Delivery Exchange transmits either the on-line goods, or delivery information about physical goods from the Delivery Handler to the Consumer, and

- o Authentication. The Authentication Exchange can be used by any Trading Role to authenticate another Trading Role to check that they are who they appear to be.

IOTP Transactions are composed of various combinations of these Trading Exchanges. For example, an IOTP Purchase transaction includes Offer, Payment, and Delivery Trading Exchanges. As another example, an IOTP Value Exchange transaction is composed of an Offer Trading Exchange and two Payment Trading Exchanges.

Trading Exchanges consist of Trading Components that are transmitted between the various Trading Roles. Where possible, the number of round-trip delays in an IOTP Transaction is minimised by packing the Components from several Trading Exchanges into combination IOTP Messages. For example, the IOTP Purchase transaction combines a Delivery Organisation Component with an Offer Response Component in order to avoid an extra Consumer request and response.

Each of the IOTP Trading Exchanges is described in more detail below. For clarity of description, these describe the Trading Exchanges as though they were standalone operations. For performance reasons, the Trading Exchanges are intermingled in the actual IOTP Transaction definitions.

2.2.1 Offer Exchange

The goal of the Offer Exchange is for the Merchant to provide the Consumer with information about the trade so that the Consumer can decide whether to continue with the trade. This is illustrated in the figure below.

```
 *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*
   Consumer
     |  Merchant
STEP |     |
 1.          Consumer decides to trade and sends information about the
             transaction (requests an offer) to the Merchant e.g.,
             using HTML.

     C --> M Data: Information on what is being purchased (Offer Request)
             - outside scope of IOTP

 2.          Merchant checks the information provided by the Consumer,
             creates an Offer optionally signs it and sends it to the
             Consumer.

     C <-- M OFFER RESPONSE. Components: Status; Organisation(s)
             (Consumer, DelivTo, Merchant, Payment Handler, Customer
             Care); Order; Payment; Delivery; TradingRoleData (optional)
             Offer Response Signature (optional) that signs other
             components

 3.          Consumer checks the information from the Merchant and
             decides whether to continue.

 *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 2 Offer Exchange

An Offer Exchange uses the following Trading Components that are passed between the Consumer and the Merchant:

- o the Status component is used to indicate to other parties that a valid Offer Response has been generated

- o the Organisation Component contains information which describes the Organisations which are taking a role in the trade:

  - the consumer provides information, about who the consumer is and, if goods or services are being delivered, where the goods or services are to be delivered to

  - the merchant augments this information by providing information about the merchant, the Payment Handler, the customer care provider and, if goods or services are being delivered, the Delivery Handler

- the Order Component contains descriptions of the goods or services which will result from the trade if the consumer agrees to the offer. This information is sent by the Merchant to the consumer who should verify it

- the Payment Component generated by the Merchant, contains details of how much to pay, the currency and the payment direction, for example the consumer could be asking for a refund. Note that there may be more than one payment in a trade

- the Delivery Component, also generated by the Merchant, is used if goods or services are being delivered. This contains information about how delivery will occur, for example by post or using e-mail

- the Trading Role Data component contains data the Merchant wants to forward to another Trading Role such as a Payment Handler or Delivery Handler

- the "Offer Response" Signature Component, if present, digitally signs all of the above components to ensure their integrity.

The exact content of the information provided by the Merchant to the Consumer will vary depending on the type of IOTP Transaction. For example:

- low value purchases may not need a signature

- the amount to be paid may vary depending on the payment brand and payment protocol used

- some offers may not involve the delivery of any goods

- a value exchange will involve two payments

- a merchant may not offer customer care.

Information provided by the consumer to the merchant is provided using a variety of methods, for example, it could be provided:

- using [HTML] pages as part of the "shopping experience" of the consumer.

- o Using the Open Profiling Standard [OPS] which has recently been proposed,
- o in the form of Organisation Components associated with an authentication of a Consumer by a Merchant
- o as Order Components in a later version of IOTP.

2.2.2 Payment Exchange

The goal of the Payment Exchange is for a payment to be made from the Consumer to a Payment Handler or vice versa using a payment brand and payment protocol selected by the Consumer. A secondary goal is to optionally provide the Consumer with a digitally signed Payment Receipt which can be used to link the payment to the reason for the payment as described in the Offer Exchange.

Payment Exchanges can work in a variety of ways. The most general case where the trade is dependent on the payment brand and protocol used is illustrated in the diagram below. Simpler payment exchanges are possible.

```
 *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*
  Consumer  Pay Handler
     |  Merchant |
STEP |     |     |
 1.                  Consumer decides to trade and sends information
                     about the transaction (requests an offer) to the
                     Merchant e.g., using HTML.

     C --> M         Information on what is being paid for (outside
                     scope of IOTP

 2.                  Merchant decides which payment brand, payment
                     protocols and currencies/amounts to offer,
                     places then in a Brand List Component and sends
                     them to the Consumer

     C <-- M         Components: Brand List

 3.                  Consumer selects the payment brand, protocol and
                     currency/amount to use, creates a Brand Selection
                     component and sends it to the Merchant

     C --> M         Component: Brand List Selection
 4.                  Merchant checks Brand Selection, creates a Payment
                     Amount information, optionally signs it to
                     authorise payment and sends it to the Consumer

     C <-- M         Component: Payment; Organisation(s) (Merchant and
                     Payment Handler); Optional Offer Response Signature
                     that signs other components

 5.                  Consumer checks the Payment Amount information and
                     if OK requests that the payment starts by sending
                     information to the Payment Handler

     C --------> P  PAYMENT REQUEST. Components: Status, Payment;
                     Organisations (Merchant and Payment Handler);
                     Trading Role Data (optional); Optional Offer
                     Response Signature that signs other components;
                     Pay Scheme Data

 6.                  Payment Handler checks information including
                     optional signature and if OK starts exchanging Pay
                     Scheme Data components for selected payment brand
                     and payment protocol

     C <-------> P  PAYMENT EXCHANGE. Component: Pay Scheme Data

 7.                  Eventually payment protocol messages finish so
                     Payment Handler sends Pay Receipt and optional
                     signature to the Consumer as proof of payment

     C <-------> P  PAYMENT RESPONSE. Components: Status, Pay Receipt;
                     Payment Note; Trading Role Data (optional);
                     Optional Offer Response Signature; Optional
                     Payment Receipt Signature that binds the payment
                     to the Offer

 8.                  Consumer checks Payment Receipt is OK

*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 3 Payment Exchange

A Payment Exchange uses the following Trading Components that are passed between the Consumer, the Merchant and the Payment Handler:

o The Brand List Component contains a list of payment brands (for example, MasterCard, Visa, Mondex, GeldKarte), payment protocols (for example SET Version 1.0, Secure Channel Credit Debit (SCCD - the name used for a credit or debit card payment where unauthorised access to account information is prevented through use of secure channel transport mechanisms such as SSL/TLS) as well as currencies/amounts that apply. The Merchant sends the Brand List to the Consumer. The consumer compares the payment brands, protocols and currencies/amounts on offer with those that the Consumer supports and makes a selection.

- The Brand Selection Component contains the Consumer’s selection. Payment brand, protocol, currency/amount and possibly protocol-specific information is sent back to the Merchant. This information may be used to change information in the Offer Exchange. For example, a merchant could choose to offer a discount to encourage the use of a store card.

- the Status component is used to indicate to the Payment Handler that an earlier exchange (e.g., an Offer Exchange) has successfully completed and by the Payment Handler to indicate the completion status of the Payment Exchange.

- The Organisation Components are generated by the Merchant. They contain details of the Merchant and Payment Handler Roles:

  - the Merchant role is required so that the Payment Handler can identify which Merchant initiated the payment. Typically, the result of the Payment Handler accepting (or making) a payment on behalf of the Merchant will be a credit or debit transaction to the Merchant’s account held by the Payment Handler. These transactions are outside the scope of this version of IOTP

  - the Payment Handler role is required so that the Payment Handler can check that it is the correct Payment Handler to be used for the payment

- The Payment Component contains details of how much to pay, the currency and the payment direction

- The "Offer Response" Signature Component, if present, digitally signs all of the above components to ensure their integrity. Note that the Brand List and Brand Selection Components are not signed until the payment information is created (step 4 in the diagram)

- the Trading Role Data component contains from other roles (e.g., a Merchant) that needs to be forwarded to the Payment Handler

- The Payment Scheme Component contains messages from the payment protocol used in the Trade. For example they could be SET messages, Mondex messages, GeldKarte Messages or one of the other payment methods supported by IOTP. The content of the Payment

Scheme Component is defined in the supplements that describe how IOTP works with various payment protocols.

- o The Payment Receipt Component contains a record of the payment. The content depends upon the payment protocol used.

- o The "Payment Receipt" Signature Component provides proof of payment by digitally signing both the Payment Receipt Component and the Offer Response Signature. The signature on the offer digitally signs the Order, Organisation and Delivery Components contained in the Offer. This signature effectively binds the payment to the offer.

The example of a Payment Exchange above is the most general case. Simpler cases are also possible. For example, if the amount paid is not dependent on the payment brand and protocol selected then the payment information generated by step 3 can be sent to the Consumer at the same time as the Brand List Component generated by step 1. These and other variations are described in the Baseline Purchase IOTP Transaction (see section 9.1.8).

2.2.3 Delivery Exchange

The goal of the Delivery Exchange is to cause purchased goods to be delivered to the consumer either online or via physical delivery. A second goal is to provide a "delivery note" to the consumer, providing details about the delivery, such as shipping tracking number. The result of the delivery may also be signed so that it can be used for customer care in the case of problems with physical delivery. The message flow is illustrated in the diagram below.

```
 *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*
  CONSUMER  DELIVERY
     |        HANDLER
     |  Merchant |
STEP |     |     |
 1.                   Consumer decides to trade and sends information
                      about what to deliver and who is to take delivery,
                      to the Merchant e.g., using HTML.

     C --> M          Information on what is being delivered (outside
                      scope of IOTP)

 2.                   Merchant checks the information provided by the
                      Consumer, adds information about how the delivery
                      will occur, information about the Organisations
                      involved in the delivery and optionally sings it
                      and sends it to the Consumer
     C <-- M          Components: Delivery; Organisations (Delivery
                      Handler, Deliver To); Order, Optional Offer
                      Response Signature

  3.                  Consumer checks delivery information is OK,
                      obtains authorisation for the delivery, for
                      example by making a payment, and sends the
                      delivery information to the Delivery Handler

     C --------> D    DELIVERY REQUEST. Components: Status; Delivery,
                      Organisations: (Merchant, Delivery Handler,
                      DelivTo); Order, Trading Role Data (optional);
                      Optional Offer Response Signature, Optional
                      Payment Receipt Signature (from Payment Exchange)

  4.                  Delivery Handler checks information and
                      authorisation. Starts or schedules delivery and
                      creates and then sends a delivery not tot the
                      Consumer which can optionally be signed.

     C <-------- D    DELIVERY RESPONSE. Components: Status; Delivery
                      Note, Trading Role Data (optional); Optional
                      Delivery Response Signature

  5.                  Consumer checks delivery note is OK and accepts or
                      waits for delivery as described in the the Delivery
                      Note.

  *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 4 Delivery Exchange

A Delivery Exchange uses the following Trading Components that are passed between the Consumer, the Merchant and the Delivery Handler:

- o the Status component is used to indicate to the Delivery Handler that an earlier exchange (e.g., an Offer Exchange or Payment Exchange) has successfully completed and by the Delivery Handler to indicate the completion status of the Delivery Exchange.

- o The Organisation Component(s) contain details of the Deliver To, Delivery Handler and Merchant Roles:

  - the Deliver To role indicates where the goods or services are to be delivered to

  - the Delivery Handler role is required so that the Delivery Handler can check that she is the correct Delivery Handler to do the delivery

  - the Merchant role is required so that the Delivery Handler can identify which Merchant initiated the delivery

- The Order Component, contains information about the goods or services to be delivered

- The Delivery Component contains information about how delivery will occur, for example by post or using e-mail.

- The "Offer Response" Signature Component, if present, digitally signs all of the above components to ensure their integrity.

- The "Payment Receipt" Signature Component provides proof of payment by digitally signing the Payment Receipt Component and the Offer Signature. This is used by the Delivery Handler to check that delivery is authorised

- The Delivery Note Component contains customer care information related to a physical delivery, or alternatively the actual "electronic goods". The Consumer’s software does not interpret information about a physical delivery but should have the ability to display the information, both at the time of the delivery and later if the Consumer selects the Trade to which this delivery relates from a transaction list

- The "Delivery Response" Signature Component, if present, provides proof of the results of the Delivery by digitally signing the Delivery Note and any Offer Response or Payment Response signatures that the Delivery Handler received.

2.2.4 Authentication Exchange

The goal of the Authentication Exchange is to allow one Organisation, for example a financial institution, to be able to check that another Organisation, for example a consumer, is who they appear to be.

An Authentication Exchange involves:

- an Authenticator - the Organisation which is requesting the authentication, and

- an Authenticatee - the Organisation being authenticated.

This is illustrated in the diagram below.

```
 +*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*
 Organisation 1
 (Authenticatee)
     |   Organisation 2
     |  (Authenticator)
STEP |     |
 1.          First Organisation, e.g., a Consumer, takes an action (for
             example by pressing a button on an HTML page) which
             requires that the Organisation is authenticated

     1 --> 2 Need for Authentication (outside scope of IOTP)

 2.          The second Organisation generates an Authentication
             Request - including challenge data, and a list of the
             algorithms that may be used for the authentication -
             and/or a request for the Organisation information then
             sends it to the first Organisation

     1 <-- 2 AUTHENTICATION REQUEST. Components: Authentication
             Request, Trading Role Information Request

 3.          The first Organisation optionally checks any signature
             associated with the Authentication Request then uses the
             specified authentication algorithm to generate an
             Authentication Response which is sent back to the second
             Organisation together with details of any Organisation
             information requested

     1 --> 2 AUTHENTICATION RESPONSE. Component: Authentication
             Response, Organisation(s)

 4.          The Authentication Response is checked against the
             challenge data to check that the first Organisation is
             who they appear to be and the result recorded in a Status
             Component which is then sent back to the first
             Organisation.

     1 <-- 2 AUTHENTICATION STATUS. Component: Status

 5.          The first Organisation then optionally checks the results
             indicated by the Status and any associated signature and
             takes the appropriate action or stops.

 *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 5 Authentication Exchange

An Authentication Exchange uses the following Trading Components that are passed between the two Organisations:

- the Authentication Request Component that requests an Authentication and indicates the authentication algorithm and optional challenge data to be used.
- A Trading Role Information Request Component that requests information about an Organisation, for example a ship to address.
- The Authentication Response Component which contains the challenge response generated by the recipient of the Authentication Request Component.
- Organisation Components that contain the result of the Trading Role Information Request
- the Status Component which contains the results of the second party’s verification of the Authentication Response.

2.3 Scope of Baseline IOTP

This specification describes the IOTP Transactions which make up Baseline IOTP. As described in the preface, IOTP will evolve over time. This section defines the initial conformance criteria for implementations that claim to "support IOTP."

The main determinant on the scope of an IOTP implementation is the roles which the solution is designed to support. The roles within IOTP are described in more detail in section 2.1 Trading Roles. To summarise the roles are: Merchant, Consumer, Payment Handler, Delivery Handler and Customer Care Provider.

Payment Handlers who can be of three types:

- those who accept a payment as part of a purchase or make a payment as part of a refund,
- those who accept value as part of a deposit transaction, or
- those that issue value a withdrawal transaction

The following table defines, for each role, the IOTP Transactions and Trading Blocks which must be supported for that role.

	Merchants					
	Store	ECash Value Issuer	ECash Value Acquirer	Consumer	Payment Handler	Delivery Handler
TRANSACTIONS						
Purchase	Must			Must		
Refund	Must			b) Depends		
Authentication	May	Must	May	b) Depends		
Value Exchange	May			Must		
Withdrawal		Must		b) Depends		
Deposit			Must	b) Depends		
Inquiry	Must	Must	Must	May	Must	Must
Ping	Must	Must	Must	May	Must	Must
TRADING BLOCKS						
TPO	Must	Must	Must	Must		
TPO Selection	Must	Must	Must	Must		
Auth-Request	a) Depends		a) Depends	a) Depends		
Auth-Reply	a) Depends		a) Depends	a) Depends		
Offer Response	Must	Must	Must	Must		

Payment Request				Must	Must	
Payment Exchange				Must	Must	
Payment Response				Must	Must	
Delivery Request				Must		Must
Delivery Response				Must		Must

	Merchants					
	Store	ECash Value Issuer	ECash Value Acquirer	Consumer	Payment Handler	Delivery Handler
Inquiry Request	Must	Must	Must	Must	Must	Must
Inquiry Response	Must	Must	Must	Must	Must	Must
Ping Request	Must	Must	Must	Must	Must	Must
Ping Response	Must	Must	Must	Must	Must	Must
Signature	Must	Must	Must	Limited	Must	Must
Error	Must	Must	Must	Must	Must	Must

In the above table:

- o "Must" means that a Trading Role must support the Transaction or Trading Block.

- o "May" means that an implementation may support the Transaction or Trading Block at the option of the developer.

- o "Depends" means implementation of the Transaction or Trading Block depends on one of the following conditions:

  - if Baseline Authentication IOTP Transaction is supported;

   - if required by a Payment Method as defined in its IOTP Supplement document.

- "Limited" means the Trading Block must be understood and its content manipulated but not in every respect. Specifically, on the Signature Block, Consumers do not have to be able to validate digital signatures.

An IOTP solution must support all the IOTP Transactions and Trading Blocks required by at least one role (column) as described in the above table for that solution to be described as "supporting IOTP".

3. Protocol Structure

The previous section provided an introduction which explained:

- Trading Roles which are the different roles which Organisations can take in a trade: Consumer, Merchant, Payment Handler, Delivery Handler and Customer Care Provider, and

- Trading Exchanges where each Trading Exchange involves the exchange of data, between Trading Roles, in the form of a set of Trading Components.

This section describes:

- how Trading Components are constructed into Trading Blocks and the IOTP Messages which are physically sent in the form of [XML] documents between the different Trading Roles,

- how IOTP Messages are exchanged between Trading Roles to create an IOTP Transaction

- the XML definitions of an IOTP Message including a Transaction Reference Block - an XML element which identifies an IOTP Transaction and the IOTP Message within it

- the definitions of the XML ID Attributes which are used to identify IOTP Messages, Trading Blocks and Trading Components and how these are referred to using Element References from other XML elements

- how extra XML Elements and new user defined values for existing IOTP codes can be used when Extending IOTP,

- how IOTP uses the Packaged Content Element to embed data such as payment protocol messages or detailed order definitions within an IOTP Message

- o how IOTP Identifies Languages so that different languages can be used within IOTP Messages

- o how IOTP handles both Secure and Insecure Net Locations when sending messages

- o how an IOTP Transaction can be cancelled.

3.1 Overview

3.1.1 IOTP Message Structure

The structure of an IOTP Message and its relationship with Trading Blocks and Trading Components is illustrated in the diagram below.

```
*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*

IOTP MESSAGE  <---------- IOTP Message - an XML Document which is
 |                        transported between the Trading Roles
 |-Trans Ref Block <----- Trans Ref Block - contains information which
 |  |                     describes the IOTP Transaction and the IOTP
 |  |                     Message.
 |  |-Trans Id Comp. <--- Transaction Id Component - uniquely
 |  |                     identifies the IOTP Transaction. The Trans Id
 |  |                     Components are the same across all IOTP
 |  |                     messages that comprise a single IOTP
 |  |                     transaction.
 |  |-Msg Id Comp. <----- Message Id Component - identifies and
 |                        describes an IOTP Message within an IOTP
 |                        Transaction
 |-Signature Block <----- Signature Block (optional) - contains one or
 |  |                     more Signature Components and their
 |  |                     associated Certificates
 |  |-Signature Comp. <-- Signature Component - contains digital
 |  |                     signatures. Signatures may sign digests of
 |  |                     the Trans Ref Block and any Trading Component
 |  |                     in any IOTP Message in the same IOTP
 |  |                     transaction.
 |  |-Certificate Comp. < Certificate Component (Optional) Used to check
 |                        the signature.
 |-Trading Block <------- Trading Block - an XML Element within an IOTP
 |  |-Trading Comp.       Message that contains a predefined set of
 |  |-Trading Comp.      Trading Components
 |  |-Trading Comp.
 |  |-Trading Comp. <--- Trading Components - XML Elements within a
 |                        Trading Block that contain a predefined set
 |-Trading Block          of XML elements and attributes containing
 |  |-Trading Comp.       information required to support a Trading
 |  |-Trading Comp.       Exchange
 |  |-Trading Comp.
 |  |-Trading Comp.
 |  |-Trading Comp.

*-*-*-*-*-*--*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 6 IOTP Message Structure

The diagram also introduces the concept of a Transaction Reference Block. This block contains, amongst other things, a globally unique identifier for the IOTP Transaction. Also each block and component is given an ID Attribute (see section 3.4) which is unique within an IOTP Transaction. Therefore the combination of the ID attribute and

   the globally unique identifier in the Transaction Reference Block is
   sufficient to uniquely identify any Trading Block or Trading
   Component.

3.1.2 IOTP Transactions

   A predefined set of IOTP Messages exchanged between the Trading Roles
   constitute an IOTP Transaction. This is illustrated in the diagram
   below.

```
*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*

     CONSUMER                                              MERCHANT
                                                       Generate first
                                                        IOTP Message
                                   ---                        |
                                  |   |                       v
 Process incoming                 | I |                 -------------
  IOTP Message &   <------------- |   | ------------ | IOTP Message |
generate next IOTP                |   |                 -------------
     Message                      | N |
        |                         |   |
        v                         |   |
  -------------                   | T |              Process incoming
 | IOTP Message |  -------------- |   | ----------->  IOTP Message &
  -------------                   |   |                 generate next
                                  | E |                  IOTP Message
                                  |   |                       |
                                  |   |                       v
 Process incoming                 | R |                 -------------
   IOTP Message    <------------- |   | ------------ | IOTP Message |
generate last IOTP                |   |                 -------------
  Message & stop                  | N |
        |                         |   |
        v                         |   |
  -------------                   | E |                  Process last
 | IOTP Message |  -------------- |   | ------------->  incoming IOTP
  -------------                   |   |                Message & stop
        |                         | T |                       |
        v                         |   |                       v
       STOP                        ---                       STOP

*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
```

Figure 7 An IOTP Transaction

In the above diagram the Internet is shown as the transport mechanism. This is not necessarily the case. IOTP Messages can be transported using a variety of transport mechanisms.

The IOTP Transactions (see section 9) in this version of IOTP are specifically:

- o Purchase. This supports a purchase involving an offer, a payment and optionally a delivery
- o Refund. This supports the refund of a payment as a result of, typically, an earlier purchase
- o Value Exchange. This involves two payments which result in the exchange of value from one combination of currency and payment method to another
- o Authentication. This supports the remote authentication of one Trading Role by another Trading Role using a variety of authentication algorithms, and the provision of an Organisation Information about the Trading Role that is being authenticated for use in, for example, the creation of an offer
- o Withdrawal. This supports the withdrawal of electronic cash from a financial institution
- o Deposit. This supports the deposit of electronic cash at a financial institution
- o Inquiry This supports inquiries on the status of an IOTP transaction which is either in progress or is complete
- o Ping This supports a simple query which enables one IOTP aware application to determine whether another IOTP application running elsewhere is working or not.

3.2 IOTP Message

As described earlier, IOTP Messages are [XML] documents which are physically sent between the different Trading Roles that are taking part in a trade.

The XML definition of an IOTP Message is as follows.

```
<!ELEMENT IotpMessage
   ( TransRefBlk,
     SigBlk?,
     ErrorBlk?,
     ( AuthReqBlk |
       AuthRespBlk |
       AuthStatusBlk |
       CancelBlk |
       DeliveryReqBlk |
       DeliveryRespBlk |
       InquiryReqBlk |
       InquiryRespBlk |
       OfferRespBlk |
       PayExchBlk |
       PayReqBlk |
       PayRespBlk |
       PingReqBlk |
       PingRespBlk |
       TpoBlk |
       TpoSelectionBlk
     )*
   ) >
<!ATTLIST IotpMessage
  xmlns                     CDATA
  'iotp:ietf.org/iotp-v1.0'
```

Content:

TransRefBlk	This contains information which describes an IOTP Message within an IOTP Transaction (see section 3.3 immediately below)
AuthReqBlk, AuthRespBlk, DeliveryReqBlk, DeliveryRespBlk ErrorBlk InquiryReqBlk, InquiryRespBlk, OfferRespBlk, PayExchBlk, PayReqBlk, PayRespBlk, PingReqBlk, PingRespBlk, SigBlk, TpoBlk, TpoSelectionBlk	These are the Trading Blocks. The Trading Blocks present within an IOTP Message, and the content of a Trading Block itself is dependent on the type of IOTP Transaction being carried out - see the definition of each transaction in section 9 Internet Open Trading Protocol Transactions. Full definitions of each Trading Block are described in section 8.

Attributes:

xmlns	The [XML Namespace] definition for IOTP messages.

3.2.1 XML Document Prolog

The IOTP Message is the root element of the XML document. It therefore needs to be preceded by an appropriate XML Document Prolog. For example:

```
<?XML Version='1.0'?>
<!DOCTYPE IotpMessage >
<IotpMessage>
 ...
</IotpMessage>
```

3.3 Transaction Reference Block

A Transaction Reference Block contains information which identifies the IOTP Transaction and IOTP Message. The Transaction Reference Block contains:

o a Transaction Id Component which globally uniquely identifies the IOTP Transaction. The Transaction Id Components are the same across all IOTP messages that comprise a single IOTP transaction,

o a Message Id Component which provides control information about the IOTP Message as well as uniquely identifying the IOTP Message within an IOTP Transaction, and

o zero or more Related To Components which link this IOTP Transaction to either other IOTP Transactions or other events using the identifiers of those events.

The definition of a Transaction Reference Block is as follows:

```
<!ELEMENT TransRefBlk (TransId, MsgId, RelatedTo*) >
<!ATTLIST TransRefBlk
 ID                ID      #REQUIRED >
```

Attributes:

ID	An identifier which uniquely identifies the Transaction Reference Block within the IOTP Transaction (see section 3.4 ID Attributes).

Content:

TransId	See 3.3.1 Transaction Id Component immediately below.
MsgId	See 3.3.2 Message Id Component immediately below.

RelatedTo	See 3.3.3 Related To Component immediately below.

3.3.1 Transaction Id Component

This contains information which globally uniquely identifies the IOTP Transaction. Its definition is as follows:

```
<!ELEMENT TransId EMPTY >
<!ATTLIST TransId
 ID                 ID      #REQUIRED
 Version            NMTOKEN #FIXED '1.0'
 IotpTransId        CDATA   #REQUIRED
 IotpTransType      CDATA   #REQUIRED
 TransTimeStamp     CDATA   #REQUIRED >
```

Attributes:

ID	An identifier which uniquely identifies the Transaction Id Component within the IOTP Transaction.
Version	This identifies the version of IOTP, and therefore the structure of the IOTP Messages, which the IOTP Transaction is using.
IotpTransId	Contains data which uniquely identifies the IOTP Transaction. It must conform to the rules for Message Ids in [RFC 822].
IotpTransTyp	This is the type of IOTP Transaction being carried out. For Baseline IOTP it identifies a "standard" IOTP Transaction and implies the sequence and content of the IOTP Messages exchanged between the Trading Roles. The valid values for Baseline IOTP are: o BaselineAuthentication o BaselineDeposit o BaselinePurchase o BaselineRefund o BaselineWithdrawal o BaselineValueExchange o BaselineInquiry o BaselinePing Values of IotpTransType are managed under the procedure described in section 12 IANA Considerations which also allows user defined values of IotpTransType to be defined.

	In later versions of IOTP, this list will be extended to support different types of standard IOTP Transaction. It is also likely to support the type Dynamic which indicates that the sequence of steps within the transaction are non-standard.
TransTimeStamp	Where the system initiating the IOTP Transaction has an internal clock, it is set to the time at which the IOTP Transaction started in [UTC] format. The main purpose of this attribute is to provide an alternative way of identifying a transaction by specifying the time at which it started. Some systems, for example, hand held devices may not be able to generate a time stamp. In this case this attribute should contain the value "NA" for Not Available.

3.3.2 Message Id Component

The Message Id Component provides control information about the IOTP Message as well as uniquely identifying the IOTP Message within an IOTP Transaction. Its definition is as follows.

```
<!ELEMENT MsgId EMPTY >
<!ATTLIST MsgId
 ID                  ID      #REQUIRED
 RespIotpMsg         NMTOKEN #IMPLIED
 xml:lang            NMTOKEN #REQUIRED
 LangPrefList        NMTOKENS #IMPLIED
 CharSetPrefList     NMTOKENS #IMPLIED
 SenderTradingRoleRef NMTOKEN #IMPLIED
 SoftwareId          CDATA   #REQUIRED
 TimeStamp           CDATA   #IMPLIED >
```

Attributes:

ID	An identifier which uniquely identifies the IOTP Message within the IOTP Transaction (see section 3.4 ID Attributes). Note that if an IOTP Message is resent then the value of this attribute remains the same.
RespIotpMsg	This contains the ID attribute of the Message Id Component of the IOTP Message to which this IOTP Message is a response. In this way all

	the IOTP Messages in an IOTP Transaction are unambiguously linked together. This field is required on every IOTP Message except the first IOTP Message in an IOTP Transaction.
SenderTradingRoleRef	The Element Reference (see section 3.5) of the Trading Role which has generated the IOTP message. It is used to identify the Net Locations (see section 3.9) of the Trading Role to which problems Technical Errors (see section 4.1) with any of Trading Blocks should be reported.
Xml:lang	Defines the language used by attributes or child elements within this component, unless overridden by an xml:lang attribute on a child element. See section 3.8 Identifying Languages.
LangPrefList	Optional list of Language codes that conform to [XML] Language Identification. It is used by the sender to indicate, in preference sequence, the languages that the receiver of the message ideally should use when generating a response. There is no obligation on the receiver to respond using one of the indicated languages, but using one of the languages is likely to provide an improved user experience.
CharSetPrefList	Optional list of Character Set identifiers that conform to [XML] Characters. It is used by the sender to indicate, in preference sequence, the character sets that the receiver of the message ideally should use when generating a response. There is no obligation on the receiver to respond using one of the character sets indicated, but using one of the character sets is likely to provide an improved user experience.
SoftwareId	This contains information which identifies the software which generated the IOTP Message. Its purpose is to help resolve interoperability problems that might occur as a result of incompatibilities between messages produced by different software. It is a single text string in the language defined by xml:lang. It must contain, as a minimum:

	o the name of the software manufacturer o the name of the software o the version of the software, and o the build of the software
TimeStamp	Where the device sending the message has an internal clock, it is set to the time at which the IOTP Message was created in [UTC] format.

3.3.3 Related To Component

The Related To Component links IOTP Transactions to either other IOTP Transactions or other events using the identifiers of those events. Its definition is as follows.

```
<!ELEMENT RelatedTo (PackagedContent) >
<!ATTLIST RelatedTo
 ID                  ID      #REQUIRED
 xml:lang            NMTOKEN #REQUIRED
 RelationshipType    NMTOKEN #REQUIRED
 Relation            CDATA   #REQUIRED
 RelnKeyWords        NMTOKENS #IMPLIED >
```

Attributes:

ID	An identifier which uniquely identifies the Related To Component within the IOTP Transaction.
xml:lang	Defines the language used by attributes or child elements within this component, unless overridden by an xml:lang attribute on a child element. See section 3.8 Identifying Languages.
RelationshipType	Defines the type of the relationship. Valid values are: o IotpTransaction. in which case the Packaged Content Element contains an IotpTransId of another IOTP Transaction o Reference in which case the Packaged Content Element contains the reference of some other, non-IOTP document. Values of RelationshipType are controlled under the procedures defined in section 12 IANA Considerations which also allows user defined values to be defined.

Relation	The Relation attribute contains a phrase in the language defined by xml:lang which describes the nature of the relationship between the IOTP transaction that contains this component and another IOTP Transaction or other event. The exact words to be used are left to the implementers of the IOTP software. The purpose of the attribute is to provide the Trading Roles involved in an IOTP Transaction with an explanation of the nature of the relationship between the transactions. Care should be taken that the words used to in the Relation attribute indicate the "direction" of the relationship correctly. For example: one transaction might be a refund for another earlier transaction. In this case the transaction which is a refund should contain in the Relation attribute words such as "refund for" rather than "refund to" or just "refund".
RelnKeyWords	This attribute contains keywords which could be used to help identify similar relationships, for example all refunds. It is anticipated that recommended keywords will be developed through examination of actual usage. In this version of the specification there are no specific recommendations and the keywords used are at the discretion of implementers.

Content:

PackagedContent	The Packaged Content (see section 3.7) contains data which identifies the related transaction. Its format varies depending on the value of the RelationshipType.

3.4 ID Attributes

IOTP Messages, Blocks (i.e. Transaction Reference Blocks and Trading Blocks), Trading Components (including the Transaction Id Component and the Signature Component) and some of their child elements are each given an XML "ID" attribute which is used to identify an instance of these XML elements. These identifiers are used so that one element can be referenced by another. All these attributes are given the attribute name ID.

The values of each ID attribute are unique within an IOTP transaction i.e. the set of IOTP Messages which have the same globally unique Transaction ID Component. Also, once the ID attribute of an element has been assigned a value it is never changed. This means that whenever an element is copied, the value of the ID attribute remains the same.

As a result it is possible to use these IDs to refer to and locate the content of any IOTP Message, Block or Component from any other IOTP Message, Block or Component in the same IOTP Transaction using Element References (see section 3.5).

This section defines the rules for setting the values for the ID attributes of IOTP Messages, Blocks and Components.

3.4.1 IOTP Message ID Attribute Definition

The ID attribute of the Message Id Component of an IOTP Message must be unique within an IOTP Transaction. It’s definition is as follows:

```
IotpMsgId_value  ::= IotpMsgIdPrefix IotpMsgIdSuffix
IotpMsgIdPrefix  ::= NameChar (NameChar)*
IotpMsgIdSuffix  ::= Digit (Digit)*
```

IotpMsgIdPrefix   Apart from messages which contain: an Inquiry Request Trading Block, an Inquiry Response Trading Block, a Ping Request Trading Block or a Ping Response Trading Block; then the same prefix is used for all messages sent by the Merchant or Consumer role as follows:

- o "M" - Merchant
- o "C" - Consumer

For messages which contain an Inquiry Request Trading Block or a Ping Request Trading Block, the prefix is set to "I" for Inquiry.

For messages which contain an Inquiry Response Trading Block or a Ping Response Trading Block, the prefix is set to "Q".

The prefix for the other roles in a trade is contained within the Organisation Component for the role and are typically set by the Merchant. The following is recommended as a guideline and must not be relied upon:

```
                         o "P" - First (only) Payment Handler
                         o "R" - Second Payment Handler
                         o "D" - Delivery Handler
                         o "C" - Deliver To

                        As a guideline, prefixes should be limited to one
                        character.

                        NameChar has the same definition as the [XML]
                        definition of NameChar.

   IotpMsgIdSuffix      The suffix consists of one or more digits. The
                        suffix must be unique within a Trading Role within
                        an IOTP Transaction. The following is recommended
                        as a guideline and must not be relied upon:

                         o the first IOTP Message sent by a trading role
                           is given the suffix "1"
                         o the second and subsequent IOTP Messages sent
                           by the same trading role are incremented by one
                           for each message
                         o no leading zeroes are included in the suffix

                        Put more simply the Message Id Component of the
                        first IOTP Message sent by a Consumer would have
                        an ID attribute of, "C1", the second "C2", the
                        third "C3" etc.

                        Digit has the same definition as the [XML]
                        definition of Digit.
```

3.4.2 Block and Component ID Attribute Definitions

The ID Attribute of Blocks and Components must also be unique within an IOTP Transaction. Their definition is as follows:

```
   BlkOrCompId_value ::= IotpMsgId_value "." IdSuffix
   IdSuffix ::= Digit (Digit)*

   IotpMsgId_value      The ID attribute of the Message ID Component of
                        the IOTP Message where the Block or Component is
                        first used.

                        In IOTP, Trading Components and Trading Blocks are
                        copied from one IOTP Message to another. The ID
                        attribute does not change when an existing Trading
                        Block or Component is copied to another IOTP
                        Message.
```

IdSuffix	The suffix consists of one or more digits. The suffix must be unique within the ID attribute of the Message ID Component used to generate the ID attribute. The following is recommended as a guideline and must not be relied upon:

- the first Block or Component sent by a trading role is given the suffix "1"
- the ID attributes of the second and subsequent Blocks or Components are incremented by one for each new Block or Component added to an IOTP Message
- no leading zeroes are included in the suffix

Put more simply, the first new Block or Component added to the second IOTP Message sent, for example, by a consumer would have a an ID attribute of "C2.1", the second "C2.2", the third "C2.3" etc.

Digit has the same definition as the [XML] definition of Digit.

3.4.3 Example of use of ID Attributes

The diagram below illustrates how ID attribute values are used.

```
*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*

      1st  IOTP MESSAGE                          2nd IOTP MESSAGE
    (e.g., from Merchant to                    (e.g., from Consumer to
           Consumer                              Payment Handler)

IOTP MESSAGE                               IOTP MESSAGE *
 |-Trans Ref Block. ID=M1.1                 |-Trans Ref Block.ID=C1.1*
 |  |-Trans Id Comp. ID = M1.2 ------------>|  |-Trans Id Comp.
 |  |                         Copy Element  |  |  ID=M1.2
 |  |-Msg Id Comp. ID = M1                  |  |-Msg Id Comp. ID=C1 *
 |                                          |
 |-Signature Block. ID=M1.8                 |-Signature Block.ID=C1.5*
 |  |-Sig Comp. ID=M1.15 ------------------>|  |-Comp. ID=M1.15
 |                            Copy Element  |
 |-Trading Block. ID=M1.3                   |-Trading Block.ID=C1.2 *
 |  |-Comp. ID=M1.4 -------------------------->|-Comp. ID=M1.4
 |  |                         Copy Element     |
 |  |-Comp. ID=M1.5 -------------------------->|-Comp. ID=M1.5
 |  |                         Copy Element     |
 |  |-Comp. ID=M1.6                            |-Comp. ID=C1.3 *
 |  |-Comp. ID=M1.7                            |-Comp. ID=C1.4 *
 |
 |-Trading Block. ID=M1.9
    |-Comp. ID=M1.10                             * = new elements
    |-Comp. ID=M1.11
    |-Comp. ID=M1.12
    |-Comp. ID=M1.13

*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
```

Figure 8 Example use of ID attributes

3.5 Element References

A Trading Component or one of its child XML elements, may contain an XML attribute that refers to another Block (i.e. a Transaction Reference Block or a Trading Block) or Trading Component (including a Transaction Id and Signature Component). These Element References are used for many purposes, a few examples include:

- o identifying an XML element whose Digest is included in a Signature Component,

- o referring to the Payment Handler Organisation Component which is used when making a Payment

An Element Reference always contains the value of an ID attribute of a Block or Component.

Identifying the IOTP Message, Trading Block or Trading Component which is referred to by an Element Reference, involves finding the XML element which:

- o belongs to the same IOTP Transaction (i.e. the Transaction Id Components of the IOTP Messages match), and

- o where the value of the ID attribute of the element matches the value of the Element Reference.

Note: The term "match" in this specification has the same definition as the [XML] definition of match.

An example of "matching" an Element Reference is illustrated in the example below.

```
   *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*

         1st  IOTP MESSAGE                        2nd IOTP MESSAGE
       (e.g., from Merchant to                  (e.g., from Consumer to
              Consumer                              Payment Handler)

   IOTP MESSAGE                                 IOTP MESSAGE
    |-Trans Ref Block. ID=M1.1     Trans ID    |-Trans RefBlock. ID=C1.1
    |  |-Trans Id Comp. ID = M1.2 <-Components-|->|-TransId Comp.ID=M1.2
    |  |                            must be    |  |
    |  |-Msg Id Comp. ID = M1      Identical   |  |-Msg Id Comp. ID=C1
    |                                  ^       |
    |-Signature Block. ID=M1.8         |       |-Signature Block.ID=C1.5
    |  |-Sig Comp. ID=M1.15            |       |  |-Comp. ID=M1.15
    |                                 AND      |
    |-Trading Block. ID=M1.3           |       |-Trading Block. ID=C1.2
    |  |-Comp. ID=M1.4                 |          |-Comp. ID=M1.4
    |  |                               v          |
    |  |-Comp. ID=M1.5 <-------- -ID Attribute    |-Comp. ID=M1.5
    |  |                          and El Ref      |
    |  |-Comp. ID=M1.6            values must     |-Comp. ID=C1.3
    |  |                             match--------|--> El Ref=M1.5
    |  |-Comp. ID=M1.7                            |-Comp. ID=C1.4
    |
    |-Trading Block. ID=M1.9
       |-Comp. ID=M1.10
       |-Comp. ID=M1.11
       |-Comp. ID=M1.12
       |-Comp. ID=M1.13

   *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
```

Figure 9 Element References

Note: Element Reference attributes are defined as "NMTOKEN" rather than "IDREF" (see [XML]). This is because an IDREF requires that the XML element referred to is in the same XML Document. With IOTP this is not necessarily the case.

3.6 Extending IOTP

Baseline IOTP defines a minimum protocol which systems supporting IOTP must be able to accept. As new versions of IOTP are developed, additional types of IOTP Transactions will be defined. In addition to this, Baseline and future versions of IOTP will support user extensions to IOTP through two mechanisms:

- o extra XML elements, and
- o new values for existing IOTP codes.

3.6.1 Extra XML Elements

The XML element and attribute names used within IOTP constitute an [XML Namespace] as identified by the xmlns attribute on the IotpMessage element. This allows IOTP to support the inclusion of additional XML elements within IOTP messages through the use of [XML Namespaces].

Using XML Namespaces, extra XML elements may be included at any level within an IOTP message including:

- o new Trading Blocks
- o new Trading Components
- o new XML elements within a Trading Component.

The following rules apply:

- o any new XML element must be declared according to the rules for [XML Namespaces]
- o new XML elements which are either Trading Blocks or Trading Components must contain an ID attributes with an attribute name of ID.

In order to make sure that extra XML elements can be processed properly, IOTP reserves the use of a special attribute, IOTP:Critical, which takes the values True or False and may appear in extra elements added to an IOTP message.

The purpose of this attribute is to allow an IOTP aware application to determine if the IOTP transaction can safely continue. Specifically:

- o if an extra XML element has an "IOTP:Critical" attribute with a value of "True" and an IOTP aware application does not know how to process the element and its child elements, then the IOTP transaction has a Technical Error (see section 4.1) and must fail.
- o if an extra XML element has an "IOTP:Critical" attribute with a value of "False" then the IOTP transaction may continue if the IOTP aware application does not know how to process it. In this case:

   - any extra XML elements contained within an XML element defined within the IOTP namespace, must be included with that element whenever the IOTP XML element is used or copied by IOTP

   - the content of the extra element must be ignored except that it must be included when it is used in the creation of a digest as part of the generation of a signature

- if an extra XML element has no "IOTP:Critical" attribute then it must be treated as if it had an "IOTP:Critical" attribute with a value of "True"

- if an XML element contains an "IOTP:Critical" attribute, then the value of that attribute is assumed to apply to all the child elements within that element

In order to ensure that documents containing "IOTP:Critical" are valid, it is declared as part of the DTD for the extra element as:

```
IOTP:Critical     (True | False ) 'True'
```

3.6.2 Opaque Embedded Data

If IOTP is to be extended using Opaque Embedded Data then a Packaged Content Element (see section 3.7) should be used to encapsulate the data.

3.7 Packaged Content Element

The Packaged Content element supports the concept of an embedded data stream, transformed to both protect it against misinterpretation by transporting systems and to ensure XML compatibility. Examples of its use in IOTP include:

- to encapsulate payment scheme messages, such as SET messages,

- to encapsulate a description of an order, a payment note, or a delivery note.

In general it is used to encapsulate one or more data streams.

This data stream has three standardised attributes that allow for identification, decoding and interpretation of the contents. Its definition is as follows.

```
   <!ELEMENT PackagedContent (#PCDATA) >
   <!ATTLIST PackagedContent
    Name             CDATA     #IMPLIED
    Content          NMTOKEN   "PCDATA"
    Transform (NONE|BASE64)    "NONE" >
```

Attributes:

```
   Name                  Optional. Distinguishes between multiple
                         occurrences of Packaged Content Elements at the
                         same point in IOTP. For example:
                           <ABCD>
                             <PackagedContent Name='FirstPiece'>
                               snroasdfnas934k
                             </PackagedContent>
                             <PackagedContent Name='SecondPiece'>
                               dvdsjnl5poidsdsflkjnw45
                             </PackagedContent>
                           </ABCD>

                         The name attribute may be omitted, for example if
                         there is only one Packaged Content element.

   Content               This identifies what type of data is contained
                         within the Content of the Packaged Content
                         Element. The valid values for the Content
                         attribute are as follows:
                          o PCDATA. The content of the Packaged Content
                            Element can be treated as PCDATA with no
                            further processing.
                          o MIME. The content of the Packaged Content
                            Element is a complete MIME item. Processing
                            should include looking for MIME headers inside
                            the Packaged Content Element.
                          o MIME:mimetype. The content of the Packaged
                            Content Element is MIME content, with the
                            following header "Content-Type: mimetype".
                            Although it is possible to have MIME:mimetype
                            with the Transform attribute set to NONE, it is
                            far more likely to have Transform attribute set
                            to BASE64. Note that if Transform is NONE is
                            used, then the entire content must still
                            conform to PCDATA. Some characters will need to
                            be encoded either as the XML default entities,
                            or as numeric character entities.
```

- XML. The content of the Packaged Content Element can be treated as an XML document. Entities and CDATA sections, or Transform set to BASE64, must be used to ensure that the Packaged Content Element contents are legitimate PCDATA.

Values of the Content attribute are controlled under the procedures defined in section 12 IANA Considerations which also allows user defined values to be defined.

Transform	This identifies the transformation that has been done to the data before it was placed in the content. Valid values are: o NONE. The PCDATA content of the Packaged Content Element is the correct representation of the data. Note that entity expansion must occur first (i.e. replacement of &amp; and &#9;) before the data is examined. CDATA sections may legitimately occur in a Packaged Content Element where the Transform attribute is set to NONE. o BASE64. The PCDATA content of the Packaged Content Element represents a BASE64 encoding of the actual content.

Content:

PCDATA	This is the actual data which has been embedded. The format of the data and rules on how to decode it are contained in the Content and the Transform attributes

Note that any special details, especially custom attributes, must be represented at a higher level.

3.7.1 Packaging HTML

The packaged content may contain HTML. In this case the following conventions are followed:

- references to any documents, images or other things, such as sounds or web pages, which can affect the recipient’s understanding of the data which is being packaged must refer to other Packaged Elements contained within the same parent element, e.g., an Order Description

- o if more than one Packaged Content element is included within a parent element in order to meet the previous requirement, then the Name attribute of the top level Packaged Content from which references to all other Packaged Elements can be determined, should have a value of Main

- o relative references to other documents, images, etc. from one Packaged Content element to another are realised by setting the value of the relative reference to the Name attribute of another Packaged Content element at the same level and within the same parent element

- o no external references that require the reference to be resolved immediately should be used. As this could make the HTML difficult or impossible to display completely

- o [MIME] is used to encapsulate the data inside each Packaged Element. This means that the information in the MIME header used to identify the type of data which has been encapsulated and therefore how it should be displayed.

If the above conventions are not followed by, for example, including external references which must be resolved, then the recipient of the HTML should be informed.

Note: As an implementation guideline the values of the Name Attributes allocated to Packaged Content elements should make it possible to extract each Packaged Content into a directory and then display the HTML directly

3.7.2 Packaging XML

Support for XML is recommended. When XML needs to be displayed, for example to display the content of an Order Description to a Consumer, then implementers should follow the latest recommendations of the World Wide Web Consortium.

Note: At the time of writing this specification, standards are under development that specify XML style sheets that show how XML documents should be displayed. See:

- o "Extensible Stylesheet Language (XSL) Specification" at http://www.w3.org/TR/WD-xsl, and

- o "Associating stylesheets with XML documents" at http://www.w3.org/TR/xml-stylesheet.

Once these standards become W3C "Recommendations", then it is anticipated that this specification will be amended if practical.

3.8 Identifying Languages

IOTP uses [XML] Language Identification to specify which languages are used within the content and attributes of IOTP Messages.

The following principles have been used in order to determine which XML elements contain an xml:lang Attributes:

- o a mandatory xml:lang attribute is contained on every Trading Component which contains attributes or content which may need to be displayed or printed in a particular language
- o an optional xml:lang attribute is included on child elements of these Trading Components. In this case the value of xml:lang, if present, overrides the value for the Trading Component.

xml:lang attributes which follow these principles are included in the Trading Components and their child XML elements defined in section 7.

A sender of a message, typically a Consumer can indicate a preference for a language, and a character set by specifying a list of preferred languages/character sets in a Message Id Component (see section 3.3.2). Note that there is no obligation on the receiver of such a message to respond using one of the listed languages/character sets as they may not have the technology to be able to do it. It also means that the ability to handle these lists is not a requirement for conformance to this specification. However the ability to respond, for example using one of the stated languages/character sets is likely to provide a better user experience.

3.9 Secure and Insecure Net Locations

IOTP contains several "Net Locations" which identify places where, typically, IOTP Messages may be sent. Net Locations come in two types:

- o "Secure" Net Locations which are net locations where privacy of data is secured using, for example, encryption methods such as [SSL/TLS], and
- o "Insecure" Net Locations where privacy of data is not assured.

Note that either a Secure Net Location or an Insecure Net Location or both must be present.

If only one of the two Net Locations is present, then the one present must be used.

Where both types of net location are present then either may be used depending on the preference of the sender of the message.

3.10 Cancelled Transactions

Any Trading Role involved in an IOTP transaction may cancel that transaction at any time.

3.10.1 Cancelling Transactions

IOTP Transactions are cancelled by sending an IOTP message containing just a Cancel Block with an appropriate Status Component to the other Trading Role involved in the Trading Exchange.

Note: The Cancel Block can be sent asynchronously of any other IOTP Message. Specifically it can be sent either before sending or after receiving an IOTP Message from the other Trading Role

If an IOTP Transaction is cancelled during a Trading Exchange (i.e. the interval between sending a "request" block and receiving the matching "response" block) then the Cancel Block is sent to the same location as the next IOTP Message in the Trading Exchange would have been sent.

If a Consumer cancels a transaction after a Trading Exchange has completed (i.e. the "response" block for the Trading Exchange has been received), but before the IOTP Transaction has finished then the Consumer sends a Cancel Block with an appropriate Status Component to the net location identified by the SenderNetLocn or SecureSenderNetLocn contained in the Protocol Options Component (see section 7.1) contained in the TPO Block (see section 8.1) for the transaction. This is normally the Merchant Trading Role.

A Consumer should not send a Cancel Block after the IOTP Transaction has completed. Cancelling a complete transaction should be treated as a technical error.

After cancelling the IOTP Transaction, the Consumer should go to the net location specified by the CancelNetLocn attribute contained in the Trading Role Element for the Organisation that was sent the Cancel Block.

A non-Consumer Trading Role should only cancel a transaction:

o after a request block has been received and o before the response block has been sent

If a non-Consumer Trading Role cancels a transaction at any other time it should be treated by the recipient as an error.

3.10.2 Handling Cancelled Transactions

If a Cancel Block is received by a Consumer at a point in the IOTP Transaction when cancellation is allowed, then the Consumer should stop the transaction.

If a Cancel Block is received by a non-Consumer role, then the Trading Role should anticipate that the Consumer may go to the location specified by the CancelNetLocn attribute contained in the Trading Role Element for the Trading Role.

4. IOTP Error Handling

IOTP is designed as a request/response protocol where each message is composed of a number of Trading Blocks which contain a number of Trading Components. There are several interrelated considerations in handling errors, re-transmissions, duplicates, and the like. These factors mean IOTP aware applications must manage message flows more complex than the simple request/response model. Also a wide variety of errors can occur in messages as well as at the transport level or in Trading Blocks or Components.

This section describes at a high level how IOTP handles errors, retries and idempotency. It covers:

- o the different types of errors which can occur. This is divided into:
  - "technical errors" which are independent of the purpose of the IOTP Message,
  - "business errors" which indicate that there is a problem specific to the process (e.g., payment or delivery) which is being carried out, and
- o the depth of the error which indicates whether the error is at the transport, message or block/component level
- o how the different trading roles should handle the different types of messages which they may receive.

4.1 Technical Errors

Technical Errors are those which are independent of the meaning of the message. This means, they can affect any attempt at IOTP communication. Typically they are handled in a standard fashion with a limited number of standard options for the user. Specifically these are:

- o retrying the transmission, or

- o cancelling the transaction.

When communications are operating sufficiently well, a technical error is indicated by an Error Component (see section 7.21) in an Error Block (see section 8.17) sent by the party which detected the error in an IOTP message to the party which sent the erroneous message.

If communications are too poor, a message which was sent may not reach its destination. In this case a time-out might occur.

The Error Codes associated with Technical Errors are recorded in the Error Component which lists all the different technical errors which can be set.

4.2 Business Errors

Business Errors may occur when the IOTP messages are "technically" correct. They are connected with a particular process, for example, an offer, payment, delivery or authentication, where each process has a different set of possible business errors.

For example, "Insufficient funds" is a reasonable payment error but makes no sense for a delivery while "Back ordered" is a reasonable delivery error but not meaningful for a payment. Business errors are indicated in the Status Component (see section 7.16) of a "response block" of the appropriate type, for example a Payment Response Block or a Delivery Response Block. This allows whatever additional response related information is needed to accompany the error indication.

Business errors must usually be presented to the user so that they can decide what to do next. For example, if the error is insufficient funds in a Brand Independent Offer (see section 9.1.2.2), the user might wish to choose a different payment instrument/account of the same brand or a different brand or payment system. Alternatively, if the IOTP based implementation allows it and it makes sense for that instrument, the user might want to put more funds into the instrument/account and try again.

4.3 Error Depth

The three levels at which IOTP errors can occur are the transport level, the message level, and the block level. Each is described below.

4.3.1 Transport Level

This level of error indicates a fundamental problem in the transport mechanism over which the IOTP communication is taking place.

All transport level errors are technical errors and are indicated by either an explicit transport level error indication, such as a "No route to destination" error from TCP/IP, or by a time out where no response has been received to a request.

The only reasonable automatic action when faced with transport level errors is to retry and, after some number of automatic retries, to inform the user.

The explicit error indications that can be received are transport dependent and the documentation for the appropriate IOTP Transport supplement should be consulted for errors and appropriate actions.

Appropriate time outs to use are a function of both the transport being used and of the payment system if the request encapsulates payment information. The transport and payment system specific documentation should be consulted for time out and automatic retry parameters. Frequently there is no way to directly inform the other party of transport level errors but they should generally be logged and if automatic recovery is unsuccessful and there is a human user, the user should be informed.

4.3.2 Message Level

This level of error indicates a fundamental technical problem with an entire IOTP message. For example, the XML is not "Well Formed", or the message is too large for the receiver to handle or there are errors in the Transaction Reference Block (see section 3.3) so it is not possible to figure out what transaction the message relates to.

All message level errors are technical errors and are indicated by Error Components (see section 7.21) sent to the other party. The Error Component includes a Severity attribute which indicates whether the error is a Warning and may be ignored, a TransientError which indicates that a retry may resolve the problem or a HardError in which case the transaction must fail.

The Technical Errors (see section 7.21.2 Error Codes) that are Message Level errors are:

- XML not well formed. The document is not well formed XML (see [XML])
- XML not valid. The document is not valid XML (see [XML])
- block level technical errors (see section 4.3.3) on the Transaction Reference Block (see section 3.3) and the Signature Block only. Checks on these blocks should only be carried out if the XML is valid

Note that checks on the Signature Block include checking, where possible, that each Signature Component is correctly calculated. If the Signature is incorrectly calculated then the data that should have been covered by the signature can not be trusted and must be treated as erroneous. A description of how to check a signature is correctly calculated is contained in section 6.2.

4.3.3 Block Level

A Block level error indicates a problem with a block or one of its components in an IOTP message (apart from Transaction Reference or Signature Blocks). The message has been transported properly, the overall message structure and the block/component(s) including the Transaction Reference and Signature Blocks are meaningful but there is some error related to one of the other blocks.

Block level errors can be either:

- technical errors, or
- business errors

Technical Errors are further divided into:

- Block Level Attribute and Element Checks, and
- Block and Component Consistency Checks
- Transient Technical Errors

If a technical error occurs related to a block or component, then an Error Component is generated for return.

4.3.3.1 Block Level Attribute and Element Checks

Block Level Attribute and Element Checks occur only within the same block. Checks which involve cross-checking against other blocks are covered by Block and Component Consistency Checks.

The Block Level Attribute & Element checks are:

- checking that each attribute value within each element in a block conforms to any rules contained within this IOTP specification
- checking that the content of each element conforms to any rules contained within this IOTP specification
- if the previous checks are OK, then checking the consistency of attribute values and element content against other attribute values or element content within any other components in the same block.

4.3.3.2 Block and Component Consistency Checks

Block and Component Consistency Checks consist of:

- checking that the combination of blocks and/or components present in the IOTP Message are consistent with the rules contained within this IOTP specification
- checking for consistency between attributes and element content within the blocks within the same IOTP message.
- checking for consistency between attributes and elements in blocks in this IOTP message and blocks received in earlier IOTP messages for the same IOTP transaction

If the block passes the "Block Level Attribute and Element Checks" and the "Block and Component Consistency Checks" then it is processed either by the IOTP Aware application or perhaps by some "back-end" system such as a payment server.

4.3.3.3 Transient Technical Errors

During the processing of the Block some temporary failure may occur that can potentially be recovered by the other trading role re-transmitting, at some slightly later time, the original message that they sent. In this case the other role is informed of the Transient

Error by sending them an Error Component (see section 7.21) with the Severity Attribute set to TransientError and the MinRetrySecs attribute set to some value suitable for the Transport Mechanism and/or payment protocol being used (see appropriate Transport and payment protocol Supplements).

Note that transient technical errors can be generated by any of the Trading Roles involved in transaction.

4.3.3.4 Block Level Business Errors

If a business error occurs in a process such as a Payment or a Delivery, then the appropriate type of response block is returned containing a Status Component (see section 7.16) with the ProcessState attribute set to Failed and the CompletionCode indicating the nature of the problem.

Some business errors may be "transient" in that the Consumer role may be able to recover and complete the transaction in some other way. For example if the Credit Card that a consumer provided had insufficient funds for a purchase, then the Consumer may recover by using a different credit card.

Recovery from "transient" business errors is dependent on the CompletionCode. See the definition of the Status Component for what is possible.

Note that no Error Component or Error Block is generated for business errors.

4.4 Idempotency, Processing Sequence, and Message Flow

IOTP messages are actually a combination of blocks and components as described in 3.1.1 IOTP Message Structure. Especially in future extensions of IOTP, a rich variety of combinations of such blocks and components can occur. It is important that the multiple transmission/receipt of the "same" request for an action that will change state does not result in that action occurring more than once. This is called idempotency. For example, a customer paying for an order would want to pay the full amount only once. Most network transport mechanisms have some probability of delivering a message more than once or not at all, perhaps requiring retransmission. On the other hand, a request for status can reasonably be repeated and should be processed fresh each time it is received.

Correct implementation of IOTP can be modelled by a particular processing order as detailed below. Any other method that is indistinguishable in the messages sent between the parties is equally acceptable.

4.5 Server Role Processing Sequence

"Server roles" are any Trading Role which is not the Consumer role. They are "Server roles" since they typically receive a request which they must service and then produce a response. However server roles can also initiate transactions. More specifically Server Roles must be able to:

- o Initiate a transaction (see section 4.5.1). These are divided into:
  - - payment related transactions and
  - - infrastructure transactions
- o Accept and process a message received from another role (see section 4.5.2). This includes:
  - - identifying if the message belongs to a transaction that has been received before
  - - handling duplicate messages
  - - generating Transient errors if the servers that process the input message are too busy to handle it
  - - processing the message if it is error free, authorised and, if appropriate, producing a response to send back to the other role
- o Cancel a current transaction if requested (see section 4.5.3)
- o Re-transmit messages if a response was expected but has not been received in a reasonable time (see section 4.5.4).

4.5.1 Initiating Transactions

Server Roles may initiate a variety of different types of transaction. Specifically:

- o an Inquiry Transaction (see section 9.2.1)
- o a Ping Transaction (see section 9.2.2)

- an Authentication Transaction (see section 9.1.6)
- a Payment Related Transaction such as:
  - a Deposit (see section 9.1.7)
  - a Purchase (see section 9.1.8)
  - a Refund (see section 9.1.9)
  - a Withdrawal (see section 9.1.10)
  - a Value Exchange (see section 9.1.11)

4.5.2 Processing Input Messages

Processing input messages involves the following:

- checking the structure and identity of the message
- checking for and handling duplicate messages
- processing non-duplicate original messages which includes:
  - checking for errors, then if no errors are found
  - processing the message to produce an output message if appropriate

Each of these is discussed in more detail below.

4.5.2.1 Checking Structure and Message Identity

It is critical to check that the message is "well formed" XML and that the transaction identifier (IotpTransId attribute on the TransId Component) within the IOTP message can be successfully identified since an IotpTransId will be needed to generate a response.

If the input message is not well formed then generate an Error Component with a Severity of HardError and ErrorCode of XmlNotWellFrmd.

If the message is well formed but the IotpTransId cannot be identified then generate an ErrorComponent with:

- a Severity of HardError and an ErrorCode of AttMissing,

- a PackagedContent containing "IotpTransId" - the missing attribute.

Insert the Error Component inside an Error Block with a new TransactionId component with a new IotpTransId and return it to the sender of the original message.

4.5.2.2 Checking/Handling Duplicate Messages

If the input message can be identified as potentially a valid input message then check to see if an "identical" input message has been received before. Identical means that all blocks, components, elements, attribute values and element content in the input message are the same.

Note: The recommended way of checking for identical messages is to check for equal values of their [DOM-HASH]

If an identical message has been received before then check to see if the processing of the previous message has completed.

If processing has not completed then generate an Error Component with a Severity of Transient Error and an Error Code of MsgBeingProc to indicate the message is being processed and send it back to the sender of the Input Message requesting that the original message be resent after an appropriate period of time.

Otherwise, if processing has completed and resulted in an output message then retrieve the last message that was sent and send it again.

If the message is not a duplicate then it should be processed.

4.5.2.3 Processing Non-Duplicate Message

Once it’s been established that the message is not a duplicate, then it can be processed. This involves:

- checking that a server is available to handle the message, generating a Transient Error if it is not
- checking the Transaction is Not Already in error or cancelled
- validating the input message. This includes:
  - checking for message level errors
  - checking for block level errors

  - checking any encapsulated data

- checking for errors in the sequence that blocks have been received
- generating error components for any errors that result
- if neither hard errors nor transient errors result, then processing the message and generating an output message, if required, for return to the sender of the Input Message

Note: This approach to handling of duplicate input messages means, if absolutely "identical" messages are received then absolutely "identical" messages are returned. This also applies to Inquiry and Ping transactions when in reality the state of a transaction or the processing ability of the servers may have changed. If up-to-date status of transactions or servers is required, then an IOTP transaction with a new value for the ID attribute of the MsgId component must be used.

Each of the above steps is discussed below.

CHECKING A SERVER IS AVAILABLE

The process that is handling the input message should check that the rest of the system is not so busy that a response in a reasonable time cannot be produced.

If the server is too busy, then it should generate an Error Component with a Severity of Transient Error and an Error Code of SystemBusy and send it back to the sender of the Input Message requesting that the original message be resent after an appropriate period of time.

Note: Some servers may occasionally become very busy due to unexpected increases in workload. This approach allows short peaks in workloads to be handled by delaying the input of messages by asking the sender of the message to resubmit later.

CHECKING THE TRANSACTION IS NOT ALREADY IN ERROR OR CANCELLED

Check that:

- previous messages received or sent did not contain or result in Hard Errors, and
- the Transaction has not been cancelled by either the Consumer or the Server Trading Role

If it has then, ignore the message. A transaction with hard errors or that has been cancelled, cannot be restarted.

CHECK FOR MESSAGE AND BLOCK LEVEL ERRORS

If the transaction is still OK then check for message level errors. This involves:

- o checking the XML is valid
- o checking that the elements, attributes and content of the Transaction Reference Block are without error and conform to this specification
- o checking the digital signature which involves:
  - checking that the Signature value is correctly calculated, and
  - the hash values in the digests are correctly calculated where the source of the hash value is available.

Checking for block level errors involves:

- o checking within each block (apart from the Transaction Reference Block) that:
  - the attributes, elements and element contents are valid
  - the values of the attributes, elements and element contents are consistent within the block
- o checking that the combination of blocks are valid
- o checking that the values of the attribute, elements and element contents are consistent between the blocks in the input message and blocks in earlier messages either sent or received. This includes checking that the presence of a block is valid for a particular transaction type

If the message contains any encapsulated data, then if possible check the encapsulated data for errors using additional software to check the data where appropriate.

4.5.2.4 Check for Errors in Block Sequence

Note: For reasons of brevity, the following explanations of how to check for errors in Block sequence, the phrase "refers to an IOTP transaction" is interpreted as "is contained in an IOTP Message where the Trans Ref Block contains an IotpTransId that refers to". So, for example, " If an Error or Cancel Block refers to an IOTP transaction that is not recognised then ..." should be interpreted as " If an Error or Cancel Block is contained in an IOTP Message where the Trans Ref Block contains an IotpTransId that refers to an IOTP transaction that is not recognised then ...

Errors in the sequence that blocks arrive depends on the block. Blocks where checking for sequence is required are:

- Error and Cancel Blocks. If an Error or Cancel Block refers to an IOTP transaction that is not recognised then it is a Hard Error. Do not return an error if Error or Cancel Blocks have been received for the IOTP Transaction before to avoid looping.

- Inquiry Request and Response Blocks. If an Inquiry Request or an Inquiry Response Block refers to an IOTP transaction that is not recognised then it is a Hard Error

- Authentication Request Block. If an Authentication Request Block refers to an IOTP transaction that is recognised it is a Hard Error

- Authentication Response Block. Check as follows:

  - if an Authentication Response Block does not refer to an IOTP transaction that is recognised it is a Hard Error, otherwise

  - if the Authentication Response Block doesn’t refer to an Authentication Request that had been previously sent then it is a Hard Error, otherwise

  - if an Authentication Response for the same IOTP transaction has been received before and the Authentication was successful then it is a Hard Error.

- Authentication Status Block. Check as follows:

  - if an Authentication Status Block does not refer to an IOTP transaction that is recognised it is a Hard Error, otherwise

  - if the Authentication Status Block doesn’t refer to an Authentication Response that had been previously sent then it is a Hard Error, otherwise

  - if an Authentication Status for the same IOTP transaction has been received before then it is a Warning Error

- o TPO Selection Block (Merchant only). Check as follows:

  - if the TPO Selection Block doesn’t refer to an IOTP Transaction that is recognised then it is a Hard Error, otherwise

  - if the TPO Selection Block refers to an IOTP Transaction where a TPO Block and Offer Response (in one message) had previously been sent then it is a Hard Error, otherwise

  - if the TPO Selection Block does not refer to an IOTP Transaction where a TPO Block only (i.e. without an Offer Response) had previously been sent then it is a Hard Error, otherwise

  - if a TPO Selection Block for the same TPO Block has been received before then it is a Hard Error

- o Payment Request Block (Payment Handler only). Check as follows:

  - if the Payment Request Block refers to an IOTP Transaction that is not recognised then its OK, otherwise

  - if the Payment Request Block refers to IOTP Transaction that was not for a Payment then it is a Hard Error, otherwise

  - if there was a previous payment that failed with a non-recoverable Completion Code then it is a Hard Error, otherwise

  - if a previous payment is still in progress then it is a Hard Error

- o Payment Exchange Block (Payment Handler only). Check as follows:

  - if the Payment Exchange Block doesn’t refer to an IOTP Transaction that is recognised then it is a Hard Error, otherwise

  - if the Payment Exchange doesn’t refer to an IOTP Transaction where a Payment Exchange had previously been sent then it a Hard Error

- o Delivery Request (Delivery Handler Only). If the Delivery Request Block refers to an IOTP Transaction that is recognised by the Server then it is a Hard Error

If any Error Components have been generated then collect them into an Error Block for sending to the sender of the Input message. Note that Error Blocks should be sent back to the sender of the message and to the ErrorLogNetLocn for the Trading Role of the sender if one is specified.

Note: The above checking on the sequence of Authentication Responses and Payment Requests supports the Consumer re-submitting a repeat action request since the previous one failed, for example:

- because they did not know the correct response (e.g., a password) on an authentication or,
- they were unable to pay as there were insufficient funds on a credit card

PROCESS THE ERROR FREE INPUT MESSAGE

If the input message passes the previous checks then it can be processed to produce an output message if required. Note that:

- Inquiry Requests on Ping Transactions should be ignored
- if the Input message contains an Error Block with a Transient Error then wait for the required time then resend the previous message, if a response to the earlier message has not been received
- if the input message contains a Error Component with a HardError or a Cancel Block then stop all further processing of the transaction. This includes suppressing the sending of any messages currently being generated or responding to any new non-duplicate messages that are received
- processing of encapsulated messages (e.g., Payment Protocol Messages) may result in additional transient errors
- a digital signature can only safely be generated once all the blocks and components have been generated and it is known which elements in the message need to be signed.

If an output message is generated then it should be saved so that it can be resent as required if an identical input message is received again. Note that output messages that contain transient errors are not saved so that they can be processed afresh when the input message is received again.

4.5.3 Cancelling a Transaction

This process is used to cancel a transaction running on an IOTP server. It is initiated by some other process as a result of an external request from another system or server that is being run by the same Trading Role. The processing required is as follows:

- o if the IotpTransId of the transaction to be cancelled is not recognised, or complete then fail the request, otherwise
- o if the IotpTransId refers to a Ping Transaction then fail the request, otherwise
- o determine which Document Exchange to cancel and generate a Cancel Block and send it to the other party

Note: Cancelling a transaction on an IOTP server typically arises for a business reason. For example a merchant may have attempted authentication several times without success and as a result decides to cancel the transaction. Therefore the process that decides to take this action needs to send a message from the process/server that made the business decision to the IOTP server with the instruction that the IOTP transaction should be cancelled.

4.5.4 Retransmitting Messages

The server should periodically check for transactions where a message is expected in return but none has been received after a time that is dependent on factors such as:

- o the Transport Mechanism being used;
- o the time required to process encapsulated messages (e.g., Payment messages) and
- o whether or not human input is required.

If no message has been received the original message should be resent. This should occur up to a maximum number of times dependent on the reliability of the Transport Mechanism being used.

If no response is received after the required time then the Transaction should be "timed out". In this case, set the process state of the transaction to Failed, and a completion code of either:

- o TimedOutRcvr if the transaction can potentially recovered later, or

- o TimedOutNoRcvr if the transaction is non-recoverable

4.6 Client Role Processing Sequence

The "Client role" in IOTP is the Consumer Trading Role.

Note: A company or Organisation that is a Merchant, for example, may take on the Trading Role of a Consumer when making purchases or downloading or withdrawing electronic cash.

More specifically the Consumer Role must be able to:

- o Initiate a transaction (see section 4.6.1). These are divided into:
  - payment related transactions and
  - infrastructure transactions
- o Accept and process a message received from another role (see section 4.6.2). This includes:
  - identifying if the message belongs to a transaction that has been received before
  - handling duplicate messages
  - generating Transient errors if the servers that process the input message are too busy to handle it
  - processing the message if it is error free and, if appropriate, producing a response to send back to the other role
- o Cancel a current transaction if requested, for example by the User (see section 4.6.3)
- o Re-transmit messages if a response was expected but has not been received in a reasonable time (see section 4.6.4).

4.6.1 Initiating Transactions

The Consumer Role may initiate a number of different types of transaction. Specifically:

- o an Inquiry Transaction (see section 9.2.1)
- o a Ping Transaction (see section 9.2.2)

o an Authentication Transaction (see section 9.1.6)

4.6.2 Processing Input Messages

Processing of Input Messages for a Consumer Role is the same as for an IOTP Server (see section 4.5.2) except in the area of checking for Errors in Block Sequence (for an IOTP Server see section 4.5.2.4). This is described below

Note: The description of the processing for an IOTP Server includes consideration of multi-threading of input messages and multi-tasking of requests. For the Consumer Role - particularly if running on a stand-alone system such as a PC - use of multi-threading is a decision of the implementer of the consumer role IOTP solution.

4.6.2.1 Check for Errors in Block Sequence

The handling of the following blocks is the same as for an IOTP Server (see section 4.5.2.4) except that the Consumer Role is substituted for IOTP Server Role:

o Error and Cancel Blocks,

o Inquiry Request and Response Blocks,

o Authentication Request, Response and Status Blocks.

For the other blocks a Consumer role might receive, the potential errors in the sequence that blocks arrive depends on the block. Blocks where checking for sequence is required are:

o TPO Block. Check as follows:

- if the input message also contains an Authentication Request block and an Offer Response Block then there is a Hard Error, otherwise
- if the input message also contains an Authentication Request block and Authentication Status block then there is Hard Error otherwise,
- if the input message also contains an Authentication Request block and the IOTP Transaction is recognised by the Consumer role’s system, then there is a Hard Error, otherwise

  - if the input message also contains an Authentication Status block and the IOTP Transaction is not recognised by the Consumer role’s system then there is a Hard Error, otherwise

  - if input message also contains an Authentication Status Block and the Authentication Status Block has not been sent after an earlier Authentication Response message then there is a hard error

  - if input message also contains an Offer Response Block and the IOTP Transaction is recognised by the Consumer role’s system then there is a Hard Error, otherwise

  - if the TPO Block occurs on its own and the IOTP Transaction is recognised by the Consumer role’s system then there is a Hard Error

- o Offer Response Block. Check as follows:

  - if the Offer Response Block is part of a Brand Independent Offer Exchange (see section 9.1.2.2) then there is no sequence checking as it is part of the first message received, otherwise

  - if the Offer Response Block is not part of an IOTP Transaction that is recognised by the Consumer role then there is a Hard Error, otherwise

  - if the Offer Response Block does not refer to an IOTP transaction where a TPO Selection Block was the last message sent then there is a Hard Error

- o Payment Exchange Block. Check as follows:

  - if the Payment Exchange Block doesn’t refer to an IOTP Transaction that is recognised by the Consumer role’s system then there is a Hard Error, otherwise

  - if the Payment Exchange doesn’t refer to an IOTP Transaction where either a Payment Request or a Payment Exchange block was most recently sent then there is a Hard Error

- o Payment Response Block. Check as follows:

  - if the Payment Response Block doesn’t refer to an IOTP Transaction that is recognised by the Consumer role’s system then there is a Hard Error, otherwise

   - if the Payment Response doesn't refer to an IOTOP Transaction where either a Payment Request or a Payment Exchange block was most recently sent then there is a Hard Error

- Delivery Response Block. Check as follows:

   - if the Delivery Response Block doesn't refer to an IOTP Transaction that is recognised by the Consumer role's system then there is a Hard Error, otherwise

   - If the Delivery Response doesn't refer to an IOTP Transaction where either a Payment Request or a Payment Exchange block was most recently sent then there is a Hard Error

4.6.3 Cancelling a Transaction

This process cancels a current transaction on an Consumer role's system as a result of an external request from the user, or another system or server in the Consumer's role. The processing is the same as for an IOTP Server (see section 4.5.3).

4.6.4 Retransmitting Messages

The process of retransmitting messages is the same as for an IOTP Server (see section 4.5.4).

5. Security Considerations

This section considers, from an IETF perspective how IOTP addresses security. The next section (see section 6. Digital Signatures and IOTP) describes how IOTP uses Digital Signatures when these are needed.

This section covers:

- determining whether to use digital signatures

- data privacy, and

- payment protocol security.

5.1 Determining whether to use digital signatures

The use of digital signatures within IOTP are entirely optional. IOTP can work successfully entirely without the use of digital signatures.

Ultimately it is up to the Merchant, or other trading role, to decide whether IOTP Messages will include signatures, and for the Consumer to decide whether carrying out a transaction without signatures is an acceptable risk. If Merchants discover that transactions without signatures are not being accepted, then they will either:

- start using signatures,
- find a method of working which does not need signatures, or
- accept a lower volume and value of business.

A non-exhaustive list of the reasons why digital signatures might be used follows:

- the Merchant (or other trading role) wants to demonstrate that they can be trusted. If, for example, a merchant generates an Offer Response Signature (see section 7.19.2) using a certificate from a trusted third party, known to the Consumer, then the Consumer can check the signature and certificate and so more reasonably rely on the offer being from the actual Organisation the Merchant claims to be. In this case signatures using asymmetric cryptography are likely to be required
- the Merchant, or other Trading Role, want to generate a record of the transaction that is fit for a particular purpose. For example, with appropriate trust hierarchies, digital signatures could be checked by the Consumer to determine:
  - if it would be accepted by tax authorities as a valid record of a transaction, or
  - if some warranty, for example from a "Better Business Bureau" orsimilar was being provided
- the Payment Handler, or Delivery Handler, needs to know that the request is unaltered and authorised. For example, in IOTP, details of how much to pay is sent to the Consumer in the Offer Response and then forwarded to the Payment Handler in a Payment Request. If the request is not signed, the Consumer could change the amount due by, for example, removing a digit. If the Payment Handler has no access to the original payment information in the Offer Response, then, without signatures, the Payment Handler cannot be sure that the data has not been altered. Similarly, if the payment information is not digitally signed, the Payment Handler cannot be sure who is the Merchant that is requesting the payment
- a Payment Handler or Delivery Handler wants to provide a non-refutable record of the completion status of a Payment or Delivery. If a Payment Response or Delivery Response is signed,

then the Consumer can later use the record of the Payment or Delivery to prove that it occurred. This could be used, for example, for customer care purposes.

A non-exhaustive list of the reasons why digital signatures might not be used follows:

- trading roles are combined therefore changes to data made by the consumer can be detected. One of the reasons for using signatures is so that one trading role can determine if data has been changed by the Consumer or some other party. However if the trading roles have access to the necessary data, then it might be possible to compare, for example, the payment information in the Payment Request with the payment information in the Offer Response. Access to the data necessary could be realised by, for example, the Merchant and Payment Handler roles being carried out by the same Organisation on the same system, or the Merchant and Payment Handler roles being carried out on different systems but the systems can communicate in some way. (Note this type of communication is outside the current scope of IOTP)
- the processing cost of the cryptography is too high. For example, if a payment is being made of only a few cents, the cost of carrying out all the cryptography associated with generating and checking digital signatures might make the whole transaction uneconomic. Co-locating trading roles, could help avoid this problem.

5.2 Symmetric and Asymmetric Cryptography

The advantage of using symmetric keys with IOTP is that no Public Key Infrastructure need be set up and just the Merchant, Payment Handler and Delivery Handler need to agree on the shared secrets to use.

However the disadvantage of symmetric cryptography is that the Consumer cannot easily check the credentials of the Merchant, Payment Handler, etc. that they are dealing with. This is likely to reduce, somewhat, the trust that the Consumer will have carrying out the transaction.

However it should be noted that even if asymmetric cryptography is being used, the Consumer does not NEED to be provided with any digital certificates as the integrity of the transaction is determined by, for example, the Payment Handler checking the Offer Response Signature copied to the Payment Request.

Note that symmetric, asymmetric or both types of cryptography may be used in a single transaction.

5.3 Data Privacy

Privacy of information is provided by sending IOTP Messages between the various Trading Roles using a secure channel such as [SSL/TLS]. Use of a secure channel within IOTP is optional.

5.4 Payment Protocol Security

IOTP is designed to be completely blind to the payment protocol being used to effect a payment. From the security perspective, this means that IOTP neither helps, nor hinders, the achievement of payment security.

If it is necessary to consider payment security from an IOTP perspective, then this should be included in the payment protocol supplement which describes how IOTP supports that payment protocol.

However what IOTP is designed to do is to use digital signatures to bind together the record, contained in a "response" message, of each trading exchange in a transaction. For example IOTP can bind together: an Offer, a Payment and a Delivery.

6. Digital Signatures and IOTP

IOTP can work successfully without using any digital signatures although in an open networking environment it will be less secure - see 5. Security Considerations for a description of the factors that need to be considered.

However, this section describes how to use digital signatures in the many situations when they will be needed. Topics covered are:

- an overview of how IOTP uses digital signatures
- how to check a signature is correctly calculated
- how Payment Handlers and Delivery Handlers check they can carry out payments or deliveries on behalf of a Merchant.

6.1 How IOTP uses Digital Signatures

In general, signatures when used with IOTP:

- are always treated as IOTP Components (see section 7)
- contain digests of one or more IOTP Components or Trading Blocks, possibly including other Signature Components, in any IOTP message within the same IOTP Transaction

- identify:

  - which Organisation signed (originated) the signature, and

  - which Organisation(s) should process the signature in order to check that the Action the Organisation should take can occur.

Digital certificates may be associated with digital signatures if asymmetric cryptography is being used. However if symmetric cryptography is being used, then the digital certificate will be replaced by some identifier of the secret key to use.

The way in which Signatures Components digest one or more elements is illustrated in the figure below.

```
 *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*

 IOTP MESSAGE                                  SIGNATURE COMPONENT

 IOTP Message                                   Signature Id = P1.3
  |-Trans Ref Block        digest TransRefBlk   |-Manifest
  |  |      ID=P1.1-----------------------------|->|-Digest of P1.1--
  |  |-Trans Id Comp       digest TransIdComp   |  |                 |
  |  |     ID = M1.2----------------------------|->|-Digest of M1.2--|
  |  |-Msg Id Comp.           digest Signature  |  |                 |
  |  |      ID = P1          -------------------|->|-Digest of M1.5--|
  |                         |   digest element  |  |                 |
  |-Signatures Block        |  -----------------|->|-Digest of M1.7--|
  |  |       ID=P1.2        | |  digest element |  |                 |
  |  |-Signature ID=P1.3    | |  ---------------|->|-Digest of C1.4--|
  |  |-Signature ID=M1.5----  | |               |  |                 |
  |  |-Signature ID=P1.4      | | Points to     |   -RecipientInfo*  |
  |  |-Certificate ID=M1.6<---|-|---------------|------CertRef=M1.6  |
  |  |                        | | Certs to use  |  Sig.ValueRef=P1.4 |
  |  |                        | |               |        |           |
  |  |                        | |               |        |           |
  |-Trading Block. ID=P1.5    | |               |        v           |
  |  |-Comp. ID=M1.7----------  |                -Value* ID=P1.4:    |
  |  |                          |                   JtvwpMdmSfMbhK<--
  |  |-Comp. ID=P1.6            |                   r1Ln3vovbMQttbBI
  |  |                          |                   J8pxLjoSRfe1o6k
  |  |-Comp. ID=C1.4------------                    OGG7nTFzTi+/0<-
  |  |-Comp. ID=C1.5
                             Digital signature of Manifest element
                             using certificate identified by CertRef

   Elements that are digested can be in any IOTP Message
        within the same IOTP Transaction
 *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 10 Signature Digests

Note: The classic example of one signature signing another in IOTP, is when an Offer is first signed by a Merchant creating an "Offer Response" signature, which is then later signed by a Payment Handler together with a record of the payment creating a "Payment Receipt" signature. In this way, the payment in an IOTP Transaction is bound to the Merchant’s offer.

Note that one Manifest may be associated with multiple signature "Value" elements where each Value element contains a digital signature over the same Manifest, perhaps using the same (or different) signature algorithm but using a different certificate or shared secret key. Specifically it will allow the Merchant to agree on different shared secrets keys with their Payment Handler and Delivery Handler.

The detailed definitions of a Signature component are contained in section 7.19.

The remainder of this section contains:

- an example of how IOTP uses signatures
- how the OriginatorInfo and RecipientInfo elements within a Signature Component are used to identify the Organisations associated with the signature
- how IOTP uses signatures to prove actions complete successfully

6.1.1 IOTP Signature Example

An example of how signatures are used is illustrated in the figure below which shows how the various components and elements in a Baseline Purchase relate to one another. Refer to this example in the later description of how signatures are used to check a payment or delivery can occur (see section 6.3).

Note: A Baseline Purchase transaction has been used for illustration purposes. The usage of the elements and attributes is the same for all types of IOTP Transactions.

```
*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*

TPO SELECTION BLOCK          TPO BLOCK          IOTPSIGNATURE BLOCK
                                                 | (Offer Response)
 Brand Selection             Organisation<---    |------Signature
   Component                 Component       |   |      Component
      |                       |              |           -Manifest
      |BrandList               -Trading Role |            |
      |  Ref                     Element     | Originator |-Orig.
      v                         (Merchant)    ------------|--Info
    Brand List                                    Ref     |
  >Component                                              |
 | |-Protocol       ------>  Organisation     Recipient   |-Recipient
 | | Amount Elem   |         Component <------------------|--Info
 | |   |           |          |                 Refs      |
 | |Pay|Protocol   |Action     -Trading Role              |
 | |   | Ref       |OrgRef       Element                  |
 | |   v           |          (Payment Handler)           |
 |  -PayProtocol--                                        |
 |    Elem                  ->Organisation    Recipient   |-Recipient
 |                         |  Component <--------------------Info
 |                         |  |                 Refs
 |                         |   -Trading Role
 |                         |     Element
 |                         | (Delivery Handler
 |
 |           OFFER RESPONSE BLOCK
 |                         |
 |BrandListRef             |ActionOrgRef
 |                         |
  --Payment                 ---Delivery
   Component                  Component
```

The Manifest element in the Signature Component contains digests of:
the Trans Ref Block (not shown); the Transaction ID Component (not
shown); Organisation Components (Merchant, Payment Handler, Delivery
Handler); the Brand List Component; the Order Component, the Payment
Component the Delivery Component and the Brand Selection Component (if a
Brand Dependent Purchase).

```
*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 11 Example use of Signatures for Baseline Purchase

6.1.2 OriginatorInfo and RecipientInfo Elements

The OriginatorRef attribute of the OriginatorInfo element in the Signature Component contains an Element Reference (see section 3.5) that points to the Organisation Component of the Organisation which generated the Signature. In this example its the Merchant.

Note that the value of the content of the Attribute element with a Type attribute set to IOTP Signature Type must match the Trading Role of the Organisation which signed it. If it does not, then it is an error. Valid combinations are given in the table below.

IOTP Signature Type	Valid Trading Role
OfferResponse	Merchant
PaymentResponse	PaymentHandler
DeliveryResponse	DeliveryHandler
AuthenticationRequest	any role
AuthenticationResponse	any role
PingRequest	any role
PingResponse	any role

The RecipientRefs attribute of the RecipientInfo element in the Signature Component contains Element References to the Organisation Components of the Organisations that should use the signature to verify that:

- o they have a pre-existing relationship with the Organisation that generated the signature,
- o the data which is secured by the signature has not been changed,
- o the data has been signed correctly, and
- o the action they are required to undertake on behalf of the Merchant is therefore authorised.

Note that if symmetric cryptography is being used then a separate RecipientInfo and Value elements for each different set of shared secret keys are likely within the Signature Component.

Alternatively if asymmetric cryptography is being used then the RecpientRefs attribute of one RecipientInfo element may refer to multiple Organisation Components if they are all using the same certificates.

6.1.3 Using signatures to Prove Actions Complete Successfully

Proving an action completed successfully, is achieved by signing data on Response messages. Specifically:

- o on the Offer Response, when a Merchant is making an Offer to the Consumer which can then be sent to either:
  - a Payment Handler to prove that the Merchant authorises Payment, or
  - a Delivery Handler to prove that Merchant authorises Delivery, provided other necessary authorisations are complete (see below)
- o on the Payment Response, when a Payment Handler is generating a Payment Receipt which can be sent to either:
  - a Delivery Handler, in a Delivery Request Block to authorise Delivery together with the Offer Response signature, or
  - another Payment Handler, in a second Payment Request, to authorise the second payment in a Value Exchange IOTP Transaction
- o Delivery Response, when a Delivery Handler is generating a Delivery Note. This can be used to prove after the event what the Delivery Handler said they would do
- o Authentication Response. One method of authenticating another party to a trade is to send an Authentication Request specifying that a Digital Signature should be used for authentication
- o Transaction Status Inquiry. The Inquiry Response Block may be digitally signed to attest to the authenticity of the response
- o Ping. The Ping Response may be digitally signed so that checks can be made that the signature can be understood.

This proof of an action may, in future versions of IOTP, also be used to prove after the event that the IOTP transaction occurred. For example to a Customer Care Provider.

6.2 Checking a Signature is Correctly Calculated

Checking a signature is correctly calculated is part of checking for Message Level Errors (see section 4.3.2). It is included here so that all signature and security related considerations are kept together.

Before a Trading Role can check a signature it must identify which of the potentially multiple Signature elements should be checked. The steps involved are as follows:

- o check that a Signature Block is present and it contains one or more Signature Components
- o identify the Organisation Component which contains an OrgId attribute for the Organisation which is carrying out the signature check. If no or more than one Organisation Component is found then it is an error
- o use the ID attribute of the Organisation Component to find the RecipientInfo element that contains a RecipientRefs attribute that refers to that Organisation Component. Note there may be no signatures to verify
- o check the Signature Component that contains the identified RecipientInfo element as follows:
  - use the SignatureValueRef and the SignatureAlgorithmRef attributes to identify, respectively: the Value element that contains the signature to be checked and the Signature Algorithm element that describes the signature algorithm to be used to verify the Signature, then
  - if the Signature Algorithm element indicates that asymmetric cryptography is being used then use the SignatureCertRef to identify the Certificate to be used by the signature algorithm
  - if Signature Algorithm element indicates that symmetric cryptography is being used then the content of the RecipientInfo element is used to identify the correct shared secret key to use
  - use the specified signature algorithm to check that the Value Element correctly signs the Manifest Element
  - check that the Digest Elements in the Manifest Element are correctly calculated where Components or Blocks referenced by the Digest have been received by the Organisation checking the signature.

6.3 Checking a Payment or Delivery can occur

This section describes the processes required for a Payment Handler or Delivery Handler to check that a payment or delivery can occur. This may include checking signatures if this is specified by the Merchant.

In outline the steps are:

- check that the Payment Request or Delivery Request has been sent to the correct Organisation
- check that correct IOTP components are present in the request, and
- check that the payment or delivery is authorised

For clarity and brevity the following terms or phrases are used in this section:

- a "Request Block" is used to refer to either a Payment Request Block (see section 8.7) or a Delivery Request Block (see section 8.10) unless specified to the contrary
- a "Response Block" is used to refer to either a Payment Response Block (see section 8.9) or a Delivery Response Block (see section 8.11)
- an "Action" is used to refer to an action which occurs on receipt of a Request Block. Actions can be either a Payment or a Delivery
- an "Action Organisation", is used to refer to the Payment Handler or Delivery Handler that carries out an Action
- a "Signer of an Action", is used to refer to the Organisations that sign data about an Action to authorise the Action, either in whole or in part
- a "Verifier of an Action", is used to refer to the Organisations that verify data to determine if they are authorised to carry out the Action
- an ActionOrgRef attribute contains Element References which can be used to identify the "Action Organisation" that should carry out an Action

6.3.1 Check Request Block sent Correct Organisation

Checking the Request Block was sent to the correct Organisation varies depending on whether the request refers to a Payment or a Delivery.

6.3.1.1 Payment

In outline a Payment Handler checks if it can accept or make a payment by identifying the Payment Component in the Payment Request Block it has received, then using the ID of the Payment Component to track through the Brand List and Brand Selection Components to identify the Organisation selected by the Consumer and then checking that this Organisation is itself.

The way data is accessed to do this is illustrated in the figure below.

```
   *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*
                                                     Start
                                                      |
                                                      v
   Brand List<--------------------------+-----------Payment
   Component         BrandListRef       |          Component
    |                                   |
    |-Brand<--------------------------  |
    | Element        BrandRef         | |
    |  |                          Brand Selection
    |  |Protocol                     Component
    |  | AmountRefs                   | |
    |  v                  Protocol    | |
    |-Protocol Amount<----------------  |
    | Element----------  AmountRef      |
    |  |               |                |
    |  |Currency       |Pay             |
    |  | AmountRefs    |Protocol        |
    |  v               |Ref             |
    |-Currency Amount  |                |
    | Element<---------|----------------
    |                  |
     -PayProtocol<-----
      Element---------------------->Organisation
                     Action         Component
                     OrgRef          |
                                      -Trading Role
                                        Element
                                     (Payment Handler)

   *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 12 Checking a Payment Handler can carry out a Payment

The following describes the steps involved and the checks which need to be made:

- o Identify the Payment Component (see section 7.9) in the Payment Request Block that was received.

- o Identify the Brand List and Brand Selection Components for the Payment Component. This involves:

  - identifying the Brand List Component (see section 7.7) where the value of its ID attribute matches the BrandListRef attribute of the Payment Component. If no or more than one Brand List Component is found there is an error.

  - identifying the Brand Selection Component (see section 7.8) where the value of its BrandListRef attribute matches the BrandListRef of the Payment Component. If no or more than one matching Brand Selection Component is found there is an error.

- o Identify the Brand, Protocol Amount, Pay Protocol and Currency Amount elements within the Brand List that have been selected by the Consumer as follows:

  - the Brand Element (see section 7.7.1) selected is the element where the value of its Id attribute matches the value of the BrandRef attribute in the Brand Selection. If no or more than one matching Brand Element is found then there is an error.

  - the Protocol Amount Element (see section 7.7.3) selected is the element where the value of its Id attribute matches the value of the ProtocolAmountRef attribute in the Brand Selection Component. If no or more than one matching Protocol Amount Element is found there is an error

  - the Pay Protocol Element (see section 7.7.5) selected is the element where the value of its Id attribute matches the value of the PayProtocolRef attribute in the identified Protocol Amount Element. If no or more than one matching Pay Protocol Element is found there is an error

  - the Currency Amount Element (see section 7.7.4) selected is the element where the value of its Id attribute matches the value of the CurrencyAmountRef attribute in the Brand Selection Component. If no or more than one matching Currency Amount element is found there is an error

- o Check the consistency of the references in the Brand List and Brand Selection Components:

  - check that an Element Reference exists in the ProtocolAmountRefs attribute of the identified Brand Element that matches the Id attribute of the identified Protocol Amount Element. If no or more than one matching Element Reference can be found there is an error

  - check that the CurrencyAmountRefs attribute of the identified Protocol Amount element contains an element reference that matches the Id attribute of the identified Currency Amount element. If no or more than one matching Element Reference is found there is an error.

  - check the consistency of the elements in the Brand List. Specifically, the selected Brand, Protocol Amount, Pay Protocol and Currency Amount Elements are all child elements of the identified Brand List Component. If they are not there is an error.

- o Check that the Payment Handler that received the Payment Request Block is the Payment Handler selected by the Consumer. This involves:

  - identifying the Organisation Component for the Payment Handler. This is the Organisation Component where its ID attribute matches the ActionOrgRef attribute in the identified Pay Protocol Element. If no or more than one matching Organisation Component is found there is an error

  - checking the Organisation Component has a Trading Role Element with a Role attribute of PaymentHandler. If not there is an error

  - finally, if the identified Organisation Component is not the same as the Organisation that received the Payment Request Block, then there is an error.

6.3.1.2 Delivery

The way data is accessed by a Delivery Handler in order to check that it may carry out a delivery is illustrated in the figure below.

```
*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*
                         Start
                           |
                           v
                        Delivery
                        Component
                           |
                           |ActionOrgRef
                           |
                           v
                        Organisation
                        Component
                        |
                         -Trading Role
                           Element
                        (Delivery Handler)

*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 13 Checking a Delivery Handler can carry out a Delivery

The steps involved are as follows:

- Identify the Delivery Component in the Delivery Request Block. If there is no or more than one matching Delivery Component there is an error
- Use the ActionOrgRef attribute of the Delivery Component to identify the Organisation Component of the Delivery Handler. If there is no or more than one matching Organisation Component there is an error
- If the Organisation Component for the Delivery Handler does not have a Trading Role Element with a Role attribute of DeliveryHandler there is an error
- Finally, if the Organisation that received the Delivery Request Block does not identify the Organisation Component for the Delivery Handler as itself, then there is an error.

6.3.2 Check Correct Components present in Request Block

Check that the correct components are present in the Payment Request Block (see section 8.7) or in the Delivery Request Block (see section 8.10).

If components are missing, there is an error.

6.3.3 Check an Action is Authorised

The previous steps identified the Action Organisation and that all the necessary components are present. This step checks that the Action Organisation is authorised to carry out the Action.

In outline the Action Organisation will identifies the Merchant, checks that it has a pre-existing agreement with the Merchant that allows it carry out the Action and that any constraints implied by that agreement are being followed, then, if signatures are required, it checks that they sign the correct data.

The steps involved are as follows:

- o Identify the Merchant. This is the Organisation Component with a Trading Role Element which has a Role attribute with a value of Merchant. If no or more than one Trading Role Element is found, there is an error
- o Check the Action Organisation’s agreements with the Merchant allows the Action to be carried out. To do this the Action Organisation must check that:
  - - the Merchant is known and a pre-existing agreement exists for the Action Organisation to be their agent for the payment or delivery
  - - they are allowed to take part in the type of IOTP transaction that is occurring. For example a Payment Handler may have agreed to accept payments as part of a Baseline Purchase, but not make payments as part of a Baseline Refund
  - - any constraints in their agreement with the Merchant are being followed, for example, whether or not an Offer Response signature is required
- o Check the signatures are correct. If signatures are required then they need to be checked. This involves:

  - Identifying the correct signatures to check. This involves the Action Organisation identifying the Signature Components that contain references to the Action Organisation (see 6.3.1). Depending on the IOTP Transaction being carried out (see section 9) either one or two signatures may be identified

  - checking that the Signature Components are correct. This involves checking that Digest elements exist within the Manifest Element that refer to the necessary Trading Components (see section 6.3.3.1).

6.3.3.1 Check the Signatures Digests are correct

All Signature Components contained within IOTP Messages must include Digest elements that refer to:

- o the Transaction Id Component (see section 3.3.1) of the IOTP message that contains the Signature Component. This binds the globally unique IotpTransId to other components which make up the IOTP Transaction

- o the Transaction Reference Block (see section 3.3) of the first IOTP Message that contained the signature. This binds the IotpTransId with information about the IOTP Message contained inside the Message Id Component (see section 3.3.2).

Check that each Signature Component contains Digest elements that refer to the correct data required.

The Digest elements that need to be present depend on the Trading Role of the Organisation which generated (signed) the signature:

- o if the signer of the signature is a Merchant then:

  - Digest elements must be present for all the components in the Request Block apart from the Brand Selection Component which is optional

- o if the signer of the signature is a Payment Handler then Digest elements must be present for:

  - the Signature Component signed by the Merchant, and optionally

  - one or more Signature Components signed by the previous Payment Handler(s) in the Transaction.

7. Trading Components

This section describes the Trading Components used within IOTP. Trading Components are the child XML elements which occur immediately below a Trading Block as illustrated in the diagram below.

```
   *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*

             IOTP MESSAGE  <----------- IOTP Message - an XML Document
              |                         which is transported between the
              |                         Trading Roles
              |-Trans Ref Block <-----  Trans Ref Block - contains
              |  |                      information which describes the
              |  |                      IOTP Transaction and the IOTP
                                        Message.
    --------> |  |-Trans Id Comp. <---  Transaction Id Component -
   |          |  |                      uniquely identifies the IOTP
   |          |  |                      Transaction. The Trans Id
   |          |  |                      Components are the same across
   |          |  |                      all IOTP messages that comprise
   |          |  |                      a single IOTP transaction.
   |          |  |-Msg Id Comp. <-----  Message Id Component -
   |          |                         identifies and describes an IOTP
   |          |                         Message within an IOTP
   |          |                         Transaction
   |          |-Signature Block <-----  Signature Block (optional) -
   |          |  |                      contains one or more Signature
   |          |  |                      Components and their associated
   |          |  |                      Certificates
   |     ---> |  |-Signature Comp. <--  Signature Component - contains
   |    |     |  |                      digital signatures. Signatures
   |    |     |  |                      may sign digests of the Trans Ref
   |    |     |  |                      Block and any Trading Component
   |    |     |  |                      in any IOTP Message in the same
   |    |     |  |                      IOTP Transaction.
   |    |     |  |-Certificate Comp. <- Certificate Component. Used to
   |    |     |                         check the signature.
     Trading  |-Trading Block <-------- Trading Block - an XML Element
   Components |  |-Trading Comp.        within an IOTP Message that
   |    |     |  |-Trading Comp.        contains a predefined set of
   |     ---> |  |-Trading Comp.        Trading Components
   |          |  |-Trading Comp.
   |          |  |-Trading Comp. <----- Trading Components - XML
   |          |                         Elements within a Trading Block
   |          |-Trading Block           that contain a predefined set of
    --------> |  |-Trading Comp.        XML elements and attributes
              |  |-Trading Comp.        containing information required
              |  |-Trading Comp.        to support a Trading Exchange
              |  |-Trading Comp.
              |  |-Trading Comp.
   *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 14 Trading Components

The Trading Components described in this section are listed below in approximately the sequence they are likely to be used:

- Protocol Options Component
- Authentication Request Component
- Authentication Response Component
- Trading Role Information Request Component
- Order Component
- Organisation Component
- Brand List Component
- Brand Selection Component
- Payment Component
- Payment Scheme Component
- Payment Receipt Component
- Delivery Component
- Delivery Data Component
- Delivery Note Component
- Signature Component
- Certificate Component
- Error Component

Note that the following components are listed in other sections of this specification:

- Transaction Id Component (see section 3.3.1)
- Message Id Component (see section 3.3.2)

7.1 Protocol Options Component

Protocol options are options which apply to the IOTP Transaction as a whole. Essentially it provides a short description of the entire transaction and the net location which the Consumer role should branch to if the IOTP Transaction is successful.

The definition of a Protocol Options Component is as follows.

```
<!ELEMENT ProtocolOptions EMPTY >
<!ATTLIST ProtocolOptions
 ID                 ID      #REQUIRED
 xml:lang           NMTOKEN #REQUIRED
 ShortDesc          CDATA   #REQUIRED
 SenderNetLocn      CDATA   #IMPLIED
 SecureSenderNetLocn CDATA  #IMPLIED
 SuccessNetLocn     CDATA   #REQUIRED >
```

Attributes:

ID	An identifier which uniquely identifies the Protocol Options Component within the IOTP Transaction.
Xml:lang	Defines the language used by attributes or child elements within this component, unless overridden by an xml:lang attribute on a child element. See section 3.8 Identifying Languages.
ShortDesc	This contains a short description of the IOTP Transaction in the language defined by xml:lang. Its purpose is to provide an explanation of what type of IOTP Transaction is being conducted by the parties involved. It is used to facilitate selecting an individual transaction from a list of similar transactions, for example from a database of IOTP transactions which has been stored by a Consumer, Merchant, etc.
SenderNetLocn	This contains the non secured net location of the sender of the TPO Block in which the Protocol Options Component is contained. It is the net location to which the recipient of the TPO block should send a TPO Selection Block if required.

	The content of this attribute is dependent on the Transport Mechanism see the Transport Mechanism Supplement.
SecureSenderNetLocn	This contains the secured net location of the sender of the TPO Block in which the Protocol Options Component is contained. The content of this attribute is dependent on the Transport Mechanism see the Transport Mechanism Supplement.
SuccessNetLocn	This contains the net location that should be displayed after the IOTP Transaction has successfully completed. The content of this attribute is dependent on the Transport Mechanism see the Transport Mechanism Supplement.

Either SenderNetLocn, SecureSenderNetLocn or both must be present.

7.2 Authentication Request Component

This Trading Component contains parameter data that is used in an Authentication of one Trading Role by another. Its definition is as follows.

```
<!ELEMENT AuthReq (Algorithm, PackagedContent*)>
<!ATTLIST AuthReq
 ID                 ID      #REQUIRED
 AuthenticationId   CDATA   #REQUIRED
 ContentSoftwareId  CDATA   #IMPLIED >
```

If required the Algorithm may use the challenge data, contained in the Packaged Content elements within the Authentication Request Component in its calculation. The format of the Packaged Contents are Algorithm dependent.

Attributes:

ID	An identifier which uniquely identifies the Authentication Request Component within the IOTP Transaction.
AuthenticationId	An identifier specified by the Authenticator which, if returned by the Organisation that receives the Authentication Request, will enable

	the Authenticator to identify which Authentication is being referred to.
ContentSoftwareId	See section 14.Glossary

Content:

PackagedContent	This contains the challenge data as one or more Packaged Content (see section 3.7) that is to be responded to using the Algorithm defined by the Algorithm element.
Algorithm	This contains information which describes the Algorithm (see 7.19 Signature Components) that must be used to generate the Authentication Response. The Algorithms that may be used are identified by the Name attribute of the Algorithm element. For valid values see section 12. IANA Considerations.

7.3 Authentication Response Component

The Authentication Response Component contains the results of an authentication request. It uses the Algorithm contained in the Authentication Request Component (see section 7.2) selected from the Authentication Request Block (see section 8.4).

Depending on the Algorithm selected, the results of applying the algorithm will either be contained in a Signature Component that signs both the Authentication Response and potentially other data, or in the Packaged Content elements within the Authentication Response Component. Its definition is as follows.

```
<!ELEMENT AuthResp (PackagedContent*) >
<!ATTLIST AuthResp
 ID                  ID      #REQUIRED
 AuthenticationId   CDATA   #REQUIRED
 SelectedAlgorithmRef NMTOKEN #REQUIRED
 ContentSoftwareId  CDATA   #IMPLIED >
```

Attributes:

ID	An identifier which uniquely identifies the Authentication Response Component within the IOTP Transaction.

AuthenticationId	The Authentication identifier specified by the Authenticator that was included in the Authentication Request Component(see section 7.2). This will enable the Authenticator to identify the Authentication that is being referred to.
SelectedAlgorithmRef	An Element Reference that identifies the Algorithm element used to generate the Authentication Response.
ContentSoftwareId	See section 14.Glossary.

Content:

PackagedContent	This may contain the response generated as a result of applying the Algorithm selected from the Authentication Request Component see section 7.2. For example, for a payment specific scheme, it may contain scheme-specific data. Refer to the scheme-specific supplemental documentation for definitions of its content.

7.4 Trading Role Information Request Component

This Trading Component contains a list of Trading Roles (see section 2.1) about which information is being requested. The result of a Trading Role Request is a set of Organisation Components (see section 7.6) that describe each of the Trading Roles requested.

Example usage includes:

o a Merchant requesting that a Consumer provides Organisation Components for the Consumer and DelivTo Trading Roles

o a Consumer requesting from a Merchant, information about the Payment Handlers and Delivery Handlers that the Merchant uses.

Its definition is as follows.

```
<!ELEMENT TradingRoleInfoReq EMPTY>
<!ATTLIST TradingRoleInfoReq
 ID                ID      #REQUIRED
 TradingRoleList    NMTOKENS #REQUIRED >
```

Attributes:

ID	An identifier which uniquely identifies the Trading Role Information Request Component within the IOTP Transaction.
TradingRoleList	Contains a list of one or more Trading Roles (see the TradingRole attribute of the Trading Role Element - section 7.6.2) for which information is being requested.

7.5 Order Component

An Order Component contains information about an order. Its definition is as follows.

```
<!ELEMENT Order (PackagedContent*) >
<!ATTLIST Order
 ID                 ID      #REQUIRED
 xml:lang           NMTOKEN #REQUIRED
 OrderIdentifier    CDATA   #REQUIRED
 ShortDesc          CDATA   #REQUIRED
 OkFrom             CDATA   #REQUIRED
 OkTo               CDATA   #REQUIRED
 ApplicableLaw      CDATA   #REQUIRED
 ContentSoftwareId  CDATA   #IMPLIED >
```

Attributes:

ID	An identifier which uniquely identifies the Order Component within the IOTP Transaction.
xml:lang	Defines the language used by attributes or child elements within this component, unless overridden by an xml:lang attribute on a child element. See section 3.8 Identifying Languages.
OrderIdentifier	This is a code, reference number or other identifier which the creator of the Order may use to identify the order. It must be unique within an IOTP Transaction. If it is used in this way, then it may remove the need to specify any content for the Order element as the reference can be used to look up the necessary information in a database.
ShortDesc	A short description of the order in the language defined by xml:lang. It is used to facilitate selecting an individual order from a list of

	orders, for example from a database of orders which has been stored by a Consumer, Merchant, etc.
OkFrom	The date and time in [UTC] format after which the offer made by the Merchant lapses.
OkTo	The date and time in [UTC] format before which a Value Acquirer may accept the offer made by the Merchant is not valid.
ApplicableLaw	A phrase in the language defined by xml:lang which describes the state or country of jurisdiction which will apply in resolving problems or disputes.
ContentSoftwareId	See section 14.Glossary.

Content:

PackagedContent	An optional description of the order information as one or more Packaged Contents (see section 3.7).

7.5.1 Order Description Content

The Packaged Content element will normally be required, however it may be omitted where sufficient information about the purchase can be provided in the ShortDesc attribute. If the full Order Description requires it several Packaged Content elements may be used.

Although the amount and currency are likely to appear in the Packaged Content of the Order Description it is the amount and currency contained in the payment related trading components (Brand List, Brand Selection and Payment) that is authoritative. This means it is important that the amount actually being paid (as contained in the payment related trading components) is prominently displayed to the Consumer.

For interoperability, implementations must support Plain Text, HTML and XML as a minimum so that it can be easily displayed.

7.5.2 OkFrom and OkTo Timestamps

Note that:

o the OkFrom date may be later than the OkFrom date on the Payment Component (see section 7.9) associated with this order, and

- similarly, the OkTo date may be earlier that the OkTo date on the Payment Component (see section 7.9).

Note: Disclaimer. The following information provided in this note does not represent formal advice of any of the authors of this specification. Readers of this specification must form their own views and seek their own legal counsel on the usefulness and applicability of this information.

The merchant in the context of Internet commerce with anonymous consumers initially frames the terms of the offer on the web page, and in order to obtain the goods or services, the consumer must accept them.

If there is to be a time-limited offer, it is recommended that merchants communicate this to the consumer and state in the order description in a manner which is clear to the consumer that:

- the offer is time limited

- the OkFrom and OkTo timestamps specify the validity of the offer

- the clock, e.g., the merchant’s clock, that will be used to determine the validity of the offer

Also note that although the OkFrom and OkTo dates are likely to appear in the Packaged Content of the Order Description it is the dates contained in the Order Component that is authoritative. This means it is important that the OkFrom and OkTo dates actually being used is prominently displayed to the Consumer.

7.6 Organisation Component

The Organisation Component provides information about an individual or an Organisation. This can be used for a variety of purposes. For example:

- to describe the merchant who is selling the goods,

- to identify who made a purchase,

- to identify who will take delivery of goods,

- to provide a customer care contact,

- to describe who will be the Payment Handler.

Note that the Organisation Components which must be present in an IOTP Message are dependent on the particular transaction being carried out. Refer to section 9. Internet Open Trading Protocol Transactions, for more details.

Its definition is as follows.

```
<!ELEMENT Org (TradingRole+, ContactInfo?,
     PersonName?, PostalAddress?)>
<!ATTLIST Org
 ID                 ID      #REQUIRED
 xml:lang           NMTOKEN #REQUIRED
 OrgId              CDATA   #REQUIRED
 LegalName          CDATA   #IMPLIED
 ShortDesc          CDATA   #IMPLIED
 LogoNetLocn        CDATA   #IMPLIED >
```

Attributes:

ID	An identifier which uniquely identifies the Organisation Component within the IOTP Transaction.
xml:lang	Defines the language used by attributes or child elements within this component, unless overridden by an xml:lang attribute on a child element. See section 3.8 Identifying Languages.
OrgId	A code which identifies the Organisation described by the Organisation Component. See 7.6.1 Organisation IDs, below.
LegalName	For Organisations which are companies this is their legal name in the language defined by xml:lang. It is required for Organisations who have a Trading Role other than Consumer or DelivTo.
ShortDesc	A short description of the Organisation in the language defined by xml:lang. It is typically the name by which the Organisation is commonly known. For example, if the legal name was "Blue Meadows Financial Services Inc.". Then its short name would likely be "Blue Meadows". It is used to facilitate selecting an individual Organisation from a list of Organisations, for example from a database of Organisations involved

	in IOTP Transactions which has been stored by a consumer.
LogoNetLocn	The net location which can be used to download the logo for the Organisation. See section 10 Retrieving Logos. The content of this attribute must conform to [RFC1738].

Content:

TradingRole	See 7.6.2 Trading Role Element below.
ContactInfo	See 7.6.3 Contact Information Element below.
PersonName	See 7.6.4 Person Name below.
PostalAddress	See 7.6.5 Postal Address below.

7.6.1 Organisation IDs

Organisation IDs are used by one IOTP Trading Role to identify another. In order to avoid confusion, this means that these IDs must be globally unique.

In principle this is achieved in the following way:

- o the Organisation Id for all trading roles, apart from the Consumer Trading Role, uses a domain name as their globally unique identifier,
- o the Organisation Id for a Consumer Trading Role is allocated by one of the other Trading Roles in an IOTP Transaction and is made unique by concatenating it with that other roles' Organisation Id,
- o once a Consumer is allocated an Organisation Id within an IOTP Transaction the same Organisation Id is used by all the other trading roles in that IOTP transaction to identify that Consumer.

Specifically, the content of the Organisation ID is defined as follows:

```
OrgId ::= NonConsumerOrgId | ConsumerOrgId
NonConsumerOrgId ::= DomainName
ConsumerOrgId ::= ConsumerOrgIdPrefix (namechar)+ "/" NonConsumerOrgId
ConsumerOrgIdPrefix ::= "Consumer:"
```

ConsumerOrgId	The Organisation ID for a Consumer consists of: o a standard prefix to identify that the Organisation Id is for a consumer, followed by o one or more characters which conform to the definition of an XML "namechar". See [XML] specifications, followed by o the NonConsumerOrgId for the Organisation which allocated the ConsumerOrgId. It is normally the Merchant role. Use of upper and lower case is not significant.
NonConsumerOrgId	If the Role is not Consumer then this contains the Canonical Name for the non-consumer Organisation being described by the Organisation Component. See [DNS] optionally followed by additional characters, if required, to make the NonConsumerOrgId unique. Note that a NonConsumerOrgId may not start with the ConsumerOrgIdPrefix. Use of upper and lower case is not significant.

Examples of Organisation Ids follow:

o newjerseybooks.com - a merchant Organisation id

o westernbank.co.uk - a Payment Handler Organisation id

o consumer:1000247ABH/newjerseybooks.com - a consumer Organisation id allocated by a merchant

7.6.2 Trading Role Element

This identifies the Trading Role of an individual or Organisation in the IOTP Transaction. Note, an Organisation may have more than one Trading Role and several roles may be present in one Organisation element. Its definition is as follows:

```
<!ELEMENT TradingRole EMPTY >
<!ATTLIST TradingRole
 ID                 ID      #REQUIRED
 TradingRole        NMTOKEN #REQUIRED
 IotpMsgIdPrefix    NMTOKEN #REQUIRED
 CancelNetLocn      CDATA   #IMPLIED
 ErrorNetLocn       CDATA   #IMPLIED
```

```
    ErrorLogNetLocn    CDATA   #IMPLIED >
```

Attributes:

ID	An identifier which uniquely identifies the Trading Role Element within the IOTP Transaction.
TradingRole	The trading role of the Organisation. Valid values are: o Consumer. The person or Organisation that is acting in the role of a consumer in the IOTP Transaction. o Merchant. The person or Organisation that is acting in the role of merchant in the IOTP Transaction. o PaymentHandler. The financial institution or other Organisation which is a Payment Handler for the IOTP Transaction o DeliveryHandler. The person or Organisation that is the delivering the goods or services for the IOTP Transaction o DelivTo. The person or Organisation that is receiving the delivery of goods or services in the IOTP Transaction o CustCare. The Organisation and/or individual who will provide customer care for an IOTP Transaction. Values of TradingRole are controlled under the procedures defined in section 12 IANA Considerations which also allows user defined values to be defined.
IotpMsgIdPrefix	Contains the prefix which must be used for all IOTP Messages sent by the Trading Role in this IOTP Transaction. The values to be used are defined in 3.4.1 IOTP Message ID Attribute Definition.
CancelNetLocn	This contains the net location of where the Consumer should go to if the Consumer cancels the transaction for some reason. It can be used by the Trading Role to provide a response which is more tailored to the circumstances of a particular transaction.

	This attribute: o must not be present when TradingRole is set to Consumer role or DelivTo, o must be present when TradingRole is set to Merchant, PaymentHandler or DeliveryHandler. The content of this attribute is dependent on the Transport Mechanism see the Transport Mechanism Supplement.
ErrorNetLocn	This contains the net location that should be displayed by the Consumer after the Consumer has either received or generated an Error Block containing an Error Component with the Severity attribute set to either: o HardError, o Warning but the Consumer decides to not continue with the transaction o TransientError and the transaction has subsequently timed out. See section 7.21.1 Error Processing Guidelines for more details. This attribute: o must not be present when TradingRole is set to Consumer or DelivTo, o must be present when TradingRole is set to Merchant, PaymentHandler or DeliveryHandler. The content of this attribute is dependent on the Transport Mechanism see the Transport Mechanism Supplement.
ErrorLogNetLocn	Optional. This contains the net location that Consumers should send IOTP Messages that contain Error Blocks with an Error Component with the Severity attribute set to either: o HardError, o Warning but the Consumer decides to not continue with the transaction o TransientError and the transaction has subsequently timed out. This attribute: o must not be present when TradingRole is set to Consumer role,

- o must be present when TradingRole is set to Merchant, PaymentHandler or DeliveryHandler.

The content of this attribute is dependent on the Transport Mechanism see the Transport Mechanism Supplement.

The ErrorLogNetLocn can be used to send error messages to the software company or some other Organisation responsible for fixing problems in the software which sent the incoming message. See section 7.21.1 Error Processing Guidelines for more details.

7.6.3 Contact Information Element

This contains information which can be used to contact an Organisation or an individual. All attributes are optional however at least one item of contact information should be present. Its definition is as follows.

```
<!ELEMENT ContactInfo EMPTY >
<!ATTLIST ContactInfo
 xml:lang           NMTOKEN #IMPLIED
 Tel                CDATA   #IMPLIED
 Fax                CDATA   #IMPLIED
 Email              CDATA   #IMPLIED
 NetLocn            CDATA   #IMPLIED >
```

Attributes:

xml:lang	Defines the language used by attributes within this element. See section 3.8 Identifying Languages.
Tel	A telephone number by which the Organisation may be contacted. Note that this is a text field and no validation is carried out on it.
Fax	A fax number by which the Organisation may be contacted. Note that this is a text field and no validation is carried out on it.
Email	An email address by which the Organisation may be contacted. Note that this field should conform to the conventions for address specifications contained in [RFC822].

NetLocn	A location on the Internet by which information about the Organisation may be obtained that can be displayed using a web browser. The content of this attribute must conform to [RFC1738].

7.6.4 Person Name Element

This contains the name of an individual person. All fields are optional however as a minimum either the GivenName or the FamilyName should be present. Its definition is as follows.

```
<!ELEMENT PersonName EMPTY >
<!ATTLIST PersonName
 xml:lang          NMTOKEN #IMPLIED
 Title             CDATA   #IMPLIED
 GivenName         CDATA   #IMPLIED
 Initials          CDATA   #IMPLIED
 FamilyName        CDATA   #IMPLIED >
```

Attributes:

xml:lang	Defines the language used by attributes within this element. See section 3.8 Identifying Languages.
Title	A distinctive name; personal appellation, hereditary or not, denoting or implying office (e.g., judge, mayor) or nobility (e.g., duke, duchess, earl), or used in addressing or referring to a person (e.g., Mr, Mrs, Miss)
GivenName	The primary or main name by which a person is known amongst and identified by their family, friends and acquaintances. Otherwise known as first name or Christian Name.
Initials	The first letter of the secondary names (other than the Given Name) by which a person is known amongst or identified by their family, friends and acquaintances.
FamilyName	The name by which family of related individuals are known. It is typically the part of an individual’s name which is passed on by parents to their children.

7.6.5 Postal Address Element

This contains an address which can be used, for example, for the physical delivery of goods, services or letters. Its definition is as follows.

```
<!ELEMENT PostalAddress EMPTY >
<!ATTLIST PostalAddress
 xml:lang           NMTOKEN #IMPLIED
 AddressLine1       CDATA   #IMPLIED
 AddressLine2       CDATA   #IMPLIED
 CityOrTown         CDATA   #IMPLIED
 StateOrRegion      CDATA   #IMPLIED
 PostalCode         CDATA   #IMPLIED
 Country            CDATA   #IMPLIED
 LegalLocation (True | False) 'False' >
```

Attributes:

xml:lang	Defines the language used by attributes within this element. See section 3.8 Identifying Languages.
AddressLine1	The first line of a postal address. e.g., "The Meadows"
AddressLine2	The second line of a postal address. e.g., "Sandy Lane"
CityOrTown	The city of town of the address. e.g., "Carpham"
StateOrRegion	The state or region within a country where the city or town is placed. e.g., "Surrey"
PostalCode	The code known as, for example a post code or zip code, that is typically used by Postal Organisations to organise postal deliveries into efficient sequences. e.g., "KT22 1AA"
Country	The country for the address. e.g., "UK"
LegalLocation	This identifies whether the address is the Registered Address for the Organisation. At least one address for the Organisation must have a value set to True unless the Trading Role is either Consumer or DeliverTo.

7.7 Brand List Component

Brand List Components are contained within the Trading Protocol Options Block (see section 8.1) of the IOTP Transaction. They contains lists of:

- o payment Brands (see also section 11.1 Brand Definitions and Brand Selection),
- o amounts to be paid in the currencies that are accepted or offered by the Merchant,
- o the payment protocols which can be used to make payments with a Brand, and
- o the net locations of the Payment Handlers which accept payment for a payment protocol

The definition of a Brand List Component is as follows.

```
<!ELEMENT BrandList (Brand+, ProtocolAmount+,
 CurrencyAmount+, PayProtocol+) >
<!ATTLIST BrandList
 ID                 ID      #REQUIRED
 xml:lang           NMTOKEN #REQUIRED
 ShortDesc          CDATA   #REQUIRED
 PayDirection (Debit | Credit) #REQUIRED >
```

Attributes:

ID	An identifier which uniquely identifies the Brand List Component within the IOTP Transaction.
xml:lang	Defines the language used by attributes or child elements within this component, unless overridden by an xml:lang attribute on a child element. See section 3.8 Identifying Languages.
ShortDesc	A text description in the language defined by xml:Lang giving details of the purpose of the Brand List. This information must be displayed to the receiver of the Brand List in order to assist with making the selection. It is of particular benefit in allowing a Consumer to distinguish the purpose of a Brand List when an IOTP Transaction involves more than one payment.

PayDirection	Indicates the direction in which the payment for which a Brand is being selected is to be made. Its values may be: o Debit The sender of the Payment Request Block (e.g., the Consumer) to which this Brand List relates will make the payment to the Payment Handler, or o Credit The sender of the Payment Request Block to which this Brand List relates will receive a payment from the Payment Handler.

Content:

Brand	This describes a Brand. The sequence of the Brand elements (see section 7.7.1) within the Brand List does not indicate any preference. It is recommended that software which processes this Brand List presents Brands in a sequence which the receiver of the Brand List prefers.
ProtocolAmount	This links a particular Brand to: o the currencies and amounts in CurrencyAmount elements that can be used with the Brand, and o the Payment Protocols and Payment Handlers, which can be used with those currencies and amounts, and a particular Brand
CurrencyAmount	This contains a currency code and an amount.
PayProtocol	This contains information about a Payment Protocol and the Payment Handler which may be used with a particular Brand.

The relationships between the elements which make up the content of the Brand List is illustrated in the diagram below.

```
   *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*

                    Brand List Component

                      |                   ProtocolAmountRefs
                      |-Brand Element-----------------------------
                      |  |                                        |
                      |   - Protocol Brand Element--------        |
                      |                                   |       |
                      |                         ProtocolId|       |
                      |                                   |       |
                      |-Protocol Amount Element<----------+-------
                      |  |                      |         |
                      |  |                      |         |
                      |  |CurrencyAmountRefs    |Pay      |
                      |  |                      |Protocol |
                      |  v                      |Ref      |
                      |-Currency Amount Element |         |
                      | Element                 |         |
                      |                         |         |
                       -PayProtocolElement<------<--------

   *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 15 Brand List Element Relationships

Examples of complete Brand Lists are contained in section 11.2 Brand List Examples.

7.7.1 Brand Element

A Brand Element describes a brand that can be used for making a payment. One or more of these elements is carried in each Brand List Component that has the PayDirection attribute set to Debit. Exactly one Brand Element may be carried in a Brand List Component that has the PayDirection attribute set to Credit.

```
   <!ELEMENT Brand (ProtocolBrand*, PackagedContent*) >
   <!ATTLIST Brand
    ID                  ID      #REQUIRED
    xml:lang            NMTOKEN #IMPLIED
    BrandId             CDATA   #REQUIRED
    BrandName           CDATA   #REQUIRED
    BrandLogoNetLocn    CDATA   #REQUIRED
    BrandNarrative      CDATA   #IMPLIED
    ProtocolAmountRefs  IDREFS  #REQUIRED
    ContentSoftwareId   CDATA   #IMPLIED >
```

Attributes:

ID	Element identifier, potentially referenced in a Brand Selection Component contained in a later Payment Request message and uniquely identifies the Brand element within the IOTP Transaction.
xml:lang	Defines the language used by attributes and content of this element. See section 3.8 Identifying Languages.
BrandId	This contains a unique identifier for the brand (or promotional brand). It is used to match against a list of Payment Instruments which the Consumer holds to determine whether or not the Consumer can pay using the Brand. Values of BrandId are managed under the procedure described in section 12 IANA Considerations. As values of BrandId are controlled under the procedures defined in section 12 IANA Considerations user defined values may be defined.
BrandName	This contains the name of the brand, for example MasterCard Credit. This is the description of the Brand which is displayed to the consumer in the Consumers language defined by xml:lang. For example it might be "American Airlines Advantage Visa". Note that this attribute is not used for matching against the payment instruments held by the Consumer.
BrandLogoNetLocn	The net location which can be used to download the logo for the Organisation. See section Retrieving Logos (see section 10). The content of this attribute must conform to [RFC1738].
BrandNarrative	This optional attribute is designed to be used by the Merchant to indicate some special conditions or benefit which would apply if the Consumer selected that brand. For example "5% discount", "free shipping and handling", "free breakage insurance for 1 year", "double air miles apply", etc.

ProtocolAmountRefs	Identifies the protocols and related currencies and amounts which can be used with this Brand. Specified as a list of ID's of Protocol Amount Elements (see section 7.7.3) contained within the Brand List.
ContentSoftwareId	See section 14.Glossary.

Content:

ProtocolBrand	Protocol Brand elements contain brand information to be used with a specific payment protocol (see section 7.7.2)
PackagedContent	Optional Packaged Content (see section 3.7) elements containing information about the brand which may be used by the payment protocol. The content of this information is defined in the supplement for a payment protocol which describes how the payment protocol works with IOTP.

Example Brand Elements are contained in section 11.2 Brand List Examples.

7.7.2 Protocol Brand Element

The Protocol Brand Element contains information that is specific to the use of a particular Protocol with a Brand. Its definition is as follows.

```
<!ELEMENT ProtocolBrand (PackagedContent*) >
<!ATTLIST ProtocolBrand
 ProtocolId         CDATA   #REQUIRED
 ProtocolBrandId    CDATA   #REQUIRED >
```

Attributes:

ProtocolId	This must match the value of a ProtocolId attribute in a Pay Protocol Element (see section 7.7.5). The values of ProtocolId should be unique within a Brand Element otherwise there is an error.

ProtocolBrandId	This is the Payment Brand Id to be used with a particular payment protocol. For example, SET and EMV have their own well defined, yet different, values for the Brand Id to be used with each protocol. The valid values of this attribute are defined in the supplement for the payment protocol identified by ProtocolId that describes how the payment protocol works with IOTP.

Content:

PackagedContent	Optional Packaged Content (see section 3.7) elements containing information about the protocol/brand which may be used by the payment protocol. The content of this information is defined in the supplement for a payment protocol which describes how the payment protocol works with IOTP.

7.7.3 Protocol Amount Element

The Protocol Amount element links a Brand to:

- o the currencies and amounts in Currency Amount Elements (see section 7.7.4) that can be used with the Brand, and
- o the Payment Protocols and Payment Handlers defined in a Pay Protocol Element (see section 7.7.5), which can be used with those currencies and amounts.

Its definition is as follows:

```
<!ELEMENT ProtocolAmount (PackagedContent*) >
<!ATTLIST ProtocolAmount
 ID                 ID      #REQUIRED
 PayProtocolRef     IDREF   #REQUIRED
 CurrencyAmountRefs IDREFS  #REQUIRED
 ContentSoftwareId  CDATA   #IMPLIED >
```

Attributes:

ID	Element identifier, potentially referenced in a Brand element; or in a Brand Selection Component contained in a later Payment Request message which uniquely identifies the Protocol Amount element within the IOTP Transaction.

PayProtocolRef	Contains an Element Reference (see section 3.5) that refers to the Pay Protocol Element (see section 7.7.5) that contains the Payment Protocol and Payment Handlers that can be used with the Brand.
CurrencyAmountRefs	Contains a list of Element References (see section 3.5) that refer to the Currency Amount Element (see section 7.7.4) that describes the currencies and amounts that can be used with the Brand.
ContentSoftwareId	See section 14. Glossary.

Content:

PackagedContent	Optional Packaged Content (see section 3.7) elements containing information about the protocol amount which may be used by the payment protocol. The content of this information is defined in the supplement for a payment protocol which describes how the payment protocol works with IOTP.

Examples of Protocol Amount Elements are contained in section 11.2 Brand List Examples.

7.7.4 Currency Amount Element

A Currency Amount element contains:

o a currency code (and its type), and

o an amount.

One or more of these elements is carried in each Brand List Component. Its definition is as follows:

```
<!ELEMENT CurrencyAmount EMPTY >
<!ATTLIST CurrencyAmount
 ID                 ID      #REQUIRED
 Amount             CDATA   #REQUIRED
 CurrCodeType       NMTOKEN ’ISO4217-A’
 CurrCode           CDATA   #REQUIRED >
```

Attributes:

ID	Element identifier, potentially referenced in a Brand element; or in a Brand Selection Component

	contained in a later Payment Request message which uniquely identifies the Currency Amount Element within the IOTP Transaction.
Amount	Indicates the amount to be paid in whole and fractional units of the currency. For example $245.35 would be expressed "245.35". Note that values smaller than the smallest denomination are allowed. For example one tenth of a cent would be "0.001".
CurrCodeType	Indicates the domain of the CurrCode. This attribute is included so that the currency code may support non-standard "currencies" such as frequent flyer points, trading stamps, etc. Its values may be: o ISO4217-A (the default) indicates the currency code is a three character alphabetic currency code that conforms to [ISO 4217] o IOTP indicates that values of CurrCode are managed under the procedure described in section 12 IANA Considerations
CurrCode	A code which identifies the currency to be used in the payment. The domain of valid currency codes is defined by CurrCodeType As values of CurrCodeType are managed under the procedure described in section 12 IANA Considerations user defined values of CurrCodeType may be defined.

Examples of Currency Amount Elements are contained in section 11.2 Brand List Examples.

7.7.5 Pay Protocol Element

A Pay Protocol element specifies details of a Payment Protocol and the Payment Handler that can be used with a Brand. One or more of these elements is carried in each Brand List.

```
<!ELEMENT PayProtocol (PackagedContent*) >
<!ATTLIST PayProtocol
 ID                 ID      #REQUIRED
 xml:lang           NMTOKEN #IMPLIED
 ProtocolId         NMTOKEN #REQUIRED
 ProtocolName       CDATA   #REQUIRED
 ActionOrgRef       NMTOKEN #REQUIRED
```

```
     PayReqNetLocn      CDATA   #IMPLIED
     SecPayReqNetLocn   CDATA   #IMPLIED
     ContentSoftwareId  CDATA   #IMPLIED >
```

Attributes:

ID	Element identifier, potentially referenced in a Brand element; or in a Brand Selection Component contained in a later Payment Request message which uniquely identifies the Pay Protocol element within the IOTP Transaction.
xml:lang	Defines the language used by attributes and content of this element. See section 3.8 Identifying Languages.
ProtocolId	Consists of a protocol name and version. For example "SETv1.0". The values of ProtocolId are defined by the payment scheme/method owners in the document that describes how to encapsulate a payment protocol within IOTP.
ProtocolName	A narrative description of the payment protocol and its version in the language identified by xml:lang. For example "Secure Electronic Transaction Version 1.0". Its purpose is to help provide information on the payment protocol being used if problems arise.
ActionOrgRef	An Element Reference (see section 3.5) to the Organisation Component for the Payment Handler for the Payment Protocol.
PayReqNetLocn	The Net Location indicating where an unsecured Payment Request message should be sent if this protocol choice is used. The content of this attribute is dependent on the Transport Mechanism (such must conform to [RFC1738].
SecPayReqNetLocn	The Net Location indicating where a secured Payment Request message should be sent if this protocol choice is used.

	A secured payment involves the use of a secure channel such as [SSL/TLS] in order to communicate with the Payment Handler. The content of this attribute must conform to [RFC1738]. See also See section 3.9 Secure and Insecure Net Locations.
ContentSoftwareId	See section 14. Glossary.

Content:

PackagedContent	Optional Packaged Content elements (see section 3.7) containing information about the protocol which is used by the payment protocol. The content of this information is defined in the supplement for a payment protocol which describes how the payment protocol works with IOTP. An example of its use could be to include a payment protocol message.

Examples of Pay Protocol Elements are contained in section 11.2 Brand List Examples.

7.8 Brand Selection Component

A Brand Selection Component identifies the choice of payment brand, payment protocol and the Payment Handler. This element is used:

- o in Payment Request messages within Baseline Purchase and Baseline Value Exchange IOTP Transactions to identify the brand, protocol and payment handler for a payment, or
- o to, optionally, inform a merchant in a purchase of the payment brand being used so that the offer and order details can be amended accordingly.

In Baseline IOTP, the integrity of Brand Selection Components is not guaranteed. However, modification of Brand Selection Components can only cause denial of service if the payment protocol itself is secure against message modification, duplication, and swapping attacks.

The definition of a Brand Selection Component is as follows.

```
<!ELEMENT BrandSelection (BrandSelBrandInfo?,
     BrandSelProtocolAmountInfo?,
     BrandSelCurrencyAmountInfo?) >
<!ATTLIST BrandSelection
```

```
     ID                  ID      #REQUIRED
     BrandListRef        NMTOKEN #REQUIRED
     BrandRef            NMTOKEN #REQUIRED
     ProtocolAmountRef   NMTOKEN #REQUIRED
     CurrencyAmountRef   NMTOKEN #REQUIRED >
```

Attributes:

ID	An identifier which uniquely identifies the Brand Selection Component within the IOTP Transaction.
BrandListRef	The Element Reference (see section 3.5) of the Brand List Component from which a Brand is being selected
BrandRef	The Element Reference of a Brand element within the Brand List Component that is being selected that is to be used in the payment.
ProtocolAmountRef	The Element Reference of a Protocol Amount element within the Brand List Component which is to be used when making the payment.
CurrencyAmountRef	The Element Reference of a Currency Amount element within the Brand List Component which is to be used when making the payment.

Content:

BrandSelBrandInfo, BrandSelProtocolAmountInfo, BrandSelCurrencyAmountInfo	This contains any additional data that may be required by a particular payment brand or protocol. See sections 7.8.1, 7.8.2, and 7.8.3.

The following rules apply:

o the BrandListRef must contain the ID of a Brand List Component in the same IOTP Transaction

o every Brand List Component in the Trading Protocol Options Block (see section 8.1) must be referenced by one and only one Brand Selection Component

o the BrandRef must refer to the ID of a Brand contained within the Brand List Component referred to by BrandListRef

- o the ProtocolAmountRef must refer to one of the Element IDs listed in the ProtocolAmountRefs attribute of the Brand element identified by BrandRef

- o the CurrencyAmountRef must refer to one of the Element IDs listed in the CurrencyAmountRefs attribute of the Protocol Amount Element identified by ProtocolAmountRef.

An example of a Brand Selection Component is included in 11.2 Brand List Examples.

7.8.1 Brand Selection Brand Info Element

The Brand Selection Brand Info Element contains any additional data that may be required by a particular payment brand. See the IOTP payment method supplement for a description of how and when it used.

```
<!ELEMENT BrandSelBrandInfo (PackagedContent+) >
<!ATTLIST BrandSelBrandInfo
 ID                 ID      #REQUIRED
 ContentSoftwareId  CDATA   #IMPLIED >
```

Attributes:

ContentSoftwareId	See section 14. Glossary.

Content:

PackagedContent	Packaged Content elements (see section 3.7) that contain additional data that may be required by a particular payment brand. See the payment method supplement for IOTP for rules on how this is used.

7.8.2 Brand Selection Protocol Amount Info Element

The Brand Selection Protocol Amount Info Element contains any additional data that is payment protocol specific that may be required by a particular payment brand or payment protocol. See the IOTP payment method supplement for a description of how and when it used.

```
<!ELEMENT BrandSelProtocolAmountInfo (PackagedContent+) >
<!ATTLIST BrandSelProtocolAmountInfo
 ID                 ID      #REQUIRED
 ContentSoftwareId  CDATA   #IMPLIED >
```

Attributes:

ContentSoftwareId	See section 14. Glossary.

Content:

PackagedContent	Packaged Content elements (see section 3.7) that may contain additional data that may be required by a particular payment brand. See the payment method supplement for IOTP for rules on how this is used.

7.8.3 Brand Selection Currency Amount Info Element

The Brand Selection Currency Amount Info Element contains any additional data that is payment brand and currency specific that may be required by a particular payment brand. See the IOTP payment method supplement for a description of how and when it used.

```
<!ELEMENT BrandSelCurrencyAmountInfo (PackagedContent+) >
<!ATTLIST BrandSelCurrencyAmountInfo
 ID                 ID      #REQUIRED
 ContentSoftwareId  CDATA   #IMPLIED >
```

Attributes:

ContentSoftwareId	See section 14. Glossary.

Content:

PackagedContent	Packaged Content elements (see section 3.7) that contain additional data relating to the payment brand and currency. See the payment method supplement for IOTP for rules on how this is used.

7.9 Payment Component

A Payment Component contains information used to control how a payment is carried out. Its provides information on:

- o the times within which a Payment with a Payment Handler may be started
- o a reference to the Brand List (see section 7.7) which identifies the Brands, protocols, currencies and amounts which can be used to make a payment
- o whether or not a payment receipt will be provided

o whether another payment precedes this payment.

Its definition is as follows.

```
<!ELEMENT Payment EMPTY >
<!ATTLIST Payment
 ID                 ID      #REQUIRED
 OkFrom             CDATA   #REQUIRED
 OkTo               CDATA   #REQUIRED
 BrandListRef       NMTOKEN #REQUIRED
 SignedPayReceipt (True | False) #REQUIRED
 StartAfterRefs     NMTOKENS #IMPLIED >
```

Attributes:

ID	An identifier which uniquely identifies the Payment Component within the IOTP Transaction.
OkFrom	The date and time in [UTC] format after which a Payment Handler may accept for processing a Payment Request Block (see section 8.7) containing the Payment Component.
OkTo	The date and time in [UTC] format before which a Payment Handler may accept for processing a Payment Request Block containing the Payment Component.
BrandListRef	An Element Reference (see section 3.5) of a Brand List Component (see section 7.7) within the TPO Trading Block for the IOTP Transaction. The Brand List identifies the alternative ways in which the payment can be made.
SignedPayReceipt	Indicates whether or not the Payment Response Block (see section 8.9) generated by the Payment Handler for the payment must be digitally signed.
StartAfter	Contains Element References (see section 3.5) of other Payment Components which describe payments which must be complete before this payment can start. If no StartAfter attribute is present then there are no dependencies and the payment can start immediately

7.10 Payment Scheme Component

A Payment Scheme Component contains payment protocol information for a specific payment scheme which is transferred between the parties involved in a payment for example a [SET] message. Its definition is as follows.

```
<!ELEMENT PaySchemeData (PackagedContent+) >
<!ATTLIST PaySchemeData
 ID                  ID      #REQUIRED
 PaymentRef          NMTOKEN #IMPLIED
 ConsumerPaymentId   CDATA   #IMPLIED
 PaymentHandlerPayId CDATA   #IMPLIED
 ContentSoftwareId   CDATA   #IMPLIED >
```

Attributes:

ID	An identifier which uniquely identifies the Payment Scheme Component within the IOTP Transaction.
PaymentRef	An Element Reference (see section 3.5) to the Payment Component (see section 7.9) to which this Payment Scheme Component relates. It is required unless the Payment Scheme Component is part of an Transaction Inquiry Status Transaction (see section 9.2.1).
ConsumerPaymentId	An identifier specified by the Consumer which, if returned by the Payment Handler in another Payment Scheme Component or by other means, will enable the Consumer to identify which payment is being referred to.
PaymentHandlerPayId	An identifier specified by the Payment Handler which, if returned by the Consumer in another Payment Scheme Component, or by other means, will enable the Payment Handler to identify which payment is being referred to. It is required on every Payment Scheme Component apart from the one contained in a Payment Request Block.
ContentSoftwareId	See section 14. Glossary.

Content:

PackagedContent	Contains payment scheme protocol information as Packaged Content elements (see section 3.7). See the payment scheme supplement for the definition of its content. Note that: o the values of the Name attribute of each packaged content element are defined by the Payment Protocol Supplement o the value of each Name must be unique within a Payment where a Payment is defined as all Payment Scheme or Payment Receipt Components with the same value of the PaymentRef attribute

7.11 Payment Receipt Component

A Payment Receipt is a record of a payment which demonstrates how much money has been paid or received. It is distinct from a purchase receipt in that it contains no record of what was being purchased.

Typically the content of a Payment Receipt Component will contain data which describes:

- o the amount paid and its currency
- o the date and time of the payment
- o internal reference numbers which identify the payment to the payment system
- o potentially digital signatures generated by the payment method which can be used to prove after the event that the payment occurred.

If the Payment Method being used provides the facility then the Payment Receipt Component should contain payment protocol messages, or references to messages, which prove the payment occurred.

The precise definition of the content is Payment Method dependent. Refer to the supplement for the payment method being used to determine the rules that apply.

Information contained in the Payment Receipt Component should be displayed or otherwise made available to the Consumer.

Note: If the Payment Receipt Component contains Payment Protocol Messages, then the Messages will need to be processed by Payment Method software to convert it into a format which can be understood by the Consumer

The definition of a Payment Receipt Component is as follows.

```
<!ELEMENT PayReceipt (PackagedContent*) >
<!ATTLIST PayReceipt
 ID                 ID      #REQUIRED
 PaymentRef         NMTOKEN #REQUIRED
 PayReceiptNameRefs NMTOKENS #IMPLIED
 ContentSoftwareId  CDATA   #IMPLIED >
```

Attributes:

```
ID                    An identifier which uniquely identifies the
                      Payment Receipt Component within the IOTP
                      Transaction.

PaymentRef            Contains an Element Reference (see section 3.5)
                      to the Payment Component (see section 7.9) to
                      which this payment receipt applies

PayReceiptNameRefs    Optionally contains a list of the values of the
                      Name attributes of Packaged Content elements that
                      together make up the receipt. The Packaged
                      Content elements are contained either within:
                       o Payment Scheme Data components exchanged
                         between the Payment Handler and the Consumer
                         roles during the Payment, and/or
                       o the Payment Receipt component itself.
                      Note that:
                       o each payment scheme defines in its supplement
                         the Names of the Packaged Content elements
                         that must be listed in this attribute (if
                         any).
                       o if a Payment Scheme Component contains
                         Packaged Content elements with a name that
                         matches a name within PayReceiptNameRefs, then
                         those Payment Scheme Components must be
                         referenced by Digests in the Payment Response
                         signature component (if such a signature is
                         being used)

                      The client software should save all the
                      components referenced so that the payment receipt
                      can be reconstructed when required.
```

ContentSoftwareId	See section 14. Glossary.

Content:

PackagedContent	Optionally contains payment scheme payment receipt information as Packaged Content elements (see section 3.7). See the payment scheme supplement for the definition of its content. Note that: o the values of the Name attribute of each packaged content element are defined by the Payment Protocol Supplement o the value of each Name must be unique within a Payment where a Payment is defined as all Payment Scheme or Payment Receipt Components, with the same value of the PaymentRef attribute

Note that either the PayReceiptNameRefs attribute, the PackagedContent element, or both must be present.

7.12 Payment Note Component

The Payment Note Component contains additional, non payment related, information which the Payment Handler wants to provide to the Consumer. For example, if a withdrawal or deposit were being made then it could contain information on the remaining balance on the account after the transfer was complete. The information should duplicate information contained within the Payment Receipt Component.

Information contained in the Payment Note Component should be displayed or otherwise made available to the Consumer. For interoperability, the Payment Note Component should support, as a minimum, the content types of "Plain Text", HTML and XML. Its definition is as follows.

```
<!ELEMENT PaymentNote (PackagedContent+) >
<!ATTLIST PaymentNote
  ID                ID      #REQUIRED
  ContentSoftwareId CDATA   #IMPLIED >
```

Attributes:

ID	An identifier which uniquely identifies the Payment Receipt Component within the IOTP Transaction.
ContentSoftwareId	See section 14. Glossary.

Content:

PackagedContent	Contains additional, non payment related, information which the Payment Handler wants to provide to the Consumer as one or more Packaged Content elements (see section 3.7).

7.13 Delivery Component

The Delivery Element contains information required to deliver goods or services. Its definition is as follows.

```
<!ELEMENT Delivery (DeliveryData?, PackagedContent*) >
<!ATTLIST Delivery
 ID                ID      #REQUIRED
 xml:lang          NMTOKEN #REQUIRED
 DelivExch         (True | False) #REQUIRED
 DelivAndPayResp   (True | False) #REQUIRED
 ActionOrgRef      NMTOKEN #IMPLIED >
```

Attributes:

ID	An identifier which uniquely identifies the Delivery Component within the IOTP Transaction.
xml:lang	Defines the language used by attributes or child elements within this component, unless overridden by an xml:lang attribute on a child element. See section 3.8 Identifying Languages.
DelivExch	Indicates if this IOTP Transaction includes the messages associated with a Delivery Exchange. Valid values are: o True indicates it does include a Delivery Exchange o False indicates it does not include a Delivery Exchange If set to true then a DeliveryData element must be present. If set to false it may be absent.
DelivAndPayResp	Indicates if the Delivery Response Block (see section 8.11) and the Payment Response Block (see section 8.9) are combined into one IOTP Message. Valid values are: o True indicates both blocks will be in the same IOTP Message, and

```
                             o False indicates each block will be in a
                               different IOTP Message

                            DelivAndPayResp should not be true if DelivExch
                            is False.

                            In practice combining the Delivery Response Block
                            and Payment Response Block is only likely to be
                            practical if the Merchant, the Payment Handler
                            and the Delivery Handler are the same
                            Organisation since:
                             o the Payment Handler must have access to Order
                               Component information so that they know what
                               to deliver, and
                             o the Payment Handler must be able to carry out
                               the delivery

   ActionOrgRef             An Element Reference to the Organisation
                            Component of the Delivery Handler for this
                            delivery.

   Content:

   DeliveryData            Contains details about how the delivery will be
                           carried out. See 7.13.1 Delivery Data Element
                           below.

   PackagedContent         Contains "user" data defined for the Merchant
                           which is required by the Delivery Handler as one
                           or more Packaged Content Elements see section 3.7.
```

7.13.1 Delivery Data Element

The DeliveryData element contains information about where and how goods are to be delivered. Its definition is as follows.

```
   <!ELEMENT DeliveryData (PackagedContent*) >
   <!ATTLIST DeliveryData
    xml:lang           NMTOKEN #IMPLIED
    OkFrom             CDATA   #REQUIRED
    OkTo               CDATA   #REQUIRED
    DelivMethod        NMTOKEN #REQUIRED
    DelivToRef         NMTOKEN #REQUIRED
    DelivReqNetLocn    CDATA   #REQUIRED
    SecDelivReqNetLocn CDATA   #REQUIRED
    ContentSoftwareId  CDATA   #IMPLIED >
```

Attributes:

xml:lang	Defines the language used by attributes within this component. See section 3.8 Identifying Languages.
OkFrom	The date and time in [UTC] format after which the Delivery Handler may accept for processing a Delivery Request Block (see section 8.10).
OkTo	The date and time in [UTC] format before which the Delivery Handler may accept for processing a Delivery Request Block.
DelivMethod	Indicates the method by which goods or services may be delivered. Valid values are: o Post the goods will be delivered by post or courier o Web the goods will be delivered electronically in the Delivery Note Component o Email the goods will be delivered electronically by e-mail Values of DelivMethod are managed under the procedure described in section 12 IANA Considerations which allows user defined codes to be defined.
DelivToRef	The Element Reference (see section 3.4) of an Organisation Component within the IOTP Transaction which has a role of DelivTo. The information in this block is used to determine where delivery is to be made. It must be compatible with DelivMethod. Specifically if the DelivMethod is: o Post, then the there must be a Postal Address Element containing sufficient information for a postal delivery, o Web, then there are no specific requirements. The information will be sent in a web page back to the Consumer o Email, then there must be Contact Information Element with a valid e-mail address
DelivReqNetLocn	This contains the Net Location to which an unsecured Delivery Request Block (see section 8.10) which contains the Delivery Component should be sent.

	The content of this attribute is dependent on the Transport Mechanism and must conform to [RFC1738].
SecDelivReqNetLocn	This contains the Net Location to which a secured Delivery Request Block (see section 8.10) which contains the Delivery Component should be sent. A secured delivery request involves the use of a secure channel such as [SSL/TLS] in order to communicate with the Payment Handler. The content of this attribute is dependent on the Transport Mechanism must conform to [RFC1738]. See also Section 3.9 Secure and Insecure Net Locations.
ContentSoftwareId	See section 14. Glossary.

Content:

PackagedContent	Additional information about the delivery as one or more Packaged Content elements (see section 3.7) provided to the Delivery Handler by the merchant.

7.14 Consumer Delivery Data Component

A Consumer Delivery Data Component is used by a Consumer to specify an identifier that can be used by the Consumer to identify the Delivery.

Its definition is as follows:

```
<!ELEMENT ConsumerDeliveryData EMPTY >
<!ATTLIST ConsumerDeliveryData
 ID                  ID      #REQUIRED
 ConsumerDeliveryId CDATA   #REQUIRED>
```

Attributes:

ID	An identifier which uniquely identifies the Consumer Delivery Data Component within the IOTP Transaction.

ConsumerDeliveryId	An identifier specified by the Consumer which, if returned by the Delivery Handler will enable the Consumer to identify which Delivery is being referred to.

7.15 Delivery Note Component

A Delivery Note contains delivery instructions about the delivery of goods or services or potentially the actual Delivery Information itself. It is information which the person or Organisation receiving the Delivery Note can use when delivery occurs.

For interoperability, the Delivery Note Component Packaged Content should support both Plain Text, HTML and XML.

It’s definition is as follows.

```
<!ELEMENT DeliveryNote (PackagedContent+) >
<!ATTLIST DeliveryNote
 ID                 ID      #REQUIRED
 xml:lang           NMTOKEN #REQUIRED
 DelivHandlerDelivId CDATA  #IMPLIED
 ContentSoftwareId  CDATA   #IMPLIED >
```

Attributes:

ID	An identifier which uniquely identifies the Delivery Note Component within the IOTP Transaction.
xml:lang	Defines the language used by attributes or child elements within this component, unless overridden by an xml:lang attribute on a child element. See section 3.8 Identifying Languages.
DelivHandlerDelivId	An optional identifier specified by the Delivery Handler which, if returned by the Consumer in another Delivery Component, or by other means, will enable the Delivery Handler to identify which Delivery is being referred to. It is required on every Delivery Component apart from the one contained in a Delivery Request Block. An example use of this attribute is to contain a delivery tracking number.
ContentSoftwareId	See section 14. Glossary.

Content:

PackagedContent	Contains actual delivery note information as one or more Packaged Content elements (see section 3.7).

Note: If the content of the Delivery Message is a Mime message then the Delivery Note may trigger an application which causes the actual delivery to occur.

7.16 Status Component

A Status Component contains status information about the business success or failure (see section 4.2) of a process.

Its definition is as follows.

```
<!ELEMENT Status EMPTY >
<!ATTLIST Status
 ID                  ID      #REQUIRED
 xml:lang            NMTOKEN #REQUIRED
 StatusType          NMTOKEN #REQUIRED
 ElRef               NMTOKEN #IMPLIED
 ProcessState (NotYetStarted | InProgress |
     CompletedOk | Failed | ProcessError) #REQUIRED
 CompletionCode      NMTOKEN #IMPLIED
 ProcessReference    CDATA   #IMPLIED
 StatusDesc          CDATA   #IMPLIED >
```

Attributes:

ID	An identifier which uniquely identifies the Status Component within the IOTP Transaction.
xml:lang	Defines the language used by attributes within this component. See section 3.8 Identifying Languages.
StatusType	Indicates the type of Document Exchange which the Status is reporting on. It may be set to either Offer, Payment, Delivery, Authentication or Undefined. Undefined means that the type of document exchange could not be identified. This is caused by an error in the initial input message of the exchange.

	Values of StatusType are managed under the procedure described in section 12 IANA Considerations which also allows user defined values of StatusType to be defined.
ElRef	If the StatusType is not set to Undefined then ElRef contains an Element Reference (see section 3.5) to the Component for which the Status is being described. It must refer to either: o an Order Component (see section 7.5), if the StatusType is Offer, o a Payment Component (see section 7.9), if the StatusType is Payment, or o a Delivery Component (see section 7.13), if the StatusType is Delivery o an Authentication Request Component (see section 7.2) if the StatusType is Authentication.
ProcessState	Contains a State Code which indicates the current state of the process being carried out. Valid values for ProcessState are: o NotYetStarted. A Request Block has been received but the process has not yet started o InProgress. Processing of the Request Block has started but it is not yet complete o CompletedOk. The processing of the Request Block has completed successfully without any errors o Failed. The processing of the Request Block has failed because of a Business Error (see section 4.2) o ProcessError. This value is only used when the Status Component is being used in connection with an Inquiry Request Trading Block (see section 8.12). It indicates there was a Technical Error (see section 4.1) in the Request Block which is being processed or some internal processing error. Note that this code reports on the processing of a Request Block. Further, asynchronous processing may occur after the Response Block associated with the Process has been sent.

CompletionCode	Indicates how the process completed. Valid values for the CompletionCode are given below together with the conditions when it must be present and indications on when recovery from failures are possible. A CompletionCode is a maximum of 14 characters long.
ProcessReference	This optional attribute holds a reference for the process whose status is being reported. It may hold the following values: o when StatusType is set to Offer, it should contain the OrderIdentifier from the Order Component o when StatusType is set to Payment, it should contain the PaymentHandlerPayId from the Payment Scheme Data Component o when StatusType is set to Delivery, it should contain the DelivHandlerDelivId from the Delivery Note Component o when StatusType is set to Authentication, it should contain the AuthenticationId from the Authentication Request Component This attribute should be absent in the Inquiry Request message when the Consumer has not been given such a reference number by the IOTP Service Provider. This attribute can be used inside an Inquiry Response Block (see section 8.13) to give the reference number for a transaction which has previously been unavailable. For example, the package tracking number might not be assigned at the time a delivery response was received. However, if the Consumer issues a Baseline Transaction Status Inquiry later, the Delivery Handler can put the package tracking number into this attribute in the Inquiry Response message and send it back to the Consumer.
StatusDesc	An optional textual description of the current status of the process in the language identified by xml:lang.

7.16.1 Offer Completion Codes

The Completion Code is only required if the ProcessState attribute is set to Failed. The following table contains the valid values for the CompletionCode that may be used and indicates whether or not recovery might be possible. It is recommended that the StatusDesc attribute is used to provide further explanation where appropriate.

Value	Description
AuthError	Authentication Error. The check of the Authentication Response which was carried out has failed. Recovery may be possible by the Consumer re-submitting a new Authentication Response Block with corrected information.
ConsCancelled	Consumer Cancelled. The Consumer decides to cancel the transaction for some reason. This code is only valid in a Status Component contained in a Cancel Block or an Inquiry Response Block. No recovery possible.
MerchCancelled	Offer Cancelled. The Merchant declines to generate an offer for some reason and cancels the transaction. This code is only valid in a Status Component contained in a Cancel Block or an Inquiry Response Block. No recovery possible.
Unspecified	Unspecified error. There is some unknown problem or error which does not fall into one of the other CompletionCodes. No recovery possible.
TimedOutRcvr	Recoverable Time Out. Messages were resent but no response received. The document exchange has therefore "Timed Out". This code is only valid on a Transaction Inquiry. Recovery is possible if the last message from the other Trading Role is received again.

TimedOutNoRcvr	Non Recoverable Time Out. Messages were resent but no response received. The document exchange has therefore "Timed Out". This code is only valid on a Transaction Inquiry. No recovery possible.

7.16.2 Payment Completion Codes

The CompletionCode is only required if the ProcessState attribute is set to Failed. The following table contains the valid values for the CompletionCode that may be used and indicates where recovery may be possible. It is recommended that the StatusDesc attribute is used by individual payment schemes to provide further explanation where appropriate.

Value	Description
BrandNotSupp	Brand not supported. The payment brand is not supported by the Payment Handler. See below for recovery options.
CurrNotSupp	Currency not supported. The currency in which the payment is to be made is not supported by either the Payment Instrument or the Payment Handler. If the payment is Brand Independent, then the Consumer may recover by selecting a different currency, if available, or a different brand. Note that this may involve a different Payment Handler.
ConsCancelled	Consumer Cancelled. The Consumer decides to cancel the payment for some reason. This code is only valid in a Status Component contained in a Cancel Block or an Inquiry Response Block. Recovery is not possible.
PaymtCancelled	Payment Cancelled. The Payment Handler declines to complete the payment for some reason and cancels the transaction. This code is only valid in a Status Component contained in a Cancel Block or an Inquiry Response Block. See below for recovery options.

AuthError	Authentication Error. The Payment Scheme specific authentication check which was carried out has failed. Recovery may be possible. See the payment scheme supplement to determine what is allowed.
InsuffFunds	Insufficient funds. There are insufficient funds available for the payment to be made. See below for recovery options.
InstBrandInvalid	Payment Instrument not valid for Brand. A Payment Instrument is being used which does not correspond with the Brand selected. For example a Visa credit card is being used when MasterCard was selected as the Brand. See below for recovery options.
InstNotValid	Payment instrument not valid for trade. The Payment Instrument cannot be used for the proposed type of trade, for some reason. See below for recovery options.
BadInstrument	Bad instrument. There is a problem with the Payment Instrument being used which means that it is unable to be used for the payment. See below for recovery options.
Unspecified	Unspecified error. There is some unknown problem or error which does not fall into one of the other CompletionCodes. The StatusDesc attribute should provide the explanation of the cause. See below for recovery options.
TimedOutRcvr	Recoverable Time Out. Messages were resent but no response received. The document exchange has therefore "Timed Out". This code is only valid on a Transaction Inquiry. Recovery is possible if the last message from the other Trading Role is received again.

TimedOutNoRcvr	Non Recoverable Time Out. Messages were resent but no response received. The document exchange has therefore "Timed Out". This code is only valid on a Transaction Inquiry. No recovery possible.

If the Payment is Brand Independent, then recovery may be possible for some values of the Completion Code, by the Consumer selecting either a different payment brand or a different payment instrument for the same brand. Note that this might involve a different Payment Handler. The codes to which this applies are: BrandNotSupp, PaymtCancelled, InsuffFunds, InstBrandInvalid, InstNotValid, BadInstrument and Unspecified.

Recovery from Payments associated with Brand Dependent purchases is only possible, if the Brand Selection component sent by the Merchant to the Consumer does not change. In practice this means that the same Brand, Protocol Amount and PayProtocol elements must be used. All that can change is the Payment Instrument. Any other change will invalidate the Merchant's Offer as a changed selection will invalidate the Offer Response.

7.16.3 Delivery Completion Codes

The following table contains the valid values for the CompletionCode attribute for a Delivery. It is recommended that the StatusDesc attribute is used to provide further explanation where appropriate.

Value	Description
BackOrdered	Back Ordered. The goods to be delivered are on order but they have not yet been received. Shipping will be arranged when they are received. This is only valid if ProcessState is CompletedOk. Recovery is not possible.
PermNotAvail	Permanently Not Available. The goods are permanently unavailable and cannot be re-ordered. This is only valid if ProcessState is Failed. Recovery is not possible.
TempNotAvail	Temporarily Not Available. The goods are temporarily unavailable and may become available if they can be ordered. This is only valid if ProcessState is CompletedOk.

```
                        Recovery is not possible.

   ShipPending          Shipping Pending. The goods are available and are
                        scheduled for shipping but they have not yet been
                        shipped. This is only valid if ProcessState is
                        CompletedOk.

                        Recovery is not possible.

   Shipped              Goods Shipped. The goods have been shipped.
                        Confirmation of delivery is awaited. This is only
                        valid if ProcessState is CompletedOk.

                        Recovery is not possible.

   ShippedNoConf        Shipped - No Delivery Confirmation. The goods have
                        been shipped but it is not possible to confirm
                        delivery of the goods. This is only valid if
                        ProcessState is CompletedOk.

                        Recovery is not possible.

   ConsCancelled        Consumer Cancelled. The Consumer decides to cancel
                        the delivery for some reason. This code is only valid
                        in a Status Component contained in a Cancel Block or
                        an Inquiry Response Block.

                        Recovery is not possible.

   DelivCancelled       Delivery Cancelled. The Delivery Handler declines to
                        complete the Delivery for some reason and cancels the
                        transaction. This code is only valid in a Status
                        Component contained in a Cancel Block or an Inquiry
                        Response Block.

                        Recovery is not possible.

   Confirmed            Confirmed. All goods have been delivered and
                        confirmation of their delivery has been received.
                        This is only valid if ProcessState is CompletedOk.

                        Recovery is not possible.

   Unspecified          Unspecified error. There is some unknown problem or
                        error which does not fall into one of the other
                        CompletionCodes. The StatusDesc attribute should
                        provide the explanation of the cause.
```

	Recovery is not possible.
TimedOutRcvr	Recoverable Time Out. Messages were resent but no response received. The document exchange has therefore "Timed Out". This code is only valid on a Transaction Inquiry. Recovery is possible if the last message from the other Trading Role is received again.
TimedOutNoRcvr	Non Recoverable Time Out. Messages were resent but no response received. The document exchange has therefore "Timed Out". This code is only valid on a Transaction Inquiry. No recovery possible.

Note: Recovery from failed, or partially completed deliveries is not possible. The Consumer should use the Transaction Status Inquiry Transaction (see section 9.2.1) to determine up-to- date information on the current state.

7.16.4 Authentication Completion Codes

The Completion Code is only required if the ProcessState attribute is set to Failed. The following table contains the valid values for the CompletionCode that may be used. It is recommended that the StatusDesc attribute is used to provide further explanation where appropriate.

Value	Description
AutEeCancel	Authenticatee Cancel. The Organisation being authenticated declines to be authenticated for some reason. This could be, for example because the signature on an Authentication Request was invalid or the Authenticator was not known or acceptable to the Authenticatee. Recovery is not possible.
AutOrCancel	Authenticator Cancel. The Organisation requesting authentication declines to validate the Authentication Response received for some reason and cancels the transaction. Recovery is not possible.

NoAuthReq	Authentication Request Not Available. The Authenticatee does not have the data that must be provided so that they may be successfully authenticated. For example a password may have been forgotten, the Authenticatee has not yet become a member, or a smart card token is not present. Recovery is not possible
AuthFailed	Authentication Failed. The Authenticator checked the Authentication Response but the authentication failed for some reason. For example a password may have been incorrect. Recovery may be possible by the Authenticatee re-sending a revised Authentication Response with corrected data.
TradRolesIncon	Trading Roles Inconsistent. The Trading Roles contained within the TradingRoleList attribute of the Trading Role Information Request Component (see section 7.4) are inconsistent with the Trading Role which the Authenticatee is taking in the IOTP Transaction or is able to take. Examples of inconsistencies include: o asking a PaymentHandler for DeliveryHandler information o asking a Consumer for Merchant information Recovery may be possible by the Authenticator re-sending a revised Authentication Request Block with corrected information.
Unspecified	Unspecified error. There is some unknown problem or error which does not fall into one of the other CompletionCodes. Recovery is not possible.
TimedOutRcvr	Recoverable Time Out. Messages were resent but no response received. The document exchange has therefore "Timed Out". This code is only valid on a Transaction Inquiry. Recovery is possible if the last message from the other Trading Role is received again.

Value	Description
TimedOutNoRcvr	Non Recoverable Time Out. Messages were resent but no response received. The document exchange has therefore "Timed Out". This code is only valid on a Transaction Inquiry. No recovery possible.

7.16.5 Undefined Completion Codes

The Completion Code is only required if the ProcessState attribute is set to Failed. The following table contains the valid values for the CompletionCode that may be used. It is recommended that the StatusDesc attribute is used to provide further explanation where appropriate.

Value	Description
InMsgHardError	Input Message Hard Error. The type of Request Block could not be identified or was inconsistent. Therefore no single Document Exchange could be identified. This will cause a Hard Error in the transaction

7.16.6 Transaction Inquiry Completion Codes

The Completion Code is only required if the ProcessState attribute is set to Failed. The following table contains the valid values for the CompletionCode that may be used. It is recommended that the StatusDesc attribute is used to provide further explanation where appropriate.

Value	Description
UnAuthReq	Unauthorised Request. The recipient of the Transaction Status Request declines to respond to the request.

7.17 Trading Role Data Component

The Trading Role Data Component contains opaque data which needs to be communicated between the Trading Roles involved in an IOTP Transaction.

Trading Role Components identify:

o the Organisation that generated the component, and

o the Organisation that is to receive it.

They are first generated and included in a "Response" Block, and then copied to the appropriate "Request" Block. For example a Payment Handler might need to inform a Delivery Handler that a credit card payment had been authorised but not captured. There may also be other information that the Payment Handler has generated where the format is privately agreed with the Delivery Handler which needs to be communicated. In another example a Merchant might need to provide a Payment Handler with some specific information about a Consumer so that consumer can acquire double loyalty points with the payment.

Its definition is as follows.

```
<!ELEMENT TradingRoleData (PackagedContent+) >
<!ATTLIST TradingRoleData
  ID                ID      #REQUIRED
  OriginatorElRef   NMTOKEN #REQUIRED
  DestinationElRefs NMTOKENS #REQUIRED >
```

Attributes:

ID	An identifier which uniquely identifies the Trading Role Data Component within the IOTP Transaction.
OrginatorElRef	Contains an element reference to the Organisation Component of the Organisation that created the Trading Role Data Component and included it in a "Response" Block (e.g., an Offer Response or a Payment Response Block).
DestinationElRefs	Contains element references to the Organisation Components of the Organisations that are to receive the Trading Role Data Component in a "Request" Block (e.g., either a Payment Request or a Delivery Request Block).

Content:

PackagedContent	This contains the data which is to be sent between the various Trading Roles as one or more PackagedContent elements see section 3.7.

7.17.1 Who Receives a Trading Role Data Component

The rules for deciding what to do with Trading Role Data Components are described below.

- whenever a Trading Role Data Component is received in a "Response" block identify the Organisation Components of the Organisations that are to receive it as identified by the DestinationElRefs attribute.

- whenever a "Request" Block is being sent, check to see if it is being sent to one of the Organisations identified by the DestinationElRefs attribute. If it is then include in the "Request" block:

  - the Trading Role Data Component as well as,

  - the Organisation Component of the Organisation identified by the OriginatorElRef attribute (if not already present)

7.18 Inquiry Type Component

The Inquiry Type Component contains the information which indicates the type of process that is being inquired upon. Its definition is as follows.

```
<!ELEMENT InquiryType EMPTY >
<!ATTLIST InquiryType
 ID                 ID      #REQUIRED
 Type               NMTOKEN #REQUIRED
 ElRef              NMTOKEN #IMPLIED
 ProcessReference   CDATA   #IMPLIED >
```

Attributes:

ID	An identifier which uniquely identifies the Inquiry Type Component within the IOTP Transaction.
Type	Contains the type of inquiry. Valid values for Type are: o Offer. The inquiry is about the status of an offer and is addressed to the Merchant. o Payment. The inquiry is about the status of a payment and is addressed to the Payment Handler. o Delivery. The inquiry is about the status of a delivery and addressed to the Delivery Handler.
ElRef	Contains an Element Reference (see section 3.5) to the component to which this Inquiry Type Component applies. That is, o TPO Block when Type is Offer

	o Payment Component when Type is Payment o Delivery Component when Type is Delivery
ProcessReference	Optionally contains a reference to the process being inquired upon. It should be set if the information is available. For the definition of the values it may contain, see the ProcessReference attribute of the Status Component (see section 7.16).

7.19 Signature Component

Note: Definitions of the XML structures for signatures and certificates are described in the document titled "Digital Signatures for the Internet Open Trading Protocol" by Kent Davidson and Yoshiaki Kawatsura published at the same time as this document - see [IOTPDSIG].

In the future it is anticipated that future versions of IOTP will adopt a whatever method for digitally signing XML becomes the standard.

Each Signature Component digitally signs one or more Blocks or Components including other Signature Components.

The Signature Component:

- contains digests of one or more Blocks or Components in one or more IOTP Messages within the same IOTP Transaction and places the result in a Digest Element

- concatenates these Digest elements with other information on the type of signature, the originator and potential recipients of the signature and details of the signature algorithms being used and places them in a Manifest element, and

- signs the Manifest element using the optional certificate identified in the Certificate element within the Signature Block placing the result in a Value element within a Signature Component

Note that there may be multiple Value elements that contain signatures of a Manifest Element.

A Signature Component can be one of four types either:

- an Offer Response Signature,

- a Payment Response Signature,

o a Delivery Response Signature, or

o an Authentication Response Signature.

For a general explanation of signatures see section 6 Digital Signatures.

7.19.1 IOTP usage of signature elements and attributes

Definitions of the elements and attributes are contained in [IOTPDSIG]. The following contains additional information that describes how these elements and attributes are used by IOTP.

SIGNATURE ELEMENT

The ID attribute is mandatory.

MANIFEST ELEMENT

The optional LocatorHrefBase attribute contains text which should be concatenated before the text contained in the LocatorHREF attribute of all Digest elements within the Manifest.

Its purpose is to reduce the size of LocatorHREF attribute values since the first part of the LocatorHREF attributes in the same signature are likely to be the same.

Typically, within IOTP, it will contain all the characters in a LocatorHref attribute up to the sharp ("#") character (see immediately below).

ALGORITHM AND PARAMETER ELEMENTS

The algorithm element identifies the algorithms used in generating the signature. The type of the algorithm is defined by the value of the Type attribute which indicates if it is to be used as a Digest algorithm, a Signature algorithm or a Key Agreement algorithm.

The following Digest algorithms must be implemented:

o a [DOM-HASH] algorithm. This is identified by setting the Name attribute of the Algorithm element to "urn:ibm:dom-hash"

o a [SHA1] algorithm. This is identified by setting the Name attribute of the Algorithm element to "urn:fips:sha1", and

o a [MD5] algorithm. This is identified by setting the Name attribute of the Algorithm element to "urn:rsa:md5" o The following Signature algorithms must be implemented:

o a [DSA] algorithm. This is identified by setting the Name attribute of the Algorithm element to "urn:us.gov:dsa"

o a [HMAC] algorithm. This is identified by setting the Name attribute of the Algorithm element to "urn:ibm:hmac"

It is recommended that the following Signature algorithm is also implemented:

o a [RSA] algorithm. This is identified by setting the Name attribute of the Algorithm element to "urn:rsa:rsa"

In addition other payment scheme specific algorithms may be used. In this case the value of the name attribute to use is specified in the payment scheme supplement for that algorithm.

One algorithm may make use of other algorithms by use of the Parameter element, for example:

```
<Algorithm ID=A1 type="digest" name="urn:ibm:dom-hash">
  <Parameter type='AlgorithmRef'>A2</Parameter>
</Algorithm>
<Algorithm ID=A2 type="digest" name="urn:fips:sha1">
</Algorithm>
<Algorithm ID=A3 type="signature" name="urn:ibm:hmac">
    <Parameter type='AlgorithmRef'>A1</Parameter>
</Algorithm>
```

DIGEST ELEMENT

The LocatorHREF attribute identifies the IOTP element which is being digitally signed. Specifically it consists of:

o the value of the IotpTransId attribute of the Transaction ID Component, followed by:

o a sharp character, i.e. "#", followed by

o an Element Reference (see section 3.5) to the element within the IOTP Transaction which is the subject of the digest.

Before analysing the structure of the LocatorHREF attribute, it must be concatenated with the value of the LocatorHrefBase attribute of the Manifest element (see immediately above).

ATTRIBUTE ELEMENT

There must be one and only one Attribute Element that contains a Type attribute with a value of IOTP Signature Type and with content set to either: OfferResponse, PaymentResponse, DeliveryResponse,

AuthenticationRequest, AuthenticationResponse, PingRequest or PingResponse; depending on the type of the signature.

Values of the content of the Attribute element are controlled under the procedures defined in section 12 IANA Considerations which also allows user defined values to be defined.

The Critical attribute must be set to true.

ORIGINATORINFO ELEMENT

The OriginatorRef attribute of the OriginatorInfo element must always be present and contain an Element Reference (see section 3.5) to the Organisation Component of the Organisation that generated the Signature Component.

RECIPIENTINFO ELEMENT

The RecipientRefs attribute contains a list of Element References (see section 3.5), that point to the Organisations that might need to validate the signature. For details see below.

7.19.2 Offer Response Signature Component

The Manifest Element of a signature which has a type of OfferResponse should contain Digest elements for the following Components:

- the Transaction Id Component (see section 3.3.1) of the IOTP message that contains the Offer Response Signature
- the Transaction Reference Block (see section 3.3) of the IOTP Message that contains the Offer Response Signature
- from the TPO Block:
  - the Protocol Options Component
  - each of the Organisation Components
  - each of the Brand List Components

- o optionally, all the Brand Selection Components if they were sent to the Merchant in a TPO Selection Block

- o from the Offer Response Block:

  - the Order Component

  - each of the Payment Components

  - the Delivery Component

  - each of the Authentication Request Components

  - any Trading Role Data Components

The Offer Response Signature should also contain Digest elements for the components that describe each of the Organisations that may or will need to verify the signature. This involves:

- o if the Merchant has received a TPO Selection Block containing Brand Selection Components, then generate a Digest element for the Payment Handler identified by the Brand Selection Component and the Delivery Handler identified by the Delivery Component. See section 6.3.1 Check Request Block sent Correct Organisation for a description of how this can be done.

- o if the Merchant is not expecting to receive a TPO Selection Block then generate a Digest element for the Delivery Handler and all the Payment Handlers that are involved.

7.19.3 Payment Receipt Signature Component

The Manifest Element of the Payment Receipt Signature Component should contain Digest Elements for the following Components:

- o the Transaction Id Component (see section 3.3.1) of the IOTP message that contains the Payment Receipt Signature

- o the Transaction Reference Block (see section 3.3) of the IOTP Message that contains the Payment Receipt Signature

- o the Offer Response Signature Component

- o the Payment Receipt Component

- o the Payment Note Component

- o the Status Component

- the Brand Selection Component.

- any Trading Role Data Components

7.19.4 Delivery Response Signature Component

The Manifest Element of the Delivery Response Signature Component should contain Digest Elements for the following Components:

- the Transaction Id Component (see section 3.3.1) of the IOTP message that contains the Delivery Response Signature

- the Transaction Reference Block (see section 3.3) of the IOTP Message that contains the Delivery Response Signature

- the Consumer Delivery Data component contained in the preceding Delivery Request (if any)

- the Signature Components contained in the preceding Delivery Request (if any)

- the Status Component

- the Delivery Note Component

7.19.5 Authentication Request Signature Component

The Manifest Element of the Authentication Request Signature Component should contain Digest Elements for the following Components:

- the Transaction Reference Block (see section 3.3) for the IOTP Message that contains information that describes the IOTP Message and IOTP Transaction

- the Transaction Id Component (see section 3.3.1) which globally uniquely identifies the IOTP Transaction

- the following components of the TPO Block :

  - the Protocol Options Component

  - the Organisation Component

- the following components of the Authentication Request Block:

  - the Authentication Request Component(s) (if present)

  - the Trading Role Information Request Component (if present)

7.19.6 Authentication Response Signature Component

The Manifest Element of the Authentication Response Signature Component should contain Digest Elements for the following Components:

- o the Transaction Reference Block (see section 3.3) for the IOTP Message that contains information that describes the IOTP Message and IOTP Transaction
- o the Transaction Id Component (see section 3.3.1) which globally uniquely identifies the IOTP Transaction
- o the following components of the Authentication Request Block:
  - the Authentication Request Component that was used in the Authentication (if present)
  - the Trading Role Information Request Component (if present)
- o the Organisation Components contained in the Authentication Response Block

7.19.7 Inquiry Request Signature Component

If the Inquiry Request is being signed (see section 9.2.1) the Manifest Element of the Inquiry Request Signature Component should contain Digest elements of the Inquiry Type Component, and if present, the Payment Scheme Component.

7.19.8 Inquiry Response Signature Component

If the Inquiry Response is being signed (see section 9.2.1) the Manifest Element of the Inquiry Response Signature Component should contain Digest elements of the Trading Response Block and the Status Component.

7.19.9 Ping Request Signature Component

If the Ping Request is being singed (see section 9.2.2), the Manifest Element of the Ping Request Signature Component should contain Digest elements for all the Organisation Components.

7.19.10 Ping Response Signature Component

If the Ping Response is being singed (see section 9.2.2), the Manifest Element of the Ping Response Signature Component should contain Digest elements fir all the Organisation Components.

7.20 Certificate Component

Note: Definitions of the XML structures for signatures and certificates are described in the paper "Digital Signatures for the Internet Open Trading Protocol", see [IOTPDSIG].

See note at the start of section 7.19 Signature Component for more details.

A Certificate Component contains a Digital Certificate. They are used only when required, for example, when asymmetric cryptography is being used and the recipient of the signature that needs to check has not already received the Public Key.

The structure of a Certificate Component is defined in [IOTPDSIG].

7.20.1 IOTP usage of signature elements and attributes

Detailed definitions of the above elements and attributes are contained in [IOTPDSIG]. The following contains additional information that describes how these elements and attributes are used by IOTP.

CERTIFICATE COMPONENT

The ID attribute is mandatory.

VALUE ELEMENT

The ID attribute is mandatory.

7.21 Error Component

The Error Component contains information about Technical Errors (see section 4.1) in an IOTP Message which has been received by one of the Trading Roles involved in the trade.

For clarity two phrases are defined which are used in the description of an Error Component:

o message in error. An IOTP message which contains or causes an error of some kind o message reporting the error. An IOTP message that contains an Error Component that describes the error found in a message in error.

The definition of the Error Component is as follows.

```
<!ELEMENT ErrorComp (ErrorLocation+, PackagedContent*) >
<!ATTLIST ErrorComp
 ID                 NMTOKEN #REQUIRED
 xml:lang           NMTOKEN #REQUIRED
 ErrorCode          NMTOKEN #REQUIRED
 ErrorDesc          CDATA   #REQUIRED
 Severity (Warning|TransientError|HardError) #REQUIRED
 MinRetrySecs       CDATA   #IMPLIED
 SwVendorErrorRef   CDATA   #IMPLIED >
```

Attributes:

ID	An identifier which uniquely identifies the Error Component within the IOTP Transaction.
xml:lang	Defines the language used by attributes or child elements within this component, unless overridden by an xml:lang attribute on a child element. See section 3.8 Identifying Languages.
ErrorCode	Contains an error code which indicates the nature of the error in the message in error. Valid values for the ErrorCode are given in section 7.21.2 Error Codes.
ErrorDesc	Contains a narrative description of the error in the language defined by xml:lang. The content of this attribute is defined by the vendor/developer of the software which generated the Error Component
Severity	Indicates the severity of the error. Valid values are: o Warning. This indicates that although there is a message in error the IOTP Transaction can still continue. o TransientError. This indicates that the error in the message in error may be recovered if the message in error that is referred to by the ErrorLocation element is resent

	o HardError. This indicates that there is an unrecoverable error in the message in error and the IOTP Transaction must stop.
MinRetrySecs	This attribute should be present if Severity is set to TransientError. It is the minimum number of whole seconds which the IOTP aware application which received the message reporting the error should wait before re-sending the message in error identified by the ErrorLocation element. If Severity is not set to TransientError then the value of this attribute is ignored.
SwVendorErrorRef	This attribute is a reference whose value is set by the vendor/developer of the software which generated the Error Component. It should contain data which enables the vendor to identify the precise location in their software and the set of circumstances which caused the software to generate a message reporting the error. See also the SoftwareId attribute of the Message Id element in the Transaction Reference Block (section 3.3).

Content:

ErrorLocation	This identifies the IOTP Transaction Id of the message in error and, where possible, the element and attribute in the message in error that caused the Error Component to be generated. If the Severity of the error is not TransientError, more than one ErrorLocation may be specified as appropriate depending on the nature of the error (see section 7.21.2 Error Codes) and at the discretion of the vendor/developer of the IOTP Aware Application.
PackagedContent	This contains additional data which can be used to understand the error. Its content may vary as appropriate depending on the nature of the error (see section 7.21.2 Error Codes) and at the discretion of the vendor/developer of the IOTP Aware Application. For a definition of PackagedContent see section 3.7.

7.21.1 Error Processing Guidelines

If there is more than one Error Component in a message reporting the error, carry out the actions appropriate for the Error Component with the highest severity. In this context, HardError has a higher severity than TransientError, which has a higher severity than Warning.

7.21.1.1 Severity - Warning

If an IOTP aware application is generating a message reporting the error with an Error Component where the Severity attribute is set to Warning, then if the message reporting the error does not contain another Error Component with a severity higher than Warning, the IOTP Message must also include the Trading Blocks and Trading Components that would have been included if no error was being reported.

If a message reporting the error is received with an Error Component where Severity is set to Warning, then:

- it is recommended that information about the error is either logged, or otherwise reported to the user,
- the implementer of the IOTP aware application must either, at their or the user’s discretion:
  - continue the IOTP transaction as normal, or
  - fail the IOTP transaction by generating a message reporting the error with an Error Component with Severity set to HardError (see section 7.21.1.3).

If the intention is to continue the IOTP transaction then, if there are no other Error Components with a higher severity, check that the necessary Trading Blocks and Trading Components for normal processing of the transaction to continue are present. If they are not then generate a message reporting the error with an Error Component with Severity set to HardError.

7.21.1.2 Severity - Transient Error

If an IOTP Aware Application is generating a message reporting the error with an Error Component where the Severity attribute is set to TransientError, then there should be only one Error Component in the message reporting the error. In addition, the MinRetrySecs attribute should be present.

If a message reporting the error is received with an Error Component where Severity is set to TransientError then:

- o if the MinRetrySecs attribute is present and a valid number, then use the MinRetrySecs value given. Otherwise if MinRetrySecs is missing or is invalid, then:
  - - generate a message reporting the error containing an Error Component with a Severity of Warning and send it on the next IOTP message (if any) to be sent to the Trading Role which sent the message reporting the error with the invalid MinRetrySecs, and
  - - use a value for MinRetrySecs which is set by the vendor/developer of the IOTP Aware Application.
- o check that only one ErrorLocation element is contained within the Error Component and that it refers to an IOTP Message which was sent by the recipient of the Error Component with a Severity of TransientError. If more than one ErrorLocation is present then generate a message reporting the error with a Severity of HardError.

7.21.1.3 Severity - Hard Error

If an IOTP Aware Application is generating a message reporting the error with an Error Component where the Severity attribute set to HardError, then there should be only one Error Component in the message reporting the error.

If a message reporting the error is received with an Error Component where Severity is set to HardError then terminate the IOTP Transaction.

7.21.2 Error Codes

The following table contains the valid values for the ErrorCode attribute of the Error Component. The first sentence of the description contains the text that should be used to describe the error when displayed or otherwise reported. Individual implementations may translate this into alternative languages at their discretion.

An Error Code must not be more that 14 characters long.

Value	Description
Reserved	Reserved. This error is reserved by the vendor/developer of the software. Contact the vendor/developer of the software for more information See the SoftwareId attribute of the Message Id element in the Transaction Reference Block(section 3.3).
XmlNotWellFrmd	XML not well formed. The XML document is not well formed. See [XML] for the meaning of "well formed". Even if the XML is not well formed, it should still be scanned to find the Transaction Reference Block so that a properly formed Error Response may be generated.
XmlNotValid	XML not valid. The XML document is well formed but the document is not valid. See [XML] for the meaning of "valid". Specifically: o the XML document does not comply with the constraints defined in the IOTP document type declaration (DTD) (see section 13 Internet Open Trading Protocol Data Type Definition), and o the XML document does not comply with the constraints defined in the document type declaration of any additional [XML Namespace] that are declared. As for XML not well formed, attempts should still be made to extract the Transaction Reference Block so that a properly formed Error Response may be generated.
ElUnexpected	Unexpected element. Although the XML document is well formed and valid, an element is present that is not expected in the particular context according to the rules and constraints contained in this specification.
ElNotSupp	Element not supported. Although the document is well formed and valid, an element is present that: o is consistent with the rules and constraints contained in this specification, but o is not supported by the IOTP Aware Application which is processing the IOTP Message.

ElMissing	Element missing. Although the document is well formed and valid, an element is missing that should have been present if the rules and constraints contained in this specification are followed. In this case set the PackagedContent of the Error Component to the type of the missing element.
ElContIllegal	Element content illegal. Although the document is well formed and valid, the element Content contains values which do not conform to the rules and constraints contained in this specification.
EncapProtErr	Encapsulated protocol error. Although the document is well formed and valid, the PackagedContent of an element contains data from an encapsulated protocol which contains errors.
AttUnexpected	Unexpected attribute. Although the XML document is well formed and valid, the presence of the attribute is not expected in the particular context according to the rules and constraints contained in this specification.
AttNotSupp	Attribute not supported. Although the XML document is well formed and valid, and the presence of the attribute in an element is consistent with the rules and constraints contained in this specification, it is not supported by the IOTP Aware Application which is processing the IOTP Message.
AttMissing	Attribute missing. Although the document is well formed and valid, an attribute is missing that should have been present if the rules and constraints contained in this specification are followed. In this case set the PackagedContent of the Error Component to the type of the missing attribute.
AttValIllegal	Attribute value illegal. The attribute contains a value which does not conform to the rules and constraints contained in this specification.
AttValNotRecog	Attribute Value Not Recognised. The attribute contains a value which the IOTP Aware Application generating the message reporting the error could not recognise.

MsgTooLarge	Message too large. The message is too large to be processed by the IOTP Aware Application.
ElTooLarge	Element too large. The element is too large to be processed by the IOTP Aware Application
ValueTooSmall	Value too small or early. The value of all or part of the Content of an element or an attribute, although valid, is too small.
ValueTooLarge	Value too large or in the future. The value of all or part of the Content of an element or an attribute, although valid, is too large.
ElInconsistent	Element Inconsistent. Although the document is well formed and valid, according to the rules and constraints contained in this specification: o the content of an element is inconsistent with the content of other elements or their attributes, or o the value of an attribute is inconsistent with the value of one or more other attributes. In this case create ErrorLocation elements which identify all the attributes or elements which are inconsistent.
TransportError	Transport Error. This error code is used to indicate that there is a problem with the Transport Mechanism which is preventing the message from being received. It is typically associated with a Transient Error. Explanation of the Transport Error is contained within the ErrorDesc attribute. The values which can be used inside ErrorDesc with a TransportError is specified in the IOTP supplement for the Transport mechanism.
MsgBeingProc	Message Being Processed. This error code is only used with a Severity of Transient Error. It indicates that the previous message, which may be an exchange message or a request message, is being processed and, if no response is received by the time indicated by the MinRetrySecs attribute, then the original message should be resent.
SystemBusy	System Busy. This error code is only used with a Severity of Transient Error. It indicates that the server that received a message is currently too busy to handle the message. If no response is received by

```
                       the time indicated by the MinRetrySecs attribute,
                       then the original message should be resent.

   Note: If the server/system handling the Transport Mechanism (e.g.,
   HTTP) is busy then a Transport Specific error message should be used
   instead of an IOTP Error message. This code should be used in
   association with IOTP servers/systems or other servers/systems to
   which the IOTP server is connected.

   UnknownError    Unknown Error. Indicates that the transaction cannot
                   complete for some reason that is not covered
                   explicitly by any of the other errors.  The ErrorDesc
                   attribute should be used to indicate the nature of
                   the problem.

                   This could be used to indicate, for example, an
                   internal error in a backend server or client process
                   of some kind.
```

7.21.3 Error Location Element

An Error Location Element identifies an element and optionally an attribute in the message in error which is associated with the error. It contains a reference to the IOTP Message, Trading Block, Trading Component, element and attribute, which is in error.

```
   <!ELEMENT ErrorLocation EMPTY >
   <!ATTLIST ErrorLocation
    ElementType        NMTOKEN #REQUIRED
    IotpMsgRef         NMTOKEN #IMPLIED
    BlkRef             NMTOKEN #IMPLIED
    CompRef            NMTOKEN #IMPLIED
    ElementRef         NMTOKEN #IMPLIED
    AttName            NMTOKEN #IMPLIED >
```

Attributes:

```
   ElementType        This is the name of the type of the element where
                      the error is located. For example if the element
                      was declared as <!ELEMENT Org ... then its name is
                      "Org".

   IotpMsgRef         This is the value of the ID attribute of the of
                      the Message Id Component (see section 3.3.2) of
                      the message in error to which this Error Component
                      applies.
```

BlkRef	If the error is associated with a specific Trading Block, then this is the value of the ID attribute of the Trading Block where the error is located.
CompRef	If the error is associated with a specific Trading Component, then this is the value of the ID attribute of the Trading Component where the error is located.
ElementRef	If the error is associated with a specific element within a Trading Component then, if the element has an attribute with an "attribute type" (see [XML]) of "ID", then this is the value of that attribute.
AttName	If the error is associated with the value of an attribute, then this is the name of that attribute. In this case the PackagedContent of the Error Component should contain the value of the attribute.

Note that as many as the attributes as possible should be included. For example if an attribute in a child element of a Trading Component contains an incorrect value, then all the attributes of ErrorLocation should be present.

8. Trading Blocks

Trading Blocks are child elements of the top level IOTP Messages that are sent in the form of [XML] documents directly between the different Trading Roles that are taking part in a trade.

Each Trading Blocks consist of one or more Trading Components (see section 7). This is illustrated in the diagram below.

```
   *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*

             IOTP MESSAGE  <-----------IOTP Message - an XML Document
              |                        which is transported between the
              |                        Trading Roles
              |-Trans Ref Block <----- Trans Ref Block - contains
              |  |                     information which describes the
              |  |                     IOTP Transaction and the IOTP
              |  |                     Message.
              |  |-Trans Id Comp. <--- Transaction Id Component -
              |  |                     uniquely identifies the IOTP
              |  |                     Transaction. The Trans Id
              |  |                     Components are the same across
              |  |                     all IOTP messages that comprise a
              |  |                     single IOTP transaction.
              |  |-Msg Id Comp. <----- Message Id Component - identifies
              |                        and describes an IOTP Message
              |                        within an IOTP Transaction
              |-Signature Block <----- Signature Block (optional) -
              |  |                     contains one or more Signature
              |  |                     Components and their associated
              |  |                     Certificates
              |  |-Signature Comp. <-- Signature Component - contains
              |  |                     digital signatures. Signatures
              |  |                     may sign digests of the Trans Ref
              |  |                     Block and any Trading Component
              |  |                     in any IOTP Message in the same
              |  |                     IOTP Transaction.
              |  |-Certificate Comp. <-Certificate Component. Used to
              |                        check the signature. (Optional)
      ------> |-Trading Block <--------Trading Block - an XML Element
        |     |  |-Trading Comp.       within an IOTP Message that
     Trading  |  |-Trading Comp.       contains a predefined set of
     Blocks   |  |-Trading Comp.       Trading Components
        |     |  |-Trading Comp.
        |     |  |-Trading Comp. <-----Trading Components - XML Elements
        |     |                        within a Trading Block that
      ------> |-Trading Block          contain a predefined set of XML
              |  |-Trading Comp.       elements and attributes
              |  |-Trading Comp.       containing information required
              |  |-Trading Comp.       to support a Trading Exchange
              |  |-Trading Comp.
              |  |-Trading Comp.
              |
   *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 16 Trading Blocks

Trading Blocks are defined as part of the definition of an IOTP Message (see section 3.1.1). The definition of an IOTP Message element is repeated here:

```
<!ELEMENT IotpMessage
   ( TransRefBlk,
     SigBlk?,
     ErrorBlk?,
     ( AuthReqBlk |
       AuthRespBlk |
       AuthStatusBlk |
       CancelBlk |
       DeliveryReqBlk |
       DeliveryRespBlk |
       InquiryReqBlk |
       InquiryRespBlk |
       OfferRespBlk |
       PayExchBlk |
       PayReqBlk |
       PayRespBlk |
       PingReqBlk |
       PingRespBlk |
       TpoBlk |
       TpoSelectionBlk
     )*
   ) >
```

The remainder of this section defines the Trading Blocks in this version of IOTP. They are:

- o Authentication Request Block
- o Authentication Response Block
- o Authentication Status Block
- o Cancel Block
- o Delivery Request Block
- o Delivery Response Block
- o Error Block
- o Inquiry Request Block
- o Inquiry Response Block

- o Offer Response Block
- o Payment Exchange Block
- o Payment Request Block
- o Payment Response Block
- o Signature Block
- o Trading Protocol Options Block
- o TPO Selection Block

The Transaction Reference Block is described in section 3.3.

8.1 Trading Protocol Options Block

The TPO Trading Block contains options which apply to the IOTP Transaction. The definition of a TPO Trading Block is as follows.

```
<!ELEMENT TpoBlk ( ProtocolOptions, BrandList*, Org* ) >
<!ATTLIST TpoBlk
 ID                 ID      #REQUIRED >
```

Attributes:

ID	An identifier which uniquely identifies the Trading Protocol Options Block within the IOTP Transaction (see section 3.4 ID Attributes).

Content:

ProtocolOptions	The Protocol Options Component (see section 7.1)defines the options which apply to the whole IOTP Transaction (see section 9).
BrandList	This Brand List Component contains one or more payment brands and protocols which may be selected (see section 7.7).
Org	The Organisation Components (see section 7.6) identify the Organisations and their roles in the IOTP Transaction. The roles and Organisations which must be present will depend on the particular type of IOTP Transaction. See the definition of each transaction in section 9. Internet Open Trading Protocol Transactions.

The TPO Block should contain:

- the Protocol Options Component
- the Organisation Component with the Trading Role of Merchant
- the Organisation Component with the Trading Role of Consumer
- optionally, the Organisation Component with the Trading Role of DeliverTo, if there is a Delivery included in the IOTP Transaction
- Brand List Components for each payment in the IOTP Transaction
- Organisation Components for all the Payment Handlers involved
- optionally, Organisation Components for the Delivery Handler (if any) for the transaction
- additional Organisation Components that the Merchant may want to include. For example
  - a Customer Care Provider
  - an Certificate Authority that offers Merchant "Credentials" or some other warranty on the goods or services being offered.

8.2 TPO Selection Block

The TPO Selection Block contains the results of selections made from the options contained in the Trading Protocol Options Block (see section 8.1).The definition of a TPO Selection Block is as follows.

```
<!ELEMENT TpoSelectionBlk (BrandSelection+) >
<!ATTLIST TpoSelectionBlk
 ID                ID      #REQUIRED >
```

Attributes:

ID	An identifier which uniquely identifies the TPO Selection Block within the IOTP Transaction.

Content:

BrandSelection	This identifies the choice of payment brand and payment protocol to be used in a payment within the IOTP Transaction. There is one Brand Selection Component (see section 7.8) for each payment to be made in the IOTP Transaction.

The TPO Selection Block should contain one Brand Selection Component for each Brand List in the TPO Block.

8.3 Offer Response Block

The Offer Response Block contains details of the goods, services, amount, delivery instructions or financial transaction which is to take place. Its definition is as follows.

```
<!ELEMENT OfferRespBlk (Status, Order?, Payment*,
             Delivery?, TradingRoleData*) >
<!ATTLIST OfferRespBlk
 ID                 ID      #REQUIRED >
```

Attributes:

ID	An identifier which uniquely identifies the Offer Response Block within the IOTP Transaction.

Content:

Status	Contains status information about the business success (see section 4.2) or failure of the generation of the Offer. Note that in an Offer Response Block, a ProcessState of NotYetStarted or InProgress are illegal values.
Order	The Order Component contains details about the goods, services or financial transaction which is taking place see section 7.5. The Order Component must be present unless the ProcessState attribute of the Status Component is set to Failed.
Payment	The Payment Components contain information about the payments which are to be made see section 7.9.
Delivery	The Delivery Component contains details of the delivery to be made (see section 7.13).
TradingRoleData	The Trading Role Data Component contains opaque data which is needs to be communicated between the Trading Roles involved in an IOTP Transaction (see section 7.17).

The Offer Response Block should contain:

- o the Order Component for the IOTP Transaction
- o Payment Components for each Payment in the IOTP Transaction
- o the Delivery Component the IOTP Transaction requires (if any).

8.4 Authentication Request Block

The Authentication Request Block contains the data which is used by one Trading Role to obtain information about and optionally authenticate another Trading Role.

In outline it contains:

- o information about how the authentication itself will be carried out, and/or
- o a request for additional information about the Organisation being authenticated.

Its definition is as follows.

```
<!ELEMENT AuthReqBlk (AuthReq*, TradingRoleInfoReq?) >
<!ATTLIST AuthReqBlk
 ID                 ID      #REQUIRED >
```

Attributes:

ID	An identifier which uniquely identifies the Authentication Request Block within the IOTP Transaction.

Content:

AuthReq	Each Authentication Request (see section 7.2) component describes an alternative way in which the recipient of the Authentication Request may authenticate themselves by generating an Authentication Response Component (see section 7.3). If one Authentication Request Component is present then that Authentication Request Component should be used.

	If more than one Authentication Request Component is present then the recipient should choose one of the components based on personal preference of the recipient or their software. If no Authentication Request Component is present it means that the Authentication Request Block is requesting the return of Organisation Components as specified in the Trading Role Information Request Component.
TradingRoleInfoReq	The Trading Role Information Request Component (see section 7.4) contains a list of Trading Roles about which information is being requested

There must be at least one Component (either an Authentication Request or a Trading Role Information Request) within the Authentication Block otherwise it is an error.

8.5 Authentication Response Block

The Authentication Response Block contains the response which results from processing the Authentication Request Block. Its definition is as follows.

```
<!ELEMENT AuthRespBlk (AuthResp?, Org*) >
<!ATTLIST AuthRespBlk
 ID                 ID      #REQUIRED >
```

Attributes:

ID	An identifier which uniquely identifies the Authentication Response Block within the IOTP Transaction.

Content:

AuthResp	The optional Authentication Response Component which contains the results of processing the Authentication Request Component - see section 7.3.
Org	Optional Organisation Components that contain information corresponding to the Trading Roles as requested by the TradingRoleList attribute of the Trading Role Information Request component.

The components present in the Authentication Response Block must match the requirement of the corresponding Authentication Request Block otherwise it is an error.

8.6 Authentication Status Block

The Authentication Status Block indicates the success or failure of the validation of an Authentication Response Block by an Authenticator. Its definition is as follows.

```
<!ELEMENT AuthStatusBlk (Status) >
<!ATTLIST AuthStatusBlk
 ID                 ID      #REQUIRED >
```

Attributes:

ID	An identifier which uniquely identifies the Authentication Status Block within the IOTP Transaction.

Content:

Status	Contains status information about the business success (see section 4.2) or failure of the authentication

8.7 Payment Request Block

The Payment Request Block contains information which requests that a payment is started. Its definition is as follows.

```
<!ELEMENT PayReqBlk (Status+, BrandList, BrandSelection,
     Payment, PaySchemeData?, Org*, TradingRoleData*) >
<!ATTLIST PayReqBlk
 ID                 ID      #REQUIRED >
```

Attributes:

ID	An identifier which uniquely identifies the Payment Request Block within the IOTP Transaction.

Content:

Status	Contains the Status Components (see section 7.13) of the responses of the steps (e.g., an Offer Response and/or a Payment Response) on which this

	step depends. It is used to indicate the success or failure of those steps. Payment should only occur if the previous steps were successful.
BrandList	The Brand List Component contains a list of one or more payment brands and protocols which may be selected (see section 7.7).
BrandSelection	This identifies the choice of payment brand, the payment protocol and the Payment Handler to be used in a payment within the IOTP Transaction. There is one Brand Selection Component (see section 7.8) for each payment to be made in the IOTP Transaction.
Payment	The Payment Components contain information about the payment which is being made see section 7.9.
PaySchemeData	The Payment Scheme Component contains payment scheme specific data see section 7.10.
Org	The Organisation Component contains details of Organisations involved in the payment (see section 7.6). The Organisations present are dependent on the IOTP Transaction and the data which is to be signed. See section 6 Digital Signatures for more details.
TradingRoleData	The Trading Role Data Component contains opaque data which is needs to be communicated between the Trading Roles involved in an IOTP Transaction (see section 7.17).

The Payment Request Block should contain:

o the Organisation Component with a Trading Role of Merchant

o the Organisation Component with the Trading Role of Consumer

o the Payment Component for the Payment

o the Brand List Component for the Payment

o the Brand Selection Component for the Brand List

o the Organisation Component for the Payment Handler of the Payment

- the Organisation Component (if any) for the Organisation which carried out the previous step, for example another Payment Handler

- the Organisation Component for the Organisation which is to carry out the next step, if any. This may be, for example, either a Delivery Handler or a Payment Handler.

- the Organisation Components for any additional Organisations that the Merchant has included in the Offer Response Block

- an Optional Payment Scheme Data Component, if required by the Payment Method as defined in the IOTP supplement for the payment method

- any Trading Role Data Components that may be required (see section 7.17.1).

8.8 Payment Exchange Block

The Payment Exchange Block contains payment scheme specific data which is exchanged between two of the roles in a trade. Its definition is as follows.

```
<!ELEMENT PayExchBlk (PaySchemeData+) >
<!ATTLIST PayExchBlk
 ID                 ID      #REQUIRED >
```

Attributes:

ID	An identifier which uniquely identifies the Payment Exchange Block within the IOTP Transaction.

Content:

PaySchemeData	This Trading Component contains payment scheme specific data see section 7.10 Payment Scheme Component.

8.9 Payment Response Block

This Payment Response Block contains a information about the Payment Status, an optional Payment Receipt, and an optional payment protocol message. Its definition is as follows.

```
<!ELEMENT PayRespBlk (Status, PayReceipt?, PaySchemeData?,
     PaymentNote?, TradingRoleData*) >
<!ATTLIST PayRespBlk
 ID                 ID      #REQUIRED >
```

Attributes:

ID	An identifier which uniquely identifies the Payment Response Block within the IOTP Transaction.

Content:

Status	Contains status information about the business success (see section 4.2) or failure of the payment. Note that in a Pay Response Block, a ProcessState of NotYetStarted or InProgress are illegal values.
PayReceipt	Contains payment scheme specific data which can be used to verify the payment occurred. See section 7.11 Payment Receipt Component. It must be present if the ProcessState attribute of the Status Component is set to CompletedOk. PayReceipt is optional for other values as specified by the appropriate Payment Scheme supplement.
PaySchemeData	Contains payment scheme specific data see section, for example a payment protocol message. See 7.10 Payment Scheme Component.
PaymentNote	Contains additional, non payment related, information which the Payment Handler wants to provide to the Consumer. For example, if a withdrawal or deposit were being made then it could contain information on the remaining balance on the account after the transfer was complete. See section 7.12 Payment Note Component.
TradingRoleData	The Trading Role Data Component contains opaque data which is needs to be communicated between the Trading Roles involved in an IOTP Transaction (see section 7.17).

8.10 Delivery Request Block

The Delivery Request Block contains details of the goods or services which are to be delivered together with a signature which can be used to check that delivery is authorised. Its definition is as follows.

```
<!ELEMENT DeliveryReqBlk (Status+, Order, Org*, Delivery,
     ConsumerDeliveryData?, TradingRoleData*) >
<!ATTLIST DeliveryReqBlk
 ID                 ID      #REQUIRED >
```

Attributes:

ID	An identifier which uniquely identifies the Delivery Request Block within the IOTP Transaction.

Content:

Status	Contains the Status Components (see section 7.13) of the responses of the steps (e.g., a Payment Response) on which this step is dependent. It is used to indicate the success or failure of those steps. Delivery should only occur if the previous steps were successful.
Order	The Order Component contains details about the goods, services or financial transaction which is taking place see section 7.5.
Org	The Organisation Components (see section 7.6) identify the Organisations and their roles in the IOTP Transaction. The roles and Organisations which must be present will depend on the particular type of IOTP Transaction. See the definition of each transaction in section 9. Internet Open Trading Protocol Transactions.
Delivery	The Delivery Component contains details of the delivery to be made (see section 7.13).
ConsumerDeliveryData	Optional. Contains an identifier specified by the Consumer which, if returned by the Delivery Handler will enable the Consumer to identify which Delivery is being referred to.

TradingRoleData	The Trading Role Data Component contains opaque data which is needs to be communicated between the Trading Roles involved in an IOTP Transaction (see section 7.17).

The Delivery Request Block contains:

- o the Organisation Component with a Trading Role of Merchant
- o the Organisation Component for the Consumer and DeliverTo Trading Roles
- o the Delivery Component for the Delivery
- o the Organisation Component for the Delivery Handler. Specifically the Organisation Component identified by the ActionOrgRef attribute on the Delivery Component
- o the Organisation Component (if any) for the Organisation which carried out the previous step, for example a Payment Handler
- o the Organisation Components for any additional Organisations that the Merchant has included in the Offer Response Block
- o any Trading Role Data Components that may be required (see section 7.17.1).

8.11 Delivery Response Block

The Delivery Response Block contains a Delivery Note containing details on how the goods will be delivered. Its definition is as follows. Note that in a Delivery Response Block a Delivery Status Element with a DeliveryStatusCode of NotYetStarted or InProgress is invalid.

```
<!ELEMENT DeliveryRespBlk (Status, DeliveryNote) >
<!ATTLIST DeliveryRespBlk
 ID                 ID      #REQUIRED >
```

Attributes:

ID	An identifier which uniquely identifies the Delivery Response Block within the IOTP Transaction.

Content:

Status	Contains status information about the business success (see section 4.2) or failure of the delivery. Note that in a Delivery Response Block, a ProcessState of NotYetStarted or InProgress are illegal values.
DeliveryNote	The Delivery Note Component contains details about how the goods or services will be delivered (see section 7.15).

8.12 Inquiry Request Trading Block

The Inquiry Request Trading Block contains an Inquiry Type Component and an optional Payment Scheme Component to contain payment scheme specific inquiry messages.

```
<!ELEMENT InquiryReqBlk ( InquiryType, PaySchemeData? ) >
<!ATTLIST InquiryReqBlk
 ID                 ID      #REQUIRED >
```

Attributes:

ID	An identifier which uniquely identifies the Inquiry Request Trading Block within the IOTP Transaction.

Content:

InquiryType	Inquiry Type Component (see section 7.18) that contains the type of inquiry.
PaySchemeData	Payment Scheme Component (see section 7.10) that contains payment scheme specific inquiry messages for inquiries on payments. This is present when the Type attribute of Inquiry Type Component is Payment.

8.13 Inquiry Response Trading Block

The Inquiry Response Trading Block contains a Status Component and an optional Payment Scheme Component to contain payment scheme specific inquiry messages. Its purpose is to enquire on the current status of an IOTP transaction at a server.

```
   <!ELEMENT InquiryRespBlk (Status, PaySchemeData?) >
   <!ATTLIST InquiryRespBlk
    ID                 ID      #REQUIRED
    LastReceivedIotpMsgRef NMTOKEN #IMPLIED
    LastSentIotpMsgRef  NMTOKEN #IMPLIED >
```

Attributes:

ID	An identifier which uniquely identifies the Inquiry Response Trading Block within the IOTP Transaction.
LastReceivedIotpMsgRef	Contains an Element Reference (see section 3.5) to the Message Id Component (see section 3.3.2) of the last message this server has received from the Consumer. If there is no previously received message from the Consumer in the pertinent transaction, this attribute should be contain the value Null. This attribute exists for debugging purposes.
LastSentIotpMsgRef	Contains an Element Reference (see section 3.5) to the Message Id Component (see section 3.3.2) of the last message this server has sent to the Consumer. If there is no previously sent message to the Consumer in the pertinent transaction, this attribute should contain the value Null. This attribute exists for debugging purposes.

Content:

Status	Contains status information about the business success (see section 4.2) or failure of a certain trading exchange (i.e., Offer, Payment, or Delivery).
PaySchemeData	Payment Scheme Component (see section 7.10) that contains payment scheme specific inquiry messages for inquiries on payments. This is present when the Type attribute of StatusType attribute of the Status Component is set to Payment.

8.14 Ping Request Block

The Ping Request Block is used to determine if a Server is operating and whether or not cryptography is compatible.

The definition of a Ping Request Block is as follows.

```
<!ELEMENT PingReqBlk (Org*)>
<!ATTLIST PingReqBlk
 ID                ID      #REQUIRED>
```

Attributes:

ID	An identifier which uniquely identifies the Ping Request Trading Block within the IOTP Transaction.

Content:

Org	Optional Organisation Components (see section 7.6). If no Organisation Component is present then the Ping Request is anonymous and simply determines if the server is operating. However if Organisation Components are present, then it indicates that the sender of the Ping Request wants to verify that digital signatures can be handled. In this case the sender includes: o an Organisation Component that identifies itself specifying the Trading Role(s) it is taking in IOTP transactions (Merchant, Payment Handler, etc.) o an Organisation Component that identifies the intended recipient of the message. These are then used to generate a signature over the Ping Response Block.

8.15 Ping Response Block

The Ping Response Trading Block provides the result of a Ping Request.

It contains an Organisation Component that identifies the sender of the Ping Response.

If the Ping Request to which this block is a response contained Organisation Components, then it also contains those Organisation Components.

```
<!ELEMENT PingRespBlk (Org+)>
<!ATTLIST PingRespBlk
 ID                 ID      #REQUIRED
 PingStatusCode (Ok | Busy | Down) #REQUIRED
 SigVerifyStatusCode (Ok | NotSupported | Fail) #IMPLIED
 xml:lang           NMTOKEN #IMPLIED
 PingStatusDesc     CDATA   #IMPLIED>
```

Attributes:

ID	An identifier which uniquely identifies the Ping Request Trading Block within the IOTP Transaction.
PingStatusCode	Contains a code which shows the status of the sender software which processes IOTP messages. Valid values are: o Ok. Everything with the service is working normally, including the signature verification. o Busy. Things are working normally but there may be some delays. o Down. The server is not functioning fully but can still provide a Ping response.
SigVerifyStatusCode	Contains a code which shows the status of signature verification. This is present only when the message containing the Ping Request Block also contains a Signature Block. Valid values are: o Ok. The signature has successfully been verified and proved compatible. o NotSupported The receiver of this Ping Request Block does not support validation of signatures. o Fail. Signature verification failed.
Xml:lang	Defines the language used in PingStatusDesc. This is present when PingStatusDesc is present.
PingStatusDesc	Contains a short description of the status of the server which sends this Ping Response Block. Servers, if their designers want, can use this

attribute to send more refined status information than PingStatusCode which can be used for debugging purposes, for example.

Content:

Org	These are Organisation Components (see section 7.6). The Organisation Components of the sender of the Ping Response is always included in addition to the Organisation Components sent in the Ping Request.

Note: Ping Status Code values do not include a value such as Fail, since, when the software receiving the Ping Request message is not working at all, no Ping Response message will be sent back.

8.16 Signature Block

The Signature Block contains one or more Signature Components and associated Certificates (if required) which sign data associated with the IOTP Transaction. For a general discussion and introduction to how IOTP uses signatures, see section 6 Digital Signatures. The definition of the Signature Component and certificates is contained in the paper "Digital Signatures for the Internet Open Trading Protocol", see [IOTPDSIG]. Descriptions of how these are used by IOTP is contained in sections 7.19 and 7.20.

The definition of a Signature Block is as follows:

```
<!ELEMENT IotpSignatures (Signature+, Certificate*) >
<!ATTLIST IotpSignatures
  ID                ID      #IMPLIED >
```

Attributes:

ID	An identifier which uniquely identifies the Signature Block within the IOTP Transaction.

Content:

Signature	A Signature Component. See section 7.19.
Certificate	A Certificate Component. See section 7.20.

The contents of a Signature Block depends on the Trading Block that is contained in the same IOTP Message as the Signature Block.

8.16.1 Signature Block with Offer Response

A Signature Block which is in the same message as an Offer Response Block contains just an Offer Response Signature Component (see section 7.19.2).

8.16.2 Signature Block with Payment Request

A Signature Block which is in the same message as a Payment Request Block contains:

- an Offer Response Signature Component (see section 7.19.2), and
- if the Payment is dependent on an earlier step (as indicated by the StartAfter attribute on the Payment Component), then the Payment Receipt Signature Component (see section 7.19.3) generated by the previous step

8.16.3 Signature Block with Payment Response

A Signature Block which is in the same message as a Payment Response Block contains just a Payment Receipt Signature Component (see section 7.19.3) generated by the step.

8.16.4 Signature Block with Delivery Request

A Signature Block which is in the same message as a Delivery Request Block contains:

- an Offer Response Signature Component (see section 7.19.2), and
- the Payment Receipt Signature Component (see section 7.19.3) generated by the previous step.

8.16.5 Signature Block with Delivery Response

A Signature Block which is in the same message as a Delivery Response Block contains just a Delivery Response Signature component (see section 7.19.4) generated by the step.

8.17 Error Block

The Error Trading Block contains one or more Error Components (see section 7.21) which contain information about Technical Errors (see section 4.1) in an IOTP Message which has been received by one of the Trading Roles involved in the trade.

For clarity two phrases are defined which are used in the description of an Error Trading Block:

- message in error. An IOTP message which contains or causes an error of some kind
- message reporting the error. An IOTP message that contains an Error Trading Block that describes the error found in a message in error.

An Error Trading Block may be contained in any message reporting the error. The action which then follows depends on the severity of the error. See the definition of an Error Component, for an explanation of the different types of severity and the actions which can then occur.

in3 Note: Although, an Error Trading Block can report multiple different errors using multiple Error Components, there is no obligation on a developer of an IOTP Aware Application to do so.

The structure of an Error Trading Block is as follows.

```
<!ELEMENT ErrorBlk (ErrorComp+, PaySchemeData*) >
<!ATTLIST ErrorBlk
 ID                 ID      #REQUIRED >
```

Attributes:

ID	An identifier which uniquely identifies the Error Trading Block within the IOTP Transaction.

Content:

ErrorComp	An Error Components (see section 7.21) that contains information about an individual Technical Error.
PaySchemeData	An optional Payment Scheme Component (see section 7.10) which contains a Payment Scheme Message. See the appropriate payment scheme supplement to

determine whether or not this component needs to be present and for the definition of what it must contain.

8.18 Cancel Block

The Cancel Block is used by one Trading Role to inform any other that a transaction has been cancelled. Example usage includes:

- a Consumer Role informing a non-Consumer role that it no longer plans to continue with the transaction. This will allow the server to close down the transaction tidily without a waiting for a time-out to occur
- a non-Consumer Role to inform a Consumer role that the Transaction is being stopped. In this case, the Consumer is then unlikely to re-send the previous message that was sent in the mistaken understanding that the original was not received.

Its definition is as follows.

```
<!ELEMENT CancelBlk (Status) >
<!ATTLIST CancelBlk
 ID                 ID      #REQUIRED >
```

Attributes:

ID	An identifier which uniquely identifies the Cancel Block within the IOTP Transaction.

Content:

Status	Contains status information indicating that the IOTP transaction has been cancelled.

9. Internet Open Trading Protocol Transactions

The Baseline Internet Open Trading Protocol supports three types of transactions for different purposes. These are

- an Authentication IOTP transaction which supports authentication of one party in a trade by another and/or requests information about another Trading Role

- o IOTP Transactions that involve one or more payments. Specifically:

  - Deposit
  - Purchase
  - Refund
  - Withdrawal, and
  - Value Exchange

- o IOTP Transactions designed to check the correct function of the IOTP infrastructure. Specifically:

  - Transaction Status Inquiry, and
  - Ping

Although the Authentication IOTP Transaction can operate on its own, authentication can optionally precede any of the "payment" transactions. Therefore, the rest of this section is divided into two parts covering:

- o Authentication and Payment transactions (Authentication, Deposit, Purchase, Refund, Withdrawal and Value Exchange)

- o Infrastructure Transactions (Transaction Status Inquiry and Ping) that are designed to support inquiries on whether or not a transaction has succeeded or a Trading Role’s servers are operating correctly, and

9.1 Authentication and Payment Related IOTP Transactions

The Authentication and Payment related IOTP Transactions consist of six Document Exchanges which are then combined in sequence to implement a specific transaction.

Generally, there is a close, but not exact, correspondence between a Document Exchange and a Trading Exchange. The main difference is that some Document Exchanges implement part or all of two Trading Exchanges simultaneously in order to minimise the number of actual IOTP Messages which must be sent over the Internet.

The six Document Exchanges are:

- o Authentication. This is a direct implementation of the Authentication Trading Exchange

- o Brand Dependent Offer. This is the Offer Trading Exchange combined with the Brand Selection part of the Payment Trading Exchange. Its purpose is to provide the Merchant with information on the Brand selected so that the content of the Offer Response may be adapted accordingly

- o Brand Independent Offer. This is also an Offer Trading Exchange. However, in this instance, the content of the Offer Response does not depend on the Brand selected.

- o Payment. This is a direct implementation of the Payment part of a Payment Trading Exchange

- o Delivery. This is a direct implementation of the Delivery Exchange

- o Delivery with Payment. This is an implementation of combined Payment and Delivery Trading Exchanges

These Document Exchanges are combined together in different sequences to implement each IOTP Transaction. The way in which they may be combined is illustrated by the diagram below.

```
   *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*

     START ------------------------------------------------------
      |                                                         v
      |                                                ----------------
      |                                               | AUTHENTICATION |
      |                                                ----------------
       --------------------------------------               |    |
                       |                     |              |    |
                       |      -------------- | -------------     |
                       v      v              v      v            |
                  -------------------     -----------------      |
                 | BRAND INDEPENDENT |   | BRAND DEPENDENT |     |
                 |       OFFER       |   |      OFFER      |     |
                  -------------------     -----------------      |
                        |    |                   |   |           |
                        |     ---------------    |   |           |
                        |                    |   |   |           |
                        |     -------------- | --    |           |
                        v    v               v       v           |
                      ---------           --------------         |
                     | PAYMENT |         | PAYMENT WITH |        |
                     | (first) |         |   DELIVERY   |        |
                      ---------           --------------         |
                          |                      |               |
              -----------------------------      |               |
              v                v           |     |               |
         ----------        ---------       |     |               |
        | DELIVERY |      | PAYMENT |      |     |               |
        |          |      | {second)|      |     |               |
         ----------        ---------       |     |               |
              |                |           |     |               v
               ----------------------------------------------> STOP

   *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 17 Payment and Authentication Message Flow Combinations

The combinations of Document Exchanges that are valid depend on the particular IOTP transaction.

The remainder of this sub-section describes:

- o each Document Exchange in more detail including descriptions of the content of each Trading Block in the Document Exchanges, and
- o descriptions of how each IOTP Transaction uses the Document Exchanges to effect the desired result.

Note: The descriptions of the Document Exchanges which follow describe the ways in which various Business Errors (see section 4.2) are handled. No reference is made however to the handling of Technical Errors (see section 4.1) in any of the messages since these are handled the same way irrespective of the context in which the message is being sent. See section 4 for more details.

9.1.1 Authentication Document Exchange

The Authentication Document Exchange is a direct implementation of the Authentication Trading Exchange (see section 2.2.4). It involves:

- o an Authenticator - the Organisation which is requesting the authentication, and
- o an Authenticatee - the Organisation being authenticated.

The authentication consists of:

- o an Authentication Request being sent by the Authenticator to the Authenticatee,
- o an Authentication Response being sent in return by the Authenticatee to the Authenticator which is then checked, and
- o an Authentication Status being sent by the Authenticator to the Authenticatee to provide an indication of the success or failure of the authentication.

An Authentication Document Exchange also:

- o provides an Authenticatee with an Organisation Component which describes the Authenticator, and
- o optionally provides the Authenticator with Organisation Components which describe the Authenticatee.

The Authentication Request may also be digitally signed which allows the Authenticatee to verify the credentials of the Authenticator.

The IOTP Messages which are involved are illustrated by the diagram below.

```
 *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*
 Organisation 1
 (Authenticatee)
     |   Organisation 2
     |  (Authenticator)
STEP |     |
 1.          First Organisation takes an action (for example by
             pressing a button on an HTML page) which requires that
             the Organisation is authenticated

     1 --> 2 Authentication Need (outside scope of IOTP)

 2.          The second Organisation generates: an Authentication
             Request Block containing one or more Authentication
             Request Components and/or a Trading Role Information
             Request Component, then sends it to the first
             Organisation

     1 <-- 2 TPO & AUTHENTICATION REQUEST. IotpMsg: Trans Ref Block;
             Signature Block (optional); TPO Block; Auth Request Block

 3.          IOTP aware application started. If a Signature Block is
             present, the first Organisation may use this to check the
             credentials of the second Organisation. If credentials are
             OK, the first Organisation selects an Authentication
             Request to use (if present and more than one), then uses
             the authentication algorithm selected to generate an
             Authentication Response Block. If present, the Trading
             Role Information Request Component is used to generate
             Organisation Components. Finally a Signature Component is
             created if required and all components are then sent back
             to the second Organisation for validation.

     1 --> 2 AUTHENTICATION RESPONSE. IotpMsg; Trans Ref Block;
             Signature Block (optional) ; Auth Response Block

 4.          The second Organisation checks the Authentication
             Response against the data in the Authentication Request
             Block to check that the first Organisation is who they
             appear to be, and sends an Authentication Status Block to
             the first Organisation to indicate the result then
             stops.

     1 <-- 2 AUTHENTICATION STATUS. IotpMsg: Trans Ref Block;
             Signature Block (optional); Auth Response Block
```

5. The first Organisation checks the authentication Status Block and optionally keeps information on the IOTP transaction for record keeping purposes and stops.

```
*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 18 Authentication Document Exchange

9.1.1.1 Message Processing Guidelines

On receiving a TPO & Authentication Request IOTP Message (see below), an Authenticatee may either:

- o generate and send an Authentication Response IOTP Message back to the Authenticator, or
- o indicate failure to comply with the Authentication Request by sending a Cancel Block back to the Authenticator containing a Status Component with a StatusType of Authentication a ProcessState of Failed and the CompletionCode (see section 7.16.4) set to either: AutEeCancel, NoAuthReq, TradRolesIncon or Unspecified.

On receiving an Authentication Response IOTP Message (see below), an Authenticator should send in return, an Authentication Status IOTP Message (see below) containing a Status Block with a Status Component where the StatusType is set to Authentication, and:

- o the ProcessState attribute of the Status Component is set to CompletedOk which indicates a successful completion, or
- o the ProcessState attribute is set to Failed and the CompletionCode attribute is set to either: AutOrCancel, AuthFailed or Unspecified which indicates a failed authentication,

On receiving an Authentication Status IOTP Message (see below), the Authenticatee should check the Status Component in the Status Block. If this indicates:

- o a successful authentication, then the Authenticatee should either:
  - - continue with the next step in the IOTP Transaction of which the Authentication Document Exchange is part (if any), or

- indicate a failure to continue with the rest of the IOTP Transaction, by sending back to the Authenticator a Cancel Block containing a Status Component with a StatusType of Authentication, a ProcessState of Failed and the CompletionCode (see section 7.16.4) set to AutEeCancel.

o a failed authentication, then the failure should be reported to the Authenticatee and any further processing stopped.

If the Authenticator receives an IOTP Message containing a Cancel block from a Consumer, then the Authenticatee may go to the CancelNetLocn specified on the Trading Role Element in the Organisation Component for the Authenticator contained in the Trading Protocol Options Block.

9.1.1.2 TPO & Authentication Request IOTP Message

Apart from a Transaction Reference Block (see section 3.3), this message consists of:

o a Trading Protocol Options Block (see section 8.1)

o an Authentication Request Block (see section 8.4), and

o an optional Signature Block (see section 8.16).

Each of these are described below.

TRADING PROTOCOL OPTIONS BLOCK

The Trading Protocol Options Block (see section 8.1) must contain the following Trading Components:

o one Protocol Options Component (see Section 7.1) which defines the options which apply to the whole Authentication Document Exchange.

o one Organisation Component (see section 7.6) which describes the Authenticator. The Trading Role on the Organisation Component should indicate the role which the Authenticator is taking in the Trade, for example a Merchant or a Consumer.

AUTHENTICATION REQUEST BLOCK

The Authentication Request Block (see section 8.4) must contain the following Trading Components:

o one Authentication Request Component (see section 7.2), and

SIGNATURE BLOCK (AUTHENTICATION REQUEST)

If the Authentication Request is being digitally signed then a Signature Block must be included. It contains Digests of the following XML elements:

- o the Transaction Reference Block (see section 3.3) for the IOTP Message that contains information that describes the IOTP Message and IOTP Transaction
- o the Transaction Id Component (see section 3.3.1) which globally uniquely identifies the IOTP Transaction
- o the following components of the TPO Block :
  - the Protocol Options Component
  - the Organisation Component
- o the following components of the Authentication Request Block:
  - the Authentication Request Component
  - the Trading Role Information Request Component

9.1.1.3 Authentication Response IOTP Message

Apart from a Transaction Reference Block (see section 3.3), this message consists of:

- o an Authentication Response Block (see section 8.5), and
- o an optional Signature Block (see section 8.16).

Each of these are described below.

AUTHENTICATION RESPONSE BLOCK

The Authentication Response Block must contain the following Trading Component:

- o one Authentication Response Component (see section 7.3)
- o one Organisation Component for every Trading Role identified in the TradingRoleList attribute of the Trading Role Information Request Component contained in the Authentication Request Block.

SIGNATURE BLOCK (AUTHENTICATION RESPONSE)

If the Algorithm element (see section 12. IANA Considerations) within the Authentication Request Component contained in the Authentication Request Block indicates that the Authentication Response should consist of a digital signature then a Signature Block must be included in the same IOTP message that contains an Authentication Response Block. The Signature Component contains Digest Elements for the following XML elements:

- o the Transaction Reference Block (see section 3.3) for the IOTP Message that contains information that describes the IOTP Message and IOTP Transaction
- o the Transaction Id Component (see section 3.3.1) which globally uniquely identifies the IOTP Transaction
- o the following components of the Authentication Request Block:
  - - the Authentication Request Component
  - - the Trading Role Information Request Component
- o the Organisation Components contained in the Authentication Response Block

Note: It should not be assumed that all trading roles can support the signing of data. Particularly it should not be assumed that Consumers support the signing of data.

9.1.1.4 Authentication Status IOTP Message

Apart from a Transaction Reference Block (see section 3.3), this message consists of:

- o an Authentication Status Block (see section 8.5), and
- o an optional Signature Block (see section 8.16).

Each of these are described below.

AUTHENTICATION STATUS BLOCK

The Authentication Status Block (see section 8.6) must contain the following Trading Components:

- o one Status Component (see section 7.16) with a ProcessState attribute set to CompletedOk.

SIGNATURE BLOCK (AUTHENTICATION STATUS)

If the Authentication Status Block is being digitally signed then a Signature Block must be included that contains a Signature Component with Digest elements for the following XML elements:

- o the Transaction Reference Block (see section 3.3) for the IOTP Message that contains information that describes the IOTP Message and IOTP Transaction
- o the Transaction Id Component (see section 3.3.1) which globally uniquely identifies the IOTP Transaction
- o the following components of the Authentication Status Block:
  - - the Status Component (see section 7.16).

Note: If the Authentication Document Exchange is followed by an Offer Document Exchange (see section 9.1.2) then the Authentication Status Block and the Signature Block (Authentication Status) may be combined with either:

o a TPO IOTP Message (see section 9.1.2.3), or

o a TPO and Offer Response IOTP Message (see section 9.1.2.6)

9.1.2 Offer Document Exchange

The Offer Document Exchange occurs in two basic forms:

- o Brand Dependent Offer Exchange. Where the content of the offer, e.g., the order details, amount, delivery details, etc., are dependent on the payment brand and protocol selected by the consumer, and
- o Brand Independent Offer Exchange. Where the content of the offer is not dependent on the payment brand and protocol selected.

Each of these types of Offer Document Exchange may be preceded by an Authentication Document Exchange (see section 9.1.1).

9.1.2.1 Brand Dependent Offer Document Exchange

In a Brand Dependent Offer Document Exchange the TPO Block and the Offer Response Block are sent separately by the Merchant to the Consumer, i.e.:

- the Brand List Component is sent to the Consumer in a TPO Block,
- the Consumer selects a Payment Brand, Payment Protocol and optionally a Currency and amount from the Brand List Component
- the Consumer sends the selected brand, protocol and currency/amount back to the Merchant in a TPO Selection Block, and
- the Merchant uses the information received to define the content of and then send the Offer Response Block to the Consumer.

This is illustrated by the diagram below.

```
 *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*
   Consumer
     |  Merchant
STEP |     |
 1.         Consumer decides to trade and sends to the Merchant
            information (e.g., using HTML) that enables the Merchant
            to create an offer,

    C --> M Offer information - outside scope of IOTP

 2.         Merchant decides which payment brand protocols,
            currencies and amounts apply, places then in a Brand List
            Component inside a TPO Block and sends to Consumer

    C <-- M TPO. IotpMsg: Trans Ref Block; TPO Block

 3.         IOTP aware application started. Consumer selects the
            payment brand, payment protocol and currency/amount to
            use. Records selection in a Brand Selection Component and
            sends back to Merchant.

    C --> M TPO SELECTION. IotpMsg: Trans Ref Block; TPO Selection
            Block

 4.         Merchant uses selected payment brand, payment protocol,
            currency/amount and the offer information to create an
            Offer Response Block containing details about the IOTP
            Transaction including price, etc. Optionally signs it and
            sends to the Consumer

    C <-- M OFFER RESPONSE. IotpMsg: Trans Ref Block; Signature Block
            (optional); Offer Response Block

 5.         Consumer checks the Offer is OK, then combines components
            from the TPO Block, the TPO Selection Block and the Offer
            Response Block to create the next IOTP Message for the
            Transaction and sends it together with the Signature
            block if present to the required Trading Role

    CONTINUED ...

 *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 19 Brand Dependent Offer Document Exchange

Note, a Consumer identifies a Brand Dependent Offer Document Exchange, by the absence of an Offer Response Block in the first IOTP Message.

MESSAGE PROCESSING GUIDELINES

On receiving a TPO IOTP Message (see below), the Consumer may either:

- o generate and send a TPO Selection IOTP Message back to the Merchant, or
- o indicate failure to continue with the IOTP Transaction by sending a Cancel Block back to the Merchant containing a Status Component with a StatusType of Offer, a ProcessState of Failed and the CompletionCode (see section 7.16.4) set to either: ConsCancelled or Unspecified.

On receiving a TPO Selection IOTP Message (see below) the Merchant may either:

- o generate and send an Offer Response IOTP Message back to the Consumer, or
- o indicate failure to continue with the IOTP Transaction by sending a Cancel Block back to the Consumer containing a Status Component with a StatusType of Offer, a ProcessState of Failed and the CompletionCode (see section 7.16.4) set to either: MerchCancelled or Unspecified.

On receiving an Offer Response IOTP Message (see below) the Consumer may either:

- o generate and send the next IOTP Message in the IOTP transaction and send it to the required Trading Role. This is dependent on the IOTP Transaction, or
- o indicate failure to continue with the IOTP Transaction by sending a Cancel Block back to the Merchant containing a Status Component with a StatusType of Offer, a ProcessState of Failed and the CompletionCode (see section 7.16.4) set to either: ConsCancelled or Unspecified.

If the Merchant receives an IOTP Message containing a Cancel block, then the Consumer is likely to go to the CancelNetLocn specified on the Trading Role Element in the Organisation Component for the Merchant.

If the Consumer receives an IOTP Message containing a Cancel block, then the information contained in the IOTP Message should be reported to the Consumer but no further action taken.

9.1.2.2 Brand Independent Offer Document Exchange

In a Brand Independent Offer Document Exchange the TPO Block and the Offer Response Block are sent together by the Merchant to the Consumer, i.e. there is one IOTP Message that contains both a TPO Block, and an Offer Response Block.

The message flow is illustrated by the diagram below:

```
 *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*
 Consumer
     |  Merchant
STEP |     |
 1.          Consumer decides to trade and sends to the Merchant
             information (e.g., using HTML) that enables the Merchant
             to create an offer,

    C --> M Offer information - outside scope of IOTP

 2.          Merchant decides which payment brand protocols,
             currencies and amounts apply, places then in a Brand List
             Component inside a TPO Block, creates an Offer Response
             containing details about the IOTP Transaction including
             price, etc., optionally signs it  and sends to Consumer

    C <-- M TPO & OFFER RESPONSE. IotpMsg: Trans Ref Block; Signature
             Block; TPO Block; Offer Response Block

 3.          IOTP aware application started. Consumer selects the
             payment brand, payment protocol and currency/amount to
             use. Records selection in a Brand Selection Component,
             checks offer is OK, combines the Brand Selection
             Component with information from the TPO Block and Offer
             Response Block to create the next IOTP Message for the
             Transaction and sends it together with the Signature
             Block if present to the required Trading Role.

    CONTINUED ...

 *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 20 Brand Independent Offer Exchange

Note that a Brand Independent Offer Document Exchange always occurs when only one payment brand, protocol and currency/amount is being offered to the Consumer by the Merchant. It is also likely to, but will not necessarily, occur when multiple brands are being offered, the Payment Handler is the same, and all brands use the same set of protocols.

Note that the TPO Block and the Offer Response Block can be sent in separate IOTP messages (see Brand Dependent Offer Document Exchange) even if the Offer Response Block does not change. However this increases the number of messages in the transaction and is therefore likely to increase transaction response times.

IOTP aware applications supporting the Consumer Trading Role must check for the existence of an Offer Response Block in the first IOTP Message to determine whether the Offer Document Exchange is brand dependent or not.

MESSAGE PROCESSING GUIDELINES

On receiving a TPO and Offer Response IOTP Message (see below), the Consumer may either:

- generate and send the next IOTP Message in the IOTP transaction and send it to the required Trading Role. This is dependent on the IOTP Transaction, or
- indicate failure to continue with the IOTP Transaction by sending a Cancel Block back to the Merchant containing a Status Component with a StatusType of Offer, a ProcessState of Failed and the CompletionCode (see section 7.16.1) set to either: ConsCancelled or Unspecified.

If the Merchant receives an IOTP Message containing a Cancel block, then the Consumer is likely to go to the CancelNetLocn specified on the Trading Role Element in the Organisation Component for the Merchant.

9.1.2.3 TPO IOTP Message

The TPO IOTP Message is only used with a Brand Dependent Offer Document Exchange. Apart from a Transaction Reference Block (see section 3.3), this message consists of just a Trading Protocol Options Block (see section 8.1) which is described below.

TPO (TRADING PROTOCOL OPTIONS) BLOCK

The Trading Protocol Options Block (see section 8.1) must contain the following Trading Components:

- one Protocol Options Component which defines the options which apply to the whole IOTP Transaction. See Section 7.1.

- one Brand List Component (see section 7.7) for each Payment in the IOTP Transaction that contain one or more payment brands and protocols which may be selected for use in each payment

- Organisation Components (see section 7.6) with the following roles:

  - Merchant who is making the offer

  - Consumer who is carrying out the transaction

  - the PaymentHandler(s) for the payment. The "ID" of the Payment Handler Organisation Component is contained within the PhOrgRef attribute of the Payment Component

If the IOTP Transaction includes a Delivery then the TPO Block must also contain:

- Organisation Components with the following roles:

  - DeliveryHandler who will be delivering the goods or services

  - DelivTo i.e. the person or Organisation which is to take delivery

AUTHENTICATION STATUS AND SIGNATURE BLOCKS

If the Offer Document Exchange was preceded by an Authentication Document Exchange, then the TPO IOTP Message may also contain:

- an Authentication Status Block (see section 8.6), and

- an optional Signature Block (Authentication Status) Signature Block

See section 9.1.1.4 Authentication Status IOTP Message for more details.

9.1.2.4 TPO Selection IOTP Message

The TPO Selection IOTP Message is only used with a Brand Dependent Offer Document Exchange. Apart from a Transaction Reference Block (see section 3.3), this message consists of just a TPO Selection Block (see section 8.1) which is described below.

TPO SELECTION BLOCK

The TPO Selection Block (see section 8.2) contains:

- one Brand Selection Component (see section 7.8) for use in a later Payment Exchange. It contains the results of the consumer selecting a Payment Brand, Payment Protocol and currency/amount from the list provided in the Brand List Component.

9.1.2.5 Offer Response IOTP Message

The Offer Response IOTP Message is only used with a Brand Dependent Offer Document Exchange. Apart from a Transaction Reference Block (see section 3.3), this message consists of:

- an Offer Response Block (see section 8.1) and
- an optional Signature Block (see section 8.16).

OFFER RESPONSE BLOCK

The Offer Response Block (see section 8.3) contains the following components:

- one Status Component (see section 7.16) which indicates the status of the Offer Response. The ProcessState attribute should be set to CompletedOk
- one Order Component (see section 7.5) which contains details about the goods and services which are being purchased or the financial transaction which is taking place
- one or more Payment Component(s) (see section 7.9) for each payment which is to be made
- zero or one Delivery Components (see section 7.13) containing details of the delivery to be made if the IOTP Transaction includes a delivery
- zero or more Trading Role Data Components (see section 7.17) if required by the Merchant.

SIGNATURE BLOCK (OFFER RESPONSE)

If the Authentication Status Block is being digitally signed then a Signature Block must be included that contains a Signature Component (see section 7.19) with Digest Elements for the following XML elements:

If the Offer Response is being digitally signed then a Signature Block must be included that contains a Signature Component (see section 7.19) with Digest Elements for the following XML elements:

- o the Transaction Reference Block (see section 3.3) for the IOTP Message that contains information that describes the IOTP Message and IOTP Transaction
- o the Transaction Id Component (see section 3.3.1) which globally uniquely identifies the IOTP Transaction
- o the following components of the TPO Block :
  - the Protocol Options Component, and
  - the Brand List Component
  - all the Organisation Components present
- o the following components of the Offer Response Block:
  - the Order Component
  - all the Payment Components present
  - the Delivery Component if present
  - any Trading Role Data Components present

9.1.2.6 TPO and Offer Response IOTP Message

The TPO and Offer Response IOTP Message is only used with a Brand Independent Offer Document Exchange. Apart from a Transaction Reference Block (see section 3.3), this message consists of:

- o a Trading Protocol Options Block (see section 8.1)
- o an Offer Response Block (see section 8.1) and
- o an optional Signature Block (see section 8.16).

TPO (TRADING PROTOCOL OPTIONS) BLOCK

This is the same as the Trading Protocol Options Block described in TPO IOTP Message (see section 9.1.2.3).

OFFER RESPONSE BLOCK

This the same as the Offer Response Block in the Offer Response IOTP Message (see section 9.1.2.5).

AUTHENTICATION STATUS

If the Offer Document Exchange was preceded by an Authentication Document Exchange, then the TPO and Offer Response IOTP Message may also contain an Authentication Status Block (see section 8.6).

SIGNATURE BLOCK

This is the same as the Signature Block in the Offer Response IOTP Message (see section 9.1.2.5) with the addition that:

- o if the Offer Document Exchange is Brand Dependent then the Signature Component in the Signature Block additionally contains a Digest Element for the Brand Selection Component contained in the TPO Selection Block

- o if the Offer Document Exchange was preceded by an Authentication Document Exchange then the Signature Component in the Signature Block additionally contains a Digest Element for the Authentication Status Block.

9.1.3 Payment Document Exchange

The Payment Document Exchange is a direct implementation of the last part of a Payment Trading Exchange (see section 2.2.2) after the Brand has been selected by the Consumer. A Payment Exchange consists of:

- o the Consumer requesting that a payment starts by generating Payment Request IOTP Message using information from previous IOTP Messages in the Transaction and then sending it to the Payment Handler

- o the Payment Handler and the Consumer then swapping Payment Exchange IOTP Messages encapsulating payment protocol messages until the payment is complete, and finally o the Payment Handler sending a Payment Response IOTP Message to the Consumer containing a receipt for the payment.

The IOTP Messages which are involved are illustrated by the diagram below.

```
 *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*
   Consumer
     |  Payment
     |  Handler
STEP |     |
 1.         Consumer generates Pay Request Block encapsulating a
            payment protocol message if required and sends to Payment
            Handler with the Signature Block if present

     C --> P PAYMENT REQUEST. IotpMsg: Trans Ref Block; Signature
            Block (optional); Pay Request Block

 2.         Payment Handler processes Pay Request Block, checks
            optional signature and starts exchanging payment protocol
            messages encapsulated in a Pay Exchange Block, with the
            Consumer

     C <-> P PAYMENT EXCHANGE. IotpMsg: Trans Ref Block; Pay Exchange
            Block

 3.         Consumer and Payment Handler keep on exchanging Payment
            Exchange blocks until eventually payment protocol
            messages finish so Payment Handler creates a Pay Receipt
            Component inside a Pay Response Block, and an optional
            Signature Component inside a Signature Block, sends them
            to the Consumer and stops.

     C <-- P PAYMENT RESPONSE. IotpMsg: Trans Ref Block; Signature
            Block (optional); Pay Response Block

 4.         Consumer checks Payment Response is OK. Optionally keeps
            information on IOTP Transaction for record keeping
            purposes and either stops or creates the next IOTP
            message for the Transaction and sends it together with
            the Signature Block, if present, to the required Trading
            Role

 *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 21 Payment Document Exchange

9.1.3.1 Message Processing Guidelines

On receiving a Payment Request IOTP Message, the Payment Handler should check that they are authorised to carry out the Payment (see section 6 Digital Signatures). They may then either:

- generate and send a Payment Exchange IOTP Message back to the Consumer, if more payment protocol messages need to be exchanged, or
- generate and send a Payment Response IOTP Message if the exchange of payment protocol messages is complete, or
- indicate failure to continue with the Payment by sending a Cancel Block back to the Consumer containing a Status Component with a StatusType of Payment, a ProcessState of Failed and the CompletionCode (see section 7.16.4) set to either: BrandNotSupp, CurrNotSupp, PaymtCancelled, AuthError, InsuffFunds, InstBrandInvalid, InstNotValid, BadInstrument or Unspecified.

On receiving a Payment Exchange IOTP Message, the Consumer may either:

- generate and send a Payment Exchange Message back to the Payment Handler or
- indicate failure to continue with the Payment by sending a Cancel Block back to the Payment Handler containing a Status Component with a StatusType of Payment, a ProcessState of Failed and the CompletionCode (see section 7.16.2) set to either: ConsCancelled or Unspecified.

On receiving a Payment Exchange IOTP Message, the Payment Handler may either:

- generate and send a Payment Exchange IOTP Message back to the Consumer, if more payment protocol messages need to be exchanged, or
- generate and send a Payment Response IOTP Message if the exchange of payment protocol messages is complete, or
- indicate failure to continue with the Payment by sending a Cancel Block back to the Consumer containing a Status Component with a StatusType of Payment, a ProcessState of Failed and the CompletionCode (see section 7.16.2) set to either: PaymtCancelled or Unspecified.

On receiving a Payment Response IOTP Message, the Consumer may either:

- o generate and send the next IOTP Message in the IOTP transaction and send it to the required Trading Role. This is dependent on the IOTP Transaction,
- o stop, since the IOTP Transaction has ended, or
- o indicate failure to continue with the IOTP Transaction by sending a Cancel Block back to the Merchant containing a Status Component with a StatusType of Payment, a ProcessState of Failed and the CompletionCode (see section 7.16.1) set to either: ConsCancelled or Unspecified.

If the Consumer receives an IOTP Message containing a Cancel block, then the information contained in the IOTP Message should be reported to the Consumer but no further action taken.

If the Payment Handler receives an IOTP Message containing a Cancel block, then the Consumer is likely to go to the CancelNetLocn specified on the Trading Role Element in the Organisation Component for the Payment Handler from which any further action may take place.

If the Merchant receives an IOTP Message containing a Cancel block, then the Consumer should have completed the payment but not continuing with the transaction for some reason. In this case the Consumer is likely to go to the CancelNetLocn specified on the Trading Role Element in the Organisation Component for the Merchant from which any further action may take place.

9.1.3.2 Payment Request IOTP Message

Apart from a Transaction Reference Block (see section 3.3), this message consists of:

- o a Payment Request Block, and
- o an optional Signature Block

PAYMENT REQUEST BLOCK

The Payment Request Block (see section 8.7) contains:

- o the following components copied from the Offer Response Block from the preceding Offer Document Exchange:
    - - the Status Component

  - the Payment Component for the payment which is being carried out

- the following components from the TPO Block:

  - the Organisation Components with the roles of Merchant and for the PaymentHandler that is being sent the Payment Request Block

  - the Brand List Component for the payment, i.e. the Brand List referred to by the BrandListRef attribute on the Payment Component

- one Brand Selection Component for the Brand List, i.e. the Brand Selection Component where BrandListRef attribute points to the Brand List. This component can be either:

  - copied from the TPO Selection Block if the payment was preceded by a Brand Dependent Offer Document Exchange (see section 9.1.2.1), or

  - created by the Consumer, containing the payment brand, payment protocol and currency/amount selected from the Brand List, if the payment was preceded by a Brand Independent Offer Document Exchange (see section 9.1.2.2)

- an optional Payment Scheme Component (see section 7.10) if required by the payment method used (see the Payment Method supplement to determine if this is needed).

- zero or more Trading Role Data Components (see section 7.17).

Note that:

- if there is more than one Payment Components in an Offer Response Block, then the second payment is the one within the Offer Response Block that contains a StartAfter attribute (see section 7.9) that identifies the Payment Component for the first payment

- the Payment Handler to include is identified by the Brand Selection Component (see section 7.8) for the payment. Also see section 6.3.1 Check Request Block sent Correct Organisation for an explanation on how Payment Handlers are identified

- the Brand List Component to include is the one identified by the BrandListRef attribute of the Payment Component for the identified payment

- o the Brand Selection Component to include from the Offer Response Block is the one that contains an BrandListRef attribute (see section 3.5) which identifies the Brand List Component for the second payment.

SIGNATURE BLOCK (PAYMENT REQUEST)

If the either the preceding Offer Document Exchange included an Offer Response Signature (see section 9.1.2.5 Offer Response IOTP Message), or a preceding Payment Exchange included a Payment Response Signature

(see section 9.1.3.4 Payment Response IOTP Message) then they should both be copied to the Signature Block in the Payment Request IOTP Message.

9.1.3.3 Payment Exchange IOTP Message

Apart from a Transaction Reference Block (see section 3.3), this message consists of just a Payment Exchange Block.

PAYMENT EXCHANGE BLOCK

The Payment Exchange Block (see section 8.8) contains:

- o one Payment Scheme Component (see section 7.10) which contains payment method specific data. See the Payment Method supplement for the payment method being used to determine what this should contain.

9.1.3.4 Payment Response IOTP Message

Apart from a Transaction Reference Block (see section 3.3), this message consists of:

- o a Payment Response Block, and
- o an optional Signature Block

PAYMENT RESPONSE BLOCK

The Payment Response Block (see section 8.9) contains:

- o one Payment Receipt Component (see section 7.11) which contains scheme specific data which can be used to verify the payment occurred

- one Payment Scheme Component (see section 7.10) if required which contains payment method specific data. See the Payment Method supplement for the payment method being used to determine what this should contain

- an optional Payment Note Component (see section 7.12)

- zero or more Trading Role Data Components (see section 7.17).

SIGNATURE BLOCK (PAYMENT RESPONSE)

If a signed Payment Receipt is being provided, indicated by the SignedPayReceipt attribute of the Payment Component being set to True, then the Signature Block should contain a Signature Component which contains Digest Elements for the following:

- the Transaction Reference Block (see section 3.3) for the IOTP Message which contains the first usage of the Payment Response Block,

- the Transaction Id Component (see section 3.3.1) within the Transaction Reference Block that globally uniquely identifies the IOTP Transaction,

- the Payment Receipt Component from the Payment Response Block,

- the Payment Note Component from the Payment Response Block,

- the other Components referenced by the PayReceiptNameRefs attribute (if present) of the Payment Receipt Component,

- the Status Component from the Payment Response Block,

- any Trading Role Data Components in the Payment Response Block, and

- all the Signature Components contained in the Payment Request Block if present.

9.1.4 Delivery Document Exchange

The Delivery Document Exchange is a direct implementation of a Delivery Trading Exchange (see section 2.2.3). It consists of:

- the Consumer requesting a Delivery by generating Delivery Request IOTP Message using information from previous IOTP Messages in the Transaction and then sending it to the Delivery Handler o the Delivery Handler sending a Delivery Response IOTP Message to the Consumer containing details about the Handler's response to the request together with an optional signature.

The message flow is illustrated by the diagram below.

```
 *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*
 Consumer
     |  Delivery
     |  Handler
STEP |     |
 1.          Consumer generates Delivery Request Block and sends it to
             the Delivery Handler with the Signature Block if present

     C --> D DELIVERY REQUEST. IotpMsg: Trans Ref Block; Signature
             Block; Delivery Request Block

 2.          Delivery Handler checks the Status and Order Components
             in the Delivery Request and the optional Signatures,
             creates a Delivery Response Block, sends to the Consumer
             and stops.

     C <-- D DELIVERY RESPONSE. IotpMsg: Trans Ref Block; Signature
             Block; Delivery Response Block

3.           Consumer checks Delivery Response Block and optional
             Signature Block are OK. Optionally keeps information on
             IOTP Transaction for record keeping purposes and stops.

 *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 22 Delivery Document Exchange

9.1.4.1 Message Processing Guidelines

On receiving a Delivery Request IOTP Message, the Delivery Handler should check that they are authorised to carry out the Delivery (see section 6 Digital Signatures). They may then either:

o generate and send a Delivery Response IOTP Message to the Consumer, or

o indicate failure to continue with the Delivery by sending a Cancel Block back to the Consumer containing a Status Component with a StatusType of Delivery, a ProcessState of Failed and the CompletionCode (see section 7.16.4) set to either: DelivCanceled, or Unspecified.

On receiving a Delivery Response IOTP Message, the Consumer should just stop since the IOTP Transaction is complete.

If the Consumer receives an IOTP Message containing a Cancel block, then the information contained in the IOTP Message should be reported to the Consumer but no further action taken.

9.1.4.2 Delivery Request IOTP Message

The Delivery Request IOTP Message consists of:

- a Delivery Request Block, and
- an optional Signature Block

DELIVERY REQUEST BLOCK

The Delivery Request Block (see section 8.10) contains:

- the following components copied from the Offer Response Block:
  - the Status Component (see section 7.16)
  - the Order Component (see section 7.5)
  - the Organisation Component (see section 7.6) with the roles of: Merchant, DeliveryHandler and DeliverTo
  - the Delivery Component (see section 7.13)
- the following Component from the Payment Response Block:
  - the Status Component (see section 7.16).
- zero or more Trading Role Data Components (see section 7.17).

SIGNATURE BLOCK (DELIVERY REQUEST)

If the preceding Offer Document Exchange included an Offer Response Signature or the Payment Document Exchange included a Payment Response Signature, then they should both be copied to the Signature Block.

9.1.4.3 Delivery Response IOTP Message

The Delivery Response IOTP Message contains a Delivery Response Block and an optional Signature Block.

DELIVERY RESPONSE BLOCK

The Delivery Response Block contains:

- o one Delivery Note Component (see section 7.15) which contains delivery instructions about the delivery of goods or services

  in3 SIGNATURE BLOCK (DELIVERY RESPONSE)

  The Signature Block should contain one Signature Component that contains Digest elements that refer to

- o the Transaction Id Component (see section 3.3.1) of the IOTP message that contains the Delivery Response Signature
- o the Transaction Reference Block (see section 3.3) of the IOTP Message that contains the Delivery Response Signature
- o the Consumer Delivery Data component contained in the Delivery Request Block (if any)
- o the Signature Components contained in the Delivery Request Block (if any)
- o the Status Component
- o the Delivery Note Component

9.1.5 Payment and Delivery Document Exchange

The Payment and Delivery Document Exchange is a combination of the last part of the Payment Trading Exchange (see section 2.2.2) and a Delivery Trading Exchange (see section 2.2.3). It consists of:

- o the Consumer requesting that a payment starts by generating Payment Request IOTP Message using information from previous IOTP Messages in the Transaction and then sending it to the Payment Handler
- o the Payment Handler and the Consumer then swapping Payment Exchange IOTP Messages encapsulating payment protocol messages until the payment is complete, and finally
- o the Payment Handler sending to the Consumer in one IOTP Message:
  - - a Payment Response Block containing a receipt for the payment, and

- a Delivery Response Block containing details of the goods or services to be delivered

The IOTP Messages which are involved are illustrated by the diagram below.

```
 *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*
 Consumer
     |  Payment
     |  Handler
STEP |     |
 1.          Consumer generates Pay Request Block encapsulating a
             payment protocol message if required and sends to Payment
             Handler with the Signature Block if present

     C --> P PAYMENT REQUEST. IotpMsg: Trans Ref Block; Signature
             Block; Pay Request Block

 2.          Payment Handler processes Pay Request Block, checks
             optional signature and starts exchanging payment protocol
             messages encapsulated in a Pay Exchange Block, with the
             Consumer

     C <-> P PAYMENT EXCHANGE. IotpMsg: Trans Ref Block; Pay Exchange
             Block

 3.          Consumer and Payment Handler keep on exchanging Payment
             Exchange blocks until eventually payment protocol
             messages finish so Payment Handler creates a Pay Receipt
             Component inside a Pay Response Block, and an optional
             Signature Component inside a Signature Block, then uses
             information from the Offer Response Bock to create a
             Delivery Response Block and sends both to the Consumer
             and stops.

     C <-- P PAYMENT RESPONSE & DELIVERY RESPONSE. IotpMsg: Trans Ref
             Block; Signature Block; Pay Response Block; Delivery
             Response Block

 4.          Consumer checks Payment Response and Delivery Response
             Blocks are OK. Optionally keeps information on IOTP
             Transaction for record keeping purposes and either stops
             or creates the next IOTP message for the Transaction and
             sends it together with the Signature Block, if present,
             to the required Trading Role

 *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 23 Payment and Delivery Document Exchange

The Delivery Response Block and the Payment Response Block may be combined into the same IOTP Message only if the Payment Handler has the information available so that she can send the Delivery Response Block. This is likely to, but will not necessarily, occur when the Merchant, the Payment Handler and the Delivery Handler Roles are combined.

The DelivAndPayResp attribute of the Delivery Component (see section 7.13) contained within the Offer Response Block (see section 8.3) is set to True if the Delivery Response Block and the Payment Response Block are combined into the same IOTP Message and is set to False if the Delivery Response Block and the Payment Response Block are sent in separate IOTP Messages.

9.1.5.1 Message Processing Guidelines

On receiving a Payment Request IOTP Message or a Payment Exchange IOTP Message, the Payment Handler should carry out the same actions as for a Payment Document Exchange (see section 9.1.3.1).

On receiving a Payment Exchange IOTP Message, the Consumer should also carry out the same actions as for a Payment Document Exchange (see section 9.1.3.1).

On receiving a Payment Response and Delivery Response IOTP Message then the IOTP Transaction is complete and should take no further action.

If the Consumer receives an IOTP Message containing a Cancel block, then the information contained in the IOTP Message should be reported to the Consumer but no further action taken.

If the Payment Handler receives an IOTP Message containing a Cancel block, then the Consumer is likely to go to the CancelNetLocn specified on the Trading Role Element in the Organisation Component for the Payment Handler from which any further action may take place.

If the Merchant receives an IOTP Message containing a Cancel block, then the Consumer should have completed the payment but not continuing with the transaction for some reason. In this case the Consumer is likely to go to the CancelNetLocn specified on the Trading Role Element in the Organisation Component for the Merchant from which any further action may take place.

9.1.5.2 Payment Request IOTP Message

The content of this message is the same as for a Payment Request IOTP Message in a Payment Document Exchange (see section 9.1.3.2).

9.1.5.3 Payment Exchange IOTP Message

The content of this message is the same as for a Payment Exchange IOTP Message in a Payment Document Exchange (see section 9.1.3.3).

9.1.5.4 Payment Response and Delivery Response IOTP Message

The content of this message consists of:

- a Payment Response Block,
- an optional Signature Block (Payment Response), and
- a Delivery Response Block.

PAYMENT RESPONSE BLOCK

The content of this block is the same as the Payment Response Block in the Payment Response IOTP Message associated with a Payment Document Exchange (see section 9.1.3.4).

SIGNATURE BLOCK (PAYMENT RESPONSE)

The content of this block is the same as the Signature Block (Payment Response) in the Payment Response IOTP Message associated with a Payment Document Exchange (see section 9.1.3.4).

DELIVERY RESPONSE BLOCK

The content of this block is the same as the Delivery Response Block in the Delivery Response IOTP Message associated with a Delivery Document Exchange (see section 9.1.4.3).

9.1.6 Baseline Authentication IOTP Transaction

A Baseline Authentication IOTP Transaction may occur at any time between any of the Trading Roles involved in IOTP Transactions. This means it could occur:

- before another IOTP Transaction
- at the same time as another IOTP Transaction
- independently of any other IOTP Transaction.

The Baseline Authentication IOTP Transaction consists of just an Authentication Document Exchange (see section 9.1.1) as illustrated by the diagram below.

```
   *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*

   START -------------------------------------------------------
                                                                v
                                                       ----------------
                                                      | AUTHENTICATION |
                                                       ----------------
                                                                 |
                                                                 |
                                                                 |
                                                                 |
                  -------------------     -----------------      |
                 | BRAND INDEPENDENT |   | BRAND DEPENDENT |     |
                 |       OFFER       |   |      OFFER      |     |
                  -------------------     -----------------      |
                                                                 |
                                                                 |
                                                                 |
                                                                 |
                                                                 |
                      ---------           --------------         |
                     | PAYMENT |         | PAYMENT WITH |        |
                     | (first) |         |   DELIVERY   |        |
                      ---------           --------------         |
                                                                 |
                                                                 |
                                                                 |
         ----------        ---------                             |
        | DELIVERY |      | PAYMENT |                            |
        |          |      | {second)|                            |
         ----------        ---------                             |
                                                                 v
                                                               STOP

   *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 24 Baseline Authentication IOTP Transaction

Example uses of the Baseline Authentication IOTP Transaction include:

- o when the Baseline Authentication IOTP Transaction takes place as an early part of a session where strong continuity exists. For example, a Financial Institution could:
  - set up a secure channel (e.g., using [SSL/TLS]) with a customer
  - authenticate the customer using the Baseline Authentication IOTP Transaction, and then

   - provide the customer with access to account information and other services with the confidence that they are communicating with a bona fide customer.

- as a means of providing a Merchant role with Organisation Components that contain information about Consumer and DelivTo Trading Roles

- so that a Consumer may authenticate a Payment Handler before starting a payment.

9.1.7 Baseline Deposit IOTP Transaction

The Baseline Deposit IOTP Transaction supports the deposit of electronic cash with a Financial Institution.

Note: The Financial Institution has, in IOTP terminology, a role of merchant in that a service (i.e. a deposit of electronic cash) is being offered in return for a fee, for example bank charges of some kind. The term "Financial Institution" is used in the diagrams and in the text for clarity.

The Baseline Deposit IOTP Transaction consists of the following Document Exchanges:

- an optional Authentication Document Exchange (see section 9.1.1)

- an Offer Document Exchange (see section 9.1.2), and

- a Payment Document Exchange (see section 9.1.3).

The way in which these Document Exchanges may be combined together is illustrated by the diagram below.

```
   *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*

   START -----------------------------------------------------
      |                                                       v
      |                                                ----------------
      |                                               | AUTHENTICATION |
      |                                                ----------------
       --------------------------------------                |
                       |                     |              |
                       |      -------------- | -------------
                       v      v              v      v
                  -------------------     -----------------
                 | BRAND INDEPENDENT |   | BRAND DEPENDENT |
                 |       OFFER       |   |      OFFER      |
                  -------------------     -----------------
                        |                        |
                        |                        |
                        |                        |
                        |     -------------------
                        v    v
                      ---------           --------------
                     | PAYMENT |         | PAYMENT WITH |
                     | (first) |         |   DELIVERY   |
                      ---------           --------------
                          |
                           ----------------
                                           |
         ----------        ---------       |
        | DELIVERY |      | PAYMENT |      |
        |          |      | {second)|      |
         ----------        ---------       |
                                           |
                                            -----------------> STOP

   *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 25 Baseline Deposit IOTP Transaction

See section 9.1.12 "Valid Combinations of Document Exchanges" to determine which combination of document exchanges apply to a particular instance of an IOTP Transaction

Note that:

o a Merchant (Financial Institution) may be able to accept a deposit in several different types of electronic cash although, since the Consumer role that is depositing the electronic cash usually knows what type of cash they want to deposit, it is usually constrained in practice to only one type. However, there may be several different protocols which may be used for the same "brand" of electronic cash. In this case a Brand Dependent Offer may be appropriate to negotiate the protocol to be used.

- o the Merchant (Financial Institution) may use the results of the authentication to identify not only the consumer but also the account to which the payment is to be deposited. If no single account can be identified, then it must be obtained by other means. For example:
  - the consumer could specify the account number prior to the Baseline Deposit IOTP Transaction starting, or
  - the consumer could have been identified earlier, for example using a Baseline Authentication IOTP Transaction, and an account selected from a list provided by the Financial Institution.
- o The Baseline Deposit IOTP Transaction without an Authentication Document Exchange might be used:
  - if a previous IOTP transaction, for example a Baseline Withdrawal or a Baseline Authentication, authenticated the consumer, and a secure channel has been maintained, therefore the authenticity of the consumer is known
  - if authentication is achieved as part of a proprietary payment protocol and is therefore included in the Payment Document Exchange
  - if authentication of the consumer has been achieved by some other means outside of the scope of IOTP, for example, by using a pass phrase, or a proprietary banking software solution.

9.1.8 Baseline Purchase IOTP Transaction

The Baseline Purchase IOTP Transaction supports the purchase of goods or services using any payment method. It consists of the following Document Exchanges:

- o an optional Authentication Document Exchange (see section 9.1.1)
- o an Offer Document Exchange (see section 9.1.2)
- o either:
  - a Payment Document Exchange (see section 9.1.3) followed by

      - a Delivery Document Exchange (see section 9.1.4)

   o a Payment Document Exchange only, or

   o a combined Payment and Delivery Document Exchange (see section 9.1.5).

   The ways in which these Document Exchanges are combined is illustrated by the diagram below.

```
   *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*

   START -----------------------------------------------------
      |                                                       v
      |                                                ----------------
      |                                               | AUTHENTICATION |
      |                                                ----------------
       --------------------------------------               |    |
                       |                     |              |    |
                       |      -------------- | -------------     |
                       v      v              v      v            |
                  -------------------     -----------------      |
                 | BRAND INDEPENDENT |   | BRAND DEPENDENT |     |
                 |       OFFER       |   |      OFFER      |     |
                  -------------------     -----------------      |
                        |    |                   |   |           |
                        |     ---------------    |   |           |
                        |                    |   |   |           |
                        |     -------------- | --    |           |
                        v    v               v       v           |
                      ---------           --------------         |
                     | PAYMENT |         | PAYMENT WITH |        |
                     | (first) |         |   DELIVERY   |        |
                      ---------           --------------         |
                          |                      |               |
              -----------------------------      |               |
              v                            |     |               |
         ----------        ---------       |     |               |
        | DELIVERY |      | PAYMENT |      |     |               |
        |          |      | {second)|      |     |               |
         ----------        ---------       |     |               |
              |                            |     |               v
               ----------------------------------------------> STOP

   *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 26 Baseline Purchase IOTP Transaction

See section 9.1.12 "Valid Combinations of Document Exchanges" to determine which combination of document exchanges apply to a particular instance of an IOTP Transaction.

9.1.9 Baseline Refund IOTP Transaction

In business terms the refund process typically consists of:

- a request for a refund being made by the Consumer to the Merchant, typically supported by evidence to demonstrate:
  - the original trade took place, for example by providing a receipt for the original transaction
  - using some type of authentication, that the consumer requesting the refund is the consumer, or a representative of the consumer, who carried out the original trade
  - the reason why the merchant should make the refund
- the merchant agreeing (or not) to the refund. This may involve some negotiation between the Consumer and the Merchant, and, if the merchant agrees,
- a refund payment by the Merchant to the Consumer.

The Baseline Refund IOTP Transaction supports a subset of the above, specifically it supports:

- stand alone authentication of the Consumer using a separate Baseline Authentication IOTP Transaction (see section 9.1.6)
- a refund payment by the Merchant to the Consumer using the following two Trading Exchanges:
  - an optional Authentication Document Exchange (see section 9.1.1)
  - an Offer Document Exchange (see section 9.1.2), and
  - a Payment Document Exchange (see section 9.1.3).

The ways in which these Document Exchanges are combined is illustrated by the diagram below.

```
   *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*

   START -----------------------------------------------------
      |                                                       v
      |                                                ----------------
      |                                               | AUTHENTICATION |
      |                                                ----------------
       ---------------------------------------                |
                       |                     |                |
                       |      -------------- | -------------
                       v      v              v      v
                  -------------------     -----------------
                 | BRAND INDEPENDENT |   | BRAND DEPENDENT |
                 |       OFFER       |   |      OFFER      |
                  -------------------     -----------------
                        |                        |
                        |                        |
                        |                        |
                        |     -------------------
                        v    v
                      ---------           --------------
                     | PAYMENT |         | PAYMENT WITH |
                     | (first) |         |   DELIVERY   |
                      ---------           --------------
                          |
                           ----------------
                                           |
         ----------        ---------       |
        | DELIVERY |      | PAYMENT |      |
        |          |      | {second)|      |
         ----------        ---------       |
                                           |
                                            ------------------> STOP

   *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 27 Baseline Refund IOTP Transaction

A Baseline Refund IOTP Transaction without an Authentication Document Exchange might be used:

o when authentication of the consumer has been achieved by some other means, for example, the consumer has entered some previously supplied code in order to identify herself and the refund to which the code applies. The code could be supplied, for example on a web page or by e-mail.

- o when a previous IOTP transaction, for example a Baseline Authentication, authenticated the consumer, and a secure channel has been maintained, therefore the authenticity of the consumer is known and therefore the previously agreed refund can be identified.

- o when the authentication of the consumer is carried out by the Payment Handler using a payment scheme authentication algorithm.

9.1.10 Baseline Withdrawal IOTP Transaction

The Baseline Withdrawal IOTP Transaction supports the withdrawal of electronic cash from a Financial Institution.

Note: The Financial Institution has, in IOTP terminology, a role of merchant in that a service (i.e. a withdrawal of electronic cash) is being offered in return for a fee, for example bank charges of some kind. The term "Financial Institution" is used in the diagrams and in the text for clarity.

The Baseline Withdrawal IOTP Transaction consists of the following Document Exchanges:

- o an optional Authentication Document Exchange (see section 9.1.1)

- o an Offer Document Exchange (see section 9.1.2), and

- o a Payment Document Exchange (see section 9.1.3).

The way in which these Document Exchanges may be combined together is illustrated by the diagram below.

```
   *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*

   START -----------------------------------------------------
      |                                                       v
      |                                                ----------------
      |                                               | AUTHENTICATION |
      |                                                ----------------
       --------------------------------------               |
                       |                     |              |
                       |      -------------- | -------------
                       v      v              v      v
                  -------------------     -----------------
                 | BRAND INDEPENDENT |   | BRAND DEPENDENT |
                 |       OFFER       |   |      OFFER      |
                  -------------------     -----------------
                        |                        |
                        |                        |
                        |                        |
                        |     -------------------
                        v    v
                      ---------           --------------
                     | PAYMENT |         | PAYMENT WITH |
                     | (first) |         |   DELIVERY   |
                      ---------           --------------
                          |
                           ----------------
                                           |
         ----------        ---------       |
        | DELIVERY |      | PAYMENT |      |
        |          |      | {second)|      |
         ----------        ---------       |
                                           |
                                            -----------------> STOP

   *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 28 Baseline Withdrawal IOTP Transaction

Note that:

o a Merchant (Financial Institution) may be able to offer withdrawal of several different types of electronic cash. In practice usually only one form of electronic cash may be offered. However, there may be several different protocols which may be used for the same "brand" of electronic cash.

- o the Merchant (Financial Institution) may use the results of the authentication to identify not only the consumer but also the account from which the withdrawal is to be made. If no single account can be identified, then it must be obtained by other means. For example:
  - the consumer could specify the account number prior to the Baseline Withdrawal IOTP Transaction starting, or
  - the consumer could have been identified earlier, for example using a Baseline Authentication IOTP Transaction, and an account selected from a list provided by the Financial Institution.
- o a Baseline Withdrawal without an authentication might be used:
  - if a previous IOTP transaction, for example a Baseline Deposit or a Baseline Authentication, authenticated the consumer, and a secure channel has been maintained, therefore the authenticity of the consumer is known
  - if authentication is achieved as part of a proprietary payment protocol and is therefore included in the Payment Document Exchange
  - if authentication of the consumer has been achieved by some other means, for example, by using a pass phrase, or a proprietary banking software solution.

9.1.11 Baseline Value Exchange IOTP Transaction

The Baseline Value Exchange Transaction uses Payment Document Exchanges to support the exchange of value in one currency obtained using one payment method with value in the same or another currency using the same or another payment method. Examples of its use include:

- o electronic cash advance on a credit card. For example the first payment could be a "dollar SET Payment" using a credit card with the second payment being a download of Visa Cash e-cash in dollars.
- o foreign exchange using the same payment method. For example the payment could be an upload of Mondex value in British Pounds and the second a download of Mondex value in Euros

- foreign exchange using different payment methods. For example the first payment could be a SET payment in Canadian Dollars followed a download of GeldKarte in Deutchmarks.

The Baseline Value Exchange uses the following Document Exchanges:

- an optional Authentication Document Exchange (see section 9.1.1)

- an Offer Document Exchange (see section 9.1.2), which provides details of what values and currencies will be exchanged, and

- two Payment Document Exchanges (see section 9.1.3) which carry out the two payments involved.

The way in which these Document Exchanges may be combined together is illustrated by the diagram below.

```
   *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*

   START ------------------------------------------------------
      |                                                         v
      |                                                ----------------
      |                                               | AUTHENTICATION |
      |                                                ----------------
       ---------------------------------------              |
                       |                     |              |
                       |      -------------- | -------------
                       v      v              v      v
                  --------------------    ------------------
                 | BRAND INDEPENDENT |   | BRAND DEPENDENT |
                 |       OFFER       |   |      OFFER      |
                  --------------------    ------------------
                        |                        |
                        |                        |
                        |                        |
                        |     -------------------
                        v    v
                      ----------          ---------------
                     | PAYMENT |         | PAYMENT WITH |
                     | (first) |         |   DELIVERY   |
                      ----------          ---------------
                          |
                           ----
                               v
         ----------        ---------
        | DELIVERY |      | PAYMENT |
        |          |      | {second)|
         ----------        ---------
                               |
                                -----------------------------> STOP

   *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 29 Baseline Value Exchange IOTP Transaction

The Baseline Value Exchange IOTP Transaction occurs in two basic forms:

- o Brand Dependent Value Exchange. Where the content of the offer, for example the rate at which one form of value is exchanged for another, is dependent on the payment brands and protocols selected by the consumer, and

- o Brand Independent Value Exchange. Where the content of the offer is not dependent on the payment brands and protocols selected.

Note: In the above the role is a Merchant even though the Organisation carrying out the Value Exchange may be a Bank or some other Financial Institution. This is because the Bank is acting as a merchant in that they are making an offer which the Consumer can either accept or decline.

The TPO Block and Offer Response Block may only be combined into the same IOTP Message if the content of the Offer Response Block does not change as a result of selecting the payment brands and payment protocols to be used in the Value Exchange.

BASELINE VALUE EXCHANGE SIGNATURES

The use of signatures to ensure the integrity of a Baseline Value Exchange is illustrated by the diagram below.

```
*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*

Signature generated                  IotpMsg (TPO)
by Merchant ensures                  - Trans Ref Block
integrity of the Offer -------->  -  - Signature Block
                                 |   - TPO Block              MERCHANT
                                 |   - Offer Response Block
                                 |
Signature generated by           |
the Payment Handler of           |   IotpMsg (Pay Resp 1)
the first payment binds          |   - Trans Ref Block          PAYMENT
Pay Receipt for the first ----->  -> - Signature Block -----    HANDLER
payment to the Offer                 - Pay Response Block 1 |      1
                                                            |
Signature generated by                                      |
the Payment Handler of           IotpMsg (Pay Resp 2)       |   PAYMENT
the second payment binds           - Trans Ref Block        |   HANDLER
the second payment to the ----->   - Signature Block <------       2
first payment and therefore        - Pay Response Block 2
to the Offer

*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 30 Baseline Value Exchange Signatures

9.1.12 Valid Combinations of Document Exchanges

The following diagram illustrates the data conditions in the various IOTP messages which can be used by a Consumer Trading Role to determine whether the combination of Document Exchanges are valid.

```
   *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*

   START
     |
     v
   Auth Request Block in  =TRUE
    first IOTP Message ? ---------------------------------------
      | = FALSE                                                |
      v                                                        v
   Offer Response Block in                             ----------------
     first IOTP Message ?                             | AUTHENTICATION |
      |=TRUE         |=FALSE                           ----------------
      |              |                                       |
      |              |                                       v
      |                ----------------------       TPO & Offer Response
       -------------                         |   Blocks in last IOTP Msg
                    |                        |     |=TRUE        |=FALSE
                    |                        |     |             v
                    |          ------------- | ----    TPO Block only if
                    |         |              |         last IOTP Message
                    |         |              |         of Authentication
                    |         |              |          |=TRUE   |=FALSE
                    v         v              v          v        |
                  -------------------     -----------------      |
                 | BRAND INDEPENDENT |   | BRAND DEPENDENT |     |
                 |       OFFER       |   |      OFFER      |     |
                  -------------------     -----------------      |
                          |                   |                  |
                          v                   v                  |
                       Offer Response Block contains             |
                             Delivery Component ?                |
                            |=FALSE        |=TRUE                |
                         ---               v                     |
                        |        Value of DelivAndPayResp        |
                        |    attribute of Delivery Component ?   |
                        |    |=FALSE         |=TRUE              |
                        |    |               |                   |
                        v    v               v                   |
                      ---------           --------------         |
                     | PAYMENT |         | PAYMENT WITH |        |
                     | (first) |         |   DELIVERY   |        |
                      ---------           --------------         |
                          |                      |               |
                          v                      |               |
            Offer and Response Block contains     -------------->|
                  Delivery Component ?                           |
                  |=TRUE           |=FALSE                       |
                  |                v                             |
                  |         Two Payment Components               |
                  |      present in Offer Response Block?        |
                  |           |=TRUE             |=FALSE         |
                  v           v                  |               |
         ----------        ---------             |               |
        | DELIVERY |      | PAYMENT |            |               |
        |          |      | {second)|            |               |
         ----------        ---------             |               |
              |                |                 |               v
               ----------------------------------------------> STOP

   *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 31 Valid Combinations of Document Exchanges

```
   1) If first IOTP Message of an IOTP Transaction contains an
      Authentication Request then:

      a) IOTP Transaction includes an Authentication Document Exchange
         (see section 9.1.1). (Note 1)

      b) If the last IOTP Message of the Authentication Document
         Exchange includes a TPO Block and an Offer Response Block then:

         i) IOTP Transaction includes a Brand Independent Offer Document
            Exchange (see section 9.1.2.2). (Note 2)

      c) Otherwise, if the last IOTP Message of the Authentication
         Exchange includes a TPO Block but NO Offer Response Block,
         then:

         i) IOTP Transaction includes a Brand Dependent Offer Document
            Exchange (see section 9.1.2.1). (Note 2)

      d) Otherwise (Authentication Status IOTP Message of the
         Authentication Document Exchange contains neither a TPO Block
         but nor an Offer Response Block)

         i) IOTP Transaction consists of just an Authentication Document
            Exchange. (Note 3)

   2) Otherwise (no Authentication Request in first IOTP Message):

      e) IOTP Transaction does not include an Authentication Document
         Exchange (Note 2)

      f) If first IOTP Message contains an Offer Response Block, then:

         i) the IOTP Transaction contains a Brand Independent Offer
            Document Exchange (Note 2)

      g) Otherwise (no Offer Response Block in first IOTP Message):

         i) the IOTP Transaction includes a Brand Dependent Offer
            Document Exchange (Note 2)

   3) If an Offer Response Block exists in any IOTP message then:

      h) If the Offer Response Block contains a Delivery Component then:

         i) If the DelivAndPayResp attribute of the Delivery Component
            is set to True, then:
```

```
          (1) the IOTP Transaction consists of a Payment And Delivery
              Document Exchange (see section 9.1.5) (Note 4)

      ii) otherwise (the DelivAndPayResp attribute of the Delivery
          Component is set to False)

          (1) the IOTP Transaction consists of a Payment Document
              Exchange (see section 9.1.3) followed by a Delivery
              Document Exchange (see section 9.1.4) (Note 4)

   i) otherwise (the Offer Response Block does not contain a Delivery
      Component)

       i) if the Offer Response Block contains just one Payment
          Component, then:

          (1) the IOTP Transaction contains just one Payment Document
              Exchange (Note 5)

      ii) if the Offer Response Block contains two Payment Components,
          then:

          (1) the IOTP Transaction contains two Payment Document
              Exchanges.  The StartAfter attribute of the Payment
              Components is used to indicate which payment occurs
              first (Note 6)

     iii) if the Offer Response Block contains no or more than two
          Payment Components, then there is an error
```

4) Otherwise (no Offer Response Block) there is an error.

The following table indicates the types of IOTP Transactions which can validly have the conditions indicated above.

Note	IOTP Transaction Validity
1.	Any Payment and Authentication IOTP Transaction
2.	Any Payment and Authentication IOTP Transaction except Baseline Authentication
3.	Either Baseline Authentication, or a Baseline Purchase, Refund, Deposit, Withdrawal or Value Exchange with a failed Authentication
4.	Baseline Purchase only
5.	Baseline Purchase, Refund, Deposit or Withdrawal

6. Baseline Value Exchange only

9.1.13 Combining Authentication Transactions with other Transactions

In the previous sections an Authentication Document Exchange is shown preceding an Offer Document Exchange as part of a single IOTP Transaction with the same IOTP Transaction Id.

It is also possible to run a separate Authentication Transaction at any point, even in parallel with another IOTP Transaction. Typically this will be used:

- by a Consumer to authenticate a Merchant, Payment Handler or a Delivery Handler, or
- by a Payment Handler or Delivery Handler to authenticate a Consumer.

In outline the basic process consists of:

- the Trading Role that decides it wants to carry out an authentication of another role suspends the current IOTP transaction being carried out
- a stand-alone Authentication transaction is then carried out. This may, at implementer’s option, be linked to the original IOTP Transaction using a Related To Component (see section 3.3.3) in the Transaction Reference Block.
- if the Authentication transaction is successful, then the original IOTP Transaction is restarted
- if the Authentication fails then the original IOTP Transaction is cancelled.

For example, a Consumer could:

- authenticate the Payment Handler for a Payment between receiving an Offer Response from a Merchant and before sending the Payment Request to that Payment Handler
- authenticate a Delivery Handler for a Delivery between receiving the Payment Response from a Payment Handler and before sending the Delivery Request

A Payment Handler could authenticate a Consumer after receiving the Payment Request and before sending the next Payment related message.

A Delivery Handler could authenticate a Consumer after receiving the Delivery Request and before sending the Delivery Response.

Note: Some Payment Methods may carry out an authentication within the Payment Exchange. In this case the information required to carry out the authentication will be included in Payment Scheme Components.

In this instance IOTP aware application will not be aware that an authentication has occurred since the Payment Scheme Components that contain authentication request information will be indistinguishable from other Payment Scheme Components.

9.2 Infrastructure Transactions

Infrastructure Transactions are designed to support inquiries about whether or not a transaction has succeeded or a Trading Role’s servers are operating correctly. There are two types of transaction:

- o a Transaction Status Inquiry Transaction which provides information on the status of an existing or complete IOTP transaction, and
- o Ping Transaction that enables one IOTP aware application to determine if the IOTP aware application at another Trading Role is operating and verify whether or not signatures can be handled.

Each of these is described below

9.2.1 Baseline Transaction Status Inquiry IOTP Transaction

The Baseline IOTP Transaction Status Inquiry provides information on the status of an existing or complete IOTP transaction.

The Trading Blocks used by the Baseline Transaction Status Inquiry Transaction are:

- o an Inquiry Request Trading Block (see section 8.12),
- o an Inquiry Response Trading Block (see section 8.13)
- o an optional Signature Block (see section 8.16).

The Inquiry IOTP Transaction can be used for a variety of reasons. For example:

- o to help in resuming a suspended transaction to determine the current state of processing of one of the other roles,

- for a merchant to determine if a payment, delivery, etc., was completed. For example, a Consumer might claim that payment was made but no signed IOTP payment receipt was available to prove it. If the Merchant makes an inquiry of the Payment Handler then the Merchant can determine whether or not payment was made.

Note: Inquiries on Baseline Ping IOTP Transactions (see section 9.2.2) are ignored.

MAKING INQUIRIES OF ANOTHER TRADING ROLE

One Trading Role may make an inquiry of any other Trading Role at any point in time.

IOTP aware software that supports the Consumer Trading Role may not:

- digitally sign a response if requested, since it may not have the capability, or

- respond to an Inquiry Request at all since it may not be on-line, or may consider that the request is not reasonable since, for example, the Request was not digitally signed.

As a guideline:

- the Consumer should send a Transaction Status Inquiry Block to a Trading Role only after the following events have occurred:

  - to the Merchant, after sending a TPO Selection Block,

  - to the Payment Handler, after sending a Payment Request Block,

  - to the Delivery Handler, after sending a Delivery Request Block,

- other Trading Roles should send a Transaction Status Inquiry Block to the Consumer only after receiving a message from the Consumer and before sending the final "Response" message to the Consumer

- there are no restrictions on non-Consumer Trading Roles sending Inquiries to other trading roles.

TRANSACTION STATUS INQUIRY TRANSPORT SESSION

For a Transaction Status Inquiry on an ongoing transaction a different transport session from the ongoing transaction is used. For a Transaction Status Inquiry on a past transaction, how the IOTP module on the software at the Trading Role is started upon the receipt of Inquiry Request message is defined in each Mapping to Transport supplement for IOTP.

TRANSACTION STATUS INQUIRY ERROR HANDLING

Errors in a Transaction Status Inquiry can be categorised into one of the following three cases:

- o Business errors (see section 4.2) in the original (inquired) messages
- o Technical errors (see section 4.1) - both IOTP and payment scheme specific ones - in the original IOTP (inquired) messages
- o Technical errors in the message containing the Inquiry Request Block itself

The following outlines what the software should do in each case

BUSINESS ERRORS IN THE ORIGINAL MESSAGES

Return an Inquiry Response Block containing the Status Component which was last sent to the Consumer Role.

TECHNICAL ERRORS IN THE ORIGINAL MESSAGES

Return an Inquiry Response Block containing a Status Component. The Status Component should contain a ProcessState attribute set to ProcessError. In this case send back an Error Block indicating where the error was found in the original message.

TECHNICAL ERRORS IN THE INQUIRY REQUEST BLOCK

Return an Error message. That is, send back an Error Block containing the Error Code (see section 7.21.2) which describes the nature of the error in the Inquiry Request message.

INQUIRY TRANSACTION MESSAGES

The following Figure outlines the Baseline IOTP Transaction Status Inquiry process.

```
 *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*

   1st Role
     |  2nd Role
STEP |     |
 1.          The first role decides to inquire on an IOTP Transaction
             by, for example, clicking on the inquiry button of an
             IOTP Aware Application. This will then generate an
             Inquiry Request Block and send it to the appropriate
             Trading Role.

     1 --> 2 INQUIRY REQUEST. IotpMsg: TransRef Block; Signature Block
             (optional); Inquiry Request Block

 2.          The Trading Role checks the digital signature (if
             present). If the recipient wants to respond, then the
             Trading Role checks the transaction status of the
             transaction that is being inquired upon by using the
             IotpTransId in the Transaction ID Component of the
             Transaction Reference Block, then generates the
             appropriate Inquiry Response Block, sends the message
             back to the 1st Role and stops

     1 <-- 2 INQUIRY RESPONSE. IotpMsg: TransRef Block; Inquiry
             Response Block; Signature Block (Optional)

3.           First role checks the Inquiry Response Block and optional
             signature, takes whatever action is appropriate or
             perhaps stops. This may include displaying status
             information to the end user.

 *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 32 Baseline Transaction Status Inquiry

The remainder of this sub-section on the Baseline Transaction Status Inquiry IOTP Transaction defines the contents of each Trading Block. Note that the term "original transaction" is the transaction which a trading role wants to discover some information about.

TRANSACTION REFERENCE BLOCK

A Trading Role making an inquiry must use a Transaction Id Component (see section 3.3.1) where both the IotpTransId and TransTimeStamp attributes are the same as in the Transaction Id Component of the original transaction that is being inquired upon. The IotpTransId attribute in this component serves as the key in querying the transaction logs maintained at the Trading Role’s site. The value of the ID attribute of the Message Id Component should be different from those of any in the original transaction (see section 3.4.1).

If up-to-date status information is required then the MsgId Component, and in particular the ID attribute for the MsgId Component must be different from any other IOTP Message that has been sent by the Trading Role. This is required because of the way that Idempotency is handled by IOTP (see section 4.5.2.2 Checking/Handling Duplicate Messages).

INQUIRY REQUEST BLOCK

The Inquiry Request Block (see section 8.12) contains the following components:

o one Inquiry Type Component (see section 7.18). This identifies whether the inquiry is on an offer, payment, or delivery.

o zero or one Payment Scheme Components (see section 7.10). This is for encapsulating payment scheme specific inquiry messages for inquiries on a payment.

SIGNATURE BLOCK (INQUIRY REQUEST)

If a signature block is present on the message containing the Inquiry Request Block then it may be checked to determine if the Inquiry Request is authorised.

If present, the Inquiry Request Signature Block (see section 8.12) contains the following components:

o one Signature Component (see section 7.19)

o one or more Certificate Components, if required.

Inquiry Response Blocks should only be generated if the Transaction is authorised.

Note: Digital signatures on an Inquiry Request is only likely to occur if the recipient of the request expects the Inquiry Request to be signed. In this version of IOTP this will require some kind of pre-existing agreement. This means that:

o Consumers are unlikely to generate requests with signatures, although it is not an error if they do

- o the other trading roles may agree that digital signatures are required. For example a Payment Handler may require that an Inquiry Request is digitally signed by the Merchant so that they can check that the request is valid.

On the other hand if the original transaction to which the Inquiry relates was carried out over a secure channel (e.g., [SSL]) then it is probably reasonable to presume that if the sender of the Inquiry knows the Transaction Id component of the original message (including for example the timestamp) then the inquiry is likely to be genuine.

INQUIRY RESPONSE BLOCK

The Inquiry Response Block (see section 8.13) contains the following components:

- o one Status Component (see section 7.16). This component holds the status information on the inquired transaction,
- o zero or one Payment Scheme Components. These contain encapsulated payment scheme specific inquiry messages for inquiries on payment.

SIGNATURE BLOCK (INQUIRY RESPONSE)

If a signature block is present on the message containing the Inquiry Response Block then it may be checked by the receiver of the block to determine if the Inquiry Response is valid.

If present, the Inquiry Response Signature Block (see section 8.13) contains the following components:

- o one Signature Component (see section 7.19)
- o one or more Certificate Components, if required.

Note: Digital signatures on an Inquiry Response is only likely to occur if the recipient of the response expects the Inquiry Request to be signed. In this version of IOTP this will require some kind of pre-existing agreement. This means that:

- o Consumers are unlikely to generate responses with signatures, although it is not an error if they do
- o the other trading roles may agree that digital signatures are required. For example a Merchant may require that an Inquiry Response is digitally signed by the Payment Handler so that they can check that the request response is valid.

9.2.2 Baseline Ping IOTP Transaction

The purpose of the Baseline IOTP Ping Transaction is to test basic connectivity between the Trading Roles that may take part in an IOTP Transaction.

It enables IOTP aware application software to:

- o determine if the IOTP aware application at another Trading Role is operating, and

- o verify whether or not the two trading roles signatures can be processed.

For example it can be used by a Merchant to determine if a Payment Handler or Delivery Handler is up and running prior to starting a Purchase transaction that uses those trading roles.

The Trading Blocks used by the Baseline Ping IOTP Transaction are:

- o a Ping Request Block (see section 8.14)

- o a Ping Response Block (see section 8.15), and

- o a Signature Block (see section 8.16).

PING MESSAGES

The following figure outlines the message flows in the Baseline IOTP Ping Transaction.

```
 *+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*+*
    1st Role
     |  2nd Role
STEP |     |
 1.          The IOTP Aware Application in the first Trading Role
             decides to check whether the counterparty IOTP
             application is up and running. It generates a Ping
             Request Block and optional Signature Block and sends them
             to the second trading role.

     1 --> 2 PING REQUEST. IotpMsg: Trans Ref Block; Signature Block
             (Optional); Ping Request Block

 2.          The second Trading Role which receives the Ping Request
             Block generates a Ping Response Block and sends it back
             to the sender of the original Ping Request with a
             signature block if required.

     1 <-- 2 PING Response. IotpMsg: Trans Ref Block; Signature Block
             (Optional); Ping Response Block

 3.          The first Trading Role checks the Ping Response Block and
             takes appropriate action, if necessary

 *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Figure 33 Baseline Ping Messages

The verification that signatures can be handled is indicated by the sender of the Ping Request Block including:

- o Organisation Components that identify itself and the intended recipient of the Ping Request Block, and
- o a Signature Block that signs data in the Ping Request.

In this way the receiver of the Ping Request:

- o knows who is sending the Ping Request and can therefore verify the Signature on the Request, and
- o knows who to generate a signature for on the Ping Response.

Note that a Ping Request:

- o does not affect any on-going transaction

- o does NOT initiate an IOTP transaction, unlike other IOTP transaction messages such as TPO or Transaction Status Inquiry.

All IOTP aware applications must return a Ping Response message to the sender of a Ping Request message when it is received.

A Baseline IOTP Ping request can also contain an optional Signature Block. IOTP aware applications can, for example, use the Signature Block to check the recipient of a Ping Request can successfully process and check signatures it has received.

For each Baseline Ping IOTP Transaction, each IOTP role shall establish a different transport session from other IOTP transactions.

Any IOTP Trading Role can send a Ping request to any other IOTP Trading Role at any time it wants. A Ping message has its own IotpTransId, which is different from other IOTP transactions.

The remainder of this sub-section on the Baseline Ping IOTP Transaction defines the contents of each Trading Block.

TRANSACTION REFERENCE BLOCK

The IotpTransId of a Ping transaction should be different from any other IOTP transaction.

PING REQUEST BLOCK

If the Ping Transaction is anonymous then no Organisation Components are included in the Ping Request Block (see section 8.7).

If the Ping Transaction is not anonymous then the Ping Request Block contains Organisation Components for:

- o the sender of the Ping Request Block, and

- o the verifier of the Signature Component

If Organisation Components are present, then it indicates that the sender of the Ping Request message has generated a Signature Block. The signature block must be verified by the Trading Role that receives the Ping Request Block.

SIGNATURE BLOCK (PING REQUEST)

The Ping Request Signature Block (see section 8.16) contains the following components:

- one Signature Component (see section 7.19)
- one or more Certificate Components, if required.

PING RESPONSE BLOCK

The Ping Response Block (see section 8.15) contains the following component:

- the Organisation Component of the sender of the Ping Response message

If the Ping Transaction is not anonymous then the Ping Response additionally contains:

- copies of the Organisation Components contained in the Ping Request Block.

SIGNATURE BLOCK (PING RESPONSE)

The Ping Response Signature Block (see section 8.16) contains the following components:

- one Signature Component (see section 7.19)
- one or more Certificate Components, if required.

10. Retrieving Logos

This section describes how to retrieve logos for display by IOTP aware software using the Logo Net Locations attribute contained in the Brand Element (see section 7.7.1) and the Organisation Component (see section 7.6).

The full address of a logo is defined as follows: Logo_address ::= Logo_net_location "/" Logo_size Logo_color_depth ".gif"

Where:

- Logo_net_location is obtained from the LogoNetLocn attribute in the Brand Element (see section 7.7.1) or the Organisation Component. Note that:
  - the content of this attribute is dependent on the Transport Mechanism (such as HTTP) that is used. See the Transport Mechanism supplement,

  - implementers should check that if the rightmost character of Logo Net Location is set to right-slash "/" then another, right slash should not be included when generating the Logo Address,

- Logo_size identifies the size of the logo,
- Logo_color_depth identifies the colour depth of the logo
- "gif" indicates that the logos are in "gif" format

Logo_size and Logo_color_depth are specified by the implementer of the IOTP software that is retrieving the logo depending on the size and colour that they want to use.

10.1 Logo Size

There are five standard sizes for logos. The sizes in pixels and the corresponding values for Logo Size are given in the table below.

Size in Pixels	Logo Size Value
32 x 32 or 32 x 20	exsmall
53 x 33	small
103 x 65	medium
180 x 114	large
263 x 166	exlarge

10.2 Logo Color Depth

There are three standard colour depths. The colour depth (including bits per pixel) and the corresponding value for Logo_Color_Depth are given in the table below.

Color Depth (bits per pixel)	Logo Color Depth Value
4 (16 colors)	4
8 (256 colors)	nothing
24 (16 million colors)	24

Note that if Logo Color Depth is omitted then a logo with the default colour depth of 256 colours will be retrieved.

10.3 Logo Net Location Examples

If Logo Net Location was set to "ftp://logos.xzpay.com", then:

- "ftp://logos.xzpay.com/medium.gif" would retrieve a medium size 256 colour logo

- "http://logos.xzpay.com/small4.gif" would retrieve a small size 16 colour logo

Note: Organisations which make logos available for use with IOTP should always make available "small" and "medium" size logos and use the "gif" format.

11. Brands

This section contains:

- a definition of Brands and an outline of Brand Selection using Brand Lists, and

- some XML examples of Brand Lists

11.1 Brand Definitions and Brand Selection

One of the key features of IOTP is the ability for a merchant to offer a list of Brands from which a consumer may make a selection. This section provides an overview of what is involved and provides guidance on how selection of a brand and associated payment instrument can be carried out by a Consumer. It covers:

- definitions of Payment Instruments and Brands - what are Payment Instruments and Brands in an IOTP context. Further categorises Brands as optionally a "Dual Brand" or a "Promotional Brand",

- identification and selection of Promotional Brands - Promotional Brands offer a Consumer some additional benefit, for example loyalty points or a discount. This means that both Consumers and Merchant must be able to correctly identify that a valid Promotional Brand is being used.

Also see the following sections:

- Brand List Component (section 7.7) which contains definitions of the XML elements which contain the list of Brands offered by a Merchant to a Consumer, and

- Brand Selection Component (section 7.8) for details of how a Consumer records the Brand, currency, amount and payment protocol that was selected.

11.1.1 Definition of Payment Instrument

A Payment Instrument is the means by which a Consumer pays for goods or services offered by a Merchant. It can be, for example:

- a credit card such as MasterCard or Visa;

- a debit card such as MasterCard’s Maestro;

- a smart card based electronic cash payment instrument such as a Mondex Card, a GeldKarte card or a Visa Cash card

- a software based electronic payment account such as a CyberCash or DigiCash account.

Most Payment Instruments have a number, typically an account number, by which the Payment Instrument can be identified.

11.1.2 Definition of Brand

A Brand is the mark which identifies a particular type of Payment Instrument. A list of Brands are the payment options which are presented by the Merchant to the Consumer and from which the Consumer makes a selection. Each Brand may have a different Payment Handler. Examples of Brands include:

- payment association and proprietary Brands, for example MasterCard, Visa, American Express, Diners Club, Mondex, GeldKarte, CyberCash, etc.

- promotional brands (see below). These include:

  - store brands, where the Payment Instrument is issued to a Consumer by a particular Merchant, for example Walmart, Sears, or Marks and Spencer (UK)

  - cobrands, for example American Advantage Visa, where an Organisation uses their own brand in conjunction with, typically, a payment association Brand.

11.1.3 Definition of Dual Brand

A Dual Brand means that a single payment instrument may be used as if it were two separate Brands. For example there could be a single Japanese "UC" MasterCard which can be used as either a UC card or a regular MasterCard. The UC card Brand and the MasterCard Brand could each have their own separate Payment Handlers. This means that:

- o the merchant treats, for example "UC" and "MasterCard" as two separate Brands when offering a list of Brands to the Consumer,
- o the consumer chooses a Brand, for example either "UC" or "MasterCard,
- o the consumer IOTP aware application determines which Payment Instrument(s) match the chosen Brand, and selects, perhaps with user assistance, the correct Payment Instrument to use.

Note: Dual Brands need no special treatment by the Merchant and therefore no explicit reference is made to Dual Brands in the DTD. This is because, as far as the Merchant is concerned, each Brand in a Dual Brand is treated as a separate Brand. It is at the Consumer, that the matching of a Brand to a Dual Brand Payment Instrument needs to be done.

11.1.4 Definition of Promotional Brand

A Promotional Brand means that, if the Consumer pays with that Brand, then the Consumer will receive some additional benefit which can be received in two ways:

- o at the time of purchase. For example if a Consumer pays with a "Walmart MasterCard" at a Walmart web site, then a 5% discount might apply, which means the consumer actually pays less,
- o from their Payment Instrument (card) issuer when the payment appears on their statement. For example loyalty points in a frequent flyer scheme could be awarded based on the total payments made with the Payment Instrument since the last statement was issued.

Note that:

- o the first example (obtaining the benefit at the time of purchase), requires that:
  - - the Consumer is informed of the benefits which arise if that Brand is selected

  - if the Brand is selected, the Merchant changes the relevant IOTP Components in the Offer Response to reflect the correct amount to be paid

- the second (obtaining a benefit through the Payment Instrument issuer) does not require that the Offer Response is changed

- each Promotional Brand should be identified as a separate Brand in the list of Brands offered by the Merchant. For example: "Walmart", "Sears", "Marks and Spencer" and "American Advantage Visa", would each be a separate Brand.

11.1.5 Identifying Promotional Brands

There are two problems which need to handled in identifying Promotional Brands:

- how does the Merchant or their Payment Handler positively identify the promotional brand being used at the time of purchase

- how does the Consumer reliably identify the correct promotional brand from the Brand List presented by the Merchant

The following is a description of how this could be achieved.

Note: Please note that the approach described here is a model approach that solves the problem. Other equivalent methods may be used.

11.1.5.1 Merchant/Payment Handler Identification of Promotional Brands

Correct identification that the Consumer is paying using a Promotional Brand is important since a Consumer might fraudulently claim to have a Promotional Brand that offers a reduced payment amount when in reality they do not.

Two approaches seem possible:

- use some feature of the Payment Instrument or the payment method to positively identify the Brand being used. For example, the SET certificate for the Brand could be used, if one is available, or

- use the Payment Instrument (card) number to look up information about the Payment Instrument on a Payment Instrument issuer database to determine if the Payment Instrument is a promotional brand.

Note that:

- o the first assumes that SET is available.

- o the second is only possible if the Merchant, or alternatively the Payment Handler, has access to card issuer information.

IOTP does not provide the Merchant with Payment Instrument information (e.g., a card or account number). This is only sent as part of the encapsulated payment protocol to a Payment Handler. This means that:

- o the Merchant would have to assume that the Payment Instrument selected was a valid Promotional Brand, or

- o the Payment Handler would have to check that the Payment Instrument was for the valid Promotional Brand and fail the payment if it was not.

A Payment Handler checking that a brand is a valid Promotional Brand is most likely if the Payment Handler is also the Card Issuer.

11.1.5.2 Consumer Selection of Promotional Brands

Two ways by which a Consumer can correctly select a Promotional Brand are:

- o the Consumer visually matching a logo for the Promotional Brand which has been provided to the Consumer by the Merchant,

- o the Consumer’s IOTP aware application matching a code for the Promotional Brand which the application has registered against a similar code contained in the list of Brands offered by the Merchant.

In the latter case, the code contained in the Consumer wallet must match exactly the code in the list offered by the Merchant otherwise no match will be found. Ways in which the Consumer’s IOTP Aware Application could obtain such a code include:

- o the Consumer types the code in directly. This is error prone and not user friendly, also the consumer needs to be provided with the code. This approach is not recommended,

- o using one of the Brand Identifiers defined by IOTP and pre-loaded into the Consumers IOTP Aware application or wallet by the developer of the Wallet,

- o using some information contained in the software or other data associated with the Payment Instrument. This could be:

  - a SET certificate for Brands which use this payment method

  - a code provided by the payment software which handles the particular payment method, this could apply to, for example, GeldKarte, Mondex, CyberCash and DigiCash,

- o the consumer making an initial "manual" link between a Promotional Brand in the list of Brands offered by the Merchant and an individual Payment Instrument, the first time the promotional brand is used. The IOTP Aware application would then "remember" the code for the Promotional Brand for use in future purchases.

11.1.5.3 Consumer Software Brand Id recommendation

New Brand Ids are allocated under IANA procedures (see section 12 IANA Considerations). Which also contains an initial list of Brand Identifiers.

It is recommended that implementers of consumer IOTP aware applications (e.g., software wallets) pre-load their software with the then current set of Brand Ids and provide a method by which they can be updated. For example, by going to the software developer’s web site.

11.2 Brand List Examples

This example contains three examples of the XML for a Brand List Component. It covers:

- o a simple credit card based example

- o a credit card based brand list including promotional credit card brands, and

- o a complex electronic cash based brand list

Note that:

- o brand lists can be as complex or as simple as required

- o all example techniques described in this appendix can be included in one brand list.

11.2.1 Simple Credit Card Based Example

This is a simple example involving:

- o only major credit card payment brands
- o a single price in a single currency
- o a single Payment Handler, and
- o a single payment protocol

```
<BrandList ID='M1.2'
  XML:Lang='us-en'
  ShortDesc='Purchase book including s&h'
  PayDirection='Debit' >
  <Brand ID ='M1.30'
    BrandId='MasterCard'
    BrandName='MasterCard Credit'
    BrandLogoNetLocn='ftp://otplogos.mastercard.com/mastercardcredit'
    ProtocolAmountRefs='M1.33'>
  </Brand>
  <Brand ID ='M.31'
    BrandId='Visa'
    BrandName='Visa Credit'
    BrandLogoNetLocn='ftp://otplogos.visa.com/visacredit'
    ProtocolAmountRefs='M1.33'>
  </Brand>
  <Brand ID ='M1.32'
    BrandId='AmericanExpress'
    BrandName='American Express'
    BrandLogoNetLocn='ftp://otplogos.amex.com'
    ProtocolAmountRefs ='M1.33' >
  </Brand >
  <ProtocolAmount ID ='M1.33'
    PayProtocolRef='M1.35'
    CurrencyAmountRefs='M1.34'>
  </ProtocolAmount>
  <CurrencyAmount ID ='M1.34'
    Amount='10.95'
    CurrCode='USD'/>
  <PayProtocol ID ='M1.35'
    ProtocolId='SCCD1.0'
    ProtocolName='Secure Channel Credit/Debit'
    PayReqNetLocn='http://www.example.com/etill/sccd1' >
  </PayProtocol>
</BrandList>
```

11.2.2 Credit Card Brand List Including Promotional Brands

An example of a Credit Card based Brand List follows. It includes:

- o two ordinary card association brands and two promotional credit card brands. The promotional brands consist of one loyalty based (British Airways MasterCard) which offers additional loyalty points and one store based (Walmart) which offers a discount on purchases over a certain amount

- o two payment protocols:

  - SET (Secure Electronic Transactions) see [SET], and

  - SCCD (Secure Channel Credit Debit) see [SCCD].

```
 <BrandList ID='M1.2'
   XML:Lang='us-en'
   ShortDesc='Purchase ladies coat'
   PayDirection='Debit' >
   <Brand ID ='M1.3'
     BrandId='MasterCard'
     BrandName='MasterCard Credit'
     BrandLogoNetLocn='ftp://otplogos.mastercard.com'
     ProtocolAmountRefs='M1.7 M1.8'>
     <ProtocolBrand ProtocolId='SET1.0' ProtocolBrandId='MasterCard:'>
     </ProtocolBrand>
   </Brand>
   <Brand ID ='M1.4'
     BrandId='Visa'
     BrandName='Visa Credit'
     BrandLogoNetLocn='ftp://otplogos.visa.com'
     ProtocolAmountRefs='M1.7 M1.8'>
     <ProtocolBrand ProtocolId='SET1.0' ProtocolBrandId='Visa:'>
     </ProtocolBrand>
   </Brand>
   <Brand ID ='M1.5'
     BrandId='BritishAirwaysMC'
     BrandName='British Airways MasterCard'
     BrandLogoNetLocn='ftp://otplogos.britishairways.co.uk'
     BrandNarrative='Double air miles with British Airways MasterCard'
     ProtocolAmountRefs ='M1.7 M1.8' >
     <ProtocolBrand ProtocolId='SET1.0' ProtocolBrandId='MasterCard:BA'>
     </ProtocolBrand>
   </Brand >
   <Brand ID ='M1.6'
     BrandId='Walmart'
     BrandName='Walmart Store Card'
     BrandLogoNetLocn='ftp://otplogos.walmart.com'

     BrandNarrative='5% off with your Walmart Card
                  on purchases over $150'
     ProtocolAmountRefs='M1.8'>
   </Brand>
   <ProtocolAmount ID ='M1.7'
     PayProtocolRef='M1.10'
     CurrencyAmountRefs='M1.9' >
     <PackagedContent Transform="BASE64">
        238djqw1298erh18dhoire
     </PackagedContent>
   </ProtocolAmount>
   <ProtocolAmount ID ='M1.8'
     PayProtocolRef='M1.11'
     CurrencyAmountRefs='M1.9' >
     <PackagedContent Transform="BASE64">
        238djqw1298erh18dhoire
     </PackagedContent>
   </ProtocolAmount>
   <CurrencyAmount ID ='M1.9'
     Amount='157.53'
     CurrCode='USD'/>
   <PayProtocol ID ='M1.10'
     ProtocolId='SET1.0'
     ProtocolName='Secure Electronic Transaction Version 1.0'
     PayReqNetLocn='http://www.example.com/etill/set1' >
     <PackagedContent Transform="BASE64">
       8ueu26e482hd82he82
     </PackagedContent>
   </PayProtocol>
   <PayProtocol ID ='M1.11'
     ProtocolId='SCCD1.0'
     ProtocolName='Secure Channel Credit/Debit'
     PayReqNetLocn='http://www.example.com/etill/sccd1' >
     <PackagedContent Transform="BASE64">
        82hd82he8226e48ueu
     </PackagedContent>
   </PayProtocol>
  </BrandList>
```

11.2.3 Brand Selection Example

In order to pay by 'British Airways' MasterCard using the example above using SET and therefore getting double air miles, the Brand Selection would be:

```
   <BrandSelection ID='C1.2'
```

```
     BrandListRef='M1.3'
     BrandRef='M1.5'
     ProtocolAmountRef='M1.7'
     CurrencyAmountRef='M1.9' >
   </BrandSelection>
```

11.2.4 Complex Electronic Cash Based Brand List

The following is an fairly complex example which includes:

- o payments using either Mondex, GeldKarte, CyberCash or DigiCash
- o in currencies including US dollars, British Pounds, Italian Lira, German Marks and Canadian Dollars
- o a discount on the price if the payment is made in Mondex using British pounds or US dollars, and
- o more than one Payment Handler is used for payments involving Mondex or CyberCash
- o support for more than one version of a CyberCash CyberCoin payment protocol.

```
   <BrandList ID='M1.2'
     XML:Lang='us-en'
     ShortDesc='Company report on XYZ Co'
     PayDirection='Debit' >
     <Brand ID ='M1.13'
       BrandId='Mondex'
       BrandName='Mondex Electronic Cash'
       BrandLogoNetLocn='ftp://otplogos.mondex.com'
       ProtocolAmountRefs='M1.17 M1.18'>
     </Brand>
     <Brand ID ='M1.14'
       BrandId='GeldKarte'
       BrandName='GeldKarte Electronic Cash'
       BrandLogoNetLocn='ftp://otplogos.geldkarte.co.de'
       ProtocolAmountRefs='M1.19'>
     </Brand>
     <Brand ID ='M1.15'
       BrandId='CyberCoin'
       BrandName='CyberCoin Eletronic Cash'
       BrandLogoNetLocn='http://otplogos.cybercash.com'
       ProtocolAmountRefs ='M1.20' >
     </Brand >
     <Brand ID ='M1.16'
       BrandId='DigiCash'
       BrandName='DigiCash Electronic Cash'
       BrandLogoNetLocn='http://otplogos.digicash.com'
       BrandNarrative='5% off with your Walmart Card
                    on purchases over $150'
       ProtocolAmountRefs='M1.22'>
     </Brand>
     <ProtocolAmount ID ='M1.17'
       PayProtocolRef='M1.31'
       CurrencyAmountRefs='M1.25 M1.29'>
     </ProtocolAmount>
     <ProtocolAmount ID ='M1.18'
       PayProtocolRef='M1.32'
       CurrencyAmountRefs='M1.26 M1.27 M1.28 M1.30'>
     </ProtocolAmount>
     <ProtocolAmount ID ='M1.19'
       PayProtocolRef='M1.35'
       CurrencyAmountRefs='M1.28'>
     </ProtocolAmount>
     <ProtocolAmount ID ='M1.20'
       PayProtocolRef='M1.34 M1.33'
       CurrencyAmountRefs='M1.23 M1.24 M1.27 M1.28 M1.29 M1.30'>
     </ProtocolAmount>
     <ProtocolAmount ID ='M1.21'
       PayProtocolRef='M1.36'
       CurrencyAmountRefs='M1.23 M1.24 M1.27 M1.28 M1.29 M1.30'>
     </ProtocolAmount>
     <CurrencyAmount ID ='M1.23'
       Amount='20.00'
       CurrCode='USD'/>
     <CurrencyAmount ID ='M1.24'
       Amount='12.00'
       CurrCode='GBP'/>
     <CurrencyAmount ID ='M1.25'
       Amount='19.50'
       CurrCode='USD'/>
     <CurrencyAmount ID ='M1.26'
       Amount='11.75'
       CurrCode='GBP'/>
     <CurrencyAmount ID ='M1.27'
       Amount='36.00'
       CurrCode='DEM'/>
     <CurrencyAmount ID ='M1.28'
       Amount='100.00'
       CurrCode='FFR'/>
     <CurrencyAmount ID ='M1.29'
       Amount='22.00'
       CurrCode='CAD'/>
     <CurrencyAmount ID ='M1.30'
```

```
       Amount='15000'
       CurrCode='ITL'/>
     <PayProtocol ID ='M1.31'
       ProtocolId='MXv1.0'
       ProtocolName='Mondex IOTP Protocol Version 1.0'
       PayReqNetLocn='http://www.mxbankus.com/etill/mx' >
     </PayProtocol>
     <PayProtocol ID ='M1.32'
       ProtocolId='MXv1.0'
       ProtocolName='Mondex IOTP Protocol Version 1.0'
       PayReqNetLocn='http://www.mxbankuk.com/vserver' >
     </PayProtocol>
     <PayProtocol ID ='M1.33'
       ProtocolId='Ccashv1.0'
       ProtocolName='CyberCoin Version 1.0'
       PayReqNetLocn='http://www.cybercash.com/ccoin' >
     </PayProtocol>
     <PayProtocol ID ='M1.34'
       ProtocolId='CCashv2.0'
       ProtocolName='CyberCoin Version 2.0'
       PayReqNetLocn='http://www.cybercash.com/ccoin' >
     </PayProtocol>
     <PayProtocol ID ='M1.35'
       ProtocolId='GKv1.0'
       ProtocolName='GeldKarte Version 1.0'
       PayReqNetLocn='http://www.example.com/pgway' >
     </PayProtocol>
     <PayProtocol ID ='M1.36'
       ProtocolId='DCashv1.0'
       ProtocolName='DigiCash Protocol Version 1.0'
       PayReqNetLocn='http://www.example.com/digicash' >
     </PayProtocol>
     </BrandList>
```

12. IANA Considerations

This section describes the codes that are controlled by IANA, and also how new codes can be created for testing purposes that are not controlled by IANA.

12.1 Codes Controlled by IANA

To help ensure interoperability, there is a need for codes used by IOTP to be maintained in a controlled environment so that their meaning and usage are well defined and duplicate codes avoided. [IANA] is the mechanism to be used for this purpose as described in RFC 2434.

The element types and attributes names to which this procedure applies is shown in the table below together with the initial values that are valid for these attributes.

Note that:

- o the IETF Trade mailing list's email address is ietf-trade@elistx.com
- o "Designated Experts" (see [IANA]) are appointed by the IESG.

Element Type/ Attribute Name	Attribute Values
Algorithm/ Name (When Algorithm is a child of an AuthReq Component)	"sha1" - indicates that a [SHA1] authentication will apply "signature" - indicates that authentication consists of the generation of a digital signature. "Pay:ppp" where "ppp" may be set to any valid value for "iotpbrand" (see below) With the exception of Algorithms that begin with "pay:", new values are allocated following review on the IETF Trade mailing list and by the Designated Expert.

Note: The Algorithm element is likely to be eventually defined within the [DSIG] name space. It is likely that the maintenance procedure defined here may need to vary over time, as the DSIG proposals become more widely adopted.

Element Type/ Attribute Name	Attribute Values
Brand/BrandId	The following list of initial BrandIds have been taken from those Organisations that have applied for SET certificates as at 1st June 1999: "Amex" - American Express "Dankort" - Dankort "JCB" - JCB "Maestro" - Maestro

	"MasterCard" - MasterCard "NICOS" - NICOS "VISA" - Visa In addition the following Brand Id values are defined: "Mondex" "GeldKarte" New values of BrandId must be announced to the IETF Trade mailing list and, if there are no objections within three weeks, are allocated on a "first come first served" basis.
CurrencyAmount/ CurrCode	Currency codes are dependent on CurrCodeType (see below). If CurrCodeType is "ISO4217-A" then the currency code is an alphabetic currency code as defined by [ISO4217]. If CurrCodeType is "IOTP" then new values must be announced to the IETF Trade mailing list and, if there are no objections within three weeks, are allocated on a "first come first served" basis.

Note: The Currency Code Type of IOTP, is designed to allow the support of "new" psuedo currencies such as loyalty or frequent flyer points. At the time of writing this specification, no currency codes of this type have been defined.

Element Type/ Attribute Name	Attribute Values
CurrencyAmount/ CurrCodeType	"ISO4217-A" "IOTP" New values of CurrCodeType attribute are allocated following review on the IETF Trade mailing list and by the Designated Expert.
DeliveryData/ DelivMethod	"Post"

	"Web"
	"Email"
	New values of Delivery Method attribute are allocated following review on the IETF Trade mailing list and by the Designated Expert. This may require the publication of additional documentation to describe how the delivery method is used.
PackagedContent/ Content	"PCDATA"
	"MIME"
	"MIME:mimetype" (where mimetype must be the same as content-type as defined by [MIME])
	"XML"
	If the Content attribute is of the form "MIME"mimetype", then control of new values for "mimetype" is as defined in [MIME].
	Otherwise, new values of the Content attribute are allocated following review on the IETF Trade mailing list and by the Designated Expert. This may require the publication of additional documentation to describe how the new attribute is used within a Packaged Content element.
RelatedTo/ RelationshipType	"IotpTransaction"
	"Reference"
	New values of the RelationshipType attribute are allocated following review on the IETF Trade Working Group mailing list and by the Designated Expert. This may require the publication of additional documentation to describe how the
Element Type/ Attribute Name	Attribute Values
	delivery method is used.
Status/ StatusType	Offer
	Payment

```
                       Delivery

                       Authentication

                       Unidentified

                       New values of the Status Type attribute are
                       allocated following:
                        o publication to the IETF Trade Working Group,
                          of an RFC describing the Trading Exchange,
                          Trading Roles and associated components that
                          relate to the Status, and
                        o review of the document on the IETF Trade
                          mailing list and by the Designated Expert.

   Note: The document describing new values for the Status Type
   attribute may be combined with documents that describe new Trading
   Roles and types of signatures (see below).

   TradingRole/        "Consumer"
   TradingRole
                       "Merchant"

                       "PaymentHandler"

                       "DeliveryHandler"

                       "DelivTo"

                       "CustCare"

                       New values of the Trading Role attribute are
                       allocated following:
                        o publication to the IETF Trade Working Group,
                          of an RFC describing the Trading Exchange,
                          Trading Roles and associated components that
                          relate to the Trading Role, and
                        o review of the document on the IETF Trade
                          mailing list and by the Designated Expert.

   Note: The document describing new values for the Trading Role
   attribute may be

     Element Type/                    Attribute Values
     Attribute Name
                                  combined with documents that describe
                                  new Status Types (see above) and
                                  types of signatures (see below).
```

TransId/ IotpTransType	"BaselineAuthentication"
	"BaselineDeposit"
	"BaselinePurchase"
	"BaselineRefund"
	"BaselineWithdrawal"
	"BaselineValueExchange"
	"BaselineInquiry"
	"BaselinePing"
	New values of the IotpTransType attribute are allocated following: o publication to the IETF Trade mailing list, of an RFC describing the new IOTP Transaction, and o review of the document on the IETF Trade Working Group mailing list and by the Designated Expert.
Attribute/ Content (see Signature Component)	"OfferResponse"
	"PaymentResponse"
	"DeliveryResponse"
	"AuthenticationRequest"
	"AuthenticationResponse"
	"PingRequest"
	"PingResponse"
	New values of the code that define the type of a signature are allocated following: o publication to the IETF Trade Working Group, of an RFC describing the Trading Exchange where the signature is being used, and o review of the document on the IETF Trade mailing list and by the Designated Expert.

Element Type/ Attribute Name	Attribute Values

Note: The document describing new values for the types of signatures may be combined with documents that describe new Status Types and Trading Roles (see above).

12.2 Codes not controlled by IANA

In addition to the formal development and registration of codes as described above, there is still a need for developers to experiment using new IOTP codes. For this reason, "user defined codes" may be used to identify additional values for the codes contained within this specification without the need for them to be registered with IANA.

The definition of a user defined code is as follows:

```
user_defined_code ::= ( "x-" | "X-" ) NameChar (NameChar)*

  NameChar           NameChar has the same definition as the [XML]
                     definition of NameChar
```

Use of domain names (see [DNS]) to make user defined codes unique is recommended although this method cannot be relied upon.

13. Internet Open Trading Protocol Data Type Definition

This section contains the XML DTD for the Internet Open Trading Protocols.

```
   <!--
   ******************************************************
   *                                                    *
   * INTERNET OPEN TRADING PROTOCOL VERSION 1.0 DTD     *
   * Filename: ietf.org/rfc/rfc2801.dtd                 *
   *                                                    *
   * Changes from version 07 (iotp-v1.0-protocol-07.dtd)*
   *   - NO CHANGES                                     *
   *                                                    *
   *                                                    *
   *                                                    *
   *                                                    *
   * Copyright Internet Engineering Task Force 1998-2000*
   *                                                    *
   ******************************************************

   ******************************************************
   * IOTP MESSAGE DEFINITION                            *
   ******************************************************
    -->

   <!ELEMENT IotpMessage
      ( TransRefBlk,
        IotpSignatures?,
        ErrorBlk?,
        ( AuthReqBlk |
          AuthRespBlk |
          AuthStatusBlk |
          CancelBlk |
          DeliveryReqBlk |
          DeliveryRespBlk |
          InquiryReqBlk |
          InquiryRespBlk |
          OfferRespBlk |
          PayExchBlk |
          PayReqBlk |
          PayRespBlk |
          PingReqBlk |
          PingRespBlk |
          TpoBlk |
          TpoSelectionBlk
        )*
      ) >
   <!ATTLIST IotpMessage
     xmlns             CDATA
      'iotp:ietf.org/iotp-v1.0' >
   <!--
   ****************************************************
   * TRANSACTION REFERENCE BLOCK DEFINITION           *
   ****************************************************
    -->

   <!ELEMENT TransRefBlk (TransId, MsgId, RelatedTo*) >
   <!ATTLIST TransRefBlk
    ID                  ID      #REQUIRED >


   <!ELEMENT TransId EMPTY >
   <!ATTLIST TransId
    ID                  ID      #REQUIRED
    Version             NMTOKEN #FIXED '1.0'
    IotpTransId         CDATA   #REQUIRED
    IotpTransType       CDATA   #REQUIRED
    TransTimeStamp      CDATA   #REQUIRED >


   <!ELEMENT MsgId EMPTY >
   <!ATTLIST MsgId
    ID                  ID      #REQUIRED
    RespIotpMsg         NMTOKEN #IMPLIED
    xml:lang            NMTOKEN #REQUIRED
    LangPrefList        NMTOKENS #IMPLIED
    CharSetPrefList     NMTOKENS #IMPLIED
    SenderTradingRoleRef NMTOKEN #IMPLIED
    SoftwareId          CDATA   #REQUIRED
    TimeStamp           CDATA   #IMPLIED >


   <!ELEMENT RelatedTo (PackagedContent) >
   <!ATTLIST RelatedTo
    ID                  ID      #REQUIRED
    xml:lang            NMTOKEN #REQUIRED
    RelationshipType    NMTOKEN #REQUIRED
    Relation            CDATA   #REQUIRED
    RelnKeyWords        NMTOKENS #IMPLIED >



   <!--
   ****************************************************
   * Packaged Content Common Element                  *
   ****************************************************
    -->
   <!ELEMENT PackagedContent (#PCDATA) >
   <!ATTLIST PackagedContent
    Name             CDATA     #IMPLIED
    Content          NMTOKEN   "PCDATA"
    Transform (NONE|BASE64)    "NONE" >

   <!--
   ******************************************************
   * TRADING COMPONENTS                                 *
   ******************************************************
    -->
   <!-- PROTOCOL OPTIONS COMPONENT -->
   <!ELEMENT ProtocolOptions EMPTY >
   <!ATTLIST ProtocolOptions
    ID                 ID      #REQUIRED
    xml:lang           NMTOKEN #REQUIRED
    ShortDesc          CDATA   #REQUIRED
    SenderNetLocn      CDATA   #IMPLIED
    SecureSenderNetLocn CDATA  #IMPLIED
    SuccessNetLocn     CDATA   #REQUIRED >


   <!-- AUTHENTICATION DATA COMPONENT -->
   <!ELEMENT AuthReq (Algorithm, PackagedContent*)>
   <!ATTLIST AuthReq
    ID                 ID      #REQUIRED
    AuthenticationId   CDATA   #REQUIRED
    ContentSoftwareId  CDATA   #IMPLIED >


   <!-- AUTHENTICATION RESPONSE COMPONENT -->
   <!ELEMENT AuthResp (PackagedContent*) >
   <!ATTLIST AuthResp
    ID                 ID      #REQUIRED
    AuthenticationId   CDATA   #REQUIRED
    SelectedAlgorithmRef NMTOKEN #REQUIRED
    ContentSoftwareId  CDATA   #IMPLIED >

   <!-- TRADING ROLE INFO REQUEST COMPONENT -->
   <!ELEMENT TradingRoleInfoReq EMPTY>
   <!ATTLIST TradingRoleInfoReq
    ID                 ID      #REQUIRED
    TradingRoleList    NMTOKENS #REQUIRED >

   <!-- ORDER COMPONENT -->
   <!ELEMENT Order (PackagedContent*) >
   <!ATTLIST Order
    ID                 ID      #REQUIRED
```

```
     xml:lang           NMTOKEN #REQUIRED
     OrderIdentifier    CDATA   #REQUIRED
     ShortDesc          CDATA   #REQUIRED
     OkFrom             CDATA   #REQUIRED
     OkTo               CDATA   #REQUIRED
     ApplicableLaw      CDATA   #REQUIRED
     ContentSoftwareId  CDATA   #IMPLIED >

    <!-- ORGANISATION COMPONENT -->
    <!ELEMENT Org (TradingRole+, ContactInfo?,
         PersonName?, PostalAddress?)>
    <!ATTLIST Org
     ID                 ID      #REQUIRED
     xml:lang           NMTOKEN #REQUIRED
     OrgId              CDATA   #REQUIRED
     LegalName          CDATA   #IMPLIED
     ShortDesc          CDATA   #IMPLIED
     LogoNetLocn        CDATA   #IMPLIED >


    <!ELEMENT TradingRole EMPTY >
    <!ATTLIST TradingRole
     ID      ID#REQUIRED
     TradingRole        NMTOKEN #REQUIRED
     IotpMsgIdPrefix    NMTOKEN #REQUIRED
     CancelNetLocn      CDATA   #IMPLIED
     ErrorNetLocn       CDATA   #IMPLIED
     ErrorLogNetLocn  CDATA           #IMPLIED >


    <!ELEMENT ContactInfo EMPTY >
    <!ATTLIST ContactInfo
     xml:lang           NMTOKEN #IMPLIED
     Tel                CDATA   #IMPLIED
     Fax                CDATA   #IMPLIED
     Email              CDATA   #IMPLIED
     NetLocn            CDATA   #IMPLIED >


    <!ELEMENT PersonName EMPTY >
    <!ATTLIST PersonName
     xml:lang           NMTOKEN #IMPLIED
     Title              CDATA   #IMPLIED
     GivenName          CDATA   #IMPLIED
     Initials           CDATA   #IMPLIED
     FamilyName         CDATA   #IMPLIED >
```

```
   <!ELEMENT PostalAddress EMPTY >
   <!ATTLIST PostalAddress
    xml:lang           NMTOKEN #IMPLIED
    AddressLine1       CDATA   #IMPLIED
    AddressLine2       CDATA   #IMPLIED
    CityOrTown         CDATA   #IMPLIED
    StateOrRegion      CDATA   #IMPLIED
    PostalCode         CDATA   #IMPLIED
    Country            CDATA   #IMPLIED
    LegalLocation (True | False) 'False' >


   <!-- BRAND LIST COMPONENT -->
   <!ELEMENT BrandList (Brand+, ProtocolAmount+,
    CurrencyAmount+, PayProtocol+) >
   <!ATTLIST BrandList
    ID                 ID      #REQUIRED
    xml:lang           NMTOKEN #REQUIRED
    ShortDesc          CDATA   #REQUIRED
    PayDirection (Debit | Credit) #REQUIRED >

   <!ELEMENT Brand (ProtocolBrand*, PackagedContent*) >
   <!ATTLIST Brand
    ID                 ID      #REQUIRED
    xml:lang           NMTOKEN #IMPLIED
    BrandId            CDATA   #REQUIRED
    BrandName          CDATA   #REQUIRED
    BrandLogoNetLocn   CDATA   #REQUIRED
    BrandNarrative     CDATA   #IMPLIED
    ProtocolAmountRefs IDREFS  #REQUIRED
    ContentSoftwareId  CDATA   #IMPLIED >

   <!ELEMENT ProtocolBrand (PackagedContent*) >
   <!ATTLIST ProtocolBrand
    ProtocolId         CDATA   #REQUIRED
    ProtocolBrandId    CDATA   #REQUIRED >

   <!ELEMENT ProtocolAmount (PackagedContent*) >
   <!ATTLIST ProtocolAmount
    ID                 ID      #REQUIRED
    PayProtocolRef     IDREF   #REQUIRED
    CurrencyAmountRefs IDREFS  #REQUIRED
    ContentSoftwareId  CDATA   #IMPLIED >

   <!ELEMENT CurrencyAmount EMPTY >
   <!ATTLIST CurrencyAmount
    ID                 ID      #REQUIRED
    Amount             CDATA   #REQUIRED
```

```
     CurrCodeType       NMTOKEN 'ISO4217-A'
     CurrCode           CDATA   #REQUIRED >

    <!ELEMENT PayProtocol (PackagedContent*) >
    <!ATTLIST PayProtocol
     ID                 ID      #REQUIRED
     xml:lang           NMTOKEN #IMPLIED
     ProtocolId         NMTOKEN #REQUIRED
     ProtocolName       CDATA   #REQUIRED
     ActionOrgRef       NMTOKEN #REQUIRED
     PayReqNetLocn      CDATA   #IMPLIED
     SecPayReqNetLocn   CDATA   #IMPLIED
     ContentSoftwareId  CDATA   #IMPLIED >


    <!-- BRAND SELECTION COMPONENT -->
    <!ELEMENT BrandSelection (BrandSelBrandInfo?,
         BrandSelProtocolAmountInfo?,
         BrandSelCurrencyAmountInfo?) >
    <!ATTLIST BrandSelection
     ID                 ID      #REQUIRED
     BrandListRef       NMTOKEN #REQUIRED
     BrandRef           NMTOKEN #REQUIRED
     ProtocolAmountRef  NMTOKEN #REQUIRED
     CurrencyAmountRef  NMTOKEN #REQUIRED >

    <!ELEMENT BrandSelBrandInfo (PackagedContent+) >
    <!ATTLIST BrandSelBrandInfo
     ID                 ID      #REQUIRED
     ContentSoftwareId  CDATA   #IMPLIED >

    <!ELEMENT BrandSelProtocolAmountInfo (PackagedContent+) >
    <!ATTLIST BrandSelProtocolAmountInfo
     ID                 ID      #REQUIRED
     ContentSoftwareId  CDATA   #IMPLIED >

    <!ELEMENT BrandSelCurrencyAmountInfo (PackagedContent+) >
    <!ATTLIST BrandSelCurrencyAmountInfo
     ID                 ID      #REQUIRED
     ContentSoftwareId  CDATA   #IMPLIED >

    <!-- PAYMENT COMPONENT -->
    <!ELEMENT Payment EMPTY >
    <!ATTLIST Payment
     ID                 ID      #REQUIRED
     OkFrom             CDATA   #REQUIRED
     OkTo               CDATA   #REQUIRED
     BrandListRef       NMTOKEN #REQUIRED
     SignedPayReceipt (True | False) #REQUIRED
     StartAfterRefs     NMTOKENS #IMPLIED >


   <!-- PAYMENT SCHEME COMPONENT -->
   <!ELEMENT PaySchemeData (PackagedContent+) >
   <!ATTLIST PaySchemeData
     ID                 ID      #REQUIRED
     PaymentRef         NMTOKEN #IMPLIED
     ConsumerPaymentId  CDATA   #IMPLIED
     PaymentHandlerPayId CDATA  #IMPLIED
     ContentSoftwareId  CDATA   #IMPLIED >


   <!-- PAYMENT RECEIPT COMPONENT -->
   <!ELEMENT PayReceipt (PackagedContent*) >
   <!ATTLIST PayReceipt
     ID                 ID      #REQUIRED
     PaymentRef         NMTOKEN #REQUIRED
     PayReceiptNameRefs NMTOKENS #IMPLIED
     ContentSoftwareId  CDATA   #IMPLIED >


   <!-- PAYMENT NOTE COMPONENT -->
   <!ELEMENT PaymentNote (PackagedContent+) >
   <!ATTLIST PaymentNote
      ID                ID      #REQUIRED
      ContentSoftwareId CDATA   #IMPLIED >


   <!-- DELIVERY COMPONENT -->
   <!ELEMENT Delivery (DeliveryData?, PackagedContent*) >
   <!ATTLIST Delivery
     ID                 ID      #REQUIRED
     xml:lang           NMTOKEN #REQUIRED
     DelivExch          (True | False) #REQUIRED
     DelivAndPayResp    (True | False) #REQUIRED
     ActionOrgRef       NMTOKEN #IMPLIED >

   <!ELEMENT DeliveryData (PackagedContent*) >
   <!ATTLIST DeliveryData
     xml:lang           NMTOKEN #IMPLIED
     OkFrom             CDATA   #REQUIRED
     OkTo               CDATA   #REQUIRED
     DelivMethod        NMTOKEN #REQUIRED
     DelivToRef         NMTOKEN #REQUIRED
     DelivReqNetLocn    CDATA   #IMPLIED
     SecDelivReqNetLocn CDATA   #IMPLIED
```

```
    ContentSoftwareId  CDATA   #IMPLIED >


   <!-- CONSUMER DELIVERY DATA COMPONENT -->
   <!ELEMENT ConsumerDeliveryData EMPTY >
   <!ATTLIST ConsumerDeliveryData
    ID                 ID      #REQUIRED
    ConsumerDeliveryId CDATA   #REQUIRED >


   <!-- DELIVERY NOTE COMPONENT -->
   <!ELEMENT DeliveryNote (PackagedContent+) >
   <!ATTLIST DeliveryNote
    ID                 ID      #REQUIRED
    xml:lang           NMTOKEN #REQUIRED
    DelivHandlerDelivId CDATA  #IMPLIED
    ContentSoftwareId  CDATA   #IMPLIED >


   <!-- STATUS COMPONENT -->
   <!ELEMENT Status EMPTY >
   <!ATTLIST Status
    ID                 ID      #REQUIRED
    xml:lang           NMTOKEN #REQUIRED
    StatusType         NMTOKEN #REQUIRED
    ElRef              NMTOKEN #IMPLIED
    ProcessState (NotYetStarted | InProgress |
        CompletedOk | Failed | ProcessError) #REQUIRED
    CompletionCode     NMTOKEN #IMPLIED
    ProcessReference   CDATA   #IMPLIED
    StatusDesc         CDATA   #IMPLIED >

   <!-- TRADING ROLE DATA COMPONENT -->
   <!ELEMENT TradingRoleData (PackagedContent+) >
   <!ATTLIST TradingRoleData
     ID                ID      #REQUIRED
     OriginatorElRef   NMTOKEN #REQUIRED
     DestinationElRefs NMTOKENS #REQUIRED >

   <!-- INQUIRY TYPE COMPONENT -->
   <!ELEMENT InquiryType EMPTY >
   <!ATTLIST InquiryType
    ID                 ID      #REQUIRED
    Type               NMTOKEN #REQUIRED
    ElRef              NMTOKEN #IMPLIED
    ProcessReference   CDATA   #IMPLIED >
   <!-- ERROR COMPONENT -->
   <!ELEMENT ErrorComp (ErrorLocation+, PackagedContent*) >
   <!ATTLIST ErrorComp
    ID                 NMTOKEN #REQUIRED
    xml:lang           NMTOKEN #REQUIRED
    ErrorCode          NMTOKEN #REQUIRED
    ErrorDesc          CDATA   #REQUIRED
    Severity (Warning|TransientError|HardError) #REQUIRED
    MinRetrySecs       CDATA   #IMPLIED
    SwVendorErrorRef   CDATA   #IMPLIED >


   <!ELEMENT ErrorLocation EMPTY >
   <!ATTLIST ErrorLocation
    ElementType        NMTOKEN #REQUIRED
    IotpMsgRef         NMTOKEN #IMPLIED
    BlkRef             NMTOKEN #IMPLIED
    CompRef            NMTOKEN #IMPLIED
    ElementRef         NMTOKEN #IMPLIED
    AttName            NMTOKEN #IMPLIED >



   <!--
   ******************************************************
   * TRADING BLOCKS                                     *
   ******************************************************
    -->

   <!-- TRADING PROTOCOL OPTIONS BLOCK -->
   <!ELEMENT TpoBlk ( ProtocolOptions, BrandList*, Org* ) >
   <!ATTLIST TpoBlk
    ID                 ID      #REQUIRED >


   <!-- TPO SELECTION BLOCK -->
   <!ELEMENT TpoSelectionBlk (BrandSelection+) >
   <!ATTLIST TpoSelectionBlk
    ID                 ID      #REQUIRED >


   <!-- OFFER RESPONSE BLOCK -->
   <!ELEMENT OfferRespBlk (Status, Order?, Payment*,
                Delivery?, TradingRoleData*) >
   <!ATTLIST OfferRespBlk
    ID                 ID      #REQUIRED >
   <!-- AUTHENTICATION REQUEST BLOCK -->
   <!ELEMENT AuthReqBlk (AuthReq*, TradingRoleInfoReq?) >
   <!ATTLIST AuthReqBlk
    ID                 ID      #REQUIRED >


   <!-- AUTHENTICATION RESPONSE BLOCK -->
   <!ELEMENT AuthRespBlk (AuthResp?, Org*) >
   <!ATTLIST AuthRespBlk
    ID                 ID      #REQUIRED >


   <!-- AUTHENTICATION STATUS BLOCK -->
   <!ELEMENT AuthStatusBlk (Status) >
   <!ATTLIST AuthStatusBlk
    ID                 ID      #REQUIRED >


   <!-- PAYMENT REQUEST BLOCK -->
   <!ELEMENT PayReqBlk (Status+, BrandList, BrandSelection,
        Payment, PaySchemeData?, Org*, TradingRoleData*) >
   <!ATTLIST PayReqBlk
    ID                 ID      #REQUIRED >


   <!-- PAYMENT EXCHANGE BLOCK -->
   <!ELEMENT PayExchBlk (PaySchemeData) >
   <!ATTLIST PayExchBlk
    ID                 ID      #REQUIRED >


   <!-- PAYMENT RESPONSE BLOCK -->
   <!ELEMENT PayRespBlk (Status, PayReceipt?, PaySchemeData?,
        PaymentNote?, TradingRoleData*) >
   <!ATTLIST PayRespBlk
    ID                 ID      #REQUIRED >
   <!-- DELIVERY REQUEST BLOCK -->
   <!ELEMENT DeliveryReqBlk (Status+, Order, Org*, Delivery,
        ConsumerDeliveryData?, TradingRoleData*) >
   <!ATTLIST DeliveryReqBlk
    ID                 ID      #REQUIRED >


   <!-- DELIVERY RESPONSE BLOCK -->
   <!ELEMENT DeliveryRespBlk (Status, DeliveryNote) >
   <!ATTLIST DeliveryRespBlk
    ID                 ID      #REQUIRED >
   <!-- INQUIRY REQUEST BLOCK -->
   <!ELEMENT InquiryReqBlk ( InquiryType, PaySchemeData? ) >
   <!ATTLIST InquiryReqBlk
    ID                 ID      #REQUIRED >


   <!-- INQUIRY RESPONSE BLOCK -->
   <!ELEMENT InquiryRespBlk (Status, PaySchemeData?) >
   <!ATTLIST InquiryRespBlk
    ID                 ID      #REQUIRED
    LastReceivedIotpMsgRef NMTOKEN #IMPLIED
    LastSentIotpMsgRef NMTOKEN #IMPLIED >


   <!-- PING REQUEST BLOCK -->
   <!ELEMENT PingReqBlk (Org*)>
   <!ATTLIST PingReqBlk
    ID                 ID      #REQUIRED>


   <!-- PING RESPONSE BLOCK -->
   <!ELEMENT PingRespBlk (Org+)>
   <!ATTLIST PingRespBlk
    ID                 ID      #REQUIRED
    PingStatusCode (Ok | Busy | Down) #REQUIRED
    SigVerifyStatusCode (Ok | NotSupported | Fail) #IMPLIED
    xml:lang           NMTOKEN #IMPLIED
    PingStatusDesc     CDATA   #IMPLIED>


   <!-- ERROR BLOCK -->
   <!ELEMENT ErrorBlk (ErrorComp+, PaySchemeData*) >
   <!ATTLIST ErrorBlk
    ID                 ID      #REQUIRED >


   <!-- CANCEL BLOCK -->
   <!ELEMENT CancelBlk (Status) >
   <!ATTLIST CancelBlk
    ID                 ID      #REQUIRED >


   <!--
   *****************************************************
   * IOTP SIGNATURES BLOCK DEFINITION                  *
   *****************************************************
   -->
   <!ELEMENT IotpSignatures (Signature+ ,Certificate*) >
   <!ATTLIST IotpSignatures
       ID        ID        #IMPLIED
   >

   <!--
   ******************************************************
   * IOTP SIGNATURE COMPONENT DEFINITION                *
   ******************************************************
   -->

   <!ELEMENT Signature (Manifest, Value+) >
   <!ATTLIST Signature
       ID         ID        #IMPLIED
   >

   <!ELEMENT Manifest
       (       Algorithm+,
               Digest+,
               Attribute*,
               OriginatorInfo,
               RecipientInfo+
       )
   >

   <!ATTLIST Manifest
       LocatorHRefBase       CDATA             #IMPLIED
   >

   <!ELEMENT Algorithm (Parameter*) >
   <!ATTLIST Algorithm
       ID                    ID                 #REQUIRED
       type            (digest|signature)      #IMPLIED
       name                  NMTOKEN           #REQUIRED
   >

   <!ELEMENT Digest (Locator, Value) >
   <!ATTLIST Digest
       DigestAlgorithmRef    IDREF             #REQUIRED
   >

   <!ELEMENT Attribute ( ANY ) >
   <!ATTLIST Attribute
       type                   NMTOKEN           #REQUIRED
       critical            ( true | false )     #REQUIRED
   >

   <!ELEMENT OriginatorInfo ANY >
   <!ATTLIST OriginatorInfo
       OriginatorRef           NMTOKEN          #IMPLIED
   >

   <!ELEMENT RecipientInfo ANY >
   <!ATTLIST RecipientInfo
       SignatureAlgorithmRef   IDREF            #REQUIRED
       SignatureValueRef       IDREF            #IMPLIED
       SignatureCertRef        IDREF            #IMPLIED
       RecipientRefs           NMTOKENS         #IMPLIED
   >

   <!ELEMENT KeyIdentifier EMPTY>
   <!ATTLIST KeyIdentifier
       value                    CDATA           #REQUIRED
   >

   <!ELEMENT Parameter ANY >
   <!ATTLIST Parameter
       type                     CDATA           #REQUIRED
   >


   <!--
   ******************************************************
   * IOTP CERTIFICATE COMPONENT DEFINITION              *
   ******************************************************
   -->

   <!ELEMENT Certificate
    (  IssuerAndSerialNumber,  ( Value | Locator ) )
   >

   <!ATTLIST Certificate
       ID                        ID                #IMPLIED
       type                      NMTOKEN           #REQUIRED
   >

   <!ELEMENT IssuerAndSerialNumber EMPTY >
   <!ATTLIST IssuerAndSerialNumber
       issuer                     CDATA            #REQUIRED
       number                     CDATA            #REQUIRED
   >

   <!--
   ******************************************************
   * IOTP SHARED COMPONENT DEFINITION                   *
   ******************************************************
   -->
   <!ELEMENT Value ( #PCDATA ) >
   <!ATTLIST Value
       ID               ID           #IMPLIED
       encoding    (base64|none)    'base64'
   >

   <!ELEMENT Locator EMPTY>
   <!ATTLIST Locator
       xml:link        CDATA        #FIXED        'simple'
       href            CDATA        #REQUIRED
   >
```

14. Glossary

This section contains a glossary of some of the terms used within this specification in alphabetical order.

NAME	DESCRIPTION
Authenticator	The Organisation which is requesting the authentication of another Organisation, and
Authenticatee	The Organisation being authenticated by an Authenticator
Business Error	See Status Component.
Brand	A Brand is the mark which identifies a particular type of Payment Instrument. A list of Brands are the payment options which are presented by the Merchant to the Consumer and from which the Consumer makes a selection. Each Brand may have a different Payment Handler. Examples of Brands include: o payment association and proprietary Brands, for example MasterCard, Visa, American Express, Diners Club, American Express, Mondex, GeldKarte, CyberCash, etc. o Promotional Brands (see below). These include: o store Brands, where the Payment Instrument is issued to a Consumer by a particular Merchant, for example Walmart, Sears, or Marks and Spencer (UK) o coBrands, for example American Advantage Visa, where an a company uses their own Brand in conjunction with, typically, a payment association Brand.

Consumer	The Organisation which is to receive the benefit of and typically pay for the goods or services.
ContentSoftwareId	This contains information which identifies the software which generated the content of the element. Its purpose is to help resolve interoperability problems that might occur as a result of incompatibilities between messages produced by different software. It is a single text string in the language defined by xml:lang. It must contain, as a minimum: o the name of the software manufacturer o the name of the software o the version of the software, and o the build of the software It is recommended that this attribute is included whenever the software which generated the content cannot be identified from the SoftwareId attribute on the Message Id Component (see section 3.3.2)
Customer Care Provider	An Organisation that is providing customer care typically on behalf of a Merchant. Examples of customer care include, responding to problems raised by a Consumer arising from an IOTP Transaction that the Consumer took part in.
Delivery Handler	The Organisation that directly delivers the goods or services to the Consumer on behalf of the Merchant. Delivery can be in the form of either digital goods (e.g., a [MIME] message), or physically delivered using the post or a courier.
Document Exchange	A Document Exchange consists of a set of IOTP Messages exchanged between two parties that implement part or all of two Trading Exchanges simultaneously in order to minimise the number of actual IOTP Messages which must be sent over the Internet. Document Exchanges are combined together in sequence to implement a particular IOTP Transaction.
Dual Brand	A Dual Brand means that a single Payment Instrument may be used as if it were two separate Brands. For example there could be a single Japanese "UC" MasterCard which can be used as

	either a UC card or a regular MasterCard. The UC card Brand and the MasterCard Brand could each have their own separate Payment Handlers. This means that: o the Merchant treats, for example "UC" and "MasterCard" as two separate Brands when offering a list of Brands to the Consumer, o the Consumer chooses a Brand, for example either "UC" or "MasterCard, o the Consumer IOTP aware application determines which Payment Instrument(s) match the chosen Brand, and selects, perhaps with user assistance, the correct Payment Instrument to use.
Error Block	An Error Block reports that a Technical Error was found in an IOTP Message that was previously received. Typically Technical Errors are caused by errors in the XML which has been received or some technical failure of the processing of the IOTP Message. Frequently the generation or receipt of an Error Block will result in failure of the IOTP Transaction. They are distinct from Business Errors, reported in a Status Component, which can also cause failure of an IOTP Transaction.
Exchange Block	An Exchange Block is sent between the two Trading Roles involved in a Trading Exchange. It contains one or more Trading Components. Exchange Blocks are always sent after a Request Block and before a Response Block in a Trading Exchange. The content of an Exchange Block is dependent on the type of Trading Exchange being carried out.
IOTP Message	An IOTP Message is the outermost wrapper for the document(s) which are sent between Trading Roles that are taking part in a trade. It is a well formed XML document. The documents it contains consist of: o a Transaction Reference Block to uniquely identify the IOTP Transaction of which the IOTP Message is part, o an optional Signature Block to digitally sign the Trading Blocks or Trading Components associated with the IOTP Transaction o an optional Error Block to report on technical errors contained in a previously received IOTP Message, and

	o a collection of IOTP Trading Blocks which carries the data required to carry out an IOTP Transaction.
IOTP Transaction	An instance of an Internet Open Trading Protocol Transaction consists of a set of IOTP Messages transferred between Trading Roles. The rules for what may be contained in the IOTP Messages is defined by the Transaction Type of the IOTP Transaction.
IOTP Transaction Type	A Transaction Type identifies the type an of IOTP Transaction. Examples of Transaction Type include: Purchase, Refund, Authentication, Withdrawal, Deposit (of electronic cash). The Transaction Type specifies for an IOTP Transaction: o the Trading Exchanges which may be included in the transaction, o how those Trading Exchanges may be combined to meet the business needs of the transaction o which Trading Blocks may be included in the IOTP Messages that make up the transaction o Consult this specification for the rules that apply for each Transaction Type.
Merchant	The Organisation from whom the service or goods are being obtained, who is legally responsible for providing the goods or services and receives the benefit of any payment made
Merchant Customer Care Provider	The Organisation that is involved with customer dispute negotiation and resolution on behalf of the Merchant
Organisation	A company or individual that takes part in a Trade as a Trading Role. The Organisations may take one or more of the roles involved in the Trade
Payment Handler	The Organisation that physically receives the payment from the Consumer on behalf of the Merchant
Payment Instrument	A Payment Instrument is the means by which Consumer pays for goods or services offered by a Merchant. It can be, for example: o a credit card such as MasterCard or Visa; o a debit card such as MasterCard’s Maestro; o a smart card based electronic cash Payment

	Instrument such as a Mondex Card, a GeldKarte card or a Visa Cash card o a software based electronic payment account such as a CyberCash’s CyberCoin or DigiCash account. All Payment Instruments have a number, typically an account number, by which the Payment Instrument can be identified.
Promotional Brand	A Promotional Brand means that, if the Consumer pays with that Brand, then the Consumer will receive some additional benefit which can be received in two ways: o at the time of purchase. For example if a Consumer pays with a "Walmart MasterCard" at a Walmart web site, then a 5% discount might apply, which means the Consumer actually pays less, o from their Payment Instrument (card) issuer when the payment appears on their statement. For example loyalty points in a frequent flyer scheme could be awarded based on the total payments made with the Payment Instrument since the last statement was issued. Each Promotional Brand should be identified as a separate Brand in the list of Brands offered by the Merchant.
Receipt Component	A Receipt Component is a record of the successful completion of a Trading Exchange. Examples of Receipt Components include: Payment Receipts, and Delivery Notes. It’s content may dependent on the technology used to perform the Trading Exchange. For example a Secure Electronic Transaction (SET) payment receipt consists of SET payment messages which record the result of the payment.
Request Block	A Request Block is Trading Block that contains a request for a Trading Exchange to start. The Trading Components in a Request Block may be signed by a Signature Block so that their authenticity may be checked and to determine that the Trading Exchange being requested is authorised. Authorisation for a Trading Exchange to start can be provided by the signatures contained on Receipt Components contained in

	Response Blocks resulting from previously completed Trading Exchanges. Examples of Request Blocks are Payment Request and Delivery Request
Response Block	A Response Block is a Trading Block that indicates that a Trading Exchange is complete. It is sent by the Trading Role that received a Request Block to the Trading Role that sent the Request Block. The Response Block contains a Status Component that contains information about the completion of the Trading Exchange, for example it indicates whether or not the Trading Exchange completed successfully. For some Trading Exchanges the Response Block contains a Receipt Component that forms a record of the Trading Exchange. Receipt Components may be digitally signed using a Signature Block to make completion non-refutable. Examples of Response Blocks include Offer Response, Payment Response and Delivery Response.
Signature Block	A Signature Block is a Trading Block that contains one or more digital signatures in the form of Signature Components. A Signature Component may digitally sign any Block or Component in any IOTP Message in the same IOTP Transaction.
Status Component	A Status Component contains information that describes the state of a Trading Exchange. Before the Trading Exchange is complete the Status Component can indicate information about how the Trading Exchange is progressing. Once a Trading Exchange is complete the Status Component can only indicate the success of the Trading Exchange or that a Business Error has occurred. A Business Error indicates that continuation with the Trading Exchange was not possible because of some business rule or logic, for example, "insufficient funds available", rather than any Technical Error associated with the content or format of the IOTP Messages in the IOTP Transaction.
Technical Error	See Error Block.

Trading Block	A Trading Block consists of one or more Trading Components. One or more Trading Blocks may be contained within the IOTP Messages which are physically sent in the form of [XML] documents between the different Trading Roles that are taking part in a trade. Trading Blocks are of three main types: o a Request Block, o an Exchange Block, or a o a Response Block
Trading Component	A Trading Component is a collection of XML elements and attributes. Trading Components are the child elements of the Trading Blocks. Examples of Trading Components are: Offer, Brand List, Payment Receipt, Delivery [information], Payment Amount [information]
Trading Exchange	A Trading Exchange consists of the exchange, between two Trading Roles, of a sequence of documents. The documents may be in the form of Trading Blocks or they may be transferred by some other means, for example through entering data into a web page. Each Trading Exchange consists of three main parts: o the sending of a Request Block by one Trading Role (the initiator) to another Trading Role (the recipient), o the optional exchange of one or more Exchange Blocks between the recipient and the initiator, until eventually, o the Trading Role that received the Request Block sends a Response Block to the initiator. A Trading Exchange is designed to implement a useful service of some kind. Examples of Trading Exchanges/services are: o Offer, which results in a Consumer receiving an offer from a Merchant to carry out a business transaction of some kind, o Payment, where a Consumer makes a payment to a Payment Handler, o Delivery, where a Consumer requests, and optionally obtains, delivery of goods or services from a Delivery Handler, and o Authentication, where any Trading Role may request and receive information about another Trading Role.

Trading Role	A Trading Role identifies the different ways in which Organisations can participate in a trade. There are five Trading Roles: Consumer, Merchant, Payment Handler, Delivery Handler, and Merchant Customer Care Provider.
Transaction Reference Block	A Transaction Reference Block identifies an IOTP Transaction. It contains data that identifies: o the Transaction Type, o the IOTP Transaction uniquely, through a globally unique transaction identifier o the IOTP Message uniquely within the IOTP Transaction, through a message identifier The Transaction Reference Block may also contain references to other transactions which may or may not be IOTP Transactions

15. References

This section contains references to related documents identified in this specification.

[Base64] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996.

[DOM-HASH] Maruyama, H., Tamura, K. and N. Uramoto, "Digest Values for DOM (DOMHASH)", RFC 2803, April 2000.

[DNS] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.

[DNS] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

[DSA] The Digital Signature Algorithm (DSA) published by the National Institute of Standards and Technology (NIST) in the Digital Signature Standard (DSS), which is a part of the US government’s Capstone project.

[ECCDSA] Elliptic Curve Cryptosystems Digital Signature Algorithm (ECCDSA). Elliptic curve cryptosystems are analogues of public-key cryptosystems such as RSA in which modular multiplication is replaced by the elliptic curve addition operation. See: V. S. Miller. Use of elliptic curves in cryptography. In Advances in Cryptology - Crypto ’85, pages 417-426, Springer-Verlag, 1986.

[HMAC] Krawczyk, H., Bellare, M. and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.

[HTML] Berners-Lee, T. and D. Connolly, "Hypertext Markup Language - 2.0", RFC 1866, November 1995.

[HTML] Hyper Text Mark Up Language. The Hypertext Mark-up Language (HTML) is a simple mark-up language used to create hypertext documents that are platform independent. See the World Wide Web (W3C) consortium web site at: http://www.w3.org/MarkUp/

[HTTP] Berners-Lee, T., Fielding, R. and H. Frystyk, "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996.

[HTTP] Fielding, R., Gettys, J., Mogul, J., Frystyk, T. and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1.", RFC 2616, June 1999.

[IANA] The Internet Assigned Numbers Authority. The organisation responsible for co-ordinating the names and numbers associated with the Internet. See http://www.iana.org/

[ISO4217] ISO 4217: Codes for the Representation of Currencies. Available from ANSI or ISO.

[IOTPDSIG] Davidson, K. and Y. Kawatsura, "Digital Signatures for the v1.0 Internet Open Trading Protocol (IOTP)", RFC 2802, April 2000.

[MD5] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, April 1992.

[MIME] Crocker, D., "Standard for the Format of ARPA Internet Text Messages", STD 11, RFC 822, August 1982.

[MIME] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996.

[MIME] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996.

[MIME] Moore, K., "MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text" RFC 2047, November 1996.

[MIME] Freed, N., Klensin, J. and J. Postel, "Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures", RFC 2048, November 1996.

[MIME] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Five: Conformance Criteria and Examples" RFC 2049, November 1996.

[OPS] Open Profiling Standard. A proposed standard which provides a framework with built-in privacy safeguards for the trusted exchange of profile information between individuals and web sites. Being developed by Netscape and Microsoft amongst others.

[RFC1738] Berners-Lee, T., Masinter, L. and M. McCahill, "Uniform Resource Locators (URL)", RFC 1738, December 1994.

[RFC2434] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 2434, October 1998.

[RSA] RSA is a public-key cryptosystem for both encryption and authentication supported by RSA Data Security Inc. See: R. L. Rivest, A. Shamir, and L.M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM, 21(2): 120-126, February 1978.

[SCCD] Secure Channel Credit Debit. A method of conducting a credit or debit card payment where unauthorised access to account information is prevented through use of secure channel transport mechanisms such as SSL/TLS. An IOTP supplement describing how SCCD works is under development.

[SET] Secure Electronic Transaction Specification, Version 1.0, May 31, 1997. Supports credit and debit card payments using certificates at the Consumer and Merchant to help ensure authenticity. Download from: <http://www.setco.org>.

[SSL/TLS] Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

[SHA1] [FIPS-180-1]"Secure Hash Standard", National Institute of Standards and Technology, US Department Of Commerce, April 1995. Also known as: 59 Fed Reg. 35317 (1994). See http://www.itl.nist.gov/div897/pubs/fip180-1.htm

[UTC]	Universal Time Co-ordinated. A method of defining time absolutely relative to Greenwich Mean Time (GMT). Typically of the form: "CCYY-MM-DDTHH:MM:SS.sssZ+n" where the "+n" defines the number of hours from GMT. See ISO DIS8601.
[UTF16]	The Unicode Standard, Version 2.0. The Unicode Consortium, Reading, Massachusetts. See ISO/IEC 10646 1 Proposed Draft Amendment 1
[X.509]	ITU Recommendation X.509 1993 | ISO/IEC 9594-8: 1995, Including Draft Amendment 1: Certificate Extensions (Version 3 Certificate)
[XML Namespace]	Recommendation for Namespaces in XML, World Wide Web Consortium, 14 January 1999, "http://www.w3.org/TR/REC-xml-names"
[XML]	Extensible Mark Up Language. A W3C recommendation. See http://www.w3.org/TR/1998/REC-xml-19980210 for the 10 February 1998 version.

16. Author’s Address

The author of this document is:

David Burdett
Commerce One
4440 Rosewood Drive, Bldg 4
Pleasanton
California 94588
USA

Phone: +1 (925) 520 4422
EMail: david.burdett@commerceone.com

The author of this document particularly wants to thank Mondex International Limited (www.mondex.com) for the tremendous support provided in the formative stages of the development of this specification.

In addition the author appreciates the following contributors to this protocol (in alphabetic order of company) without which it could not have been developed.

- Phillip Mullarkey, British Telecom plc
- Andrew Marchewka, Canadian Imperial Bank of Commerce
- Brian Boesch, CyberCash Inc.
- Tom Arnold, CyberSource
- Terry Allen, Commerce One (formally Veo Systems)
- Richard Brown, GlobeSet Inc.
- Peter Chang, Hewlett Packard
- Masaaki Hiroya, Hitachi Ltd
- Yoshiaki Kawatsura, Hitachi Ltd
- Mark Linehan, International Business Machines
- Jonathan Sowler, JCP Computer Services Ltd
- John Wankmueller, MasterCard International
- Steve Fabes, Mondex International Ltd
- Donald Eastlake 3rd, Motorola Inc (formerly International Business Machines Inc)
- Surendra Reddy, Oracle Corporation
- Akihiro Nakano, Plat Home, Inc. (ex Hitachi Ltd)
- Chris Smith, Royal Bank of Canada
- Hans Bernhard-Beykirch, SIZ (IT Development and Coordination

  Centre of the German Savings Banks Organisation)
- W. Reid Carlisle, Spyrus (ex Citibank Universal Card Services, formally AT&T Universal Card Services)
- Efrem Lipkin, Sun Microsystems

- Tony Lewis, Visa International

The author would also like to thank the following organisations for their support:

- Amino Communications
- DigiCash
- Fujitsu
- General Information Systems
- Globe Id Software
- Hyperion
- InterTrader
- Nobil I T Corp
- Mercantec
- Netscape
- Nippon Telegraph and Telephone Corporation
- Oracle Corporation
- Smart Card Integrations Ltd.
- Spyrus
- Verifone
- Unisource nv
- Wells Fargo Bank

17. Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.