TIS : SYSTEM CONNECTION AND SESSION CONTROL

Abstract

This document is part of the Technical Interface Specification (TIS) for Direct Trader Input (DTI) to CHIEF and for Inventory system linking. It defines the services provided by CHIEF to CSPs and those assumed to be provided to CHIEF by CSPs, and the method for establishing and controlling user sessions to allow access to the services.

Author:	Pradeep Agarwal
Approved by:	Jenny Arentsen
Issue:	3.6
Date:	07/12/2009

Prepared for
HM Revenue and Customs
Alexander House
SOUTHEND-ON-SEA

Table of Contents

1. INTRODUCTION

1.1. Purpose

This document is part of the Technical Interface Specification (TIS) for Direct Trader Input (DTI) to CHIEF and Inventory system linking.

Its purpose is to provide CSPs with the information needed to configure (or where necessary, to develop) their software at Transport and Session levels to achieve successful interaction with CHIEF.

1.2. Scope

The document covers the following topics:

- the services provided by CHIEF to CSPs, and the services expected to be provided by CSPs to CHIEF;
- the use of transport connections between CSPs and CHIEF;
- the use of the OSI Session Service to control sessions over Transport Connections.

It is concerned only with the upper layers of the interface: it assumes the availability of a Session Service (see Reference [5]) and an underlying Transport Service (see Reference [4]). The Network and lower levels are outside the scope of this document.

All forms of service between CHIEF and CSPs are covered: interactive screens and EDI messages, unsolicited (e.g. reports) and inter-system EDI messages. Note that services provided to HMRC users are not within the scope of this document.

The syntax of the interactive screens and EDI messages is not of concern to the definition of session control in this document.

END OF SECTION 1

2. SERVICES

The CHIEF architecture makes use of the concepts of ‘clients’ and ‘servers’. A client submits requests for processing to a server; the server processes the request as appropriate, and responds to the client. A server can process requests from many clients, and a client can make requests to many servers. Clients fall broadly into two classes: those which act as an agent for a human user and are largely driven by that user’s actions at the terminal; and those which act on behalf of the system itself and are mostly driven by defined internal or external events.

Note that the concepts of client and server are abstractions, and do not imply any particular implementation within a system: such implementation is outside the scope of this document.

As a result of this architecture, CHIEF is represented to other systems as a number of services, where the term ‘service’ is used to denote the processing capabilities of a type of server. In addition, for certain classes of interaction with CSPs, CHIEF acts as a client with the CSP system providing the required services.

Care should be taken not to confuse this use of the word ‘service’ with that in the OSI context, where a service is an abstract representation of the facilities provided to the immediately higher layer by a particular layer of the seven layer model. To avoid ambiguity, the latter usage is always qualified in this document by an explicit reference to the layer concerned, e.g. Session Service.

The general characteristics of services may be summarised as follows:

a. The facilities of a system available through one service may differ from those available through another: both may be subsets of the full system facilities.

b. Each service has an associated Transport Service Access Point (TSAP). Each service is able to support a number of concurrent transport connections, up to a limit defined for that service.

Each client also has an associated TSAP, via which it can establish transport connections to the TSAPs of appropriate services.

2.1. CHIEF Services

The services provided by CHIEF for any particular CSP system are defined in the Interchange Agreement with the CSP. CHIEF services have the following characteristics:

a. CHIEF is able to constrain external systems to access its facilities through particular services only.

b. Separate services are provided for each of the following classes of use:

- direct trader interaction via ICAB-02 screens (all permitted transactions);
- direct trader interaction via EDIFACT messages (for entry input only);
- inventory system interaction via EDIFACT messages.

c. In addition, as an aid to system management, these services can be further divided on the basis of communities of use (for example, a service can provide facilities only to interactive screen users via a particular CSP).

With this approach, each CSP views CHIEF as a number of services, each supporting one of the classes of use identified above and typically (though not essentially) dedicated to its own use. The naming convention used by CHIEF to identify services (and thus TSAPs) illustrates the point:

CIESCNSHCI1:	The first (and in fact only) ICAB-02 screen (HCI) live service for CNS users;
HMUTFCPEDI:	The EDIFACT message (EDI) trade test service for FCP users;
CIESDHBINV:	The Inventory System (EDI) live service for Dover.

Should a CSP require more simultaneous connections than CHIEF's implementation limit for a single service, then a further service must be configured (concurrency limits for each service are documented as part of the Interchange Agreements). The new service would have the same capabilities, but would be separately identified (as the addition of, or a change to, the numeric suffix). Such an additional service has a different TSAP. Whilst there is a limit to the number of additional services that can be configured in this way, it is of no practical significance since it is greatly in excess of the number of services needed to support the full set of CSPs.

2.2. CHIEF Clients

The clients provided by CHIEF for any particular CSP system are defined in the Interchange Agreement with the CSP.

A separate client is provided for each of the following classes of use within each CSP system:

- transfer to the CSP system of EDIFACT report messages for spooling and onward delivery;
- transfer to the CSP system of EDIFACT broadcast messages for spooling and onward delivery;
- EDI transactions serviced by an Inventory system.

2.3. CSP Services

Each CSP must provide services to handle the requests from the CHIEF clients (see Section 2.2). Each such service must have an associated TSAP.

2.4. CSP Clients

Each CSP must provide appropriate clients to make use of the services offered by CHIEF to that CSP.

END OF SECTION 2

3. CONNECTION TO SERVICES

The level of service for each CSP is agreed with HMRC and included in the Interchange Agreement. As part of this agreement, the number of transport connections which can be established between the TSAPs of CHIEF and a CSP is specified – the number allowed for each CSP system/CHIEF TSAP pair is based upon the maximum number of concurrent sessions that are expected.

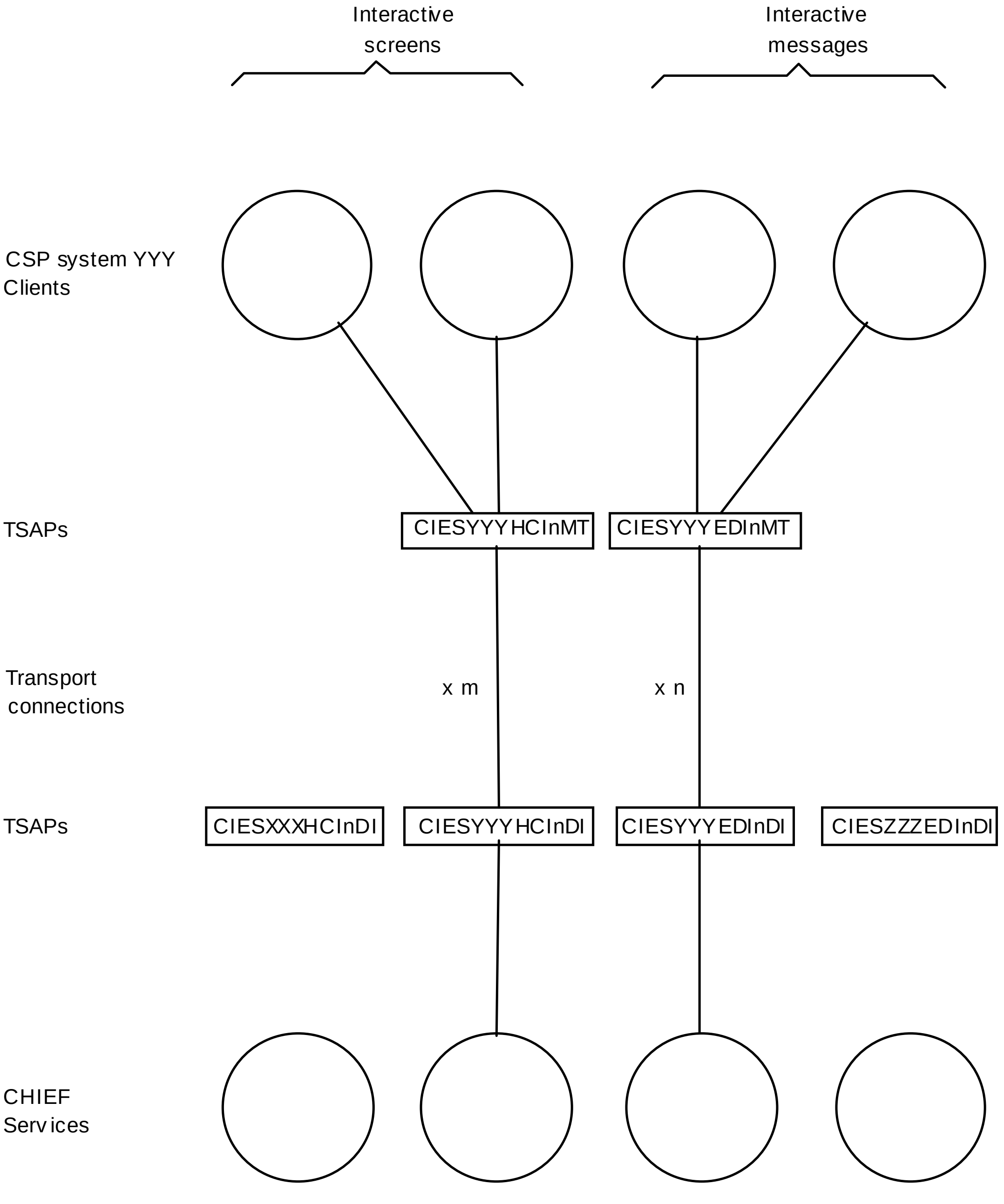

Figure 3.1. Transport Connection Configuration

Transport connections are established by the client. The responsibility is thus on CSPs to initiate the requests for transport connections from their clients to CHIEF services, and on CHIEF to initiate those from CHIEF clients to their services.

Figure 3.1 portrays schematically the connections between a CSP and CHIEF.

- CSP system 'YYY' is defined as two client TSAPs: one for interactive screens, the other for interactive Trader Input messages (ignoring the other classes of messages which would simply extend the picture).
- Corresponding service TSAPs exist within CHIEF.
- A number of transport connections are established: the number not being related directly to the total CSP system population for each class of use but rather to their anticipated maximum concurrency.

3.1. Reuse of Transport Connections

OSI permits only a single session connection over a particular transport connection at any one time. Recognising the potential performance implications of dynamically making and breaking a transport connection however, it is permitted serially to reuse a transport connection: having released one session connection, another can be established using the same underlying transport connection. An OSI Session Service implementation does not have to support this feature, but it is recommended.

Where possible, advantage should be taken of transport connection reuse to make more efficient use of the communication resources between CSPs and CHIEF. Thus when a client completes its interactions with the server, the session connection is released but the underlying transport connection is retained. The client can then establish another session connection over the same transport connection, as and when required.

Where reuse of transport connections is supported by both the client and server systems, the client system could choose to establish all of its permitted transport connections as a result of some initialisation event and once established, reuse them (without disconnection) in accordance with client session demand. In practice such an approach may prove unsatisfactory: for example, it could incur lengthened restart times following a system failure. Indeed, it is strongly recommended that transport connections are progressively established in response to session demand, up to the configured limit: it might also prove desirable that connections established during peak periods should subsequently be disconnected when demand slackens.

3.2. Concurrency Limit

Should demand for sessions exceed the concurrency specified in the Interchange Agreement, then the client system must wait or reject the implied user request. Should this become a regular occurrence, then an increased limit to the number of transport connections (possibly involving the provision of an additional service) would need to be agreed with HMRC.

3.3. Streaming by Trader Role

In order to provide the maximum flexibility and resources for handling the peak load from each CSP (which occurs at different times of the day), the application processes are not dedicated to a particular CSP but, within certain groupings by message type, are available to all CSPs. A minimum number of application processes of each type are 'connected' at all times, with others available 'on demand' if the number of queued messages demands greater processing concurrency.

CHIEF can accept and concurrently process EDI messages for the same trader. However, as part of the processing it is necessary to read and update database records relevant to the trader and these records are locked for most of the processing period, so concurrent processing of two messages for a single trader results in delays as the second process awaits release of the locks.

If, within a short period, the CSP submits many messages for the same trader over multiple connections, all the available application processes contend with each other, message queues build and additional processes could be connected which would only serve to make matters worse. Indeed, submitting messages in this way is often counterproductive in performance terms and may in some circumstances cause rejection of messages due to service overload and/or provide a ‘denial of service’ for other traders both via that CSP and from other CSPs. Perversely, this situation has greater impact when CHIEF is otherwise lightly loaded as there are no messages from other traders to break-up the logjam.

Although it is outside the scope of this document to consider how CSPs receive bulk volumes of entries from their users, entries to CHIEF for a single trader role must be ‘streamed’ across a single transport connection, by queuing the next entry until a response is received for its predecessor and sustaining the user session until all entries for the trader have been completed. This will result in optimum processing of the set of messages for the trader, reduce the likelihood of unexpected failures and not block the service to other traders or CSPs.

END OF SECTION 3

4. CLIENT SESSION MANAGEMENT

4.1. Overview

Previous chapters of this document have described how a number of transport connections can be established between a CSP system and CHIEF for each required class of usage. Whilst this provides a basic communication path, a further level of control is necessary to define client-to-server sessions: this is accomplished by use of the CHIEF Session Profile (see Reference [5]) – a subset of the OSI Session Service (see Reference [4]).

The OSI Session Service provides a mechanism for declaring the start and end of a session, including the ability to pass any data which needs to be exchanged by the communicating entities in establishing or terminating the session (for CHIEF, the exchanged data relates particularly to security parameters).

It is important to note that the term ‘client session’ is used in this document to refer to a sequence of interactions between a client and a server. Client sessions are established both for user and for inter-system interactions:

a. For a user, the client system is responsible for user authentication (sign-on) prior to establishing a client session for that user. A client session cannot span a user session (sign-on to sign-off) in the client system (see Figure 4.1).

b. For a client acting on behalf of the system, the client session is bounded by successive system or client start-ups.

With this definition, a client session is not necessarily of the same duration as an OSI session (from connect to release or abort). Indeed, as will become evident in Section 4.1.2, a client session can survive across a series of OSI sessions. The use of the word session with these two different meanings is a possible source of confusion, and thus the terms client session and OSI session are used to avoid ambiguity.

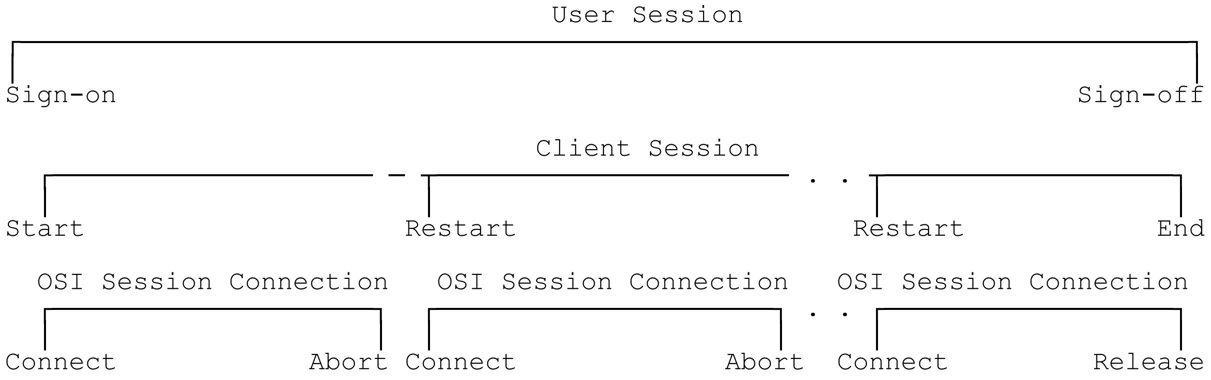

Figure 4.1. Relationship of Client Session to OSI Session Connections

4.1.1. Data Exchange within a Client Session

A Client Session provides the means by which a client can interact with a server, the syntax of this interaction is of no interest to session control. The nature of the interaction is of interest since this determines requirements for controlling whether it is the turn of the client or server to send a message and whether out-of-turn messages are allowed.

The EDIFACT syntax (see Reference [5]) is used by CHIEF for the EDI messages. The EDI Specification (see Reference [2]) defines the interactive use of EDI messages and only requires a simple exchange of messages, with the client making a request and the server responding before a further request is made by the client.

The ICL ICAB-02 syntax (see Reference [5]) is used by CHIEF for interactive screens. The HCI Guide (see Reference [3]) defines the way in which the user interacts through a VDU screen with CHIEF. This is defined to be a simple input message from the client and output screen from the server. There is currently no requirement supported by the HCI for the user to break-in while the server is processing an input message or for the server to output an unsolicited message while the client is ‘typing’.

Thus a client session is only required to support messages in alternate directions. An input from the client is followed by a response from the server or a forced session termination: an output from the server is followed by the next input from the client or a normal session termination.

The only exception to this sequence is when the underlying OSI session is aborted due to a failure in the software at the other end or in the network connection. An abort can be indicated at any time (i.e. when expecting to send or receive a message). In this case the client session may be recoverable on another OSI session (see Section 4.1.2).

It should be noted that an end session command from the client always ends the session and will abort any incomplete transaction(s) – there is no response to an end session request. The user should be discouraged from ending a session without completing all his current transactions.

4.1.2. Client Session Recovery

A client session requires an OSI session in order to exchange data with the server. However, a client session can survive the failure of an OSI session and be restarted on connecting a new OSI session.

While a client session is awaiting recovery on a new OSI session it is still subject to time-out (see Section 4.1.3) or may be terminated by the server system for other reasons. A restart request for a client session that is not currently awaiting recovery will fail – the client system must explicitly start a new client session if required. However, a start session for the same end-user will succeed even if there is a recoverable client session for the same user (all roles are permitted concurrent sessions). The recoverable client session will eventually be timed-out or terminated for concurrency reasons.

The number of extant client sessions at a particular point in time cannot exceed the maximum OSI session concurrency (i.e. the transport connection limit) for the service. When client sessions are pending recovery, it is possible that an OSI session connection request may be received for a new client session to be started when the client session concurrency is at its limit. It this case the client session closest to time-out is ended and the new client session started.

4.1.3. Client Session Timeout

Client sessions are timed-out by CHIEF after the period agreed for each service and defined in the Interchange Agreement.

It is good practise for the client to end the client session before the time-out occurs. Thus the time-out period defined for the server system should be longer than the time-out period enforced by the client system.

4.2. Use of OSI Session Service

This section describes how the OSI Session Service is used to manage client sessions between CHIEF and CSPs, by passing defined values known as Client Session Management Commands (CSMCs) in the User Data or Reason Code parameters of the various OSI Session Service primitives.

The commands and their underlying OSI Session primitives are shown in Table 4.1. The interactive protocol between client and server using these CSMCs is shown in Figure 4.2. It should be noted that an abort indication can occur at any time and is not depicted – the abort indication need not be observed by the client or server until the next normal action is attempted (e.g. as a error response to a send data attempt).

CSM Command/Action	CI	OSI Session Primitive	Text reference
START-SESSION-REQUEST	1	S-CONNECT request	Section 5.2.1
START-SESSION-RESPONSE	2	S-CONNECT response	Section 5.2.1
Send data		S-DATA request	Section 5.2.2
RESTART-SESSION-REQUEST	4	S-CONNECT request	Section 5.2.3
RESTART-SESSION-RESPONSE	5	S-CONNECT response	Section 5.2.3
END-SESSION	6	S-RELEASE request	Section 5.2.4
FORCED-END-SESSION	7	S-RELEASE request	Section 5.2.5
ABORT-SESSION	8	S-U-ABORT request	Section.2.6

Table 4.1. Client Session Management Commands

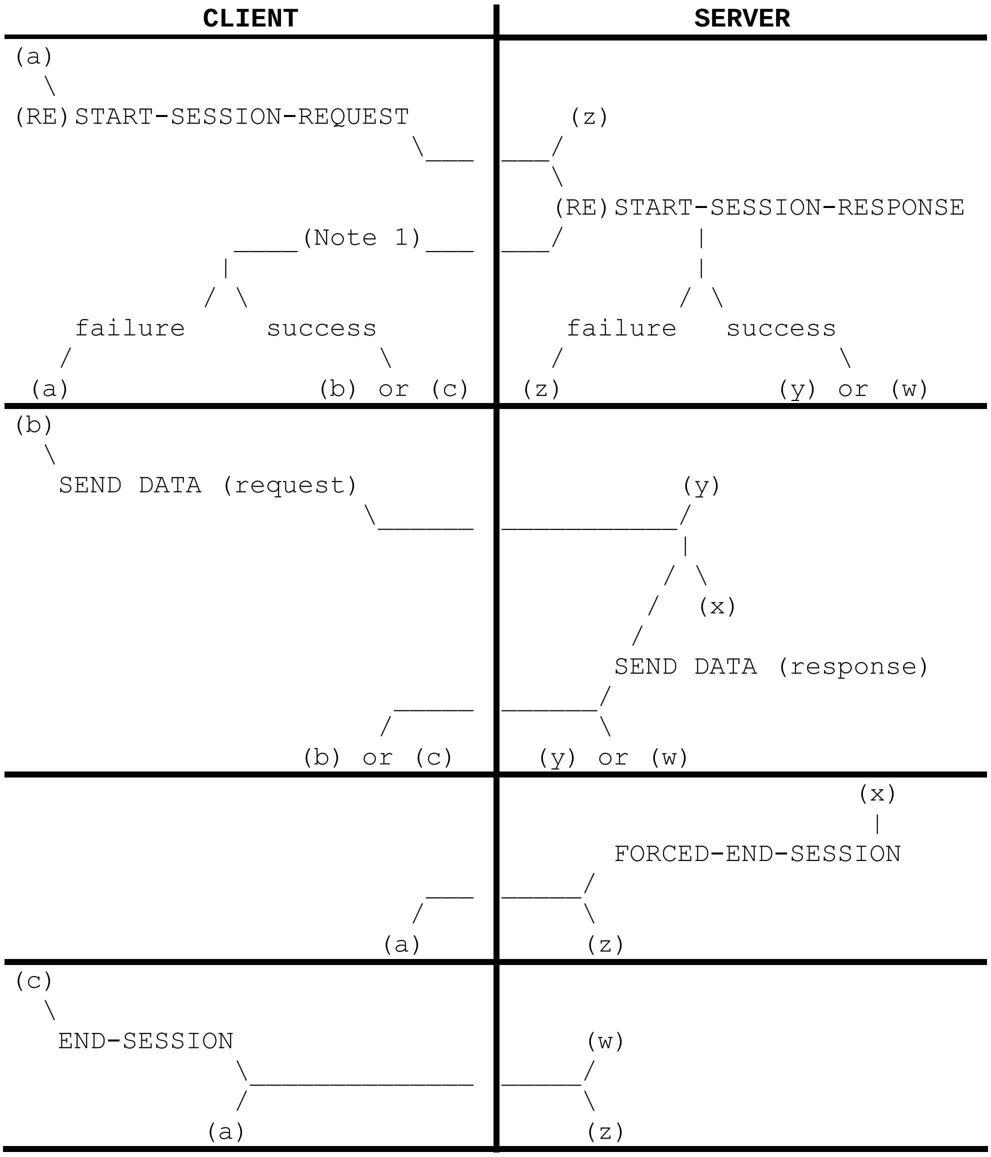

Figure 4.2. CSM Protocol Flow

Note:

1. For the HCI service only the START/RESTART-SESSION-RESPONSE commands are followed by the initial HCI Signed-On/Command Line screen as an S-DATA message.

The structure and encoding of these commands is defined in Section 5.

The following subsections describe how the CSMCs are used to perform the various aspects of client session management and how data is transferred. References are made to the OSI Session Service primitives as defined in Reference [4] but not to the underlying session protocol as defined in the OSI Session Protocol (see Reference [5]).

4.2.1. Client Session Establishment

A client session is initiated by the client issuing a START-SESSION-REQUEST command as defined in Section 5.2.1.

The START-SESSION-REQUEST command is passed in the SS-user data parameter of an S-CONNECT request primitive to the OSI Session Service, which establishes an OSI session connection (over a transport connection to the required service).

The server receives the START-SESSION-REQUEST command as the SS-user data parameter of an S-CONNECT indication primitive from the OSI Session Service. It replies with a START-SESSION-RESPONSE command as defined in Section 5.2.2.

The Result indicates whether the sign-on request has succeeded, and if not, gives a reason why. In the event of a failure the OSI session connection is not made.

The Client Session Reference is provided so the client system can restart the client session after a client system failure or an S-U-ABORT or an S-P-ABORT indication. Whenever possible, the server system retains client session details for recovery after server system failure or receipt of an S-U-ABORT or an S-P-ABORT indication.

The START-SESSION-RESPONSE command is passed in the SS-user data parameter of an S-CONNECT response primitive to the OSI Session Service.

Where the result field of the START-SESSION-RESPONSE command is ‘SUCCESS’ the client receives the START-SESSION-RESPONSE command as the SS-user data parameter of an S-CONNECT confirm primitive from the OSI Session Service.

Where the result field of the START-SESSION-RESPONSE command is ‘FAILED’ the client receives the START-SESSION-RESPONSE command in the user data field of the Reason Code parameter of the Refuse SPDU, with the Reason Code set to ‘rejection by called SS-user’. How the Client user data is presented to the user in the S-CONNECT response primitive of the local Session Service interface is local implementation dependent.

4.2.2. Data Transfer

While the client session is associated with an OSI session (i.e. has been (re)started and is not awaiting recovery), data is transferred between the communicating entities (the client and the server) by normal use of the OSI Session Service, using S-DATA request and indication primitives in accordance with the rules of References [4] and [5].

The client initiates a transaction by sending an input message and waiting for a response from the server. The server either responds with data or a FORCED-END-SESSION or ABORT-SESSION command.

After sending a response, the server waits for the next input message from the client or an END-SESSION or an abort indication.

While data is being transferred on a client session, a S-P-ABORT indication can be received by either the client or the server at any time. If possible the server retains client session context for recovery by the client system issuing a RESTART-SESSION-REQUEST. The server does not retain the last response message (which may not have been successfully received and fully processed by the client) for output on recovery of the client session. The client is responsible for repeating any input message for which a response has not been received – the application is responsible for detecting a duplicate message or ensuring that duplicates can be reprocessed.

It should be noted that at the HCI the end-user can request a screen refresh which will cause the server to repeat its last output.

The subset of the OSI Session Service defined in Reference [5] implies that the Session Service itself provides no turn control to regulate the exchange of messages between client and server; nor does it provide end-to-end acknowledgement of receipt of data. These are application layer matters.

The maximum size of Session Service Data Unit (SSDU) that can be handled by each CHIEF service or client is specified in the appropriate service or client definition in the Interchange Agreement with the relevant CSP.

4.2.3. Client Session Recovery

A client session may be recoverable by the client system in the event of the underlying OSI session being lost as a result of a failure of the client, the server or the transport connection. An attempt to recover the client session can be made by the client issuing a RESTART-SESSION-REQUEST command as defined in Section 5.2.3.

The Client Session Reference must identify a client session that is not currently associated with an OSI session and which was established for the specified role, location and purpose.

The RESTART-SESSION-REQUEST command is passed in the SS-user data parameter of an S-CONNECT request primitive to the OSI Session Service, which establishes an OSI session connection (over a transport connection to the required service).

The server receives the RESTART-SESSION-REQUEST command as the SS-user data parameter of an S-CONNECT indication primitive from the OSI Session Service. It replies with a RESTART-SESSION-RESPONSE command as defined in Section 5.2.4.

The result identifies when the client session has been recovered and if it cannot be recovered the reason why.

The RESTART-SESSION-RESPONSE command is passed in the SS-user data parameter of an S-CONNECT response primitive to the OSI Session Service.

Where the result field of the RESTART-SESSION-RESPONSE command is ‘SUCCESS’ the client receives the RESTART-SESSION-RESPONSE command as the SS-user data parameter of an S-CONNECT confirm primitive from the OSI Session Service.

Where the result field of the RESTART-SESSION-RESPONSE command is ‘FAILED’ the client receives the RESTART-SESSION-RESPONSE command in the user data field of the Reason Code parameter of the Refuse SPDU, with the Reason Code set to ‘rejection by called SS-user’ (thus rejecting the OSI session connection). How the Client user data is presented to the user in the S-CONNECT response primitive of the local Session Service interface is local implementation dependant but must end the client session as it was not successfully recovered.

4.2.4. Normal Client Session Termination

A client session is normally terminated by the client issuing an END-SESSION command. This command has no parameters.

The END-SESSION command is passed in the SS-user data parameter of an S-RELEASE request primitive to the OSI Session Service, which causes the OSI session connection to be released.

The END-SESSION command is received by the server as the SS-user data parameter of an S-RELEASE indication primitive from the OSI Session Service. The server clears all context and ends the client session. It should be noted that the client is responsible for ensuring that there are no active transactions in the server when the session is ended – if there are, they are aborted (cf. nested transactions at the HCI).

4.2.5. Forced Client Session Termination

A client session can be forcibly terminated by the server issuing a FORCED-END-SESSION command as defined in Section 5.2.6 in response to data input by the client – the server can only end the session by aborting (see Section 4.2.6) at other times.

The FORCED-END-SESSION command is passed in the SS-user data parameter of an S-RELEASE request primitive to the OSI Session Service, which causes the OSI session connection to be released.

The FORCED-END-SESSION command is received by the client as the SS-user data parameter of an S-RELEASE indication primitive from the OSI Session Service.

4.2.6. Client Session Abort

A client session can be aborted at any time by either party issuing an ABORT-SESSION command as defined in Section 5.2.7.

ABORT-SESSION is used when either the client or server detects a protocol error such as an unexpected indication from the Session Service or an invalid CSMC (e.g. the server requesting END-SESSION).

The ABORT-SESSION command is passed in the SS-user data parameter of an S-U-ABORT request primitive to the OSI Session Service, which causes the immediate release of the OSI session connection.

The ABORT-SESSION command is received by the other party in the SS-user data parameter of an S-U-ABORT indication primitive from the OSI session service.

It should be noted that the S-U-ABORT indication (or a S-P-ABORT indication) can be received by either party at any time: in particular, when a certain command is expected in response to one already sent, the abort indication may be received instead.

After an abort indication, the client session is available for restart subject to the failure being one from which both the server and client can recover. The client is responsible for attempting recovery (see Section 4.1.2).

4.2.7. Additional CHIEF Client Information

Additional information can be supplied by the CHIEF Client giving details of the current transaction and its state – this is configurable at system load time.

This information is intended to serve two purposes:

- better Session management – to allow dynamic adjustment of session timeout and improved session recovery;
- to provide a mechanism for charging for a transaction, irrespective of how it is input (stated to be essential to the commercial viability of a CSP). In many cases declarations entered via the HCI cannot be detected.

This information is carried within the SESSION-STATUS CSMC using the S-TYPED-DATA write and indication primitives. This message is sent prior to the S-DATA message to which it relates.

It should be noted that this message is only sent out prior to the '**end**' of a transaction.

END OF SECTION 4

5. CSM COMMAND DEFINITIONS

This section defines the generic structure of Client Session Management Commands (CSMCs), and for each CSMC, the parameters that it contains and their encoding.

5.1. CSMC Structure

Each CSMC consists of one or more octets that are numbered sequentially starting from 1.

Each octet within a CSMC consists of eight bits numbered 8 to 1 where 1 is the low order bit.

Where a bit is defined as reserved, it must have the value 0.

The sequence of octets within a CSMC and the sequence of bits within an octet are defined for each CSMC in Section 5.2.5.2.

The structure of a CSMC is:

<CI>[<parameter>]...

where:

<CI> The CI (Command Identifier) field that identifies the type of CSMC. This is a single octet encoded as a binary number. The values of the CI fields are defined as decimal numbers in Section 5.2.

<parameter> A parameter (in an optional list of parameters as defined for the particular CSMC) as defined below.

The structure of a parameter is:

<PI><length><value>

where:

<PI> The PI (parameter identifier) field that identifies the type of parameter. This is a single octet encoded as a binary number. The values of the PI fields are defined as decimal numbers in the Tables in Section 5.2.

<length> The length field that indicates the length in octets of the value field, encoded as a binary number; a value of 0 indicates that the associated parameter value is absent.

<value> The parameter value field.

The parameters can be in any order and optional parameters can be omitted (or supplied with a length of 0).

5.2. CSMC Identifiers and Parameter Fields

5.2.1. START-SESSION-REQUEST Command

The START-SESSION-REQUEST CSMC is:

CI	1

Parameter	PI	m/o/c	Length
Version	1	m	1 octet
Individual	2	o	12 octets maximum
Role	3	m	12 octets maximum
Location	4	m	12 octets maximum
Purpose	5	o	1 octet

m: mandatory
o: optional
c: conditional

The Version parameter must be set to 1 to identify the version of the CSM protocol defined in this document.

5.2.2. START-SESSION-RESPONSE Command

The START-SESSION-RESPONSE CSMC is:

CI	2

Parameter	PI	m/o/c	Length
Result	7	m	1 octet
Client Session Reference	8	c	16 octets maximum

m: mandatory
o: optional
c: conditional

Possible Result values are:

0 Success

1 failed – version not supported

2 failed – invalid individual (i.e. Character)

3 failed – role not recognised

4 failed – location not recognised

5 failed – purpose not recognised

6 failed – service not available

7 failed – sign-on not permitted for role/location/purpose from the client system

8 failed – system limit exceeded

The Client Session Reference is present if and only if the Result is 0 (success) and the server system supports client session restart.

5.2.3. RESTART-SESSION-REQUEST Command

The RESTART-SESSION-REQUEST CSMC is:

CI	4

Parameter	PI	m/o/c	Length
Client Session Reference	8	m	16 octets maximum
Individual	2	o	12 octets maximum
Role	3	m	12 octets maximum
Location	4	m	12 octets maximum
Purpose	5	o	1 octet

m: mandatory
o: optional
c: conditional

The Client Session Reference is the value which was returned by the server in the START-SESSION-RESPONSE command when the client session was established.

The Individual, Role, Location and Purpose parameter values must be identical to those supplied in the START-SESSION-REQUEST which established the client session.

5.2.4. RESTART-SESSION-RESPONSE Command

The RESTART-SESSION-RESPONSE CSMC is:

CI	5

Parameter	PI	m/o/c	Length
Result	7	m	1 octet

m: mandatory
o: optional
c: conditional

Possible Result values are:

0 Success

1 failed – version not supported

6 failed – service not available

7 failed – sign-on not permitted for role/location/purpose from the client system

8 failed – system limit exceeded

101 failed – Client Session Reference invalid or client session is not available for recovery (e.g. timed-out)

102 failed – individual/role/location/purpose do not match client session details supplied when the client session was started

103 failed – already active

5.2.5. END-SESSION Command

The END-SESSION CSMC is:

CI	**6**

This command has no parameters.

5.2.6. FORCED-END-SESSION Command

The FORCED-END-SESSION CSMC is:

CI	**7**

Parameter	PI	m/o/c	Length
Reason	10	m	1 octet

m: mandatory
o: optional
c: conditional

Possible Result values are:

1 security violation threshold exceeded

2 service has become unavailable (note that there are failure conditions under which the service will become unavailable but it will not be possible to inform the client via this mechanism: such failures will result in the session being aborted with an S-P-ABORT indication)

3 client session has been terminated (e.g. time-out)

5.2.7. ABORT-SESSION Command

The ABORT-SESSION CSMC is:

CI	8

Parameter	PI	m/o/c	Length
Reason	10	m	1 octet

m: mandatory
o: optional
c: conditional

Possible Result values are:

7 command format invalid

10 CSM command out of context

11 software failure (note that there are software failure conditions under which it will not be possible to inform the server via this mechanism: such failures will result in the session being aborted with an S-P-ABORT indication).

12 Session Service indication out of context

13 Abnormal Terminal disconnection

14 Application detected Time-Out

15 Message too garbled to transmit CONTRL message

16 Session abandoned by system operator

17 Unsupported event. (An event notified by the underlying service which cannot be handled).

5.2.8. SESSION-STATUS Command

The SESSION-STATUS CSMC is:

CI	9

Parameter	PI	m/o/c	Length
Transaction Identifier	11	m	4 octets
Transaction Status	12	m	1 octet
Transaction Level	13	m	1 octet
Inventory Consignment Reference	14	c	35 octets maximum

m: mandatory
o: optional
c: conditional

END OF SECTION 5

6. CSM PARAMETER DEFINITIONS

The characters specified in the format of the parameters in Table 6.1 are encoded as an octet using the UK national variant of ISO 646 in bits 7 to 1 with bit 8 set to zero.

Parameter	PI	Format	Description
Client Session Reference	8	Binary	The client session reference is a unique reference to the client session within the service. It is provided so the client session can be recovered in the event of the underlying OSI session being aborted.
Individual	2	A to Z 0 to 9 space	The Individual parameter is supplied for auditing purposes in the server system. It is an octet string which is intended to be used to uniquely identify the individual for whom the session is being started when there are a number of users who can perform the same role. It must not be supplied if the client is not acting on behalf of a human user. The parameter is optional and can range from a name or a personnel number supplied by the trade system, to an authentication reference defined by the CSP (e.g. to a record of authenticated details of a user session).
Inventory Consignment Reference	14	full level B character set	For an inventory linked entry, this is the Inventory system's reference for the consignment.
Location	4	A to Z 0 to 9 space	The Location parameter identifies the location at which the user is operating as determined by the client system. Where the client is not acting on behalf of a human user, the location is the physical location of the client system. It is an octet string which uniquely identifies the operating location of the user for whom the session is being started – in particular it may be used to assist in the delivery of any output reports associated with the session. This location may need to be registered within the Security Management Information Base of the server system.
Purpose	5	A to Z	The Purpose of the client session is either operational ('O') or training ('T'). The parameter defaults to operational.
Reason	10	Binary	The Reason parameter value is a binary number indicating the reason for the command. Values are defined for the CSMC.
Result	7	Binary	The Result parameter value is a binary number indicating the success or failure of the command. Values are defined for the CSMC.
Role	3	A to Z 0 to 9 space	The Role parameter is intended to be used by the server system to confer appropriate privileges to the user during the client session. Where the client is not acting on behalf of a human user (e.g. an Inventory system client for notifying goods arrival and other consignment changes), the role identifies the system and the function being undertaken by the client. This role may need to be registered within the Security Management Information Base of the server system.
Transaction Identifier	11	A to Z 0 to 9	The Transaction Identifier is the four character CHIEF transaction mnemonic.

Parameter	PI	Format	Description
Transaction Status	12	Binary	The Transaction Status is a binary encoded field representing the state that the transaction is passing through, i.e. 0 = Fully completed (for an entry Insert/Amend this means the entry has been committed), 1 = Aborted, 2 = Partially completed.
Transaction Level	13	Binary	The Transaction Level is a binary encoded field indicating the level of nesting in which the transaction is operating, i.e. 1 = Normal Top Level Transaction e.g. initiated from HCI Command Line, 2 = Nested Transaction e.g. initiated from HCI Downward Select.
Version	1	Binary	The Version parameter is a binary number indicating which version of the CSM protocol is being used.

Table 6.1. CSM Parameter Definitions

END OF SECTION 6

7. GLOSSARY AND REFERENCES

7.1. Glossary

See USM 102 – CHIEF GLOSSARY AND ABBREVIATIONS

7.2. References

Ref No.	Title	Document reference
1.	TIS : OVERVIEW	DES 110
2.	TIS : ELECTRONIC DATA INTERCHANGE (EDI) SPECIFICATION	DES 150
3.	TIS : HUMAN COMPUTER INTERFACE (HCI) GUIDE	DES 214
4.	OPEN SYSTEMS INTERCONNECTION: BASIC CONNECTION-ORIENTED SESSION SERVICE DEFINITION	ISO 8326
5.	OPEN SYSTEMS INTERCONNECTION: BASIC CONNECTION-ORIENTED SESSION PROTOCOL DEFINITION	ISO 8327

END OF SECTION 7

8. DOCUMENT CONTROL

8.1. Document History

Issue No.	Date of Change	IC No.	Details of changes
3.2	18/04/96		Change to Transaction Status for LP-0838 (Hotline 7534).
3.3	10/08/01		Conversion to Word97. Correction of TSAP names. Minor corrections and clarification.
3.4	31/05/06		Update to latest document standards. Addition of Section 3.3 (Streaming by Trader Role).
3.5	26/03/2009		This is part of a restructuring of the TIS documents. RFC1006 over TCP/IP added.
3.6	07/12/2009		Removed BT logo and BT specific details

8.2. Revision Record

Revision Number	Date	Name	Signature

The above table is to be used for recording the incorporation of minor revisions into the document; that is, revisions issued as changed pages only. This page must be retained in the document until such time as the complete document is re-issued.

8.3. Configuration Management

8.3.1. Document Configuration

a) Title:	TIS : SYSTEM CONNECTION AND SESSION CONTROL
b) Reference:	DES 111
c) Privacy marking:	X<>
d) Status:	Agreed for use
e) Owner:	Jenny Arentsen
f) Change Authority:	CHIEF Document Controller
g) Location of master copy:	
Paper:	CHIEF Library
Electronic:	System: http://aspireportal/sites/CHIEFTRANS/Knowledge%20Management%20%20Transfer Directory: \Redocumentation Project\TIS\ Filename: DES111 - DTI System Connection and Session Control.doc Format: Word 2003
h) Suggested Distribution:	Project Library HMRC for onward distribution to the Trade

8.3.2. Document Signatories

Approver		**Author**	
Signature:		**Signature:**	
Name:	Jenny Arentsen	**Name:**	Pradeep Agarwal
Date:		**Date :**	

END OF DOCUMENT