ATNP WG3/SG3 (Upper Layer Architecture)

Draft Upper Layer Guidance Material for CNS/ATM-1 Package

(Presented by WG3/SG3)

REVISION STATUS

Version 1.0 First draft Guidance Material for Upper Layers (Gold Coast)
Version 2.0 Second draft Guidance Material for Upper Layers (Bruxelles)
Version 3.0 Third draft Guidance Material for Upper Layers (Munich)
Version 4.0 Fourth draft Guidance Material for Upper Layers (Alexandria)

TABLE OF CONTENTS

1. Introduction

1.1.1.1 The ATN ULA

The ATN ULA has three aspects. These are as follows:

ISO 9545, edition 2 specifies the extended application layer structure (ALS). This allows modular construction of software by specification of application service elements (ASEs). An ASE is implemented as a software module. These are combined into Application Service Objects (ASOs). Interaction between ASEs and ASOs are mediated by a control function (CF).

ISO 8649, edition 2 and ISO 8650, edition 2 specify the Application Control Service Element (ACSE) needed to support the ALS. ACSE allows the establishment of associations over transport connections.

Amendments to ISO 8823 and ISO 8327 specify efficient Presentation Protocols and Session Protocols. The amendments specify minimal functionality protocols to indicate protocol functionality.

The ATN ULA uses the ATN transport service.

1.1.2 Architectural Guidance Material

1.1.2.1 Description of the Upper Layer Architecture

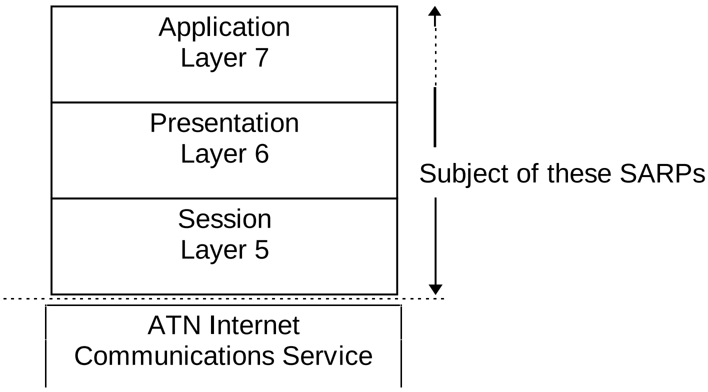


Figure 1.1: Conceptual view of the scope of the UL SARPs

Note **--** In the picture above, it is useful to note the effect of the supporting layer efficiency enhancements.
First, the connect request at the application layer is mapped to the connect request of the supporting upper layers.
Second, the data request at the application layer is mapped directly to the transport data request.
Third, there is no congruent mapping for the release request. It must be handled by the Control Function using data requests (to carry the ACSE release) and abort requests (to actually clear the connection after the graceful release.)

1.1.2.2 Description of the Application Layer

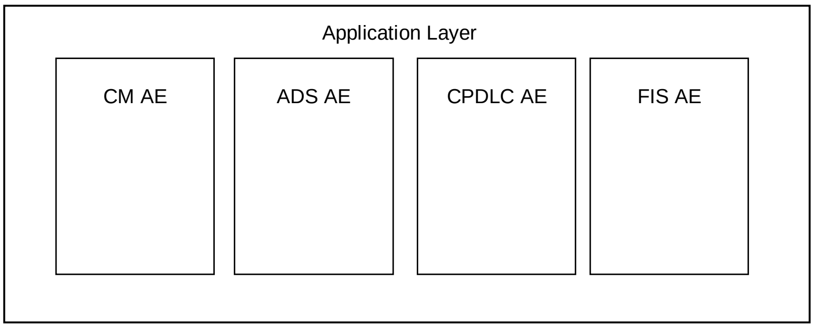


Figure 1.2: Conceptual view of Application Layer

Note **--** It is important to note the architectural decisions embodied in the above picture.
Each application is embodied in an AE.
Each AE contains an ATN ASE, which is the communications element responsible for an ATN application. The internal structure of the ATN ASE may be of arbitrary complexity. The dialogue service is the ATN ASE's view of the ATN ULA.
Thus, the type of the AE is the same as the type of the ASE. That is, an ADS AE will contain only an ADS ASE.
Thus, on connection establishment, the peer title can be completely constructed. The AE-title is provided by the peer ID value in the dialogue service, and the peer AP-qualifier (e.g., CPC) is the same as yours is.
There is no architectural capability for multiple instances of the same ATN ASE within the same AE. A requirement for another instance of an ASE (e.g., the CM contact procedure), requires spawning another AE. This implies that the ATN ASE will generate and manage only one dialogue over the lifetime of the ASE.
There is no interaction inside an AE between ASEs of different types, since these are not provided by the architecture. Any requirement for interaction between applications occurs in user space through, e.g., global data structures.
As implied in the figure, there is no upper-layer addressing. All addressing of the AE-type is complete with the TSAP address. All upper layer selectors are necessarily nil.
The application context name is also a simple construct, since the CNS/ATM-1 ASE list can be inferred from the AP-title. The CNS/ATM-1 application context name encodes only the application version.

1.1.2.3 Description of the AE Structure

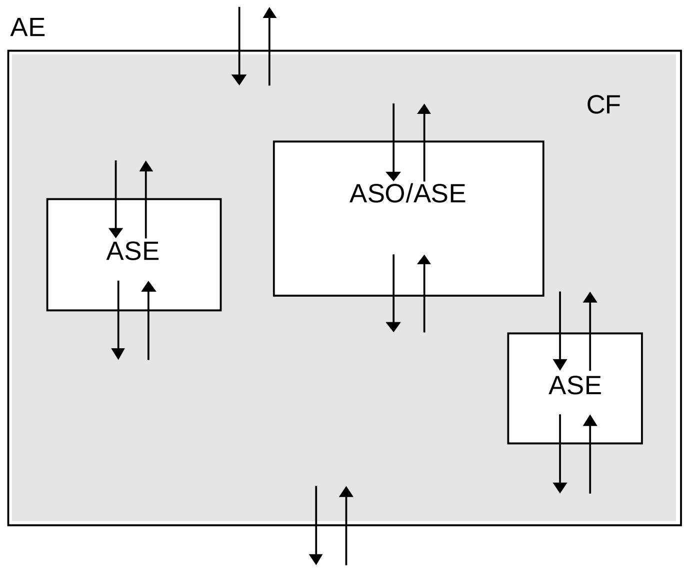


Figure 1.3: Generic Application Entity structure

The internal picture of the AE indicates that the AE comprises several ASEs. The AE picture is completed by the presence of an upper and lower service boundary.
Each ASE picture is similar in form, with an upper and lower service boundary. The job of the Control Function is solely to map inputs and outputs to and from ASE and AE service boundaries. By definition, the CF cannot produce protocol; in that case, an ASE is required.
The confirmed data service element (CDSE) is still undergoing requirements analysis. The CDSE is shown as an example of the inclusion in the ATN ULA of a common ASE constructed for use by all ATN applications.
In CNS/ATM-1, the upper AE service boundary is identical with the upper ASE service boundary. In CNS/ATM-1 all AE inputs are mapped to the ATN ASE.

1.2 Scope of document

This document is intended to describe the motivation and future migration of ATN Upper Layers.

The following sections draw together existing material on each of these areas, as well as providing additional summaries and analysis not found elsewhere.

1.3 Objectives

The aim of this document is to define the general communications architecture for ATN upper layer(s) (i.e. everything above the ATN Transport Service) and to provide reference material to aid the development and implementation of the upper layers.

The basic aim of this document is to define a set of architectural principles which will allow ATN Applications to be constructed in a standard way. This "building block" approach has many well-known advantages, including:

- the duplication of effort associated with designing and debugging similar functionality for many different application types is minimised
- the same type of design problem would otherwise have to be repeatedly solved for each new application
- the productivity of designers, programmers, system engineers and testers is increased, as they only have to deal with a single architecture
- the certification effort is eased, as experience is gained with previously accredited modules.

This work should stabilise and document the architectural basis for standardisation in ICAO, in particular considering:

- The approach to upper layer stack selection
- Preferred data encoding schemes
- Naming and Addressing
- Application Layer Structure

1.3.1 The ATN Upper Layer Environment

The OSI/ATN environment as it relates to upper layers and ATN Applications is depicted in Figure 1.1. This architecture effectively embraces two areas of standardisation:

a) Communications profiles of the upper layers to deliver the communications requirements for proposed applications; such profiles should be based on existing or new (defined by the ATN community) standards.

b) Application specifications, defining the messages and message sequence rules for applications to meet specific operational requirements, as set down in a completed application SARPs.

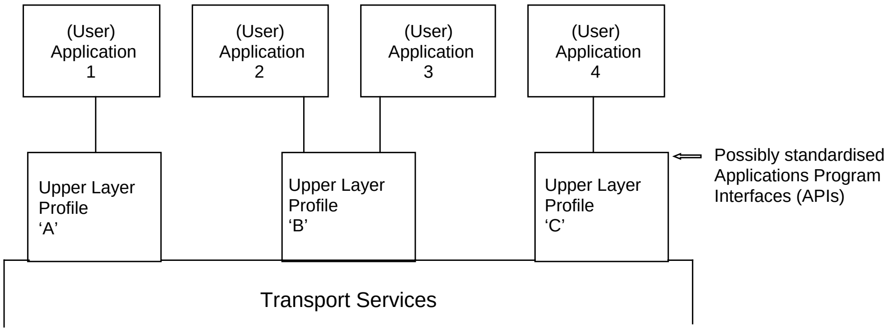


Figure 1.1 : ATN Host Software Environment

1.4 The Basic Architectural Concept

1.4.1 Scope

This section proposes generic communication services for use on the ATN. It describes the proposed services and suggests existing application layer standards which may be used to implement them. It also examines the application of the generic communication services to the ATN and considers some of the special constraints imposed on the ATN by the communications subnetworks.

1.4.2 Background

The framework for the standardisation of the upper layers puts forward the proposal that a set of standardised common communication services should be provided on which user applications, or so called Data Link Applications, would be constructed. These communication services will be used by the user applications to exchange information and, where thought appropriate, the interactions (i.e. message exchanges) which take place over these services would also be standardised in terms of the functions offered by the specific service.

The adoption of this framework means that the standardisation process may be subdivided into two parts; firstly, the standardisation of a common set of communication services, and secondly the standardisation of the DLA which uses those services. The aims of this framework are:

a) to keep to a minimum the number of standard application services

b) to use existing OSI application layer standards wherever possible, thus removing the need to define, standardise and conformance test new application layer standards

c) to tailor some of the service profiles to the underlying restrictions of some of the low bandwidth air-ground subnetworks in the ATN, but to use recognised application profiles wherever possible.

It is planned to define a limited set of communication service profiles for use in the aeronautical domain, to provide applications with access to the ATN. Each profile constitutes an upper layer "protocol stack" definition which when implemented provides the appropriate functionality in the selected upper layers.

Profiles are defined by selecting valid combinations of protocol standards and forming valid subsets in such a way as to deliver a specific level of service to the applications. A number of such profiles will be defined to provide wide applicability for the differing upper layer support requirements of different applications. Collectively, these profiles will provide a well-defined set of services which can be utilised when designing and implementing particular ATN applications. This does not imply that it is necessary or even desirable to implement the complete set of selected upper layer profiles on all end-systems. Subsets of the full set can be selected, to provide appropriate levels of functionality to meet the requirements of different classes of applications.

Inherent in this discussion is the desire to use standardised protocols whenever possible. Many of the necessary protocol standards already exist, and profiling is already being undertaken to produce International Standardized Profiles (ISPs). However, functionality or performance requirements may exist which are not satisfied by existing upper layer protocols. In such circumstances, it may be necessary to develop specialised upper layer protocol definitions, within the framework of the OSI reference model, for use in the aeronautical domain.

The adopted framework separates the communications profiling from the application standardisation and tries to standardise at the communications level only a small number of generic communications classes, which would be appropriate for use by a wide range of applications.

1.5 Benefits of this approach

The following benefits result from this approach:

a) It allows the separation of application specific and communication specific functions.

b) The certification of communications and applications software may be carried out separately.

c) It allows the use of a small number of standard communication services, based on distinct modes of interaction, by a large number of DLAs.

d) It does not require the definition of new application contexts, presentation transfer syntaxes,... whenever a new DLA is introduced or an existing DLA modified.

e) Some DLA communication requirements may be satisfied by existing ASEs and profiles.

f) The communication services may in some cases be based on COTS (Commercial Off The Shelf) products.

g) It does not require the implementation of application specific interactions (eg timeouts, message sequencing rules) within the communication service.

In the ATN Protocol Architecture, OSI application entities provide communications services to ATN applications. The service boundary between the application entities and the ATN applications is an abstract interface which may or may not be realised as an exposed interface in a real implementation. ATN profiles are defined to lie on the service-provider side of this boundary (see figure 2.1).

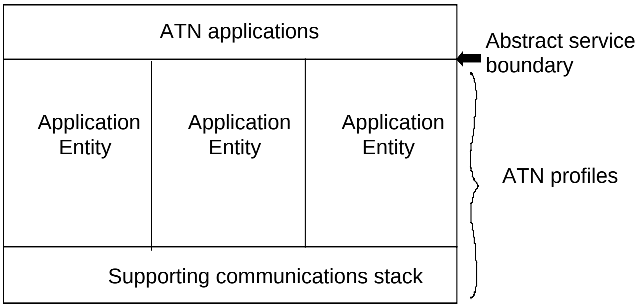


Figure 2.1 : Position of ATN Profiles within an End System

ATN applications are defined to use the services of the selected ATN profiles. The communication aspects of these applications will be defined in terms of:

a) the semantics and structure of the information to be exchanged (‘messages’)

b) the rules governing the dialogue between parties (message sequencing)

c) requirements imposed on the underlying communications services (quality of service, etc.)

1.6 Upper Layer Concepts and Structure

1.6.1 Scope

This section deals with the upper layer model. It covers the existing ASE structure and also considers the applicability of the revised OSI Application Layer Structure (ALS) model in the 2nd edition of ISO/IEC 9545 (also known as Extended Application Layer Structure or XALS) in terms of ASOs and associated Control Functions.

1.6.2 Functions of the Upper Layers

The service currently offered by the ATN Internet is a low-level communications service which corresponds to the OSI Transport Service defined in ISO 8072. Although it is possible to construct ATN Applications which make direct use the raw transport service, such applications will not benefit from the "common building block" approach.

It is therefore envisaged that a set of functions which add value to the raw ATN transport service will be standardised, and these will provide high-level services to the specific ATN Applications. Since it is a fundamental principle of the ATN that it adopts the protocols defined in the international standards for Open Systems Interconnection (OSI), it is logical to look to the standards defining the upper layers of the OSI model to provide the required value-added functions.

Before examining further alternatives, this section considers what the standard OSI upper layer (Session, Presentation and Application) protocols offer as added value on top of the raw transport service. It is these features that need to be incorporated in any ATN Application, or rejected as being unnecessary for a particular requirement.

One important, static function is to define formats and encodings for data interchange, in an unambiguous and open way, i.e. independent of any particular hardware bit-ordering or word-size conventions.

1.7 Migration Path of the ATN Upper Layers

1.7.1 The current standardisation work in the ATN Upper Layers focuses on the developments of ACSE, edition 3. The new work on ACSE allows for support of the extended application layer structure (XALS) concepts by ACSE. The work also allows ACSE to support multiple associations over a single TP4 connection. Upper Layer Structure

The fact that the upper layers (Session, Presentation and Application) are divided into three layers in the OSI reference model, and that the Application Layer is divided into discrete service elements, does not imply that this abstract division should be visible in real world implementations.

To assist in application design, a standardized application layer architecture has been developed as ISO/IEC 9545 Application Layer Structure (ALS). The architecture specifies the existence of the following major application layer components:

- *Application Process (AP)*: an element within a real Open System which performs the processing for a particular application. An example of an AP is an X.400 software package, of which some elements will be responsible for OSI communication and other elements will have responsibilities beyond the scope of the OSI environment, for example user interfaces and document filing systems. The AP can thus be considered to be partially in the OSI environment and partially in the local system environment. It may be modelled as a set of application entities (AEs).

- *Application Entity (AE)*: an aspect of an application process pertinent to OSI, and is the means by which application processes exchange information using defined application protocols and the underlying presentation service. An example of an application entity is an X.400 Message Transfer Agent (MTA) or Message Store (MS).

- *Application Service Object (ASO)*: the generic term for an object which performs some communications related task. For example, a file transfer system such as an FTAM initiator is an ASO. An AE is a particular kind of ASO, the outermost ASO. An ASO can contain further ASOs, which are thus recursively defined.

  Application Service Element (ASE): An AE can be broken down further into a number of ASEs, each of which provide a set of OSI communication functions for a particular purpose. An ASE is a particular kind of ASO which is elemental and therefore cannot be subdivided. An ASE may be thought of as a leaf node in a tree of ASOs. ASEs may provide general purpose functions which can be used by a number of applications. ASEs in general are defined in separate standards. For example, the *Association Control Service Element (ACSE)* is used to create and release associations between applications. The other application SARPs each define one or more ASEs.

- *Control Function (CF)*: this exists within an ASO to coordinate the use of the different services provided in the ASOs and ASEs, and also the use of external services such as the presentation service. It provides a mapping of the ASO to the subordinate ASOs and ASEs which it contains. This is defined in great detail in Chapter 3.

AEs, ASOs and ASEs are abstract representations of some part of an application. They cannot perform any action without first being invoked; this is equivalent to having a computer program that cannot do anything until it is run.

An *AE invocation (AEI)* can be thought of as an AE that is running; similarly an *ASO-invocation (ASOI)* and *ASE-invocation (ASEI)* are ASOs and ASEs that are running.

1.8 Application Layer

The application layer is the seventh and topmost layer in the OSI seven layer reference model. It makes use of the presentation service in order to cooperate with peer applications in other end systems, and provides services to the user (which may be a human or another application).

The application layer has a number of significant structural differences compared with the other layers. Each of the layers up to presentation provides a fixed standardized service to the layer above. The application layer however has many different functions (depending on the application), and may provide services either to the application user (which is equivalent to 'the layer above') or to other elements within the application layer itself.

Figure 1.1 illustrates at a high level the relationship between the various components of the application layer.

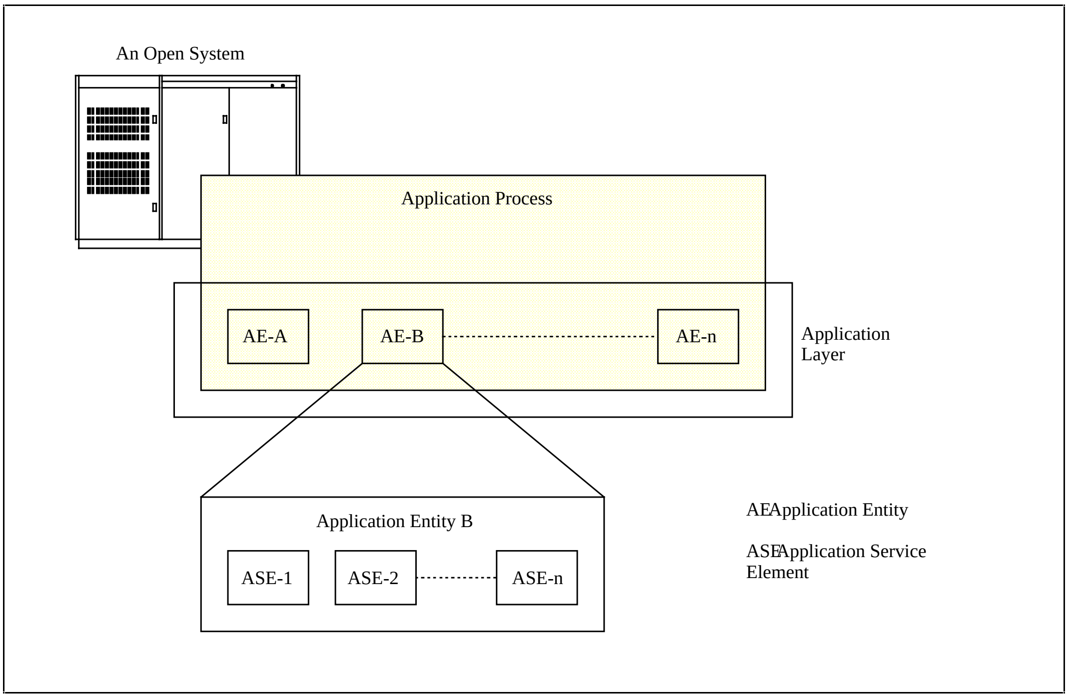


***Figure 1.1** : The Major Application Layer Components*

Figure 1.2 shows the various components of in the application layer as defined in the Application Layer Structure (ALS) model in ISO 9545, and shows how they are related.

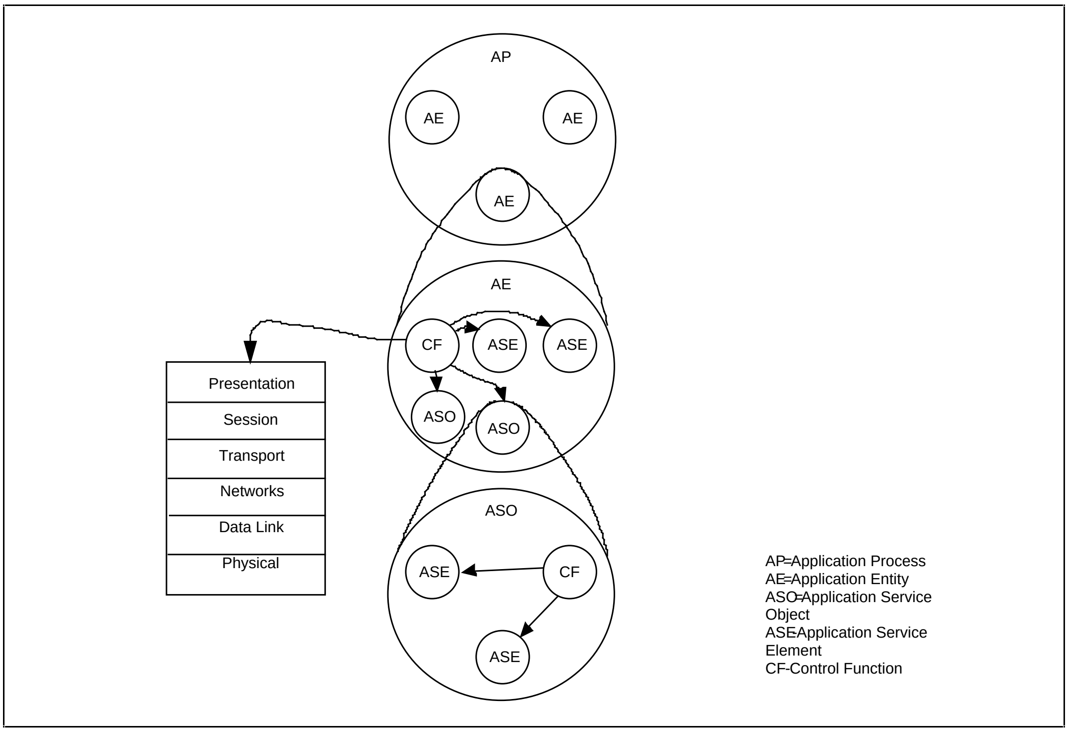


***Figure 1.2** : Components of the application layer as defined in ALS*

Associations

ASOs cannot cooperate until invoked. When two (or possibly more) ASOIs cooperate, the relationship between them is known as an ASO Association.

At any given time, an ASOI may have zero, one or more than one ASO associations with other ASOIs.

An ASO association is between two (or possibly more) ASOIs. These peer ASOIs are not necessarily of the same type, but must be of complimentary types. For example, if they are to exchange data, both must understand the same data syntax.

An ASO association is characterised by an ASO Context, which defines:

- the communications behaviour
- a set of rules and state information
- the number of ASOIs allowed in the ASO association
- how the ASO association can be started and finished

The ASOIs agree the ASO context before the ASO association is established. The ASO context may be identified by either defining it with the information listed above, or (more practically) by sending the identification of an ASO context that is well known or has been agreed beforehand.

The agreement of the ASO context is usually performed by the ACSE which is therefore one of the ASEs within the ASO.

An application association is a special type of ASO association which exists between an AE in one application and a peer AE in another. In a similar way, an application context is a special type of ASO context that characterises the application association. An application association underlies every other ASO association during the lifetime of the AEI. This is illustrated in figure 1.3.

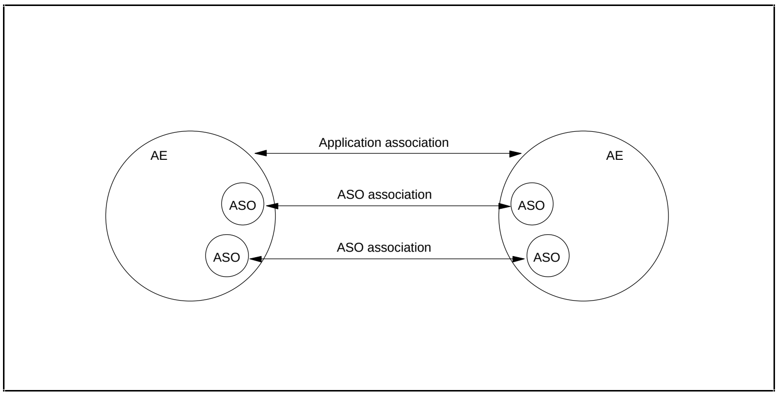


***Figure 1.3** : Application and ASO Associations*

2. Dialogue Service

2.1.1 Introduction

The dialogue service is a service that allows a user to bind to an association, send data, and unbind from the association.

2.1.2 Connection Mode

The dialogue service provides a connection mode upper layer service to the application. There is no connectionless mode upper layer architecture in CNS/ATM-1. Thus, there is no use of the connectionless transport protocol in CNS/ATM-1.

3. Application Entity

3.1.1 Control Function (Air-Ground)

The air-ground Control Function (CF) mapping function is described in the following figure. The five threads to implement the CF are described.

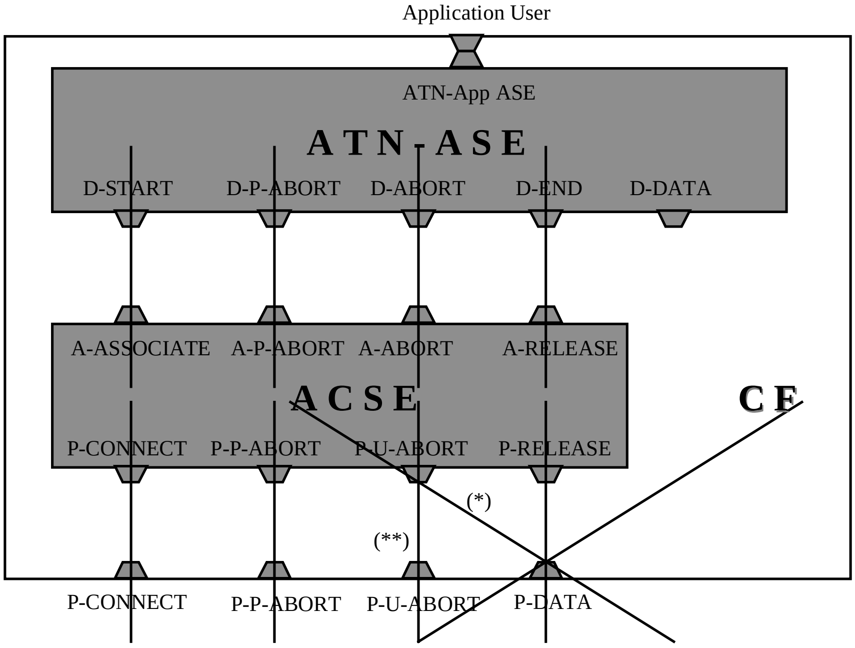
The D-START is mapped to the A-ASSOCIATE, which is mapped to the P-CONNECT by ACSE. The P-CONNECT is then mapped to the P-CONNECT (which is the T-CONNECT plus fast byte).

The D-P-ABORT occurs as an indication (‘up’) only. The P-P-ABORT is mapped to the A-P-ABORT. The A-P-ABORT is mapped to the D-P-ABORT.

The D-ABORT is mapped to the A-ABORT. The A-ABORT is mapped to the P-U-ABORT. The P-U-ABORT with user-data is mapped to the P-DATA which is followed by a P-U-ABORT. The P-U-ABORT without user-data is mapped to the P-U-ABORT.

The D-END is mapped to the A-RELEASE, which becomes a P-RELEASE. The P-RELEASE is mapped to the P-DATA.

The D-DATA is mapped to P-DATA, which in CNS/ATM-1 is syntactically equivalent to T-DATA.



(*) user data in the abort
(**) no user data in the abort

3.1.2 Naming and Addressing

3.2 Naming, addressing and registration

3.2.1 Scope

This section contains material on upper layer naming and addressing which is not covered in the ATN Manual (2nd edition).

3.2.2 Naming and Addressing Guidance

Naming refers to the identifier which must be assigned to any information object which may need to be referred to during information processing. Addressing refers to the physical location of a resource. To quote ISO/IEC TR 10730:

> *"Naming and addressing mechanisms are an essential aspect of OSI. Real Open Systems, even while being fully conformant to OSI protocols in all seven layers, may well be unable to set up a dialogue because of inconsistencies among their naming and addressing policies."*

3.2.3 Naming

Information objects which may need to be named include:

- application processes
- managed objects
- ATN application message types

AP, AE and Context Naming

In a given end system, application processes (APs) are the elements which perform the information processing for particular user applications. They are identified by AP-titles, which must be unambiguous throughout the OSI environment (OSIE), and thus require a Registration Authority to allocate names.

When APs in different end systems need to cooperate in order to perform information processing for a particular user application, they include and make use of communication capabilities which are conceptualised as application entities (AEs). An AP may make use of communication capabilities through one or more AEs, but an AE can belong to only one AP.

Application entities are each named in terms of a unique Application Entity Title (AET), which consists of an AP-title and an AE Qualifier.

AP-titles and AE-qualifiers may be assigned either an attribute-based name form or an Object Identifier name form. When an AP-title is allocated an attribute-based name form, all of the associated AE-qualifiers must also be assigned an attribute-based name form; when an AP-title is allocated an Object Identifier name form, all of the associated AE-qualifiers must also be assigned an Object Identifier name form.

It may at times be necessary to make a distinction between the various invocations of a given AP running concurrently on an Open System. Thus it is possible, say, to have a pool of generic application servers, and when a server is allocated to perform a particular task, it is identified via its Invocation-identifier. This is done through the use of AP-invocation-identifiers which must be unambiguous only within the scope of the AP, and thus do not have to be registered.

Similarly, it may be necessary to distinguish between the various invocations of a given AE running concurrently as part of a given AP. This is done through the use of AE-invocation-identifiers which must be unambiguous within the scope of the {AP-invocation, AE} pair and thus do not have to be registered.

For communication purposes, AE-invocations have to handle one or more Application Associations. These can be identified by Application-Association-Identifiers, which need only be unambiguous within the scope of the cooperating AE-invocations, and thus do not have to be registered.

In connection-mode communications, the AET can be used in called, calling and responding application address parameters in A-ASSOCIATE (and RT-OPEN) service primitives. The ACSE service provides for the optional specification of an AET value by its component values (AP-title and AE-qualifier) in A-ASSOCIATE primitives.

The calling/called AP title identifies the AP that contains the requester/acceptor of the A-ASSOCIATE service. The AE qualifier identifies the particular AE of the AP that contains the requester or the acceptor of the A-ASSOCIATE service. The AP and AE Invocation-identifiers identify the AP invocation and AE invocation that contain the requester or the acceptor of the A-ASSOCIATE service. The presence of each of these addressing parameters is defined in ISO standards as a user option.

Other information objects which must be named are the Application Context name and Presentation Context identifier which are exchanged at connect-time.

3.2.4 Addressing

According to ISO 7498-3 (Basic Reference Model - Naming and Addressing Addendum), a Presentation Address comprises a globally unique NSAP address, plus local Transport selector, Session selector and Presentation selector. The Session selector and Presentation selector are not utilized in CNS/ATM-1 Package.

The only mandatory addressing parameters in the A-ASSOCIATE service primitives are the calling, called and responding Presentation Addresses. These are passed transparently by ACSE (and RTSE) to the Presentation Service.

3.2.5 Name - Address Mapping

Any OSI layer entity or application process can be named via a title. This must be translated into an address by means of a directory function either at Application or Network Layer. Each AE is attached to one or more PSAP and hence the AET is associated with the corresponding Presentation Address(es). The AET is mapped onto a Presentation Address by means of an Application Layer directory function explained later in this document. The Application Layer directory function provides a mapping from an AET into the PSAP address required to access the referenced application entity.
The use of selectors is a local function and there may in practice be a direct correspondence between application entity titles and TSAP address or NSAP address.

3.2.6 Registration Issues

It has been agreed that the ISO 9834 registration scheme should be put forward for adoption by ICAO.

A number of situations have been identified where object identifiers (OIDs) are being interchanged; some of these are registered elsewhere, some will need registration by ICAO. A given object should only be assigned one OID, ie. it should only be registered once (either by ICAO or by some other organization).

ICAO is in the process of setting up a registration authority under ISO. ICAO Working Groups will need to register information objects including ASOs, ASEs, Application Titles, Presentation Contexts through such a registration authority. It may also be necessary to set up a registration authority for Distinguished Names, as used by the Directory service and by systems management.

3.2.7 Mapping to transport

The ATN ULA makes extensive use of the user-data capability of the transport service. The ULA attempts to map the AARQ (containing user data) to the T-CONNECT. If this is not possible, based on user-data size, the T-CONNECT is sent, and the T-DATA is used to convey the AARQ (including user data). The implementor is advised to consult the PDU calculations in the present Guidance Material, but generally if the DS-User places more than five octets of user-data in the D-START, the D-START is mapped to the T-CONNECT + T-DATA.

4. Session

A detailed evaluation of the overheads incurred by use of the OSI upper layer protocols is presented in this chapter. In making this analysis, the following assumptions are made:

4.1.1 Basic Assumptions

a) "ACSE" refers to the second edition of ISO 8650 (1995).

b) Session Version 2 is used.

c) Unless otherwise stated, all OPTIONAL items are omitted from PDU size calculations.

4.1.2 Encoding Options

e) ASN.1 Packed Encoding Rules (PER) in this paper refers to the Basic Unaligned variant of ISO/IEC 8825-2. By imposing limits (bounds) on value ranges in the abstract syntax, use can be made of constrained and semi-constrained whole number coding (for lengths and integers).

f) ASN.1 type EXTERNAL is taken to be as defined in ISO/IEC 8824-1 (1992).

g) When a SEQUENCE OF SEQUENCE is encoded by ASN.1/PER it is assumed that the lower bound is zero and the upper bound is 255. In this way just one octet is required to specify the component count.

4.1.3 Name and Address Options

h) Where object identifiers (OIDs) are used, it is assumed that they are constrained such that no more than five arcs will be present and the component values will be small. This enables the OID to be encoded in 4 octets, with a length field of 1-octet (for BER) or 2-bits (for PER).

i) An AP-title is assumed to be an OID and an AE-qualifier is assumed to be a small value Integer (less than 128).

j) Both the Presentation Selector and Session Selector have a value of NULL..

4.1.4 Presentation Context

k) When A2CSE is used, even the presentation-context-definition-list in Presentation CP and CPA PDUs are omitted 1 (A2CSE provides this information in its own PDUs).

m) It is assumed that Presentation Context Identifiers are small valued integers (less than 128). One presentation context only is specified, this is for ACSE. If the application needs to specify a distinct abstract syntax, then the addition of this adds 15 octets (when selecting ASN.1/BER) or 17 octets (when selecting ASN.1/PER), as illustrated in Annex C.

---

[1] Strictly the presence of the presentation-context-definition-list field is mandatory, as ISO 8823 Section 6.2.2.7 states "This shall be a list containing one or more items."

n) Only one transfer syntax (ASN.1/BER or ASN.1/PER as appropriate) is negotiated.

4.1.5 User Data

o) Presentation user data is simply-encoded-data, that is an OCTET STRING.

p) The effect of application user data is shown as an option. User data is sent as a fully-encoded BIT STRING and so its structure will be the subject of bilateral agreement.

4.2 Session Layer Definitions

The PPDUs defined for the short-encoding and the null-encoding options of the session protocol are indicated by redlining.

CONNECT (CN) SPDU SI = 13						
PGI	**m/nm**	**Code**	**PI**	**m/nm**	**Code**	**Octets**
Connection identifier	nm	1	Calling SS-user reference	nm	10	64 max
			Common Reference	nm	11	64 max
			Additional reference information	nm	12	4 max
Connect / Accept item	nm	5	Protocol options	m	19	1
			TSDU maximum size	nm	21	4
			Version number	m	22	1
			Initial serial number	nm	23	6 max
			Token setting item	nm	26	1
			Session user requirements	nm	20	2
			Calling session selector	nm	51	16 max
			Called session selector	nm	52	16 max
User Data	nm	193				512 max
			Data overflow	nm	60	1
Extended user data	nm	194				10240 max

SHORT CONNECT (SCN) SPDU						
	m/nm	**Octet**	**Parameter**	**m/nm**	**Value**	**Bir number**
SI&P	m	1	SI	m	101101	4-8
			Parameter indication	m	0	3

DATA TRANSFER (DT) SPDU SI = 1						
PGI	**m/nm**	**Code**	**PI**	**m/nm**	**Code**	**Octets**
			Enclosure item	nm	25	1

User Information Field			unlimited

TYPED DATA TRANSFER (TD) SPDU SI = 33						
PGI	**m/nm**	**Code**	**PI**	**m/nm**	**Code**	**Octets**
			Enclosure item	nm	25	1
User Information Field						unlimited

ACCEPT (CNA) SPDU SI = 14						
PGI	**m/nm**	**Code**	**PI**	**m/nm**	**Code**	**Octets**
Connection identifier	nm	1	Called SS-user reference	nm	9	64 max
			Common Reference	nm	11	64 max
			Additional reference information	nm	12	4 max
Connect / Accept item	nm	5	Protocol options	m	19	1
			TSDU maximum size	nm	21	4
			Version number	m	22	1
			Initial serial number	nm	23	6 max
			Token setting item	nm	26	1
			Token item	nm	16	1
			Session user requirements	nm	20	2
			Calling session selector	nm	51	16 max
			Responding session selector	nm	52	16 max
			Enclosure item	nm	25	1
User Data	nm	193				

SHORT ACCEPT (SAC) SPDU						
	m/nm	**Octet**	**Parameter**	**m/nm**	**Value**	**Bir number**
SI&P	m	1	SI	m	111101	4-8
			Parameter indication	m	0	3

FINISH (FN) SPDU SI = 9						
PGI	**m/nm**	**Code**	**PI**	**m/nm**	**Code**	**Octets**
			Transport disconnect	nm	17	1
			Enclosure item	nm	25	1
User Data	nm	193				

DISCONNECT (DN) SPDU SI = 10						
PGI	**m/nm**	**Code**	**PI**	**m/nm**	**Code**	**Octets**
			Enclosure item	nm	25	1
User Data	nm	193				

<Editor’s Note **--** New ISO short PDUs, SRFC and SACC will be added.>

<u>Encoding of the SHORT CONNECT PDU (SCN)</u>

```
-----
-- Short-encoding option (”Fast Byte”) : SHORT CONNECT PDU (SCN)
-----
SI = B`101101’                          -- Session Short CN
P  = B`0’                   -- no parameter
             -- user-information (See section 5.2)
```

With no user data, the SCN overhead is thus 1 octets.

<u>Encoding of the SHORT CONNECT ACCEPT PDU (SCA)</u>

```
-----
-- Short-encoding option (”Fast Byte”) : SHORT CONNECT PDU (SCA)
-----
SI = B`101101’                          -- Session Short Accept
P  = B`0’                   -- no parameter
```

With no user data, the SCA overhead is thus 1 octets.

4.2.1 Session Defect Report

(Not a Defect, but needs clarification) In the ISO DAM text, the short-connect/null-encoding and upper-layer-context-id approach use the short-accept/refuse and the short-accept/refuse-continue in opposite orders. Thus, short-connect sends the short-accept-continue and then the short-accept, while the upper-layer-conext-id sends the short-accept-continue and then the short-accept.

4.2.2 Mapping to transport

The ATN ULA makes extensive use of the user-data capability of the transport service. The ULA attempts to map the AARQ (containing user data) to the T-CONNECT. If this is not possible, based on user-data size, the T-CONNECT is sent, and the T-DATA is used to convey the AARQ (including user data). The implementor is advised to consult the PDU calculations in the present Guidance Material, but generally if the DS-User places more than five octets of user-data in the D-START, the D-START is mapped to the T-CONNECT + T-DATA.

5. Presentation

5.1.1 Encoding

5.2 Presentation Layer Definitions

5.2.1 Presentation PCI

The PPDUs defined for the short-encoding and the null-encoding options of the presentation protocol are indicated by redlining.

```
CP-type ::= SET                    -- Presentation Connect PDU
{
[0]          IMPLICIT Mode-selector,
[1]          IMPLICIT SET {COMPONENTS OF Reliable-Transfer-APDUs.RTORQapdu} OPTIONAL,
[2]          IMPLICIT SEQUENCE
```

```
        {
        [0]     IMPLICIT Protocol-version DEFAULT {version-1},
        [1]     IMPLICIT Calling-presentation-selector OPTIONAL,
        [2]     IMPLICIT Called-presentation-selector OPTIONAL,
        [4]     IMPLICIT Presentation-context-definition-list OPTIONAL,
        [6]     IMPLICIT Default-context-name OPTIONAL,
        [8]     IMPLICIT Presentation-requirements OPTIONAL,
        [9]     IMPLICIT User-session-requirements OPTIONAL,
                User-data OPTIONAL
        } OPTIONAL
}
```

SHORT-CP PDU

The PCI of the SHORT-CP is one octet, with the two trailing bits consisting of the encoding-choice parameter. This PCI is followed by the User-data parameter (encoded as per the encoding-choice parameter, cf next section). The encoding of the SHORT-CP is as shown in the following bit pattern:

0000 00zz

where zz identifies the encoding choice as follows:
00: bilateral agreement
01: BER
10: unaligned PER
11: aligned PER

```
CPA-PPDU ::= SET                -- Presentation Connect Accept PDU
{
[0]  IMPLICIT Mode-selector,
[1]  IMPLICIT SET {COMPONENTS OF Reliable-Transfer-APDUs.RTOACapdu} OPTIONAL,
[2]  IMPLICIT SEQUENCE
        {
        [0]     IMPLICIT Protocol-version DEFAULT {version-1},
        [3]     IMPLICIT Responding-presentation-selector OPTIONAL,
        [5]     IMPLICIT Presentation-context-definition-result-list OPTIONAL,
        [8]     IMPLICIT Presentation-requirements OPTIONAL,
        [9]     IMPLICIT User-session-requirements OPTIONAL,
                User-data OPTIONAL
        } OPTIONAL
}
```

SHORT-CPA PDU

The PCI of the SHORT-CPA is one octet, with the two trailing bits consisting of the encoding-choice parameter. This PCI is followed by the User-data parameter (encoded as per the encoding-choice parameter, cf next section). The encoding of the SHORT-CP is as shown in the following bit pattern:

0000 00zz

where zz identifies the encoding choice as follows:
00: bilateral agreement
01: BER
10: unaligned PER
11: aligned PER

5.2.2 Presentation User Data

The encoding of the presentation User data required by the CNS/ATM-1 ULA is redlined.

```
User-data ::= CHOICE
{
        [APPLICATION 0] IMPLICIT Simply-encoded-data,
        [APPLICATION 1] IMPLICIT Fully-encoded-data
}

Simply-encoded-data ::= OCTET STRING

Fully-encoded-data ::= SEQUENCE
                              SIZE (1,...)
                              OF PDV-list

PDV-list ::= SEQUENCE
          {       Transfer-syntax-name OPTIONAL,
                  Presentation-context-identifier,
                  presentation-data-values    CHOICE {
                       single-ASN.1-type     [0] ANY,
                       octet-aligned  [1] IMPLICIT OCTET STRING,
                       arbitrary             [2] IMPLICIT BIT STRING
                       }
        }
```

The Transfer-syntax-name field shall not be present in the encoded presentation User Data.
The ”arbitrary” choice for presentation-data-value shall be used in the encoded presentation User Data.
The values of the Presentation-context-identifier are predefined as follows: 0 (acse-apdu), 1 (reserved for future use), 2(user-ase-apdu), other (reserved for future use).

```
Presentation-context-definition-list ::= Context-list

Context-list ::= SEQUENCE OF SEQUENCE
{
          Presentation-context-identifier,
          Abstract-syntax-name,
          SEQUENCE OF Transfer-syntax-name
}

Presentation-context-identifier ::= INTEGER
                                    (1..127,...)
```

```
Presentation-context-definition-result-list ::= Result-list

Result-list ::= SEQUENCE OF SEQUENCE
{
          [0] IMPLICIT Result,                -- INTEGER(0..2)
                  -- Transfer-syntax-name shall be present if Result is "acceptance"
          [1] IMPLICIT Transfer-syntax-name OPTIONAL,
                  -- provider-reason shall be present if Result is "provider-rejection"
          provider-reason   [2] IMPLICIT INTEGER
                            {
                            reason-not-specified (0),
```

```
                                  abstract-syntax-not-supported (1),
                                  proposed-transfer-syntaxes-not-supported (2),
                                  local-limit-on-DCS-exceeded (3)
                                  } OPTIONAL
}

Result ::= INTEGER {
                    acceptance (0),
                    user-rejection (1),
                    provider-rejection (2)
          }
```

Encoding of the SHORT CP

```
-----
-- Short-encoding option (”Fast Byte”) : SHORT CONNECT PDU (SCP)
-----
PCI = B`000000’                          -- Presentation Short CP
zz   = B`10’                             -- transfer syntax = PER
           -- user-information (See section 5.2)
```

With no user data, the SCP overhead is thus 1 octets.

Encoding of the SHORT CPA

```
-----
-- Short-encoding option (”Fast Byte”): SHORT CONNECT ACCEPT PDU (SHORT-CPA)
-----
PCI        = B`000000’                          -- CPA-PPDU
zz         = B`10’                                 -- transfer syntax = PER
```

With no user data, the SHORT-CPA overhead is thus 1 octet.

5.2.3 Session Connection Accept [CNA]

The session user requirements are used to indicate the functional units proposed by the Session Connection Request (see previous calculation for CN-SPDU).

```
SI = 0x0E                                   -- CNA-SPDU
LI = 0x??
        PGI = 0x05                          -- Connect/Accept item
        LI = 0x06
                PI = 0x13                   -- Protocol Options
              LI = 0x01
              PF = 0x00
                PI = 0x16                   -- Version Number
              LI = 0x01
              PF = 0x02
        PI = 0x14                           -- Session User Requirements
        LI = 0x02
        PF = 0x00 0x02                            -- See Text
        PGI = 0xC1 | 0xC2                   -- User Data
        LI = 0x?? | 0xFF 0x?? 0x??
        PF = xx xx ... xx xx                -- PPDU
```

The Basic Session-CNA Header Size is 14 octets. If the Session User data is not greater than 512 octets, then the data header adds another 2 octets; otherwise 4 octets are added, giving a header size of:

- 16 octets (short user data)
- 18 octets (long user data)

5.2.4 Presentation Defect Report

The CPR and CPA have identical bit-encoding for the case where refusal code = 0. (presentation user)

The short-connect and null-encoding options are not separate options in the current text.

5.2.5 Presentation Encoding Guidance

On receipt of a S-CONNECT indication, the receiving PPM shall figure out whether or not the short-connect encoding option has been selected by the sending PPM. This determines the contents of the User Data parameter of the S-CONNECT indication primitive: either a CP or a SHORT CP.

The SCP will always be 0000 00zz (and in our case zz will always be 10 - unaligned PER).

The CP will be encoded as follows:
a) in the case of BER-encoding, the first octet will be 31H (SET)
b) in the PER-encoding case, the first two bits will define which of the 2 optional set components are present. 8823-1 is not very clear, but it appears from 6.2.2.7 that the optional sequence containing the parameters for the CP must always be present in normal mode. Therefore the first 2 bits of the CP will be 01.

So the first 4 bits of the User Data parameter of the S-CONNECT indication *are* sufficient to identify the type of PPDU:
0000 - it is a SCP
0011 - it is a CP (BER-encoded)
0100 - it is a CP (PER-encoded)

In fact the ISO PDAM text says in 6.2.6.8 "negotiation of [the PER] protocol option is not always possible. ... [Encoding the CP PPDU using PER] is only suitable if it is known by bilateral agreement that the PER protocol option is supported by the responder.

5.2.6 Presentation Fast Byte Guidance Material (OSIEFF)

Null functionality at a layer refers to the case where no functionality is required of a layer during the data transfer phase but where OSI compatibility and compliance are required. This is the case which is most clearly applicable to the ITU-T applications which use “short stacks”, to permit much greater OSI compatibility while at the same time allowing efficient communications. While it is possible to use the normal OSI layer protocol to signal that null functionality shall be required in the data phase, in certain instances, it is also possible to use a different protocol which is considerably more efficient (in terms of byte efficiency and, possibly, connection-establishment efficiency) to perform the negotiation. The term “fast byte” has been employed, as a convenient mnemonic, to refer to the insertion of a single byte PCI at connection establishment to signal that no further PCI will flow for that instance of communication. The use of the fast byte at a layer therefore serves to provide a service mapping between the layer above it and the layer below it.

NOTE — Early discussions on the fast byte concept considered the possibility of using the byte — theoretically only a bit — to serve as a *placeholder* so that it was possible, at some point during the instance of communication, to use the normal layer protocol to re-negotiate the use of some layer functionality where none was required earlier. At least for the upper layers, such a dynamic re-negotiation of layer functionality is for further study.

Thus, if a transport layer fast byte is exchanged, the layer service remains the same, i.e., the transport fast byte is a different version of the transport protocol with a one-to-one mapping of the network services to the transport services. In other words, by using the transport fast byte, one gets a QoS which is only as good as that provided by the underlying network service. The lower layer fast bytes are particularly useful for cases where the applications are communicating over a single data link, as is the case with ISDN access signalling.

For the upper layers, the typical, normal OSI implementation requires a 13 to 20 octet overhead on a single presentation data value (pdv) using the presentation and session data transfer services. This overhead is necessary to identify the state of the communication (i.e., that it is the data transfer phase as opposed to, say, the release phase), and to identify the pdv as belonging to a particular presentation context. Clearly, a null PCI optimization for the data phase implies a reduction in the layer service available to the application. For instance, in this case where all the application data is carried directly as user-data of the transport service, there is no guarantee that an encoded application pdu will not resemble a session spdu; therefore, null PCI for the session data transfer phase implies that it is not possible to distinguish session spdus from application PCI. Therefore, it is not possible to use the orderly release facility of the session layer, though, of course, the application protocol can be defined to perform this function. Similarly, null PCI for presentation data transfer implies that there can only be one presentation context for the application pdus, whose abstract transfer syntaxes are known a priori. Thus, reducing the upper layer functionality inherent in the null functionality data phase restricts the range of applications that can use this optimization.

This loss of functionality must be reflected to the user at the service interface. For the session and presentation layers, the layer services are bundled together in groups known as functional units. At this time, orderly release of the session connection is provided as a part of the mandatory kernel functional unit. The use of null encoding for the data phase requires that the users have negotiated the use of a new functional unit, the so-called no orderly release functional unit, which removes the orderly release from the kernel functional unit.

NOTE — The orderly release capability would more logically be a functional unit separate from the kernel; the new "negative" functional unit provides compatibility with the current specificatons that require the (non-negotiable) kernel to be indivisible.

To this end, ITU-T Rec. X.pfbs defines the pass-through access to the session service, in particular the (new) no orderly release functional unit. As the presentation layer uses the session layer services for release of the presentation connection, there is no reduction to the presentation services. Thus efficiency optimizations available at the presentation layer are new protocol options, i.e., alternative, efficient PCI and procedures.

ITU-T Rec. X.pfbp define two protocol options at the presentation layer that greatly reduce the quantity of presentation PCI in cases where the presentation user's requirements for presentation functionality are limited. The null-encoding protocol option provides an alternative presentation protocol option for data transfer with zero PCI which can be negotiated at connection establishment only if one of the following conditions described below is true:

a) the presentation context definition list contains precisely one item in which the abstract syntax name is known to the responding presentation protocol machine by bilateral agreement; or

b) the presentation context definition list is empty and the default context is known by bilateral agreement; or

c) the presentation context definition list is empty and the abstract syntax of the default context is known to the responding presentation protocol machine by bilateral agreement and is specified in ASN.1.

NOTE — It may be possible in the future to negotiate the null-encoding protocol option for efficient data transfer using the presentation protocol defined in ITU-T Rec. X.226 (1994) | ISO/IEC 8823-1:1994. It is left for further study to define an alternative version of the presentation protocol encoded using PER which will permit byte-efficient presentation negotiation of the full set of presentation functionality.

In addition, it is possible to use another protocol option, the short-encoding option, which defines encodings for some presentation PPDUs which are considerably shorter than the current ones if *both* conditions d) and e) described below are true:

d) the calling and called presentation selectors are null; and

e) the presentation-requirements parameter in the P-CONNECT service includes the kernel functional unit only.

The short-encoding protocol option allows the negotiation of the encoding rule which shall be used as the transfer syntax of the application PCI belonging to the single presentation context (which may be the default context) from one of BER, the aligned and unaligned variants of PER or a "transparent" encoding which is understood by bilateral agreement.

ITU-T Rec. X.sfbs specifies the no orderly release functional unit, whose selection by the session user indicates that the user has no requirements for orderly release of the session connection. Thus, either the application protocol has been chosen to perform this function, or the application association (which is one-to-one with the underlying session connection) is released by disconnecting the transport connection or by an abortive release of the session connection. The selection of this functional unit by the initiating session user permits the initiating session protocol machine to offer the use of the null-encoding protocol option on the established session connection. The responding session protocol machine can accept this option if the responding session user has selected only (and nothing other than) the kernel, full-duplex and no orderly release functional units for use on the connection.

ITU-T Rec. X.sfbp describes how the negotiation of the null encoding protocol option can be done using the protocol options field of the session establishment SPDUs defined in ITU-T Rec. X.225 (1995) | ISO/IEC 8327-1:_____. However, ITU-T Rec. X.sfbp also defines the possibility of using the short-encoding protocol option for the establishment SPDUs, which define a one byte PCI for these SPDUs which are distinct from the leading octet of the current SPDUs, which provides a byte-efficient negotiation of the null-encoding protocol option provided that there is no session layer addressing information required to be exchanged, i.e., the session selectors are null.

It is expected that the short-encoding protocol option will be used in conjunction with the transport connection set-up to achieve interworking with current implementations and, for the case where the responder also implements this protocol option, achieve an improvement in round-trip efficiency by setting up the upper layer connections concurrently with the transport connection. This is achieved as follows: the SHORT CONNECT SPDU — which is the short-encoding version of the current session CONNECT SPDU — is sent as user data of the T-CONNECT request service primitive. This requires that the SHORT CONNECT SPDU plus any accompanying user data meet the 32 octet limitation on the size of the transport user data.

NOTE — The transport protocol class 0 does not permit the carriage of user data. Therefore, for this scenario to work, the transport protocol clsss 4 should be available at both ends, or the transport fast byte protocol should be employed.

Current session implementations ignore any user data on the T-CONNECT indication primitive, or, at worst, disconnect the transport connection. Thus, absence of any user data on the T-CONNECT confirm primitive is a signal to the initiating session protocol machine that the responder is an implementation of the current standards. If the responding session entity implements the short-encoding protocol option, the SHORT ACCEPT SPDU is sent as user data of the T-CONNECT response service primitive, and its receipt by the initiating session protocol machine completes the session connection establishment in tandem with the transport connection establishment.

Of course, the short-encoding option may be used with the T-DATA service for the case where an already established transport connection is assigned to the session connection. Interworking is not fully achieveable as there is no guarantee that the responding session entity, if based on the current standards, will send a REFUSE SPDU to signal a protocol error, which is what a short-encoding for an SPDU would be.

6. ACSE

Upper Layer Services and Protocols

Application layer standards define the services and protocols supported by ASEs. Some of these standards define services which are common to a range of application layer standards, and which can be used as "building blocks" when defining new applications. The most significant of these are summarised below; others define the specific services and protocols supported by particular application layer standards such as electronic messaging and file transfer.

Association Control Service Element (ACSE)

The ACSE service allows one application-specific AE (for example, an X.400 MTA) to request that an association be established with a remote AE (in this example, an adjacent X.400 MTA) for the purpose of information transfer. ACSE will attempt to establish the association using the supporting OSI layers, and will report the success or failure of this attempt to the requesting AE. ACSE also provides for the orderly and abrupt release of the association once established.

The connection-oriented ACSE service and protocol are defined in ISO 8649 (CCITT X.217) and ISO 8650 (CCITT X.227) respectively. The connectionless ACSE protocol is defined in ISO 10035 (CCITT X.237), in which the request to establish an association, the information to be exchanged and the request to release the association are all transferred together.

The following ACSE functional units are defined:

- Kernel
- Authentication

- Application Context Negotiation

ASO-Association Control Service Element (AACSE, or "A2CSE")

There is currently a project under way within ISO/IEC JTC 1/SC 21 to modify the ACSE standards to support the revised ALS model more directly by allowing associations between named ASOs to be set up explicitly and to perform the context demarcation functions required for ASO-associations

6.1

6.1.1 Discussion of differences in ACSE editions

As the ACSE is the major part of available software in the ULA implementation, a description is provided of the evolution of the ACSE standard. The editor's preface to ISO 8649, Service Description, was amended three times from the first edition to the second edition. The three amendments are 1) Peer-entity Authentication during Association Establishment, 2) Connectionless ACSE Service, and 3) Application Context Name Negotiation. There were also three technical corrigenda (Tcs) cited. The most important of the TCs resolved a defect wherein EXTERNAL events could affect ACSE sequencing and state machine. The EXTERNAL events were added to ACSE in a TC to answer a defect that pointed out that a session resynchronization could purge (destroy) the session finish or disconnect that the A-RELEASE is traveling on. Now, clearly over the ATN ULA of strict fast-byte supporting upper layers none of this matters, since there are no external events. A classic implementation, though, must put in a few EXTERNAL hooks at the bottom of its state machine when things go wrong in classic session/presentation.

ISO 8650-1, edition 2, has two Amendments and four TCs noted in the Editor's Preface. AM1 is Authentication, AM2 is Application Context Name negotiation, and TC2 is the 'EXTERNAL' TC.

The changes in the ACSE protocol specification are roughly similar, e.g., for TC2, "A-RELEASE procedure is disrupted if P-RESYNCHRONIZE, P-U-EXCEPTION-REPORT, or P-EXCEPTION-REPORT primitives occur on the association".

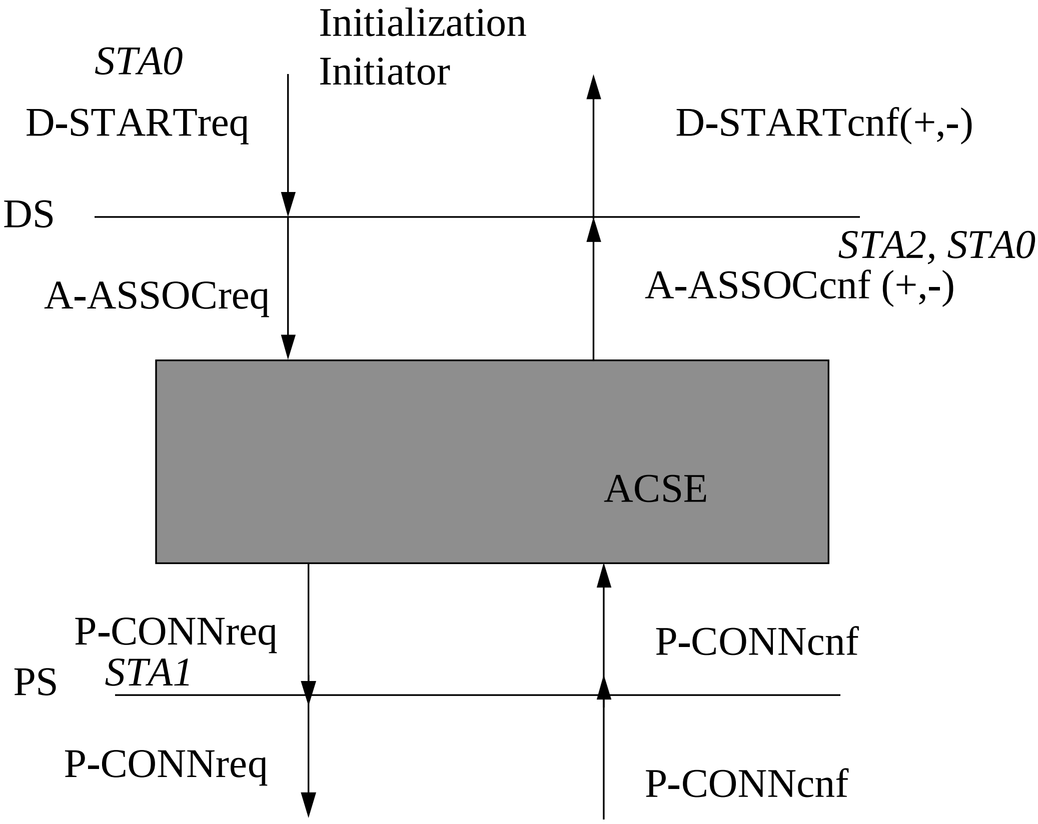
The state machine also adds two stimuli for EXTERN-1 and EXTERN-2. These stimuli cause the ACPM to return to the Associated state from one of the Attempting Release state.

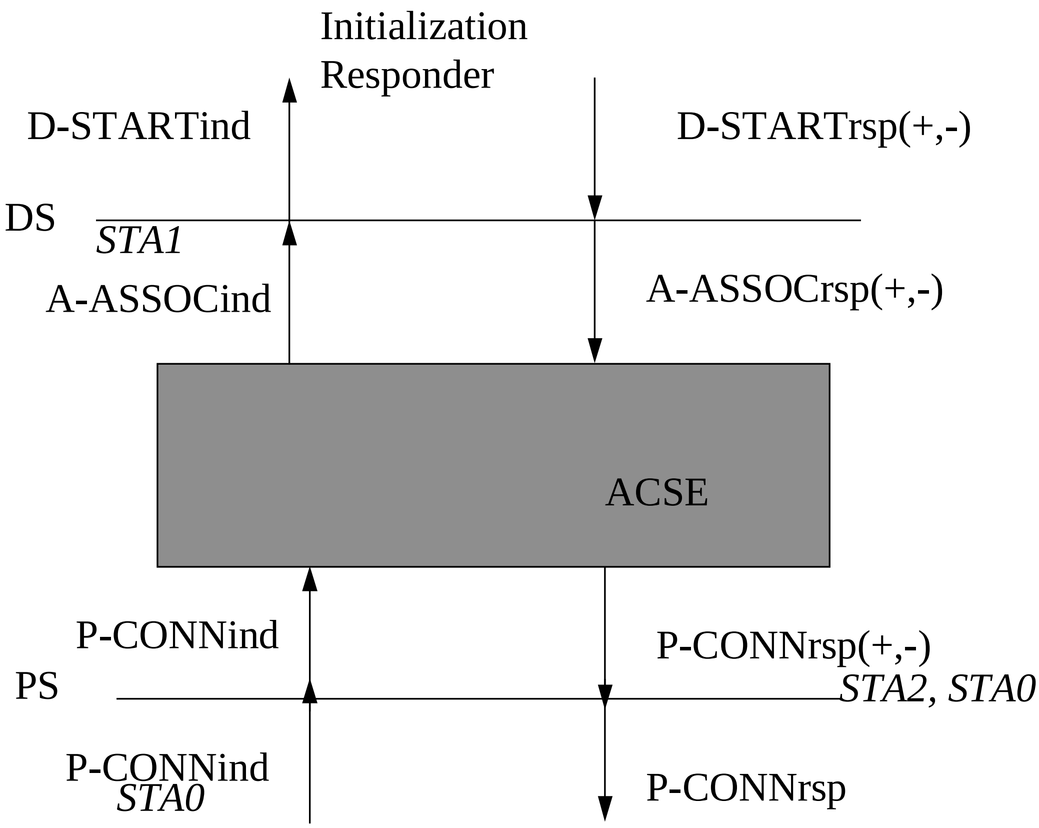
The discussion indicates that the CNS/ATM-1 ULA requires none of the changes that distinguish ACSE, edition 1 from ACSE, edition 2.

The two changes to ACSE that are required are the requirements to encode the ACSE PDUs in PER, and to map the P-RELEASE to P-DATA.

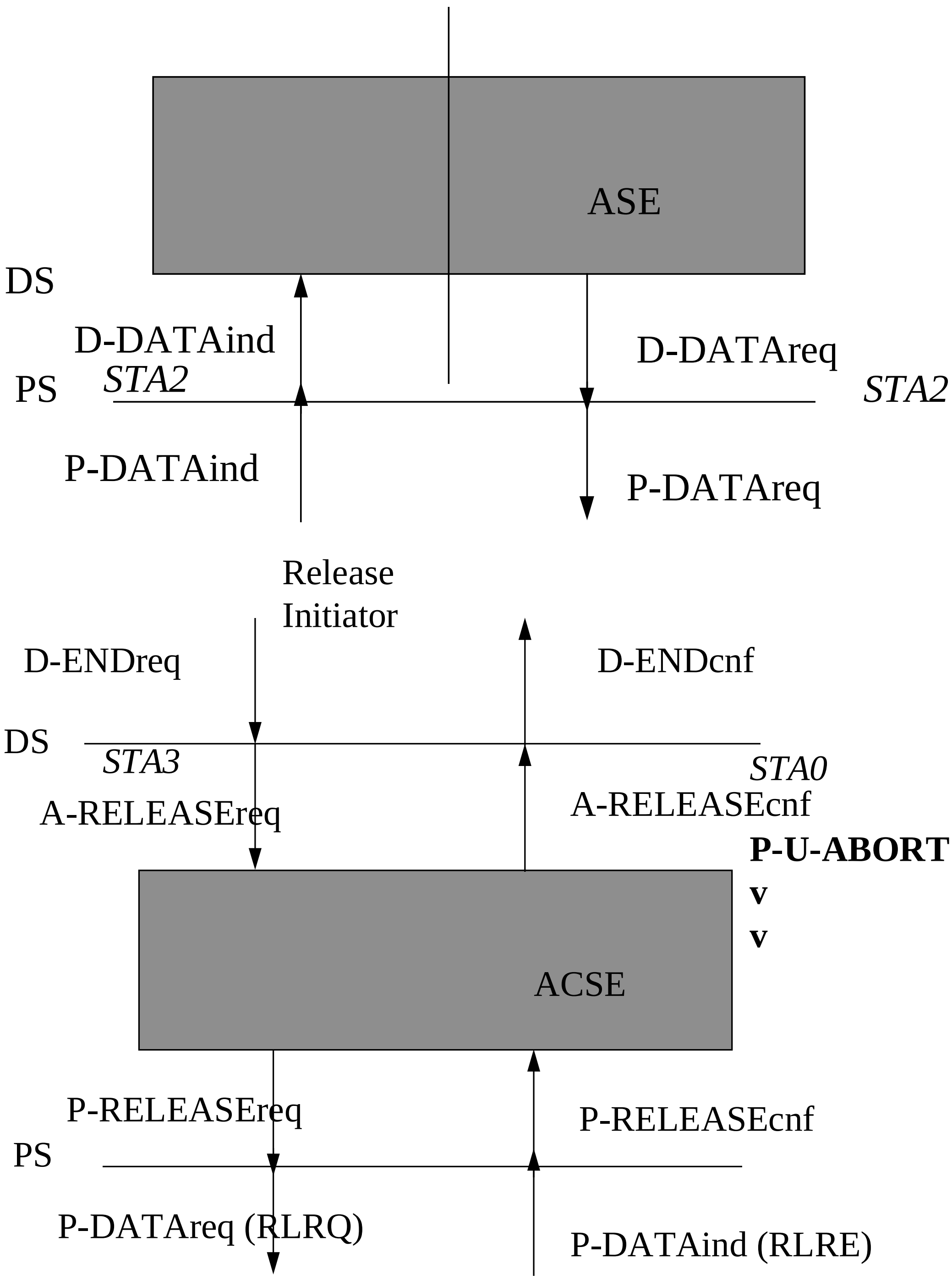
6.1.2 ACSE Primitive Flow Diagrams

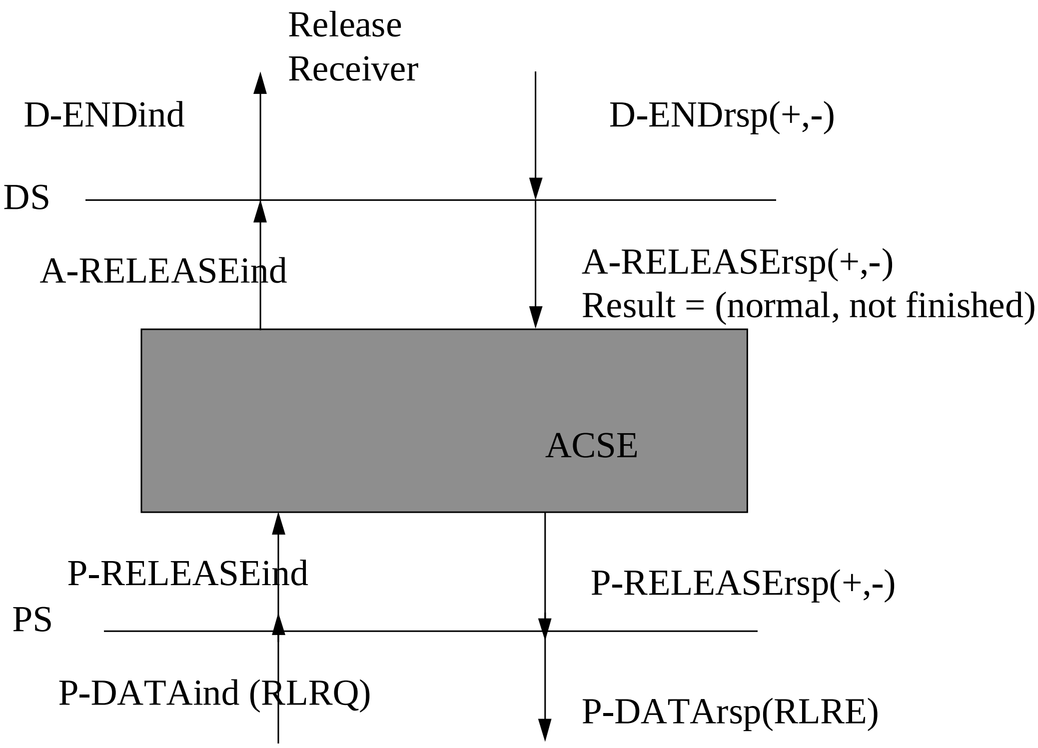
The role of the CNS/ATM-1 Control Function (CF) is to modulate the interaction of the ATN ASE and the ACSE. To that end, description of the characteristic flows involving ACSE is provided in the figures below.

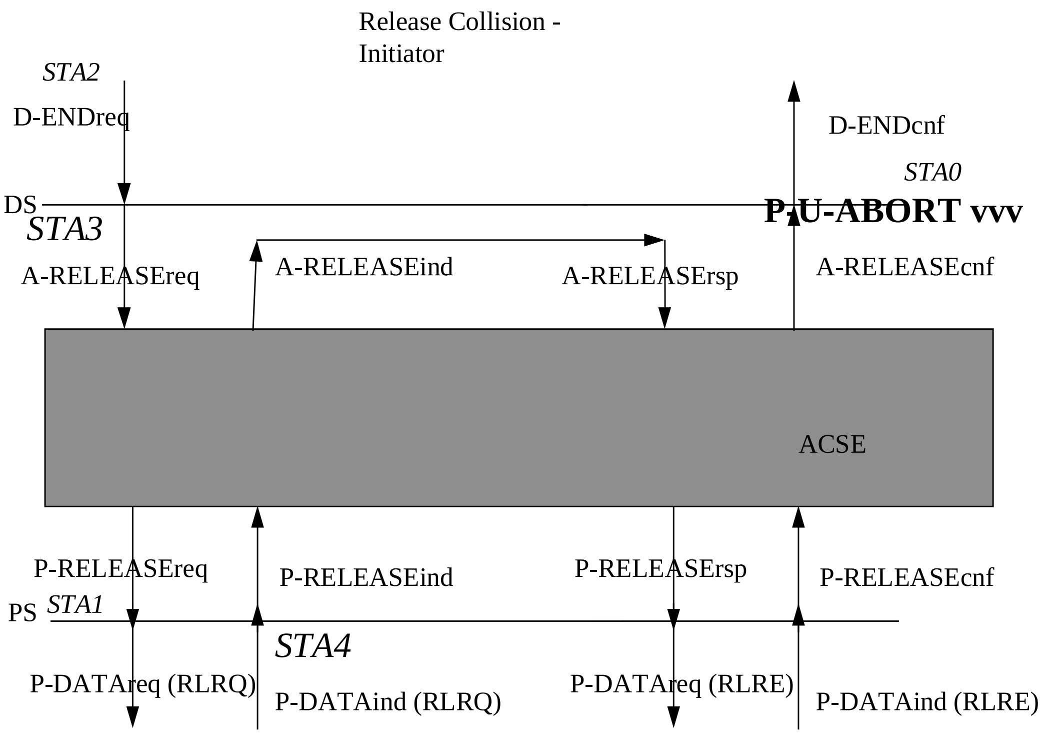

Initialization
Initiator
STA0
D-STARTreq
D-STARTcnf(+,-)
DS
STA2, STA0
A-ASSOCreq
A-ASSOCcnf (+,-)
ACSE
P-CONNreq
P-CONNcnf
PS
STA1
P-CONNreq
P-CONNcnf

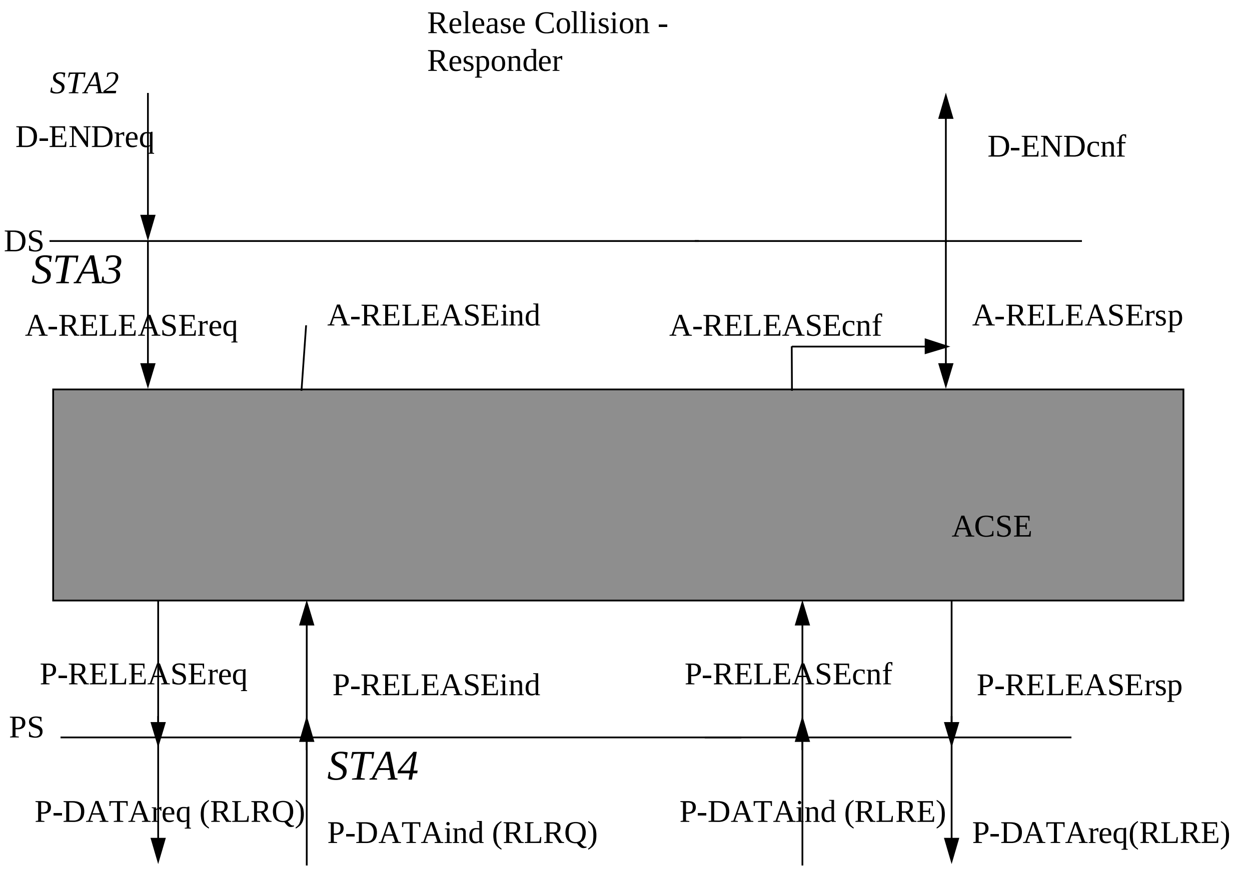

Initialization
Responder
D-STARTind
D-STARTrsp(+,-)
DS
STA1
A-ASSOCind
A-ASSOCrsp(+,-)
ACSE
P-CONNind
P-CONNrsp(+,-)
PS
STA2, STA0
P-CONNind
STA0
P-CONNrsp

Data Transmission
Data Receipt




Release
Receiver
D-ENDind
D-ENDrsp(+,-)
DS
A-RELEASEind
A-RELEASErsp(+,-)
Result = (normal, not finished)
ACSE
P-RELEASEind
P-RELEASErsp(+,-)
PS
P-DATAind (RLRQ)
P-DATArsp(RLRE)


Release Collision -
Initiator
STA2
D-ENDreq
D-ENDcnf
STA0
DS
P-U-ABORT vvv
STA3
A-RELEASEreq
A-RELEASEind
A-RELEASErsp
A-RELEASEcnf
ACSE
P-RELEASEreq
P-RELEASEind
P-RELEASErsp
P-RELEASEcnf
PS
STA1
STA4
P-DATAreq (RLRQ)
P-DATAind (RLRQ)
P-DATAreq (RLRE)
P-DATAind (RLRE)


Release Collision -
Responder
STA2
D-ENDreq
D-ENDcnf
DS
STA3
A-RELEASEreq
A-RELEASEind
A-RELEASEcnf
A-RELEASErsp
ACSE
P-RELEASEreq
P-RELEASEind
P-RELEASEcnf
P-RELEASErsp
PS
STA4
P-DATAreq (RLRQ)
P-DATAind (RLRQ)
P-DATAind (RLRE)
P-DATAreq(RLRE)

ACSE Definitions

The elements new to the ACSE edition 2 with the ASN.1extensibility notation (ISO/IEC 8650-1:1995,ed.2/PDAM2) of the connection-oriented ACSE are indicated by redlining.

```
ACSE-apdu :: = CHOICE
{
aarq            AARQ-apdu,              -- ACSE associate request pdu
aare            AARE-apdu,              -- ACSE associate response pdu
rlrq            RLRQ-apdu,              -- ACSE release request pdu
rlre            RLRE-apdu,              -- ACSE release response pdu
abrt            ABRT-apdu,              -- ACSE abort pdu
. ..
}
```

```
AARQ-apdu ::= [APPLICATION 0] IMPLICIT SEQUENCE
{
protocol-version                    [0]   IMPLICIT BIT STRING
                                              {version1(0)} DEFAULT {version1 },
application-context-name            [1]   Application-context-name,
called-AP-title                     [2]   AP-title OPTIONAL,
called-AE-qualifier                 [3]   AE-qualifier OPTIONAL,
called-AP-invocation-identifier[4]    AP-invocation-identifier OPTIONAL,
called-AE-invocation-identifier     [5]   AE-invocation-identifier OPTIONAL,
calling-AP-title                    [6]   AP-title OPTIONAL,
calling-AE-qualifier                [7]   AE-qualifier OPTIONAL,
calling-AP-invocation-identifier    [8]   AP-invocation-identifier OPTIONAL,
calling-AE-invocation-identifier    [9]   AE-invocation-identifier OPTIONAL,
-- The following field shall not be present if only the Kernel is used.
sender-acse-requirements            [10]  IMPLICIT ACSE-requirements OPTIONAL,
-- The following field shall only be present if the Authentication FU is selected.
mechanism-name                      [11]  IMPLICIT Mechanism-name OPTIONAL,
-- The following field shall only be present if the Authentication FU is selected.
calling-authentication-value        [12]  EXPLICIT Authentication-value OPTIONAL,
application-context-name-list   [13]  IMPLICIT Application-context-name-list OPTIONAL,
implementation-information          [29]  IMPLICIT Implementation-data OPTIONAL,
extensions                                SEQUENCE {...} OPTIONAL,
user-information                    [30]  IMPLICIT Association-information OPTIONAL
}
```

```
AARE-apdu ::= [APPLICATION 1] IMPLICIT SEQUENCE
{
protocol-version                      [0]   IMPLICIT BIT STRING
                                                {version1(0),} DEFAULT { version1 },
application-context-name              [1]   Application-context-name,
result                                [2]   Associate-result,
result-source-diagnostic         [3]   Associate-source-diagnostic,
responding-AP-title                   [4]   AP-title OPTIONAL,
responding-AE-qualifier               [5]   AE-qualifier OPTIONAL,
responding-AP-invocation-identifier   [6]   AP-invocation-identifier OPTIONAL,
responding-AE-invocation-identifier   [7]   AE-invocation-identifier OPTIONAL,
-- The following field shall not be present if only the Kernel is used.
responder-acse-requirements                 [8]   IMPLICIT ACSE-requirements OPTIONAL,
-- The following field shall only be present if the Authentication functional unit is selected.
mechanism-name                              [9]   IMPLICIT Mechanism-name OPTIONAL,
-- The following field shall only be present if the Authentication functional unit is selected.
responding-authentication-value       [10]  EXPLICIT Authentication-value OPTIONAL,
application-context-name-list         [11]  IMPLICIT Application-context-name-list OPTIONAL,
-- The above field shall only be present if the Application Context Negotiation functional unit is selected
implementation-information            [29]  IMPLICIT Implementation-data OPTIONAL,
extensions                                  SEQUENCE {...} OPTIONAL,
user-information                      [30]  IMPLICIT Association-information OPTIONAL,
}
```

```
RLRQ-apdu ::= [APPLICATION 2] IMPLICIT SEQUENCE
{
reason                                [0]   IMPLICIT Release-request-reason OPTIONAL,
extensions                                  SEQUENCE {...} OPTIONAL,
user-information                      [30]  IMPLICIT Association-information OPTIONAL,
}
```

```
RLRE-apdu ::= [APPLICATION 3] IMPLICIT SEQUENCE
{
reason                                [0]   IMPLICIT Release-request-reason OPTIONAL,
extensions                                  SEQUENCE {...} OPTIONAL,
user-information                      [30]  IMPLICIT Association-information OPTIONAL,
}
```

```
-- For the purposes of this and associated papers, assume that an AE-Title is defined as:

AE-Title ::= OBJECT IDENTIFIER   -- 7 arcs of low values {(1) (3) (27) a b c d}
```

AARQ

ASN.1 Record

```
ACSE-apdu
{
AARQ-apdu
        {
         application-context-name "{1 3 27 3 1}",
         calling-AP-title "{1 3 27 1 500 0}"
         calling-AE-qualifier "1 (CM)"
        }
}
```

Hexadecimal view (17 octets)

00 60 00 42 B1 B0 30 10 18 AC 6C 06 0D D0 00 01 01

Binary view

So as to make it easier to read the binary wiew of the data, blank lines are used to group fields that logically belong together (typically length/value pairs) ; a newline is used to delineate fields ; space is used to delineate characters within a character string ; a period (.) is used to mark octet boundaries ; and an ‘x’ represents a zero-bit used to pad the final octet to an octet boundary.

0	No extension values present in ACSE-apdu
000	Indicates AARQ-apdu is used
0000.0110 0000.0000	Bitmap indicates Calling AE Qualifier and Calling AP Title are present
0000.0100	Length of Application context name = 4
0010.1011 0001.1011 0000.0011 0000.0001	Application Context Name = {1 3 27 3 1 }
0	No extension value present in Calling AP-Title
0	Indicates AP-title-form2 is used
00.000110	Length of Calling AP-Title = 6
00.101011 00.011011 00.000001 10.000011 01.110100 00.000000	Calling AP-Title = {1 3 27 1 500 0}
0	No extension values present in Calling AE-qualifier
0.	Indicates AE-qualifier-form2 is used
00000001.	Length of Calling AE-qualifier = 1
00000001.	Calling AE-qualifier = 1 (CMA)

AARE

ASN.1 Record

```
ACSE-apdu
{
AARE-apdu
        {
         application-context-name "{1 3 27 3 1}",
         result "accepted (0)"
         result-source-diagnostic
                {
                 acse-service-user "null (0)"
                }
        }
}
```

Hexadecimal view (10 octets)

10 00 04 2B 1B 03 01 01 00 00

Binary view

So as to make it easier to read the binary view of the data, blank lines are used to group fields that logically belong together (typically length/value pairs) ; a newline is used to delineate fields ; space is used to delineate characters within a character string ; a period (.) is used to mark octet boundaries ; and an ‘x’ represents a zero-bit used to pad the final octet to an octet boundary.

0	No extension values present in ACSE-apdu
001	Indicates AARE-apdu is used
0000.0000 0000.	Bitmap indicates no optional field is present
00000100.	Length of Application context name = 4
00101011. 00011011. 00000011. 00000001.	Application Context Name = {1 3 27 3 1 }
00000001.	Length of Result = 1
00000000.	Result = 0 (accepted)
0	Indicates acse-service-user is used
0	No extension values present in Source Diagnostic
0000xx	Source Diagnostic = 0 (null)

RLRQ

ASN.1 Record

ACSE-apdu
{
RLRQ-apdu {}
}

Hexadecimal view

20

Binary view

So as to make it easier to read the binary view of the data, blank lines are used to group fields that logically belong together (typically length/value pairs) ; a newline is used to delineate fields ; space is used to delineate characters within a character string ; a period (.) is used to mark octet boundaries ; and an ‘x’ represents a zero-bit used to pad the final octet to an octet boundary.

0	No extension values present in ACSE-apdu
010	Indicates RLRQ-apdu is used
000x	Bitmap indicates no optional field is present

RLRE

ASN.1 Record

ACSE-apdu
{
RLRE-apdu {}
}

Hexadecimal view

30

Binary view

So as to make it easier to read the binary view of the data, blank lines are used to group fields that logically belong together (typically length/value pairs) ; a newline is used to delineate fields ; space is used to delineate characters within a character string ; a period (.) is used to mark octet boundaries ; and an ‘x’ represents a zero-bit used to pad the final octet to an octet boundary.

0	No extension values present in ACSE-apdu
011	Indicates RLRE-apdu is used
000x	Bitmap indicates no optional field is present

```
-----
--  ASN.1/PER Encoding of the CF PDV
-----

Number of PDVs       =     0x01                  --  PDV-list
Preamble             =         B`0'                  -- transfer-syntax-name absent
presentation-ctx-id  =         0x00              --  presentation-context-identification
p-d-v                =         B`10'             -- choice = arbitrary
```

The encoding of the CF PDV generates an overhead of 2 octets + 3 bits. The actual AARQ overhead is thus 9 octets + 1 bit.

6.2 ASN.1 EXTERNAL Type Definition

ASN.1 UNIVERSAL type EXTERNAL is taken to be as defined in ISO DIS 8824-1(1992):

```
EXTERNAL ::= [UNIVERSAL 8] IMPLICIT SEQUENCE
{
        direct-reference        OBJECT IDENTIFIER OPTIONAL,
        indirect-reference      INTEGER OPTIONAL,
        data-value-descriptor           ObjectDescriptor OPTIONAL,
        encoding CHOICE
        {
                        single-ASN1-type    [0] ANY,
                        octet-aligned       [1] IMPLICIT OCTET STRING,
                        arbitrary           [2] IMPLICIT BIT STRING
        }
}

ObjectDescriptor ::= [UNIVERSAL 7] GraphicString
```

6.3 Definition of Object Identifiers

Entity	Object Identifier	BER Encoding
ATN-App (application context)	{iso icd(3) icao(27) atn-ac(3) 0 (0)}	0x 2B 1B 03 00
ACSE	{joint-iso-ccitt association-control(2) abstract-syntax(1) apdus(0) version1(1)}	0x 52 01 00 01
ASN.1/BER	{joint-iso-ccitt asn1(1) basic-encoding(1)}	0x 51 01
ASN.1/PER	{joint-iso-ccitt asn1(1) packed-encoding(3) basic(0) unaligned(1)}	0x 51 03 00 01

Summary

The overhead due to upper layers depends upon the inclusion or exclusion of user-information and its ultimate encoding. The estimates are based on a number of assumptions which are detailed in section 3.2 in the body of the Standing Document.

The ASN.1/BER used in producing these estimates does not give a unique octet sequence during encoding. Different designs of encoder will produce different, but correct, encodings (cf. Distinguished Encoding Rules, which guarantee that an encoding can only be performed one way - thus encodings can be 'string compared'). If minimum size is the ultimate goal, then PER should be examined. However, PER is not currently allowed in the base standards for ACSE and Presentation.

As an example, if an application generates an A-ASSOCIATE with no user-data, then transmits 1024 octets of data, and then terminates, the overheads incurred are as summarised in the following sections.

Options:

- *The addition of called and calling address information to ACSE adds 23 octets to the ASN.1/BER encoding or 7¼ to the ASN.1/PER encodings.*
- *If Presentation is expected to carry between 126 and 65,535 octets of data this adds 4 octets to the P-DATA.*

6.4 Overhead Using a ULA stack

	CF PDV + ACSE ed. 2 + short and null encoding option of Presentation and Session (octets + bits)
Establish Connection - Request CF PDV ACSE AARQ Presentation Session	 2 + 3 6 + 6 1 1
S	11 + 1
Establish Connection - Response CF PDV ACSE AARE Presentation Session	2 + 3 8 + 2 1 1
S	10 + 7
Data Transfer Phase CF PDV Presentation P-DATA Session S-TYPED-DATA	2 + 3 0 0
S	2 + 3

Terminate Connection - Request CF PDV ACSE RLRQ Presentation P-RELEASE Session FN	2 + 3 0 + 6 0 0
S	3 + 1
Terminate Connection - Response CF PDV ACSE RLRE Presentation P-RELEASE Session DN	2 + 3 0 + 6 0 0
S	3 + 1
TOTALS	30 + 5