FFI RAPPORT

A PROTOCOL SOLUTION FOR REPLICATION OF INFORMATION IN A NATO TACTICAL DIRECTORY

EGGEN, Anders

FFI/RAPPORT-2003/01517

FFIE/840/110

Approved
Kjeller 19. December 2003

Torleiv Maseng
Director of Research

A PROTOCOL SOLUTION FOR REPLICATION OF INFORMATION IN A NATO TACTICAL DIRECTORY

EGGEN, Anders

FFI/RAPPORT-2003/01517

FORSVARETS FORSKNINGSINSTITUTT
Norwegian Defence Research Establishment
P O Box 25, NO-2027 Kjeller, Norway

FORSVARETS FORSKNINGSINSTITUTT (FFI)
Norwegian Defence Research Establishment

P O BOX 25
N0-2027 KJELLER, NORWAY

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE
(when data entered)

REPORT DOCUMENTATION PAGE

1) PUBL/REPORT NUMBER	2) SECURITY CLASSIFICATION	3) NUMBER OF PAGES
FFI/RAPPORT-2003/01517	UNCLASSIFIED	44
1a) PROJECT REFERENCE	**2a) DECLASSIFICATION/DOWNGRADING SCHEDULE**	
FFIE/840/110	-	

4) TITLE

A PROTOCOL SOLUTION FOR REPLICATION OF INFORMATION IN A NATO TACTICAL DIRECTORY

5) NAMES OF AUTHOR(S) IN FULL (surname first)

EGGEN, Anders

6) DISTRIBUTION STATEMENT

Approved for public release. Distribution unlimited. (Offentlig tilgjengelig)

7) INDEXING TERMS

IN ENGLISH:	IN NORWEGIAN:
a) Directory	a) Directory
b) Taktiske nett	b) Tactical networks
c) Lav datarate	c) Low bandwidth
d) DISP	d) DISP
e) Replisering	e) Replication

THESAURUS REFERENCE:

8) ABSTRACT

This document describes a tactical X.500 (ref. (24)) or ACP 133 (ref. (25)) Directory protocol solution, which may be used for replication of data between NATO Directory systems over links/networks with reduced throughput capacity (lower than 20 Kbps). The benefits of replication come from improved performance during retrieval, since copies of data can be located nearer to the end user. However Directories often need to be updated in order to reflect the changes in the stored information. This also applies to Directories used in tactical environments where the communications systems often have low throughput. Since we are focusing on tactical systems with reduced data-rate and which are not built for interactive traffic, we are only discussing the protocol DISP for replicating information between DSAs. Our assumption is that each tactical unit has its own DSA. We also assume that most of the Directory information of the tactical user is downloaded upfront, using higher bandwidth communication systems, CDs or other means, and that all the traffic sent over the low data-rate tactical connections are updates or "incremental refresh" only. The protocol solution proposed in this report is based on the authors protocol proposal for tactical MMHS, which is described in the NATO STANAG 4406 Ed. 1 Annex E (ref. (1)).

9) DATE	AUTHORIZED BY This page only	POSITION
19. December 2003	Torleiv Maseng	Director of Research

ISBN-82-464-0812-7

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE
(when data entered)

CONTENTS

Page

A PROTOCOL SOLUTION FOR REPLICATION OF INFORMATION IN A NATO TACTICAL DIRECTORY

1 INTRODUCTION

This document describes a tactical X.500 (ref. (24)) or ACP 133 (ref. (25)) Directory protocol solution, which may be used for replication of data between NATO Directory systems over links/networks with reduced throughput capacity (lower than 20 Kbps). The benefits of replication come from improved performance during retrieval, since copies of data can be located nearer to the end user. However Directories often need to be updated in order to reflect the changes in the stored information. This also applies to Directories used in tactical environments where the communications systems often have reduced data-rate.

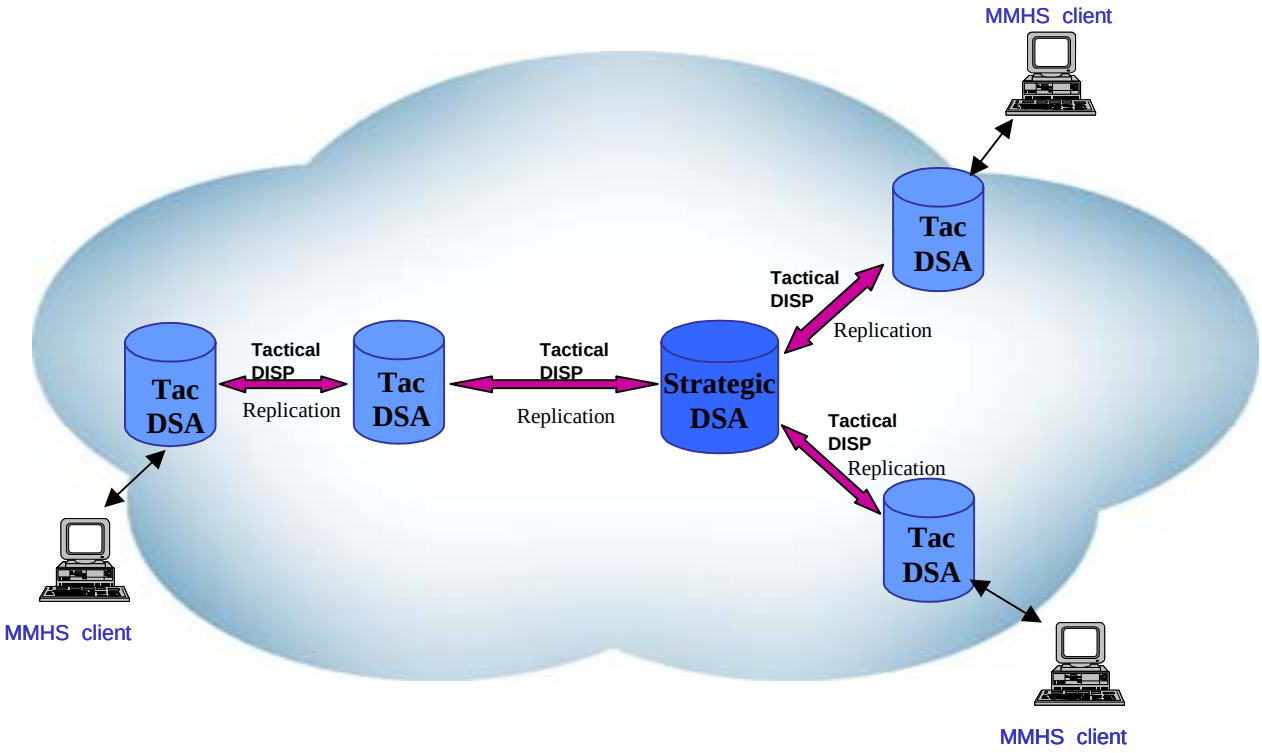
Since we are focusing on tactical systems with reduced data-rate and which are not built for interactive traffic, we are only discussing the protocol DISP for replicating information between DSAs. We do not consider the X500 protocols DAP, DSP and DOP for use over low data-rate connections. Our assumption is that each tactical unit has its own DSA. We also assume that most of the Directory information of the tactical user is downloaded upfront, using higher bandwidth communication systems, CDs or other means, and that all the traffic sent over the low data-rate tactical connections are updates or “incremental refresh” only.

The protocol solution proposed in this report is based on the author’s protocol proposal for tactical MMHS, which is described in the NATO STANAG 4406 Ed. 1 Annex E (ref. (1)). The readers who are familiar with the STANAG 4406 Annex E will therefore recognize much of the text in this report, as the ideas for tactical MMHS are reused in this proposal for a tactical version of X.500 DISP. MMHS and Directory systems will often be co-located and it therefore seems reasonable to use the same protocol stack.

This document identifies a set of interoperability-protocols. This includes application level Directory protocols and related lower layer protocols that need to be taken into account for a complete solution.

This document further describes the use of data compression and encoding techniques. The protocols defined include the ACP-133/X.500 DISP protocol, the ACP-142 (P_Mul) protocol (ref. (13)) and the WAP transport protocol WDP (Wireless Datagram Protocol, ref. (16)). For compression the Compressed Data Content Type protocol is proposed with the ZLIB [RFC 1950] [RFC 1951] algorithm.

Figur 1.1 illustrates the use of DISP for replication of information between Directory servers (DSAs) in a tactical environment.



Figur 1.1 The figure illustrates how the tactical protocol solution for DISP may be used to replicate information between strategic and tactical DSAs, and directly between tactical DSAs.

2 FUNCTIONAL COMPONENTS AND INTERFACES

This document defines the protocols to be used for a tactical version of DISP. Figure 2.1 shows a tactical Directory replication functional architecture, including the defined tactical replication interfaces.

2.1 Functional Components

Functional Components define a set of functions required in a tactical Directory system. One of the Functional Components (the TIA) is defined by the inclusion of other Functional Components. In figure 2.1, only the light DSA is shown, but a TIA may also include other functional components like i.a. a light MTA as described in ref. (1).

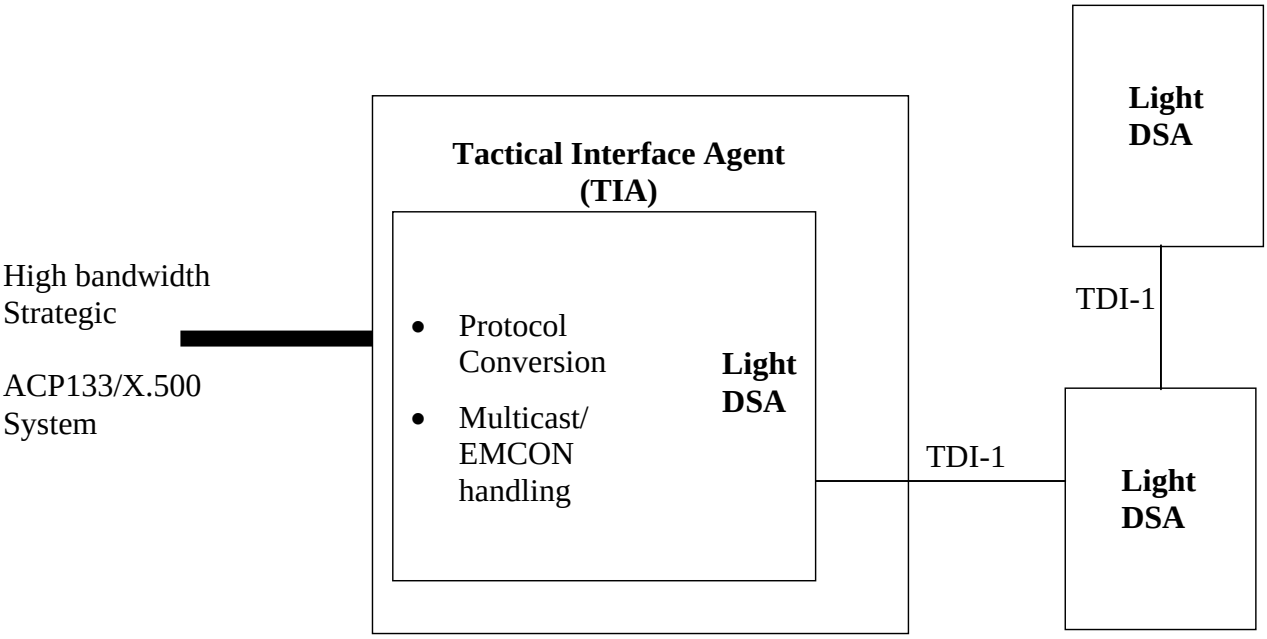


Figure 2.1 The tactical Directory replication functional architecture.

2.1.1 Tactical Interface Agent (TIA)

A TIA defines the interface point between a strategic and a tactical X.500 system. The TIA performs gateway functions in order to interconnect the strategic and the tactical Directory systems. The TIA may be integrated with one of the NATO strategic-tactical gateways.

2.1.2 Light Directory System Agent (LDSA)

A Light Directory System Agent (LDSA) is an X.500 DSA that communicates with the other LDSAs using a restricted bandwidth protocol stack defined by the TDI-1.

The LDSA may in addition contain:

- a Multicast Functional Component
  The Multicast Functional Component handles multicast of Directory information. This functional component will also handle circumstances when the communication path exists in only one direction (Unidirectional Communications).
- a Protocol/Profile Conversion Functional Component for conversion between the strategic protocols/profiles and the tactical protocol/profiles

2.2 Tactical Directory Interface (TDI)

This section describes the tactical Directory interface defined in the tactical Directory replication functional architecture.

2.2.1 TDI-1

TDI-1 is a tactical interface between an X.500 DSA from one NATO nation and an X.500 DSA of a different NATO nation. This interface defines the use of the X.500 DISP protocol with a light protocol stack, in order to increase the throughput.

The table below shows the protocols used for this interface. The different layers and protocols referred to in the table are described in section 3.

Layer	Protocol	Clause
Directory Sub-Layer	CCITT X.500 DISP Protocol	4.1
Tactical Adaption Sub-Layer	Compressed Data Type	4.2.6
P-MUL Sub-Layer	ACP-142	4.3
Transport Layer	WAP WDP	5

3 PROTOCOL ARCHITECTURE OVERVIEW

Figure 3.1 shows the tactical DISP protocol architecture for the interface TDI-1 described in section 2.

Each of the layers shown in the figure is described in the following sections.

3.1 Document Conventions

This specification uses the same keywords as specified in [ref. (22)] for defining the significance of each particular requirement. These words are:

MUST

This word, or the terms "REQUIRED" or "SHALL", mean that the definition is an absolute requirement of the specification.

MUST NOT

This phrase, or the phrase "SHALL NOT", means that the definition is an absolute prohibition of the specification.

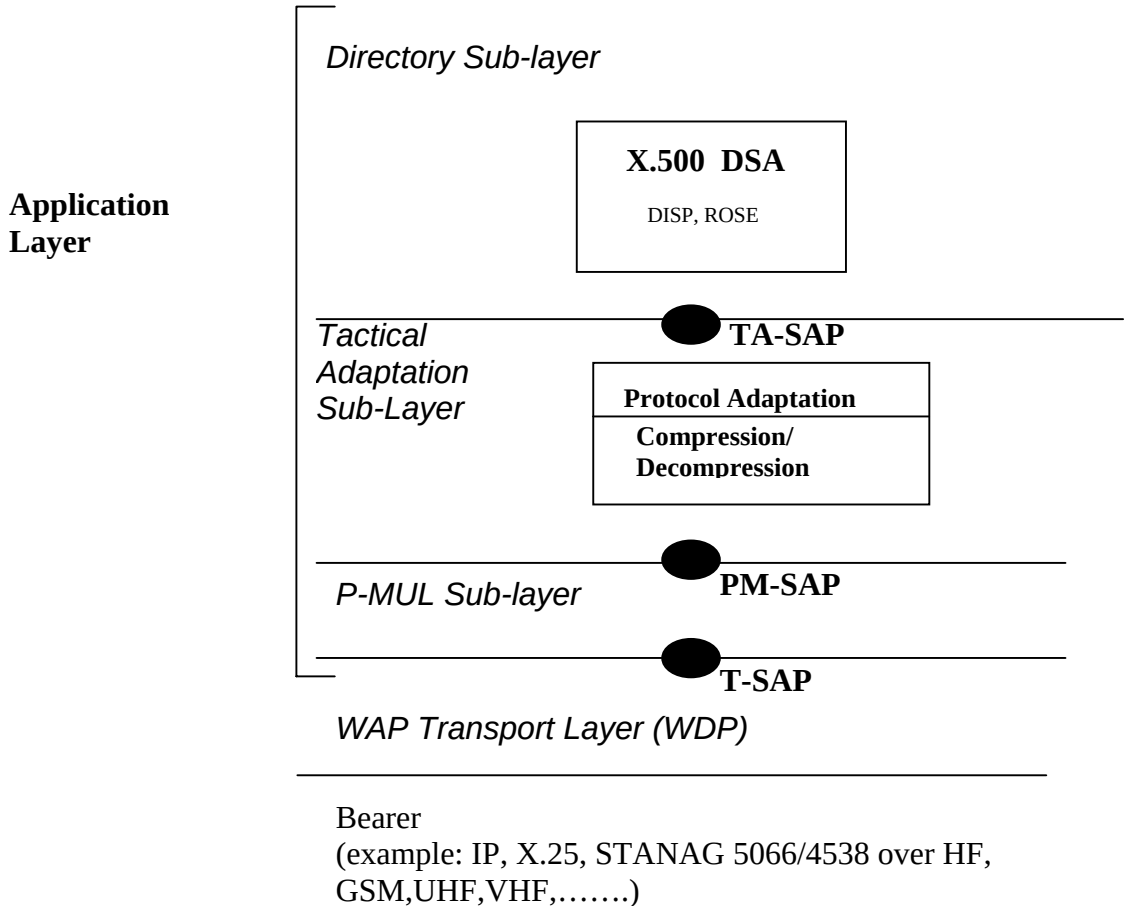


Figure 3.1 The Tactical DISP Protocol Architecture

SHOULD NOT

This phrase, or the phrase "NOT RECOMMENDED" mean that there may exist valid reasons in particular circumstances when the particular behaviour is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behaviour described with this label.

MAY

This word, or the adjective “OPTIONAL”, means that an item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because the vendor feels that it enhances the product while another vendor may omit the same item. An implementation, which does not include a particular option, MUST be prepared to interoperate with another implementation, which does include the option, though perhaps with reduced functionality. In the same way, an implementation, which does include a particular option, MUST be prepared to interoperate with another implementation, which does not include the option (except, of course, for the feature the option provides.)

3.2 Elements For Layer-to-Layer Communication

3.2.1 Definition of Service Primitives and Parameters

Communication between layers is accomplished by means of service primitives. Service primitives represent, in an abstract way, the logical exchange of information and control between the adjacent layers. Service primitives consist of commands and their respective responses associated with the services requested of another layer. The general syntax of a primitive is:

X-Service.type (Parameters)
where X designates the layer providing the service. Service primitives are not the same as an application programming interface (API) and are not meant to imply any specific method of implementing an API. Service primitives are an abstract means of illustrating the services provided by the protocol layer to the layer above. The mapping of these concepts to a real API and the semantics associated with a real API are an implementation issue and are beyond the scope of this specification.

3.2.2 Time Sequence Charts

The behaviour of service primitives is illustrated using time sequence charts, which are described in [ref. (21)].

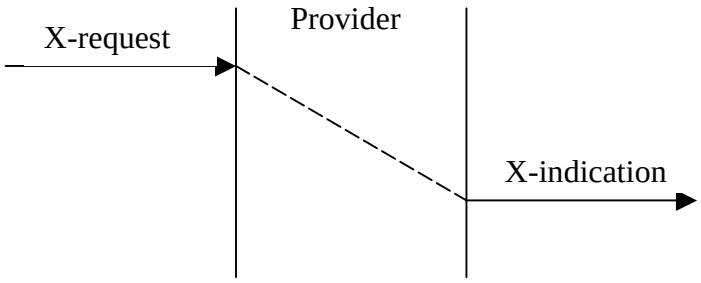


Figure 3.2 A Non-confirmed Service

Figure 3.2 illustrates a simple non-confirmed service, which is invoked using a request primitive and results in an indication primitive in the peer. The dashed line represents propagation through the provider over a period of time indicated by the vertical difference between the two arrows representing the primitives.

3.2.3 Primitive Types

The primitives types defined in this specification are:

Type	Abbreviation	Description
request	req	Used when a higher layer is requesting a service from the next lower layer
indication	ind	A layer providing a service uses this primitive type to notify the

		next higher layer of activities related to the request primitive type of the peer (such as the invocation of the request primitive) or to the provider of the service (such as a protocol generated event)
response	res	A layer uses the response primitive type to acknowledge receipt of the indication primitive type from the next lower layer
confirm	cnf	The layer providing the requested service uses the confirm primitive type to report that the activity has been completed successfully

3.2.4 Service Parameter Tables

The service primitives are defined using tables indicating which parameters are possible and how they are used with the different primitive types. For example, a simple confirmed primitive might be defined using the following:

Primitive **Parameter**	**X-primitive** ***req***	***ind***	***res***	***cnf***
Parameter 1	M	M(=)		
Parameter 2			O	C(=)

If some primitive type is not possible, the column for it will be omitted. The entries used in the primitive type columns are defined in the following table:

M	Presence of the parameter is mandatory – it MUST be present
C	Presence of the parameter is conditional depending on values of other parameters
O	Presence of the parameter is a user option – it MAY be omitted
P	Presence of the parameter is a service provider option – an implementation MAY not provide it
-	The parameter is absent
*	Presence of the parameter is determined by the lower layer protocol
(=)	The value of the parameter is identical to the value of the corresponding parameter of the preceding service primitive

In the example table above, *Parameter 1* is always present in *X-primitive.request* and corresponding *X-primitive. indicatio*n. *Parameter 2* MAY be specified in *X-primitive.response* and in that case it MUST be present and have the equivalent value also in the corresponding *X-primitive.confir*m; otherwise, it MUST NOT be present.

4 THE APPLICATION LAYER

4.1 ACP-133/X.500 (DISP) (Directory Sub-Layer)

We anticipate that it is not required that the DSAs support the DOP protocol. In a tactical environment with low data-rate communication systems, all shadowing agreement information will most likely be configured in prior to receiving the DISP data.

The ACP-133/X.500 functional component DSA may be used as defined, with some slight modifications. The connection-oriented Reliable Transmission Service Element (RTSE), the Association Control Service Element (ACSE), the OSI Presentation layer, the OSI Session layer, and the OSI Transport layer, are all replaced by the Tactical Adaptation Sub-Layer and the connection-less P_Mul Sub-Layer (see figure 3.1). The reliability functionality of the RTSE to handle retransmission and synchronisation is replaced by the P-MUL protocol (for details see section 4.3).

According to the X.500 specification, the protocol used for replication between the DSAs is DISP. This protocol uses the services of the Remote Operation Service Element (ROSE). The ROSE service elements expect to see the services provided by the RTSE, or ACSE and Presentation Layer, which are all removed in this profile. The Tactical Adaptation Sub-Layer will provide the protocols in the Messaging Sub-Layer with a service interface similar to the one of the RTSE, and map these connection-oriented services to the connection-less services provided by the P_Mul Sub-Layer. This means that the same DSA functionality, which is implemented for the strategic Directory, may be reused. How the service/protocol mapping is done, is described in section 4.2. See the ACP-133 or the ITU X.500 standards for more information about the DSA functionality and the DISP protocol.

According to the ACP 133 Annex D, the Supported Security Level equal to “None” is a valid choice for all of the DISP protocol. For the tactical Directory defined by this specification, the Supported Security Level equal to “None” SHALL be used and no “Credentials” shall be set in the “DirectoryBindArgument“ or “DirectoryBindResult”. The reason for this is that they will not be transferred. No user-data will be transmitted in the bind and unbind operations (se section 4.2 for more details).

4.2 The Tactical Adaptation Sub-Layer

The Tactical Adaptation Sub-Layer is required in order to provide the expected connection oriented service interface to the protocols at the Directory sub-layer, and to perform the mapping of these services to the connection less P_Mul protocol. In addition this sub-layer performs operations to increase the throughput, like compression/decompression. The main concern of this layer is to make it possible to use the standard ACP-133/X.500 DSA application, while reducing the bandwidth usage to a minimum.

Originally the X.500 protocol DISP use the Application Service Element; Remote Operations Service Element (ROSE) (See the ITU X.229 for details). This service element makes use of the connection oriented Application Service Elements RTSE (Reliable Transfer Service Element), which further makes use of the ACSE (Association Control Service Element) and the connection oriented Presentation and Session Layer services.

The RTSE provides a mechanism to recover from communication and end-system failure minimising the amount of retransmissions. However, the connection oriented RTSE, ACSE, Presentation-, Session- and Transport-layer protocols introduce a lot of overhead that should be avoided in a tactical network. They can neither be used in high latency simplex scenarios. We

have therefore chosen to use a connectionless protocol stack that includes the ACP 142 (P_Mul). ACP 142 will replace the reliability functionality of RTSE, in that it fragments the data into smaller PDUs, includes sequence numbers and checksums and ensures selective retransmission of lost PDUs.

To make this protocol solution transparent for the applications, we need an adaptation layer that maps the connection-oriented service interface to the actual connection-less services provided by ACP 142.

4.2.1 Functionality to Increase the Throughput Over Low Data-rate Connections

The data throughput is increased through different means:

1) Use of compression: The whole APDU is compressed (see section 4.2.6).
2) The connection-oriented protocol stack is mapped onto a connection-less protocol stack in order to avoid the overhead of connection establishment and termination at each layer.
3) We have removed the communication overhead caused by the ISO connection-oriented Reliable Transfer Service Element (RTSE), Association Control Service Element (ACSE), Presentation-, Session- and Transport-layer. The connection-oriented services of the Directory Sub-Layer (see figure 2.1) are mapped to the connection less services of the P_Mul Sub Layer.
4) For DISP, the transmission of the data related to the operations DSAShadowBind, DSAShadowUnbind and CoordinateShadowUpdate are simulated (or “faked”) in order to reduce the number of requests and responses.

4.2.2 The Tactical Adaptation Sub-Layer Service Interface

The X.500 protocol DISP, use the Application Service Element; Remote Operations Service Element (ROSE). This service element can either make use of the connection oriented Application Service Element RTSE (Reliable Transfer Service Element), or the connection oriented ACSE (Association Control Service Element) and the Presentation layer services.

The service interface provided by the Tactical Adaptation Sub-Layer is identical to the one provided by RTSE. The service primitives and parameters are the same, as well as the responses the service-user expects to see when invoking a service primitive. One difference is that the primitives have a prefix TA instead of RT. Another is that even though the Tactical Adaptation Sub-Layer provides services similar to the RTSE services, the protocol machine and the functionality are not the same in that some of the functionality is only simulated or “faked”.

The services provided by the Tactical Adaptation Sub-Layer are:

- TA-OPEN.request/indication/response/confirmation
- TA-CLOSE.request/indication/response/confirmation
- TA-TRANSFER.request/indication/ confirmation
- TA-TURN-PLEASE.request/indication
- TA-TURN-GIVE.request/indication
- TA-P-ABORT.indication

- TA-U-ABORT.request/indication

Figure 4.1 shows an example of the communication between the Messaging Sub-Layer and the Tactical Adaptation Sublayer during the data transmission. When the LDSA with the service element ROSE invokes a TA-OPEN.request primitive, it expects an APDU to be sent to the peer-entity, that the peer entity issues a TA-OPEN.indication, and that a response/confirmation is returned. What actually happens is that when the ROSE issues a TA-OPEN.request, the Tactical Adaptation Sub-Layer immediately issues a TA-OPEN.confirmation to the ROSE and waits for a TA-TRANSFER.request containing the actual shadow update data. The data is transferred using P_Mul and delivered to the Tactical Adaptation Sub-Layer at the peer side. The peer Tactical Adaptation Sub-Layer then issues a TA-OPEN.indication to the ROSE and waits for a TA-OPEN.response before the TA-TRANSFER.indication is issued.

When the P_Mul Data_PDU has been sent to the recipients and a PM_DATA.confirmation is received from the P_Mul Sublayer, a TA-TRANSFER.confirmation is issued to the ROSE. Thus the connection establishment services are “faked” and the service-user protocols need not be changed. The P_Mul Sublayer takes care of acknowledgements and retransmissions.

It is important to be aware of that the TA-TRANSFER.confirmation primitive only acknowledges that the data was sent to the P_Mul Sublayer of the next LDSA, and is not to be regarded as an end-to-end acknowledgement between the LDSA applications.

In ACP-133 Annex D, the ROSE class 2 is chosen. This means that ROSE is mapped onto ACSE and the Presentation Layer services without going through the RTSE. However, the services and the related parameters provided by the TA-service interface, is similar to the set of services provided by the ACSE and the Presentation layer and may replace their services.

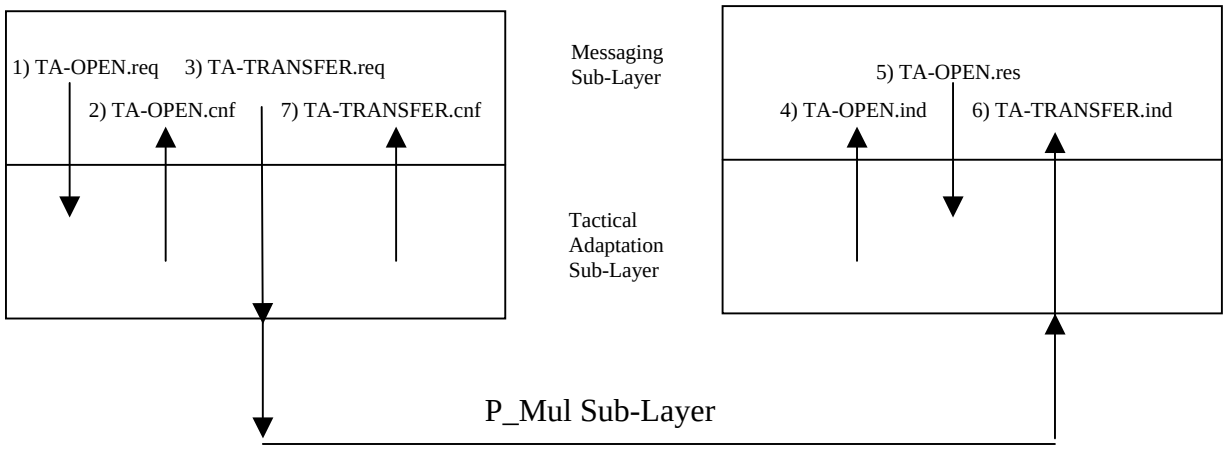


Figure 4.1 Example of the “faked” association establishment

4.2.3 The Tactical Adaptation Sublayer Service Primitives and Parameters

This service interface may be used to replace the service interface defined by the ITU RTSE (X.218) service definitions. The RTSE services mapping specification in X.219/X.229 SHOULD be followed.

For each of the service primitives, we have only included the parameters of the corresponding RTSE primitives that are Mandatory in X.218, and which are required by ROSE X.219/X.229.

If these primitives are used to replace the ITU RTSE (X.218) service interface, the parameter values SHOULD be according to the ROSE (ITU X.219/X.229) specification for the use of RTSE (normal mode). The primitive sequences are described in section 4.2.5.

4.2.3.1 TA-OPEN

This service replaces the RT-OPEN service defined in the ITU X.218 specification. The parameters are defined in the X.218 specification.

Primitive / **Parameter**	**TA-OPEN**			
	req	***ind***	***res***	***cnf***
Dialogue-mode	-	M	-	-
Initial-turn	-	M	-	-
User-data	-	M[1]	-	C(=)
Mode	-	M	-	-
Application context name	-	M	-	C(=)
Presentation context definition list	-	M	-	-
Result	-	-	M	M(=)

1) This parameter is usually used to transfer the arguments of a bind-operation. For this service primitive arguments are NOT allowed because they will not be transferred. The presence of the parameter in the primitive is only to provide the same interface as the ITU X.218 standard.

When the service-user invokes a TA-OPEN.request primitive, it expects the service to cause an APDU to be sent to the peer-entity, the peer-entity to issue a TA-OPEN.indication, and a response/confirmation to be returned. What actually happens is that when the service-user issues a TA-OPEN.request, the Tactical Adaptation Sub-Layer immediately issues a TA-OPEN.confirmation with the Result parameter set to “accepted”, and waits for a TA-TRANSFER.request from the ROSE containing the actual shadow update data.
The data is transferred to the Tactical Adaptation Sub-Layer at the peer side. The peer Tactical Adaptation Sublayer then issues a TA-OPEN.indication to the service-user with the following parameter default values:

- Mode: “normal mode”
- Application context name: “reliableShadowSupplierInitiatedAC” if the replication is initiated by the supplier, or “reliableShadowConsumerInitiatedAC” if the replication is initiated by the consumer. The term “reliable….” Indicates that RTSE is used in the application context.
- User-Data: NULL
- Presentation context definition list: “id-as-acse”, “id-as-directoryReliableShadowAS”
- Initial turn: “association-initiator”
- Dialogue-mode: “two-way-alternate”

The peer Tactical Adaptation Sublayer then waits for a TA-OPEN.response. If the Result parameter in the response primitive is “accepted”, the TA-TRANSFER.indication is issued, otherwise a PM-U-ABORT.request is issued. Thus the connection establishment services are “faked” and the service-user protocols do not need to be changed.

4.2.3.2 TA-CLOSE

This service replaces the RT-CLOSE service defined in the ITU X.218 specification. The parameters are defined in the X.218 specification. No parameters are transferred.

Primitive / **Parameter**	**TA-CLOSE**			
	req	***ind***	***res***	***cnf***
User-data	-	M[1]	-	M[1]
Reason	-	M[2]	-	M[2]

1) This parameter is used by ROSE (see ITU X.219). The User-data parameter SHOULD be set to NULL if ROSE is used. For other ASEs it SHALL not be used.
2) This parameter is used by ROSE (see ITU X.219) and SHOULD be set to “normal” if ROSE is used.

When the Tactical Adaptation Sublayer receives a TA-CLOSE.request, it immediately issues a TA-CLOSE.confirmation primitive to the service-user. On the peer side, the TA-CLOSE.response from the service-user SHOULD be ignored.

4.2.3.3 TA-TRANSFER

This service replaces the RT- TRANSFER service defined in the ITU X.218 specification. The parameters are defined in the X.218 specification, except for the parameters

- “List-Of-Destination-Entries” is a list of IP addresses of the receiving nodes, This list of addresses may be found by the DISP application by going through the configuration data of the different Shadowing agreements.
- “ROSE-Operation-ID” is needed by the TA-Adaptation Sublayer in order to “simulate” and construct the “fake” DISP operations. The Operation “CODE” defined by the ASN.1 definition of the relevant operations (see X.525), SHOULD be used to identify the operation.
- Source-ID is needed by the DISP application in order to fill in the List-Of-Destination-Entries of a DISP Result/Error.
- “Priority”, indicates the grade of Delivery.

These parameters are only used to convey this information to the TA-Adaptation Sublayer and the P_Mul Sublayer and are therefore not present in the indication primitives.

Primitive / **Parameter**	**TA-TRANSFER**		
	req	***ind***	***cnf***
List-Of-Destination-Entries	M	-	
ROSE-Operation-ID	M	M	-
Source-ID	-	M	-

Priority	M	-	
APDU	M	M(=)	-
Transfer-time [1]	M	-	-
Result	-	-	M

1) The transfer-time should be set long enough for the data to be transferred over the low data-rate connection.

The TA-TRANSFER service is used to transfer the data to the peer side (consumer LDSA). It is a confirmed service, but is not symmetric in the sense that there is no TA-TRANSFER.response primitive.

All of the DISP operations and arguments mentioned in the following text will be encapsulated in ROSE-APDUs even if this is not highlighted in the text. When it is mentioned that a DISP operation or argument is to be created by the TA-Sublayer, the corresponding ROSE-APDU also has to be created.

All “Results” SHALL be sent as ACP-142 unicast.

The TA-Adaptation Sublayer has to be able to send the Responses back to the right Requests/Updates. The TA-Adaptaion Sublayer SHOULD therefore log the outstanding Requests/Update Operations and the addresses they are sent to. This method requires that there is only one Shadowing Agreement per entity.

NOTE that the following replication solutions will not work for EMCON recipients, because the recipients need to be able to send a “result” or “error” in response to the update.

Shadow supplier initiated update:

The shadow supplier initiated update, may involve one supplier and one consumer, or one supplier and many consumers, in which the multicast properties of ACP 142 may be utilised.

If there are many recipients, who need to be updated with the same information, the supplier may look at all the recipients as one multicast group and send the update to this multicast group. This will allow for the DISP application to operate as it was communicating with only one DSA. It is then up to the TA-Adaptation Sublayer to handle the communication with each of the recipients in the multicast group, and to aggregate the individual responses of an update into one response on behalf of the multicast group.

Supplier side:

When the TA-Sublayer receives a TA-TRANSFER.request (containing a “CoordinateShadowUpdateArgument”) from the service-user, it SHALL NOT forward it to the P_Mul Sublayer for transmission, but instead issue a TA-TRANSFER.confirmation primitive.

The TA-Sublayer then generates a “fake” CoordinateShadowUpdateResult, and gives it to the service-user in a TA-TRANSFER.indication primitive. The sublayer then waits for the user to issue a TA-TRANSFER.request containing an “UpdateShadow”. This operation is forwarded by the TA-Sublayer to the P-Mul sublayer in a PM-DATA.request. The TA-Sublayer then issues a TA-TRANSFER.confirmation to the user. Note that this relies on the P_Mul Sublayer to take care of the transmission (and re-transmission) of the information.

The TA-Sublayer must then wait for the “UpdateShadowResult” responses from all of the members of the mulitcast group, before it issues a TA-TRANSFER.indication containing an “UpdateShadowResult” on behalf of the multicast group.

If the “UpdateShadow” operation fails, the TA-Sublayer will receive a PM-DATA.indication, which will include a “shadow error” (see X.525). If one or more of the recipients respond with a “shadow error”, The TA-Adaptaion Sublayer SHALL issue a TA-TRANSFER.indication containing an “shadow error” on behalf of the multicast group. If different shadow problems are reported by the recipients in a multicast group, only one (from the list below) may be reported to DISP on behalf of the group. The following precedence list SHOULD be used (1 is the highest):

1. invalidAgreementID
2. inactiveAgreement
3. invalidInformationReceived
4. unsupportedStrategy
5. missedPrevious
6. fullUpdateRequired
7. unwillingToPerform
8. unsuitableTiming
9. updateAlreadyReceived
10. invalidSequencing
11. insufficientResources

If one of the following problems are reported

- unsupportedStrategy,
- missedPrevious,
- fullUpdateRequired,
- unsuitableTiming,
- updateAlreadyReceived

they are related to the CoordinateShadowUpdate operation and are not allowed to be returned in response to the UpdateShadow operation. They are therefore to be mapped to “invalidInformationReceived” which SHOULD have the lowest precedence.

Consumer side:

When the consumer receives a PM-DATA.indication containing an "UpdateShadowArgument", the consumer SHALL issue a TA-OPEN.indication with the parameter "Application context name" set to "reliableShadowSupplierInitiatedAC".

After the reception of a successful TA-OPEN.response, the first TA-TRANSFER.indication primitive generated by the TA-Sublayer, SHALL contain a "CoordinateShadowUpdateArgument", which is built at the consumer side (by using the information in the "UpdateShadowArgument"). The "CoordinateShadowUpdateArgument" is normally sent from the shadow supplier to the shadow consumer to synchronise the DSA for the next batch of updates. All of the mandatory information for building the "CoordinateShadowUpdateArgument" is included in the "UpdateShadowArgument" (which is transferred from the supplier side) and may therefore be read out of the "UpdateShadowArgument" (AgreementID from AgreementID, RequestedStrategy from RefreshInformation). Alternatively, default values may be used for building the "CoordinateShadowUpdateArgument". As mentioned in the Chapter 1, we anticipate that the major part of the Directory information is already downloaded, and that only incremental updates are transferred over the tactical low data-rate connection.

If the positive "CoordinateShadowUpdateResult" is received in the TA-TRANSFER.request as a response to the CoordinateShadowUpdate, the TA-Sublayer SHALL NOT send this result back to the supplier side, but issue a TA-TRANSFER.confirmation to the service-user, followed by a new TA-TRANSFER.indication containing the "UpdateShadow".

If the shadow update operation went OK, the TA-Sublayer will receive a TA-TRANSFER.request with a positive "UpdateShadowResult". The TA-Sublayer SHALL then forward this to the P_Mul Sublayer for transmission, by invoking a PM-DATA.request. The TA-Sublayer then waits for a PM-DATA.confirmation before it issues a TA-TRANSFER.confirmation to the user.

If the "CoordinateShadowUpdate" operation fails, the TA-Sublayer will receive a TA-TRANSFER.request containing a "shadow Error" (see X.525). In this case the TA-Sublayer SHALL NOT issue a new TA-TRANSFER.indication containing the "UpdateShadow", but instead forward the "shadow Error" by invoking a PM-DATA.request. The TA-Sublayer then waits for a PM-DATA.confirmation before it issues a TA-TRANSFER.confirmation to the user.

If the "UpdateShadow" operation fails, the TA-Sublayer will receive a TA-TRANSFER.request from the service-user, which will include a "shadow error" (see X.525). This information shall be forwarded by the TA-Sublayer to the P-Mul sublayer in a PM-DATA.request. The TA-Sublayer then waits for a PM-DATA.confirmation, before it issues a TA-TRANSFER.confirmation to the user.

Shadow consumer initiated update

A consumer may ask for an update by issuing a “RequestShadowUpdate” operation. We anticipate that this scenario will be between one supplier and one consumer only (no mulitcast).

Consumer side:

When the TA-Sublayer receives a TA-TRANSFER.request primitive containing a “RequestShadowUpdateArgument” from the service user, it SHALL forward this to the P-Mul sublayer in a PM-DATA.request. The TA-Sublayer SHALL then waits for a PM-DATA.confirmation before it issues a TA-TRANSFER.confirmation to the user.

The sublayer then waits for the shadow update from the supplier, received as a PM-DATA.indication containing an “UpdateShadow”.

When the PM-DATA.indication containing an “UpdateShadow” is received, the TA-Sublayer then issues a TA-TRANSFER.indication containing a “RequestShadowUpdateResult”, followed by a new TA-TRANSFER.indication containing the “UpdateShadow”.

If the shadow update operation went OK, the TA-Sublayer will receive a TA-TRANSFER.request, containing a positive “UpdateShadowResult”. The TA-Sublayer forwards this to the P_Mul Sublayer for transmition, by invoking a PM-DATA.request. The TA-Sublayer then waits for a PM-DATA.confirmation before it issues a TA-TRANSFER.confirmation to the user.

If the “RequestShadowUpdate” operation fails, the TA-Sublayer will receive a PM-DATA.indication containing a “shadow Error” (see X.525). In this case the TA-Sublayer “SHALL forward the “shadow Error” by invoking a TA-TRANSFER.indication. In this case, the consumer will not receive a PM-DATA.indication containing the “UpdateShadow” from the supplier.

If the “UpdateShadow” operation fails, the TA-Sublayer will receive a TA-TRANSFER.request from the service-user, which will include a “shadow error” (see X.525). This information shall be forwarded by the TA-Sublayer to the P-Mul sublayer in a PM-DATA.request. The TA-Sublayer then waits for a PM-DATA.confirmation before it issues a TA-TRANSFER.confirmation to the service-user.

Supplier side:

When the supplier receives a PM-DATA.indication containing an “RequestShadowUpdateArgument”, the supplier SHALL issue a TA-OPEN.indication with the parameter “Application context name” set to “reliableShadowConsumerInitiatedAC”.

After the reception of a successful TA-OPEN.response, the TA-Sublayer generates a TA-TRANSFER.indication primitive containing the “RequestShadowUpdateArgument”.

If the positive “RequestShadowUpdateResult” is received in the TA-TRANSFER.request as a response to the request, the TA-Sublayer SHALL NOT send this result back to the consumer side, but instead issue a TA-TRANSFER.confirmation to the service-user.

The TA-Sublayer then waits for the service-user to issue a TA-TRANSFER.request containing an “UpdateShadowArgument”. This argument is forwarded by the TA-Sublayer to the P-Mul sublayer in a PM-DATA.request. The TA-Sublayer then waits for a PM-DATA.confirmation before it issues a TA-TRANSFER.confirmation to the user.

The TA-Sublayer then waits for the “UpdateShadowResult” response from the consumer, before it issues a TA-TRANSFER.indication containing an “UpdateShadowResult” to the user. If the “RequestShadowUpdate” operation fails, the TA-Sublayer will receive a TA-TRANSFER.request from the service-user, which will include a “shadow error” (see X.525). This information shall be forwarded by the TA-Sublayer to the P-Mul sublayer in a PM-DATA.request. The TA-Sublayer then waits for a PM-DATA.confirmation before it issues a TA-TRANSFER.confirmation to the service-user.

If the “UpdateShadow” operation fails, the TA-Sublayer will receive a PM-DATA.incication, which will include a “shadow error” (see X.525). This information shall be forwarded by the TA-Sublayer to the service-user in a TA-TRANSFER.indication primitive.

The operations MUST NOT be digitally signed, because the security signature will be broken by this protocol.

4.2.3.4 TA-TURN-PLEASE

This service replaces the RT- TURN-PLEASE service defined in the ITU X.218 specification. The parameters are defined in the X.218 specification.

Primitive **Parameter**	**TA-TURN-PLEASE**	
	req	***ind***
Priority	-	-

This service is used by the service-user to ask for the ”turn” which it must possess before it may use the TA-TRANSFER service. When receiving the TA-TURN-PLEASE.request primitive, the Tactical Adaptation Sublayer immediately issues the TA-TURN-GIVE.indication to the service-user. The corresponding APDU is never sent to the peer-entity. This service is included in order not to change the Directory applications, which expect this service to be present.

4.2.3.5 TA-TURN-GIVE

This service replaces the RT- TURN-GIVE service defined in the ITU X.218 specification. This service has no parameters.

Primitive / Parameter	TA-TURN-GIVE *req*	*ind*
-	-	-

This service is used by the service-user to give the “turn” to the requesting service-user. When receiving the TA-TURN-PLEASE.request primitive, the Tactical Adaptation Sublayer immediately issues the TA-TURN-GIVE.indication to the service-user. The TA-TURN-GIVE.request primitive is just ignored. This service is included in order not to change the Directory applications, which expect this service to be present.

4.2.3.6 TA-P-ABORT

This service replaces the RT-P-ABORT service defined in the ITU X.218 specification. This service has no parameters.

Primitive / Parameter	TA-P-ABORT *ind*
-	-

This service is used by the Tactical Adaptation Sub-Layer to report to the service-user that a local error has occurred that has caused an abortion of the data delivery process.

4.2.3.7 TA-U-ABORT

This service replaces the RT-U-ABORT service defined in the ITU X.218 specification.

Primitive / Parameter	TA-U-ABORT *req*	*ind*
User-data	O	O

User-data: This service has one parameter indicating the reason for abort.
This parameter will not be transferred to the peer side. At the recipient side the User-data may contain a default value in order to indicate to ROSE that an error has occurred at the peer side.

This service is used by the service-user to notify the peer-entity that an error has occurred that has caused the process to abort.

4.2.4 Use of The P_Mul Sub-Layer Services

The Tactical Adaptation Sublayer uses the following service primitives provided by the P_Mul Sublayer:

- PM-DATA.request/indication/confirmation,
- PM-P-ABORT.indication,
- PM-U-ABORT.request/indication

See section 4.3.2 for a description of the services.

4.2.5 Service Primitive Sequences

The table below shows the TA-service primitive sequences. The left column shows incoming primitives to the Tactical Adaptation Sublayer from the service user or the P_Mul Sublayer. The right column shows primitives issued by the Tactical Adaptation Sublayer as a response to the incoming primitive.

Incoming primitive	**Issued primitive(s)**
TA-OPEN.request	TA-OPEN.confirmation
PM-DATA.indication	TA-OPEN.indication[1] OR TA-TRANSFER.indication[2]
TA-OPEN.response	TA-TRANSFER.indication [3], TA-CLOSE indication
TA-CLOSE request	TA-CLOSE.confirmation
TA-CLOSE response	ignored
TA-TRANSFER request	PM-DATA.request[4]
PM-DATA.confirmation[5]	TA-TRANSFER.confirmation
TA-TURN-PLEASE.request	TA-TURN-GIVE.indication
TA-TURN-GIVE.request	ignored
PM-P-ABORT.indication	TA-P-ABORT.indication
TA-U-ABORT.request	PM-U-ABORT.request
PM-U-ABORT.indication	TA-U-ABORT.indication

Table 4.1

- The TA-OPEN request primitive is mapped onto the TA-OPEN.confirmation primitive, which fools the service user to believe that an association is established. The Tactical Adaptation Sublayer then waits for a TA-TRANSFER.request to be issued.
- The TA-TRANSFER.request primitive is mapped onto the PM-DATA.request primitive.

[1] If a “fake association” is not open.
[2] If a “fake association” is already open
[3] There may be several TA-TRANSFER.indication primitives issued before a TA-CLOSE.indication is issued.
[4] This is not the case if the request contains a “CoordinateShadowUpdateArgument” or an “operation result”, then the TA-Sublayer SHALL issue a TA-TRANSFER.confirmation primitive (see section 4.2.3.3).
[5] This is not a symmetric service in the sense that there is no TA-TRANSFER.response primitive. A PM-DATA.confirmation primitive is issued when the P_Mul Sub-Layer sends the Data_PDU to the recipients.

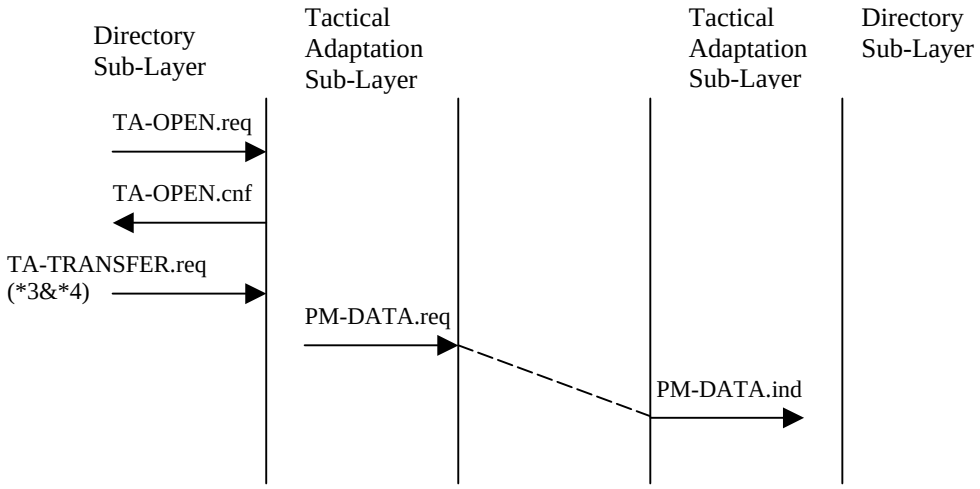


Figure 4.2 Transfer of the shadow update data (for the comment, see table 4.1 above)

- The TA-CLOSE.request primitive is mapped onto a TA-CLOSE.confirmation primitive to fool the service user to believe that an association is terminated.

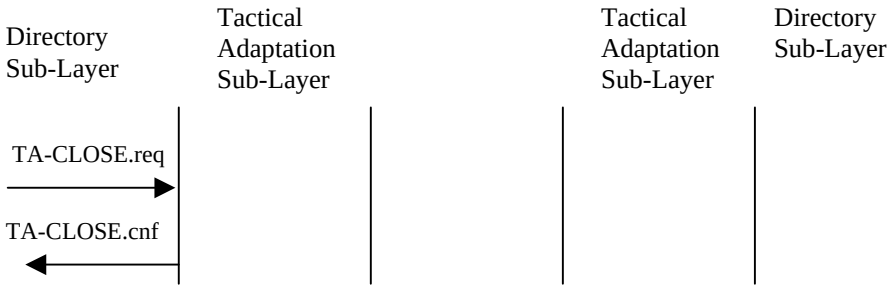


Figure 4.3 Reception of a TA-CLOSE.req

- If a “fake” association is not open, the PM-DATA.indication primitive is mapped onto the TA-OPEN.indication in order to “fake” an association establishment to the service user. The Tactical Adaptation Sublayer then waits for a TA-OPEN.response before it issues the TA-TRANSFER.indication primitive followed by a TA-CLOSE.indication primitive to fool the service user to believe that an association is requested to be terminated (note: There is no TA-TRANSFER.response primitive in order to be conformant to X.218).

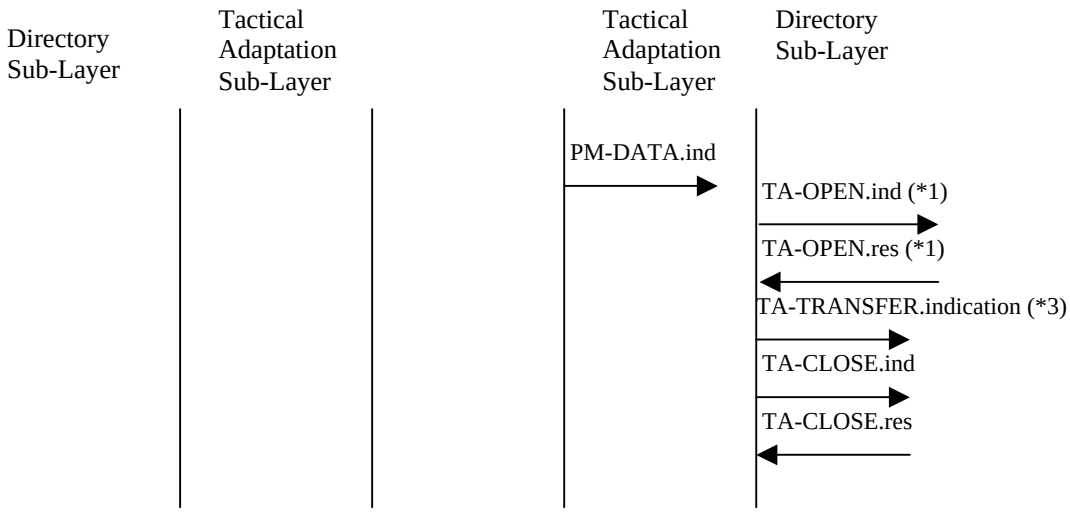


Figure 4.4 The reception of a PM-DATA.indication primitive (for the comments, see table 4.1 above)

- The TA-CLOSE.response primitive is just ignored and causes no action.
- A PM-DATA.confirmation primitive is issued from the P_Mul Sublayer when the Data_PDU is sent to the recipients. The primitive is mapped onto a TA-TRANSFER.confirmation primitive. It is important to be aware of that this only acknowledges that the data was sent to the P_Mul Sublayer of the next LDSA and is not to be regarded as an end-to-end acknowledgement between the LDSAs.

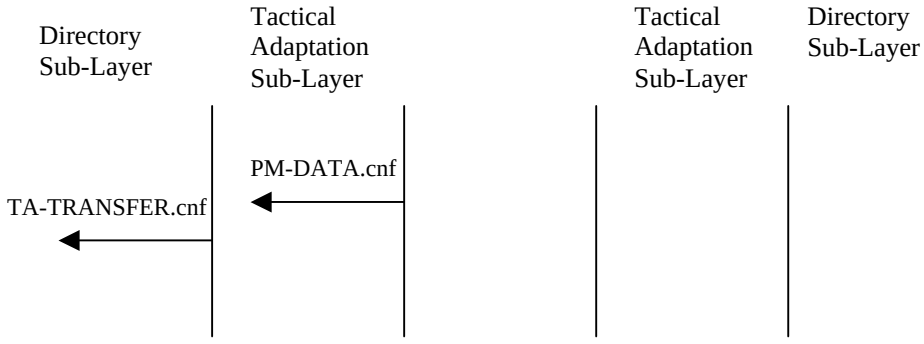


Figure 4.5 Reception of a PM-DATA.confirmation

- The TA-TURN-PLEASE.request primitive is mapped onto a TA-TURN-GIVE.indication primitive to fool the service user to believe that he has been given the send-token by the peer-entity.

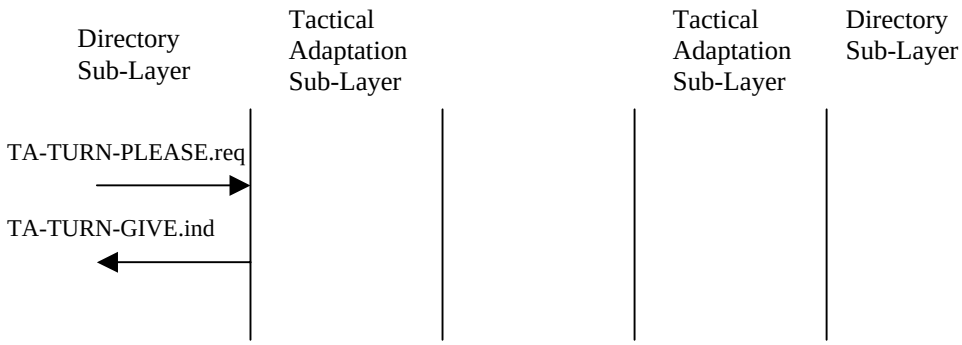


Figure 4.6 Reception of a TA-TURN-PLEASE.req

- A TA-TURN-GIVE.request is just ignored.
- A PM-P-ABORT.indication primitive indicates an error from the P_Mul Sub-layer and causes a TA-P-ABORT.indication to be issued.

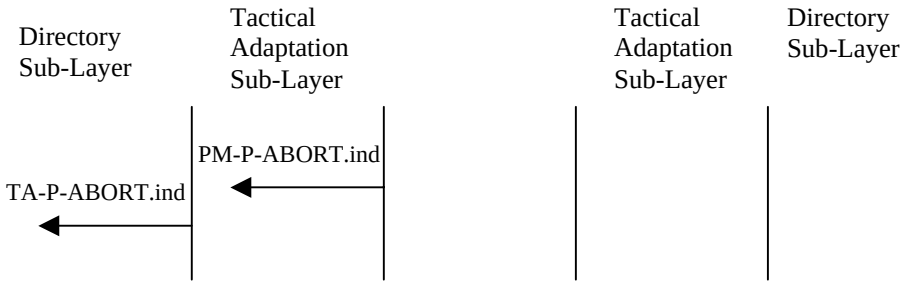


Figure 4.7 Reception of a PM-P-ABORT.indication primitive

- A TA-U-ABORT.request primitive is mapped onto a PM-U-ABORT.request primitive. The Tactical Adaptation Sub-Layer may receive the TA-U-ABORT.request primitive at any time.
- A PM-U-ABORT.indication is mapped onto a TA-U-ABORT.indication. The Tactical Adaptation Sub-Layer may issue the TA-U-ABORT.indication primitive at any time.

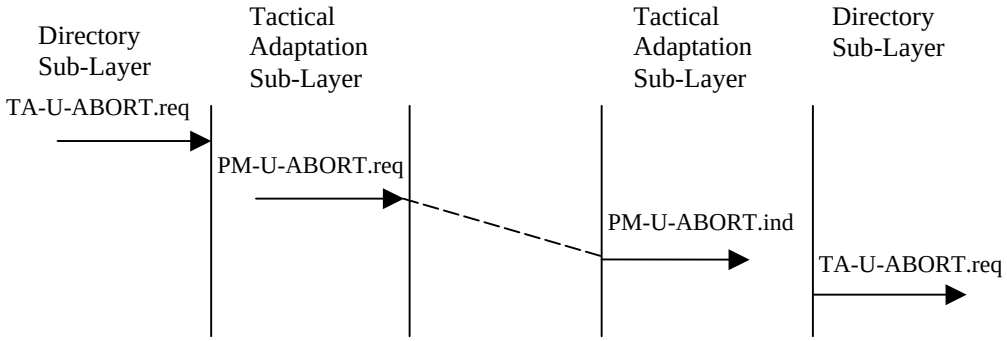


Figure 4.8 Reception of a TA-U-ABORT.request

- The Tactical Adaptation Sub-Layer may issue the TA-P-ABORT.indication primitive at any time whenever en error occurs in the sub-layer.

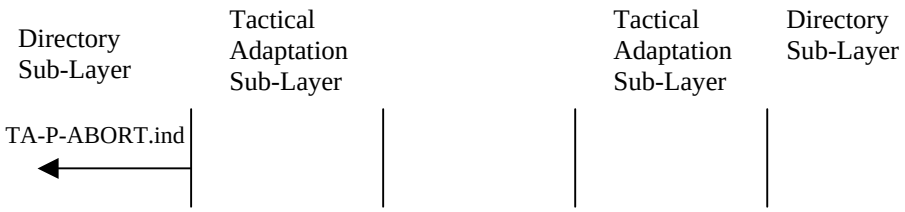


Figure 4.9 Emmision of a TA-P-ABORT.indication

4.2.6 Compression

Compression shall be performed at the Tactical Application Sub-layer in order to compress the whole APDU. In this section we have defined a compression wrapper called “Compressed Data Type”. This Compressed Data Type allows for any OCTET STRING to be compressed, and shall be used to compress the whole ROSE APDU and not just the data content.

4.2.6.1 The Compressed Data Type

The Compressed Data Type (CDT) consists of content of any type that is compressed using a specified algorithm. The following object identifier identifies the Compressed Data Type:

```
id-mmhs-CDT ID ::= {iso(1) identified-organization(3) nato(26) stanags(0)
                mmhs(4406)object-identifiers(0) id-mcont(4) 2}
```

The Compressed Data Type are defined by the following ASN.1 type:

```
DEFINITIONS ::=
BEGIN
IMPORTS ROSEapdus FROM Remote-Operations-APDUs {joint-iso-ccitt remote-
operations(4) apdus(1)}

CompressedData ::= SEQUENCE {
      compressionAlgorithm  CompressionAlgorithmIdentifier,
      compressedContentInfo CompressedContentInfo }

CompressionAlgorithmIdentifier ::= CHOICE {
      algorithmID-ShortForm  [0] AlgorithmID-ShortForm,
      algorithmID-OID        [1] OBJECT IDENTIFIER }

AlgorithmID-ShortForm ::= INTEGER  {
      zlibCompress (0) }

CompressedContentInfo ::= SEQUENCE {
      CHOICE {
            contentType-ShortForm [0] ContentType-ShortForm,
            contentType-OID       [1] OBJECT IDENTIFIER
      },
      compressedContent [0] EXPLICIT OCTET STRING }

ContentType-ShortForm ::= INTEGER  {
      unidentified (0),
      external  (1)           -- identified by the object-identifier of
                              -- the EXTERNAL content
```

```
ROSEapdus                       --See X.229 for the ASN.1 definition
       }

      }

END
```

The fields of Compressed Data Type have the following meaning:

compressionAlgorithm **(dynamically mandatory)** defines the compression algorithm to be used. The algorithm may be defined using either an INTEGER value (which is **mandatory** to support both on origination and reception) or an OBJECT IDENTIFIER (which is **optional** on origination and **mandatory** on reception).

compressedContentInfo **(dynamically mandatory)** defines the type of content that is compressed. The type of content may be indicated using either an INTEGER value (which is **mandatory** to support both on origination and reception) or an OBJECT IDENTIFIER (which is **optional** on origination and **mandatory** on reception).

compressedContent **(dynamically mandatory)** is the compressed content.

4.2.6.2 Use of the Compressed Data Type

To ensure interoperability, this section defines how the ROSE APDUs shall be encoded and conveyed within the Compressed Data Type.

The compressed ROSE APDU shall be placed in the *compressedContent* field of the *CompressedContentInfo* element. Note that this X.500/LDAP APDU SHALL be ASN.1 and BER encoded. The object identifier for the envelope type of the compressed ROSE APDU SHALL be placed in either the *contentType-ShortForm* or the *contentType-OID* field of the *CompressedContentInfo* element.

An illustration of this required wrapping convention is shown in figure 4.10.

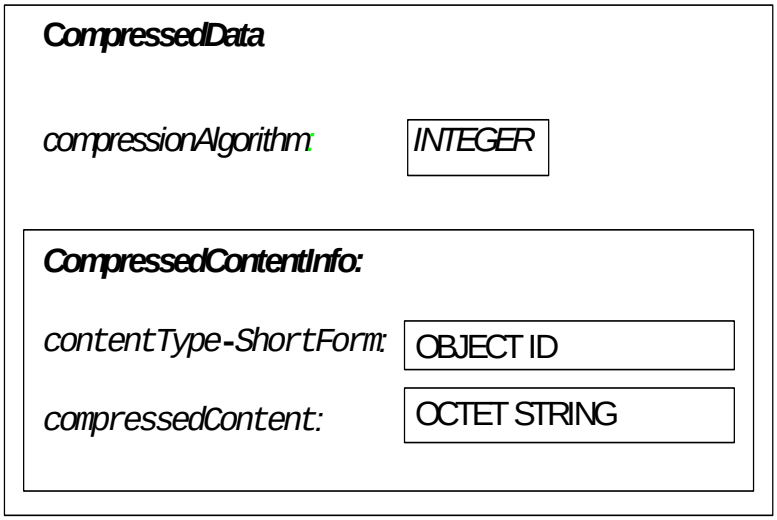


Figure 4.10 – Protocol wrapping

Analysis described in [ref. (2)] indicates that full compression of the APDU gives the best result if BER [ref. (3)] is used instead of PER [ref. (4)] to encode the ASN.1 structures of the ROSE APDU. BER encoding SHALL therefore be used before compression.

4.2.6.3 Compression Algorithm

This specification mandates the support of the compression algorithm ZLIB [ref. (18)] [ref. (19)], which is free of any intellectual property restrictions and has a freely available, portable and efficient reference implementation. The following object identifier identifies ZLIB:

id-alg-zlibCompress OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) alg(3) 8 }

The INTEGER reference SHOULD however be used and the integer value 0 identifies this algorithm.

4.3 The P_Mul Sub-Layer

P_Mul (ACP 142) [ref. (13)] is an application layer protocol that is designed to be used together with other protocols to handle EMCON (Emission Control) conditions and multicast communication techniques. It is also a provider of a reliable acknowledged, connectionless application-layer service. It operates directly above a connectionless transport layer and may operate in simplex, half-duplex or full-duplex mode. P_Mul is thus a flexible protocol that may be used by most applications.

Most of the ISO application layer protocols are based on the service support from connection oriented protocols at the top four OSI layers, for reliable data transport between application entities. This means that the standard applications cannot be used over simplex connections, or to send the replication data to users under EMCON conditions, using a connection oriented

protocol stack. When EMCON conditions apply, some nodes are only allowed to receive data and are not allowed to acknowledge them.

EMCON conditions are handled in P_Mul by allowing acknowledgements from the receiving nodes to be missing for a rather long time. The sending node has to know which of the receiving nodes that are in EMCON, and retransmissions are performed to increase the probability that the nodes in EMCON receive the data. Note that the proposed solution for a tactical DISP protocol, may not work for replication of information to recipients in EMCON (see section 4.2.3.3).

P_Mul uses an encapsulation technique where the P_Mul PDU types are wrapped around another Transfer Protocol. P_Mul replaces layers 4, 5, 6 and parts of layer 7 of the OSI protocol stack, and should be run on top of a connectionless transport-layer protocol.

By making use of the broadcast properties of a connectionless protocol stack, one data replication may be sent to N recipients instead of replicating the data N times. To handle the problem of packet flooding in broadcast networks, a multicast addressing scheme has been invented.

If there are any conflicts between this Annex and ACP-142, this annex takes precedence.
The complete specification of the P_Mul protocol with procedures, are given in ACP 142 [ref. (13)].

4.3.1 Protocol Data Units

The Tactical Task Force of the Combined Communications and Electronics Board (CCEB) has adopted P_Mul and issued the protocol as a standard defined by ACP 142 [ref. (13)].

The ACP 142 specification describes two groups of Protocol Data Units (PDUs). One group consists of those PDUs needed for the transfer of the data. These PDUs are:

- **Address_PDU**
- **Data_PDU**
- **Ack_PDU**
- **Discard_PDU**

The other group consists of PDUs for dynamic configuration of multicast groups. The concept of multicast groups is introduced to reduce the network load in situations when the sender has exact knowledge about the addresses of the recipients. The objective is that the multicast transmission of the replicated data shall involve as few recipients as possible, by forming groups of transmitting and receiving nodes within a multicast network. The Application Protocol Data Units (APDUs) used to manage multicast groups are:

- **Request_PDU** – for requesting a multicast group
- **Reject_PDU –** for rejecting a multicast group
- **Release_PDU** – for releasing a multicast group

- **Announce_PDU** – announcing a multicast group

These four PDUs will be used by a P_Mul management function and not by the data transfer service user. The specification of the P_Mul management function is out of scope for this Annex. For more details about the P_MUL protocol and the PDUs, see ACP 142 [ref. (13)].

4.3.2 The P_Mul Sub-Layer Service Interface

In order to make P_Mul more independent of the application protocols using it and to make P_Mul fit into the layered model described in this document, we have defined a service interface with a set of service primitives to be invoked by the P_Mul user. We have defined the following services for the P_Mul Sub-Layer:

- PM-DATA.request/indication/confirmation(*)
- PM-P-ABORT.indication
- PM-U-ABORT.request/indication
- PM-REQUEST.request/indication
- PM-REJECT.request/indication
- PM-RELEASE.request/indication
- PM-ANNOUNCE.request/indication

(*) This is not a symmetric service, a PM-DATA.confirmation primitive is issued when the PDU is sent to the recipients.

The PM-DATA, PM-P-ABORT and PM-U-ABORT services are used by the Tactical Adaptation Sub-Layer for data transfer and error handling.

The PM-REQUEST, PM-REJECT, PM-RELEASE and PM-ANNOUNCE services are not used by the Tactical Adaptation sub-layer, but by a P_MUL management function directly in order to set up and organise multicast groups. The reason for defining these four service primitives, is to clearly separate the P_Mul protocol machine from the P_Mul Management function which may be integrated with the user application. How to handle multicast groups is a local implementation matter and some implementations may not include these primitives for handling multicast groups.

Figure 4.12 shows the layered structure of the protocol profile where we see that some of the P_Mul services are used by the Tactical Adaptation Sublayer, and some services are used by the P_Mul Management Function to handle the multicast groups.

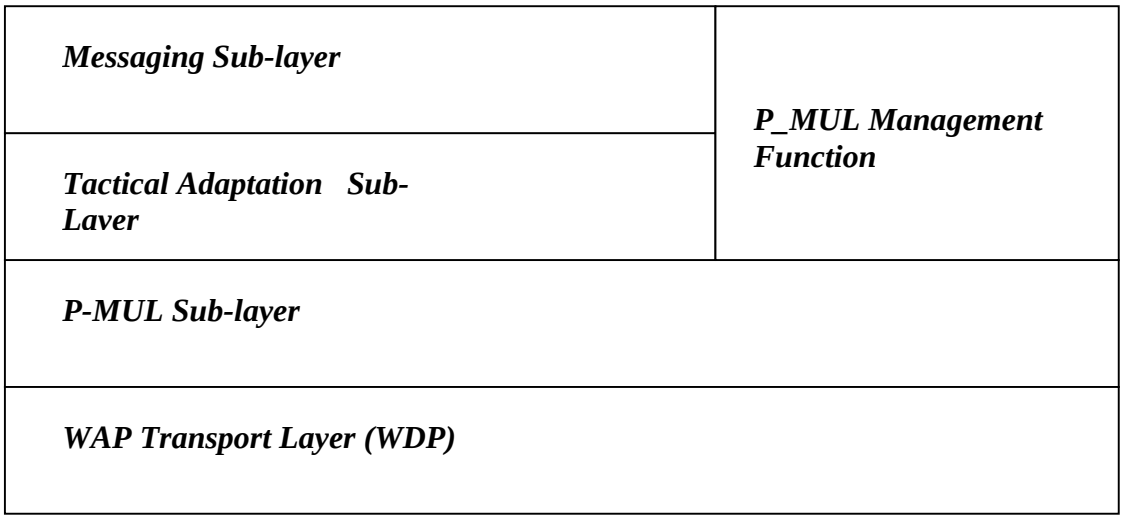


Figure 4.12 The P_Mul Sub-Layer interfaces both the Tactical Adaptation Sub-layer and a P_Mul Management Function

4.3.3 The P_Mul Sub-Layer Service Primitives and Parameters

4.3.3.1 PM-DATA

This service is used to send data from the originator to the receiver.
When the P_Mul Sub-Layer receives a PM-DATA.request primitive, it will contain the data to be sent and the sublayer SHALL create and send Address_PDUs and Data_PDUs according to the protocol description in ACP 142.

The PM-DATA.indication primitive is issued by the P_Mul Sublayer to the Tactical Adaptation Sublayer when all of the Data_PDUs belonging to a data replication are received, and an Ack_PDU is sent back indicating no missing Data_PDUs. See ACP-142 for description of the protocol.

A PM-DATA.confirmation primitive is issued by the P_Mul Sub-Layer to the Tactical Adaptation Sublayer when the Data_PDU is sent to the recipients. This is not a symmetric service in that there is no PM-DATA.response primitive. See ACP-142 for description of the protocol. It is important to be aware of that this only acknowledges that the data was sent to the P_Mul Sublayer of the next LDSA and is not to be regarded as an end-to-end acknowledgement between the LDSAs.

Primitive / **Parameter**	**PM-DATA**			
	req	***ind***	***res***	***cnf***
Priority	M	M(=)	-	M(=)
MessageID	M	M(=)	-	M(=)

Expiry_Time	O	O(=)	-	-
List_of_Destination_Entries	M	P	-	-

Priority
This parameter is to be mapped to the priority fields of the Address_PDU and the Data_PDU. See ACP-142 for description of the field and the semantic.

MessageID
See ACP-142 for details.

Expiry_Time
This parameter is to be mapped to the Expiry_Time field of the Address_PDU. See ACP-142 for description of the field and the semantic.

List_of_Destination_Entries
This parameter is to be mapped to the List_of_Destination_Entries field of the Address_PDU. See ACP-142 for description of the field and the semantic.

4.3.3.2 PM-P-ABORT

The PM-P-ABORT.indication primitive is issued by the P_Mul Sublayer to the Tactical Adaptation Sub-Layer if an error occurs in the sub-layer and the processing of the data has to be aborted. This primitive is also issued when the T-Derror.indication primitive is received from the WAP WDP layer.

Primitive **Parameter**	**PM-P-ABORT** ***req***	 ***ind***
Reason_Code	-	M

Reason_Code
The Reason_Code is a parameter indicating the reason for the abortion of the data processing caused by the P_Mul Sub-Layer. The Reason_Code may have the following values:

1 - Error receiving the data
2 - Error sending a the data
3 – Unknown error

4.3.3.3 PM-U-ABORT

The reception of a PM-U-ABORT.request indicates that an error has occurred in the above sub-layers, which has caused the data processing to be aborted. The P_Mul Sub-Layer shall create and send a Discard_Message_PDU according to the protocol description in ACP 142.
The PM-U-ABORT.indication primitive, is issued by the P_Mul Sub_Layer to the Tactical Adaptation Sub-Layer when a Discard_Message_PDU is received. See ACP-142 for description this PDU.

Primitive **Parameter**	**PM-U-ABORT** ***req***	 ***ind***

Priority	M	M(=)
MessageID	M	M(=)

Priority
This parameter is to be mapped to the priority field of the Discard_Message_PDU. See ACP-142 for description of the field and the semantic.

MessageID
See ACP-142 for details.

4.3.3.4 PM-REQUEST

This service is used to notify other transmitting nodes about a selected address to be used to send data to a multicast-group (see ACP 142 for details).
When the P_Mul Sub-Layer receives a PM-REQUEST.request primitive it shall create and send a REQUEST_PDU according to the protocol description in ACP 142.
The PM-REQUEST.indication primitive, is issued by the P_Mul Sublayer to the P_MUL Management Function when a REQUEST_PDU is received. See ACP-142 for description of the protocol.

Primitive / **Parameter**	**PM-REQUEST**	
	req	***ind***
SourceID	M	M(=)
MessageID	M	M(=)
Multicast_Group	M	M(=)

SourceID
This parameter is to be mapped to the SourceID field of the REQUEST_PDU. See ACP-142 for description of the field and the semantic.

MessageID
See ACP-142 for details.

Multicast_Group
This parameter is to be mapped to the Multicast_Group field of the REQUEST_PDU. See ACP-142 for description of the field and the semantic.

4.3.3.5 PM-REJECT

This service is used in response to the PM-REQUEST.request service to notify the sender that the selected address is occupied (see ACP 142 for details).
When the P_Mul Sublayer receives a PM-REJECT.request primitive it shall create and send a REJECT_PDU according to the protocol description in ACP 142.
The PM-REJECT.indication primitive is issued by the P_Mul Sublayer to the P_MUL Management Function when a REJECT_PDU is received. See ACP-142 for description of the protocol.

Primitive / Parameter	PM-REJECT *req*	PM-REJECT *ind*
SourceID	M	M(=)
MessageID	M	M(=)
Receipient_Address	M	M(=)

SourceID
This parameter is to be mapped to the SourceID field of the REJECT_PDU. See ACP-142 for description of the field and the semantic.

MessageID
See ACP-142 for details.

Receipient_Address
This parameter is to be mapped to the Multicast_Group field of the REJECT_PDU. See ACP-142 for description of the field and the semantic.

4.3.3.6 PM-RELEASE

This service is used to release a multicast address, which will then be available for others to use (see ACP 142 for details). When the P_Mul Sublayer receives a PM- RELEASE.request primitive it shall create and send a RELEASE_PDU according to the protocol description in ACP 142.

The PM- RELEASE.indication primitive is issued by the P_Mul Sublayer to the P_MUL Management Function when a RELEASE_PDU is received. See ACP-142 for description of the protocol.

Primitive / Parameter	PM-RELEASE *req*	PM-RELEASE *ind*
SourceID	M	M(=)
MessageID	M	M(=)
Multicast_Group	M	M(=)

SourceID
This parameter is to be mapped to the SourceID field of the RELEASE_PDU. See ACP-142 for description of the field and the semantic.¨

MessageID
See ACP-142 for details.

Multicast_Group
This parameter is to be mapped to the Multicast_Group field of the RELEASE_PDU. See ACP-142 for description of the field and the semantic.

4.3.3.7 PM-ANNOUNCE

The PM-ANNOUNCE service is used to announce the allocation of a requested multicast address to the receiving nodes.

When the P_Mul Sublayer receives a PM-ANNOUNCE.request primitive it shall create and send one or more ANNOUNCE_PDUs according to the protocol description in ACP 142. The PM-ANNOUNCE.indication primitive is issued by the P_Mul Sublayer to the P_MUL Management Function when a ANNOUNCE_PDU is received. See ACP-142 for description of the protocol.

Primitive **Parameter**	**PM-ANNOUNCE** ***req***	 ***ind***
SourceID	M	M(=)
MessageID	M	M(=)
Expiry_Time	O	O(=)
Multicast_Group_Address	M	M(=)
List_of_Destination_IDs	M	M(=)

SourceID
This parameter is to be mapped to the SourceID fields of the ANNOUNCE_PDU. See ACP-142 for description of the field and the semantic.

MessageID
See ACP-142 for details.

Expiry_Time
This parameter is to be mapped to the Expiry_Time field of the ANNOUNCE_PDU. See ACP-142 for description of the field and the semantic.¨

Multicast_Group_Address
This parameter is to be mapped to the Multicast_Group_Address field of the ANNOUNCE_PDU. See ACP-142 for description of the field and the semantic.

List_of_Destination_IDs
This parameter is to be mapped to the List_of_Destination_IDs field of the ANNOUNCE_PDU. See ACP-142 for description of the field and the semantic.

4.3.4 Use of The WAP WDP Services

The WAP WDP (Wireless Application Protocol – Wireless Datagram Protocol) [ref. (16)] defines a user service interface consisting of the following service primitives (see section 5.2 for a detailed description):

- T-DUnitdata.request/indication
- T-DError.indication

The T-DUnidata SHALL be used for transmission of all of the PDUs created by the P_Mul Sublayer. The parameters are described in section 5.2.1 and the UDP port numbers to be used are defined in ACP-142 Annex B.

The T-DError.indication primitive SHALL be mapped onto the PM-P-ABORT.indication primitive. The Error Code parameter is of local significance only and SHOULD be mapped to the Reason Code of the PM-P-ABORT.indication primitive.

5 THE WAP TRANSPORT LAYER

5.1 Introduction

The WAP (Wireless Application Protocol) connection-less transport protocol WDP (Wireless Datagram Protocol) SHALL be used in this profile as the protocol for the transport layer as specified by the WAP WDP standard [ref. (16)]. The WAP architecture and protocols have been developed by the WAP Forum. Parts of the text in this section are taken from the WAP Architecture Specification from the WAP Forum.

According to the WAP WDP specification, the User Datagram Protocol (UDP) [ref. (15)] is adopted as the WDP protocol definition for any wireless bearer network where IP is used as a routing protocol. UDP provides port based addressing and IP provides the segmentation and reassembly in a connectionless datagram service. There is no value in defining a new datagram protocol to operate over IP when the ubiquitous User Datagram Protocol (UDP) will provide the same mechanisms and functions, and is already very widely implemented. Therefore in all cases where the IP protocol is available over a bearer service the WDP Datagram service offered for that bearer will be UDP. UDP is fully specified in IETF RFC 768 [ref. (15)], while the IP networking layer is defined in the IETF RFC 791[ref. (14)].

One reason for adopting WAP WDP in this profile and not only use UDP, is that the WDP protocol, in addition to specifying the use of UDP over IP, specifies the mappings to a lot of other available bearer services. WDP also describes a service interface to the layer above, which allows for applications to operate transparently over the different bearer services. Another reason is that UDP mandates the use of IP, but in some military scenarios IP is not required and an OSI layer 2 bearer can be used directly.

5.1.1 The Wireless Datagram Protocol (WDP)

The Transport layer protocol in the WAP architecture is referred to as the Wireless Datagram Protocol (WDP). The WDP layer operates above the data capable bearer services supported by the various network types. As a general transport service, WDP offers a consistent service to the upper layer protocols of WAP and communicate transparently over one of the available bearer services. Since the WDP protocols provide a common interface to the upper layer protocols, the layers above are able to function independently of the underlying wireless network. This is accomplished by adapting the transport layer to specific features of the

underlying bearer. By keeping the transport layer interface and the basic features consistent, global interoperability can be achieved using mediating gateways.

The User Datagram Protocol (UDP) is adopted as the WDP protocol definition for any wireless bearer network where IP is used as a routing protocol (see ref. (15)for UDP protocol details). When IP is used as the network protocol, the “Robust IP header compression” SHALL be used as defined in the IETF RFC 3095 and RFC 3096.

5.1.2 Bearers

The WAP protocols are designed to operate over a variety of different bearers services, including circuit-switched data, and packet data. The bearers offer differing levels of quality of service with respect to throughput, error rate, and delays. The WAP protocols are designed to compensate for or tolerate these varying levels of service. Since the WDP layer provides the convergence between the bearer service and the rest of the WAP stack, the WDP specification [ref. (16)] lists the bearers that are supported and the techniques used to allow WAP protocols to run over each bearer. The list of supported bearers will change over time with new bearers being added as the wireless market evolves.

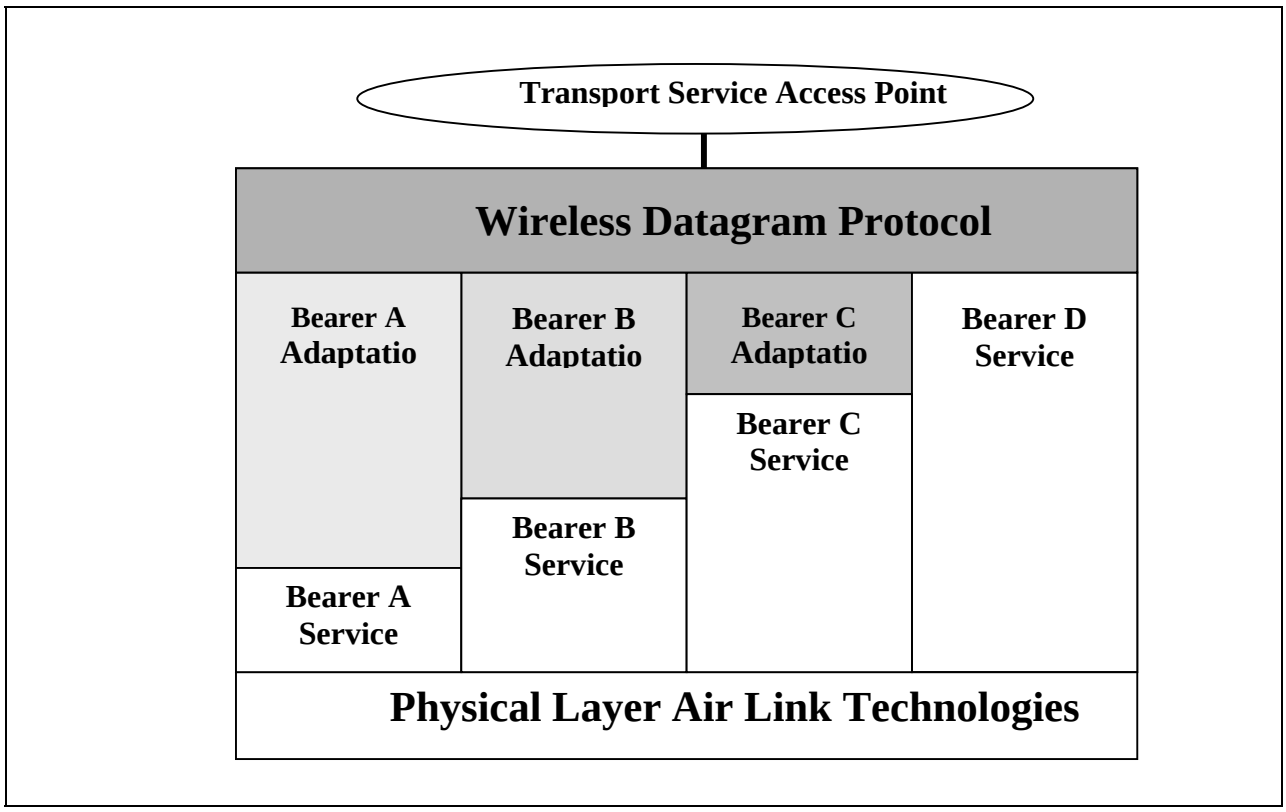


Figure 5.1 Wireless Datagram Protocol Architecture

5.1.3 Other Services and Applications

The WAP layered architecture enables other services and applications to utilise the features of the WAP stack through a set of well-defined interfaces. This allows the WAP stack to be used for applications and services not currently specified by WAP.

5.2 The WDP Service Interface

The WDP protocol uses a single service primitive T-DUnitdata. WDP may also receive a T-DError primitive if the WDP protocol layer cannot execute the requested transmission.

5.2.1 T-DUnitdata

T-DUnitdata is the primitive used to transmit data as a datagram. T-DUnitdata does not require an existing connection to be established. A T-DUnitdata.Req can be sent to the WDP layer at any time.

Primitive **Parameter**	**T-DUnitdata**	
	req	***ind***
Source Address	M	M(=)
Source Port	M	M(=)
Destination Address	M	O(=)
Destination Port	M	O(=)
User Data	M	M(=)

5.2.1.1 Source Address

The source address is the unique address of the device making a request to the WDP layer. The source address may be an MSISDN number, IP address, X.25 address or other identifier. In this profile the use of the IP address is mandatory.

5.2.1.2 Source Port

The application address associated with the source address of the requesting communication instance.

5.2.1.3 Destination Address

The destination address of the user data submitted to the WDP layer. The destination address may be an MSISDN number, IP address, X.25 address or other identifier. In this profile the support of the IP address is mandatory.

5.2.1.4 Destination Port

The application address associated with the destination address for the requested communication instance.

5.2.1.5 User Data

The user data is carried by the WDP protocol. The unit of data submitted to or received from the WDP layer is also referred to as the Service Data Unit. This is the complete unit of data, which the higher layer has submitted to the WDP layer for transmission. The WDP layer will transmit the Service Data Unit and deliver it to its destination without any manipulation of its content.

5.2.2 T-DError

The T-DError primitive is used to provide information to the higher layer when an error occurs which may impact the requested service. A T-DError Indication may be issued by the WDP layer only after the higher layer has made a request to the WDP layer, such as by issuing a T-DUnitdata.request. The T-DError primitive is used when the WDP layer is unable to complete the requested service due to a local problem. It is not used to inform the upper layer of network errors external to the device/server. An example would be if the upper layer issues a T-DUnitdata.request containing an PDU which is larger than the maximum size PDU allowed by the specific WDP implementation. In this case a T-DError Indication would be returned to the upper layer with an error code indicating the PDU size is too large.

Primitive **Parameter**	**T-DError**	
	req	***ind***
Source Address	-	O
Source Port	-	M
Destination Address	-	O
Destination Port	-	O
Error Code	-	M

5.2.2.1 Source Port

The application address associated with the source address of the requesting communication instance.

5.2.2.2 Error Code

An error code carried by the D-Error primitive to the higher layer. The error codes are of local significance only and SHOULD match the “Reason Code “ of the PM-P-ABORT.indication service of the P_Mul Sub-Layer.

6 SECURITY

The send-to-end security services described in the X.500 standard series, cannot be used in this tactical protocol solution. The reason for this is that the services for establishing and terminating a connection is not used and therefore the parameters for supporting the security information in the bind operation is not transferred to the peer side. Some of the operations are also simulated (or “faked”) and therefore the arguments containing the security information (like signatures) will not be transferred.

Security may be performed at a lower layer in the OSI model (e.g. IP krypto).

7 REFERENCES

(1) AC/322(SC/5)N/224, Ratification Draft of STANAG 4406 (Ed. 1): Military Message Handling System, Annex E: Tactical MMHS Protocol and Profile Solution, Editor Anders Eggen (FFI, Norway)

(2) Data provided by Mr. Chris Bonatti IECA Inc. (1999).

(3) (1990): ISO/IEC: IS 8825, Information Technology - Open Systems Interconnection - Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1).

(4) (1991): ISO/IEC: DIS 8825-2, Information Technology - Open Systems Interconnection - Specification of ASN.1 Encoding Rules - Part 2: Packet Encoding Rules.

(5) (1988): CCITT X.225 Session Protocol Specification for OSI for CCITT Applications.

(6) (1988): CCITT X.226 Presentation Protocol Specification for OSI for CCITT Applications.

(7) (1988): CCITT X.227 Association Control Protocol Specification for OSI for CCITT Applications.

(8) (1988): CCITT X.228 Reliable Transfer: Protocol Specification.

(9) (1998): C. Kenneth Miller, CTO: Data Distribution Over IP in High Error Rate Military Environments , IEEE MILCOM 1998.

(10) Eggen, A. MMHS WG 258 Tactical MMHS Requirements and Scenario Document.

(11) Eggen, A. MMHS WG 254 Tactical MMHS Protocol Solutions.

(12) Eggen, A. MMHS WG 168 Tactical MMHS: Sources of overhead

(13) ACP 142, P_Mul: A protocol for reliable multicast messaging in bandwidth constrained and delayed acknowledgement (EMCON) environments.

(14) IETF RFC 791 Internet Protocol (IP)

(15) IETF RFC 768 User Datagram Protocol (UDP)

(16) Wireless Application Protocol (WAP), Wireless Datagram Protocol (WDP) Specification, Wireless Application Protocol Forum Ldt. 1999

(17) STANAG 5066 (Edition 1), “Profile for High Frequency (HF) Radio Data Communications”, AC/322(SC/6)N/277

(18) IETF RFC 1950 ZLIB Compressed Data Format Specification version 3.3

(19) IETF RFC 1951 Compressed Data Format Specification version 1.3

(20) IETF RFC 1166 Internet Numbers

(21) ISO10731 “Information Technology - Open Systems Interconnection - Basic Reference Model - Conventions for the Definition of OSI Services”, ISO/IEC 10731:1994.

(22) IETF RFC 2119 “Key words for use in RFCs to Indicate Requirement Levels”

(23) IETF RFC 1236 “IP to X.121 Address Mapping for DDN”

(24) ITU-T X.500-582 Date Networks And Open System Communications – The Directory, Date 1993

(25) ACP 133: Common Directory Services And Procedures