



GSM **T**ECHNICAL **S**PECIFICATION

GSM 03.49

May 1996

Version 5.1.0

Source: ETSI TC-SMG

Reference: TS/SMG-040349QR

ICS: 33.060.50

Key words: Digital cellular telecommunications system, Global System for Mobile communications (GSM)



Digital cellular telecommunications system (Phase 2+); Example protocol stacks for interconnecting Cell Broadcast Centre (CBC) and Base Station Controller (BSC) (GSM 03.49)

ETSI

European Telecommunications Standards Institute

ETSI Secretariat

Postal address: F-06921 Sophia Antipolis CEDEX - FRANCE

Office address: 650 Route des Lucioles - Sophia Antipolis - Valbonne - FRANCE

X.400: c=fr, a=atlas, p=etsi, s=secretariat - **Internet:** secretariat@etsi.fr

Tel.: +33 92 94 42 00 - Fax: +33 93 65 47 16

*

Copyright Notification: No part may be reproduced except as authorized by written permission. The copyright and the foregoing restriction extend to reproduction in all media.

© European Telecommunications Standards Institute 1996. All rights reserved.

Contents

Foreword	5
1 Scope	7
1.1 References	7
1.2 Abbreviations	8
2 A protocol stack which utilises an application-network layer convergence function for interconnecting CBC and BSC	8
2.1 CBSE Definition	9
2.2 ASN1 Specification	10
2.3 Application Rules for Avoidance of Collision of CBSE Operations	16
2.4 Non Support of 128 bytes of NS-user-data in Network Connection and Network Connection Release phases	16
3 An OSI Protocol Stack For Interconnecting CBC and BSC	17
3.1 Service elements on the application layer	17
3.2 Detailed specification of the CBRSE services	18
3.3 Application rules.....	32
3.3.1 Application rule set 1 Semi-permanent symmetric connection	32
3.3.2 Application rule set 2 Transient asymmetric connection	32
4 An SS7 Protocol Stack For Interconnecting CBC And BSC	33
History.....	34

Blank page

Foreword

This Global System for Mobile communications Technical Specification (GTS) has been produced by the Special Mobile Group (SMG) Technical Committee (TC) of the European Telecommunications Standards Institute (ETSI).

This GTS specifies three alternative approaches to the specification of protocol stacks of communication protocols for the purpose of fulfilling the service requirements of the primitives specified for the CBC - BSC interface in GSM 03.41 within the digital cellular telecommunications system (Phase 2/Phase 2+).

This GTS is a TC-SMG approved GSM technical specification version 5, which contains GSM Phase 2+ enhancements/features to the version 4 GSM technical specification. The ETR from which this Phase 2+ GTS has evolved is Phase 2 GSM ETR 107 edition 3 (GSM 03.49 version 4.6.0).

GTS are produced by TC-SMG to enable the GSM Phase 2 + specifications to become publicly available, prior to submission for the formal ETSI standards approval procedure. This ensures the earliest possible access to GSM Phase 2 + specifications for all Manufacturers, Network operators and implementors of the Global System for Mobile communications.

The contents of this GTS are subject to continuing work within TC-SMG and may change following formal TC-SMG approval. Should TC-SMG modify the contents of this GTS it will then be republished by ETSI with an identifying change of release date and an increase in version number as follows:

Version 5.x.y

where:

- y the third digit is incremented when editorial only changes have been incorporated in the specification;

- x the second digit is incremented for all other types of changes, i.e. technical enhancements, corrections, updates, etc.

NOTE: TC-SMG has produced documents which give the technical specifications for the implementation of the digital cellular telecommunications system. Historically, these documents have been identified as GSM Technical Specifications (GSM-TSs). These TSs may have subsequently become I-ETs (Phase 1), or ETs/ETSI Technical Reports (ETRs) (Phase 2). TC-SMG has also produced ETSI GSM TSs which give the technical specifications for the implementation of Phase 2+ enhancements of the digital cellular telecommunications system. These version 5.x.x GSM Technical Specifications may be referred to as GTs.

Blank page

1 Scope

No mandatory protocol between the Cell Broadcast Centre (CBC) and the Base Station Controller (BSC) is specified by GSM; this is a matter of agreement between CBC and PLMN operators.

This Global System for Mobile communications Technical Specification (GTS) specifies three alternative approaches to the specification of protocol stacks of communication protocols for the purpose of fulfilling the service requirements of the primitives specified for the CBC - BSC interface in GSM 03.41.

One approach is based upon the use of the complete OSI reference model (see X.200), another approach is based upon the use of only the lower 3 OSI layers, and another approach is based upon the use of CCITT Signalling System No. 7 (see Q.700).

Specifications are based upon individual contributions. Any judgement concerning functionality, completeness and advantages/disadvantages of implementation is intentionally omitted.

1.1 References

This GTS incorporates by dated and undated reference, provisions from other publications. These references are cited at the appropriate places in the text and the publications are listed hereafter. For dated references, subsequent amendments to or revisions of any of these publications apply to this GTS only when incorporated in it by amendment or revision. For undated references, the latest edition of the publication referred to applies.

- [1] GSM 01.04 (ETR 100): "Digital cellular telecommunication system (Phase 2); Abbreviations and acronyms".
- [2] GSM 03.41 (ETS 300 537): "Digital cellular telecommunication system (Phase 2); Technical realization of Short Message Service Cell Broadcast (SMSCB)".
- [3] GSM 12.20 (ETS 300 622): "Digital cellular telecommunication system (Phase 2); Network Management (NM) procedures and messages".
- [4] CCITT Recommendation Q.700: "Introduction to CCITT Signalling System No.7".
- [5] CCITT Recommendation Q.931: "Integrated services digital network.(ISDN) User-Network interface layer 3 specification for basic control".
- [6] CCITT Recommendation Q.932: "Generic procedures for the control of ISDN supplementary services".
- [7] CCITT Recommendation Q.941: "ISDN user-network interface protocol profile for management".
- [8] CCITT Recommendation Q.1400: "Architecture framework for the development of signalling and organisation, administration and maintenance protocols using OSI concepts".
- [9] CCITT Recommendation X.2 (1988): "International data transmission services and optional user facilities in public data networks and ISDNs".
- [10] CCITT Recommendation X.200: "Reference Model of Open Systems Interconnection for CCITT Applications".
- [11] CCITT Recommendation X.213: "Information technology - Network service definition for Open Systems Interconnection".
- [12] CCITT Recommendation X.215: "Session service definition for open systems interconnection for CCITT applications".

- [13] CCITT Recommendation X.217: "Association control service definition for open systems interconnection for CCITT applications".
- [14] CCITT Recommendation X.219: " Remote operations: model, notation and service definition".
- [15] CCITT Recommendation X.225: "Session protocol specification for Open Systems Interconnection for CCITT Applications".
- [16] CCITT Recommendation X.227: "Information technology - Open Systems Interconnection - protocol specification for the association".
- [17] CCITT Recommendation X.229: "Remote operations Protocol specification".

1.2 Abbreviations

Abbreviations used in this report are listed in GSM 01.04.

2 A protocol stack which utilises an application-network layer convergence function for interconnecting CBC and BSC

A convergence function (see Draft CCITT Recommendation Q.941 Report R 22 May 1990) which maps an application entity protocol directly to the Network Layer service defined by X.213 can provide a practical alternative to ACSE, ROSE and OSI layers 6, 5 and 4.

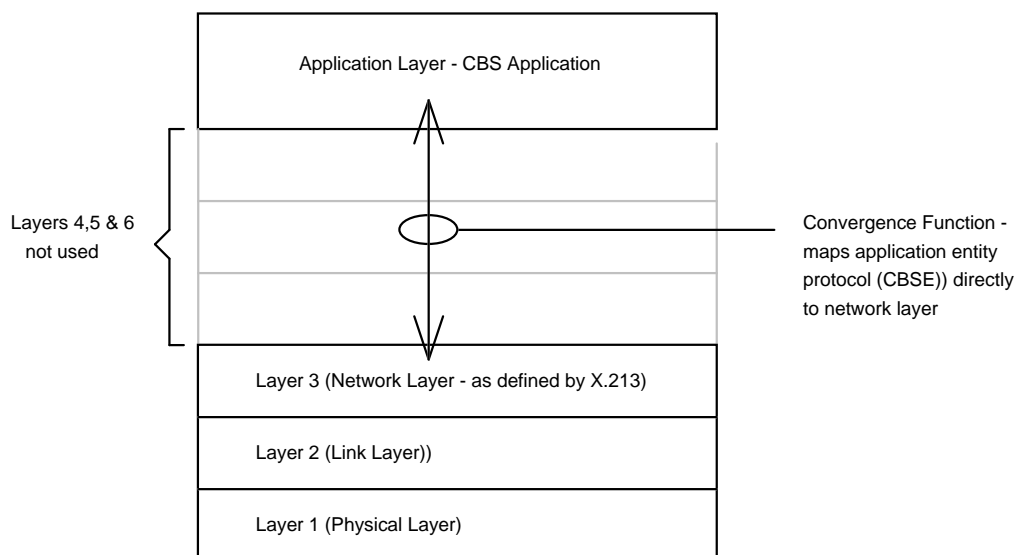


Figure 1

Draft CCITT Recommendation Q.941 proposes to map application layer protocols ACSE and ROSE via a convergence function to network layers defined by CCITT Recommendations Q.931 and Q.932.

The complexity of dealing with the many different network layer protocols is avoided by mapping the application protocols to the Network Layer Service defined by X.213. ACSE and ROSE are specifically defined in terms of the full OSI stack. The use of ACSE and ROSE is avoided by incorporating the functionality provided by ACSE and ROSE into the CBS protocol. The convergence function is embedded in the CBS protocol.

2.1 CBSE Definition

The Cell Broadcast Short Message Service Element (CBSE) is defined in terms of the following service:

CBSE-BIND

This operation must be invoked by the party which is responsible for establishing the application association; only after the application association has been established may the remaining CBSE services be used. This operation reports either success (via CBSE-Bind-Confirm) or failure (via CBSE-Bind-Failure).

CBSE-BIND will be mapped to/from N-CONNECT request/indication with CBSE-BIND parameters carried in NS-user-data (if the network layer does not support NS-user-data of 128 octets then CBSE-BIND parameters may be carried by the first N-DATA request/indication following establishment of the network layer connection - see section 2.4).

CBSE-BIND-CONFIRM

This operation must be invoked by a party to accept an application association.

CBSE-BIND-CONFIRM will be mapped to/from N-CONNECT confirm/response with CBSE-BIND-CONFIRM parameters carried in NS-user-data (if the network layer does not support NS-user-data of 128 octets then CBSE-BIND-CONFIRM may be carried as the second N-DATA request/indication following establishment of the network layer connection - see Section 2.4).

CBSE-BIND-FAILURE

This operation must be invoked by a party to reject an attempted application association.

CBSE-BIND-FAILURE will be mapped to/from N-DISCONNECT request/indication with CBSE-BIND-FAILURE parameters carried in NS-user-data (if the network layer does not support NS-user-data of 128 octets then CBSE-BIND-FAILURE parameters will not be carried by the network layer - i.e. NS-user-data will be discarded).

CBSE-UNBIND

This operation must be invoked by a party to release the application association.

CBSE-UNBIND will be mapped to/from N-DISCONNECT request/indication with CBSE-UNBIND parameters be carried in NS-user-data (if the network layer does not support NS-user-data of 128 octets then CBSE-UNBIND parameters may be carried by the N-DATA request/indication preceding N-DISCONNECT - see Section 2.4).

CBSE-WRITE-REPLACE, CBSE-KILL-MESSAGE, CBSE-REPORT-SUCCESS,

CBSE-STATUS-CBCH-QUERY, CBSE-STATUS-CBCH-QUERY-RESP., CBSE-STATUS-MESS.-QUERY, CBSE-STATUS-MESS.-QUERY-RESP., CBSE-REPORT-FAILURE, CBSE-BSC-RESTART, CBSE-SET-DRX. CBSE-SET-DRX-RESP

Application data units CBSE-WRITE-REPLACE, CBSE-KILL-MESSAGE, CBSE-REPORT-SUCCESS, CBSE-STATUS-CBCH-QUERY, CBSE-STATUS-CBCH-QUERY-RESP., CBSE-STATUS-MESS.-QUERY, CBSE-STATUS-MESS.-QUERY-RESP., CBSE-REPORT-FAILURE, CBSE-BSC-RESTART, CBSE-SET-DRX. CBSE-SET-DRX-RESP provide the services specified via primitives Write-Replace, Kill-Message, Report-Success, Status-CBCH, Status-CBCH-Response, Status-Message, Status-Message-Response, Report-Failure, BSC-Restart, Set-DRX. and Set-DRX-Response respectively in GSM 03.41.

These application data units will be mapped to/from N-DATA request/indication.

2.2 ASN1 Specification

The Abstract Syntax Notation of the Cell Broadcast Short Message Service Element

CBSE

1st module of 2:

CBS-UsefulDefinitions

```
CBS-UsefulDefinitions {  
    ccitt identified-organization (4) etsi (0) mobile-domain (0)  
    gsm-messaging (4) gsm-sms3 (12) usefulDefinitions (10) }
```

DEFINITIONS

IMPLICIT TAGS

```
::=  
BEGIN
```

```
    ID ::= OBJECT IDENTIFIER
```

```
mobile-domain ID ::= {ccitt identified-organization (4) etsi (0) mobile-domain(0)}
```

```
--    root for all sms allocations
```

```
gsm-messaging ID ::= { mobile-domain gsm-messaging(4) }
```

```
--    categories
```

```
gsm-sms3 ID ::= { gsm-messaging 12 }
```

```
END
```

2nd module of 2:

Application Protocol

```
ApplicationProtocol {
    ccitt identified-organization (4) etsi (0) mobile-domain(0)
    gsm-messaging(4) gsm-sms3 (12) applicationProtocol(11) }
```

DEFINITIONS

IMPLICIT TAGS

::=

BEGIN

- CBSE-BIND will be carried as N-CONNECT request/indication
- CBSE-BIND-Parameters will be carried in the User Data field of the N-CONNECT request/indication message.

```
CBSE-BIND-Parameters ::= SEQUENCE {
    initiatorID [0] Name,
    password [1] Password OPTIONAL
}
```

- Above and in CBSE-BIND-CONFIRM
- initiatorID/respID: identify the initiating/responding telecommunication subsystem
- password: may assist in authentication

```
Name ::= SEQUENCE {
    operator [0] Operator OPTIONAL,
    bilateralAgreem [1] BilateralAgreem OPTIONAL,
    dataNetworkAddress [2] XI2IAddress OPTIONAL,
    iSDNAddress [3] CBS-Address OPTIONAL
}
```

- operator is a text string containing the name of the CBC/PLMN operator. bilateralagreem is a text string identifying the bilateral agreement between the CBC and the PLMN operators which allows for this association to be established.
- dataNetworkAddress is the PSPDN X.121 address of the CBC/BSC issuing the BIND or CONFIRM, occurring only if a PSPDN is used.
- iSDNAddress is the PLMN address of the CBC (same datum in both BIND and CONFIRM).
- Any pair of subsets of these parameters may be used to identify the CBC and the BSC to one another.
- upper bound settings

```
Operator ::= PrintableString (SIZE (0..20))
```

```
BilateralAgreem ::= PrintableString (SIZE (0 .. 20))
```

```
XI2IAddress ::= NumericString (SIZE(0..15))
```

-- Definition of Cell Broadcast Short Message Service address

```
CBS-Address ::= [APPLICATION 0] SEQUENCE {
  address-type  INTEGER { unknown-type(0),
                        international-number(1),
                        national-number(2),
                        network-specific-number(3),
                        short-number(4) },
  numbering-plan INTEGER { unknown-numbering(0),
                        SDN-numbering(1),
                        data-network-numbering(3),
                        telex-numbering(4),
                        national-numbering(8),
                        private-numbering(9) },
  address-value CHOICE {
    octet-format
    SemiOctetString
    -- other formats are for further study
  }
}
```

-- each octet contains two binary coded decimal digits

SemiOctetString ::= OCTET STRING (SIZE(1..10))

Password ::= PrintableString (SIZE(0..20))

-- CBSE-BIND-CONFIRM will be carried as N-CONNECT response/confirm
 -- CBSE-BIND-CONFIRM parameters will be carried in User Data of the N-CONNECT
 -- response/confirm message

```
CBSE-BIND-CONFIRM-Parameters ::= SEQUENCE {
  respId [0] Name,
  password [1] Password OPTIONAL
}
```

-- The following defines the choices and tags for the N-DISCONNECT.request/indication User Data.

```
Applic-protocol-discs ::= CHOICE {
  bindfail [1] CBSE-BIND-FAILURE,
  unbindreq [2] CBSE-UNBIND
}
```

CBSE-BIND-FAILURE ::= Connect-failure-reason

-- connect-failure-reason provides one of the error indications given in the following table.

Table 1

Error indications	Reason
not-entitled	The responder is not entitled to accept a request for an association between itself and the initiator.
temporary-overload	The responder is not capable of establishing an association due to temporary overload.
temporary-failure	The responder is not capable of establishing an association due to a temporary failure.
incorrect-ID-or-password	The responder will not accept the request to establish an association between itself and the initiator due to incorrect identity or password.

--

```
Connect-failure-reason ::= INTEGER {
    not-entitled (0),
    temporary-overload (1),
    temporary-failure (2),
    incorrect-ID-or-password (3)
}
```

CBSE-UNBIND ::= NULL

-- The following defines the choices and tags for the N-DATA.request/indication User Data

```
CBSMSEapdus ::= CHOICE {
    cbse-WRITE-REPLACE [1] Write-Replace,
    cbse-KILL-MESSAGE [2] Kill-Message,
    cbse-REPORT-SUCCESS [3] Report-Success,
    cbse-STATUS-CBCH-QUERY [4] Status-CBCH,
    cbse-STATUS-CBCH-QUERY-RESP [5] Status-CBCH-Resp,
    cbse-STATUS-MESSAGE-QUERY [6] Status-Message,
    cbse-STATUS-MESS-QUERY-RESP [7] Status-Mess-Resp,
    cbse-REPORT-FAILURE [8] Report-Failure,
    cbse-BSC-RESTART [9] BSC-Restart,
    cbse-RESET [10] Reset,
    cbse-FAILURE-IND [11] Failure-Ind
}
```

```
Write-Replace ::= SEQUENCE {
    message-Identifier INTEGER (0 .. 65535),
    new-Serial-Number Serial-Number,
    no-of-Pages INTEGER (1 .. 15),
    data-coding-scheme INTEGER (0 .. 255),
    cell-list Cell-List,
    repetition-Rate INTEGER (1 .. 7),
    no-of-broadcast-req INTEGER (0 .. 2880),
    cbs-Page-Inf SEQUENCE OF Page-Inf,
    old-Serial-Number [3] Serial-Number OPTIONAL,
    category INTEGER (0..2) OPTIONAL
}
```

```
Page-Inf ::= SEQUENCE {
    message-info-useful-octets INTEGER (0..82),
    message-info-page Message-Info-Page
}
```

Message-Info-Page ::= OCTET STRING (SIZE(82))

Cell-Id-Disc ::= OCTET-STRING (SIZE(1))
-- values from the following table

```
lacAndCi ::= '00000001' -- 2 Octet lac followed by 2 Octet Cell Id
ciOnly ::= '00000010' -- Cell Id only
```

Cell-Id ::= OCTET-STRING (SIZE(4))

--Note: If Cell-Id-Disc equals ciOnly then only the last 2 octets of Cell-ID are to be considered --and the first 2 octets are filler octets.

```
Cell ::= SEQUENCE OF {
    disc Cell-Id-Disc,
    id Cell-Id
}
```

```

Cell-List ::= SEQUENCE {
    length      INTEGER, -- number of cells in the list
    disc       Cell-Id-Disc,
    list       SEQUENCE OF Cell-Id
}

Serial-Number ::= INTEGER (0 .. 65535)

Kill-Message ::= SEQUENCE {
    message-Identifier INTEGER (0 .. 65535),
    old-Serial-Number Serial-Number,
    cell-List          Cell-List
}

Report-Success ::= SEQUENCE {
    message-Identifier INTEGER (0 .. 65535),
    serial-Number      Serial-Number
    SEQUENCE OF SEQUENCE {
        cell-id          Cell,
        no-of-broadcasts-compl INTEGER
    } OPTIONAL
    SEQUENCE OF SEQUENCE {
        cell-id          Cell,
        cause            Failure-Reason,
        diagnostic       Diagnostic-Info  OPTIONAL
    } OPTIONAL
}

Status-CBCH ::= SEQUENCE{
    cell-List          Cell-List
}

Status-CBCH-Resp ::= SEQUENCE
    SEQUENCE OF SEQUENCE{
        cell-id          Cell,
        cbch-loading     INTEGER (0 .. 1019)
        -- indicates total number of messages broadcast
        -- across the air interface within the last 32
        -- minutes (min: 0, max: 1019)
    }
    SEQUENCE OF SEQUENCE{
        cell-id          Cell,
        cause            Failure-Reason,
        diagnostic       Diagnostic-Info  OPTIONAL
    } OPTIONAL
}

Status-Message ::= SEQUENCE {
    message-Identifier INTEGER (0..65535),
    current-Serial-No  Serial-Number,
    cell-List          Cell-List
}

```

```
Status-Mess-Resp ::= SEQUENCE {
    message-Identifier    INTEGER (0 .. 65535),
    old-serial-number    Serial-Number,
    SEQUENCE OF SEQUENCE {
        cell-id          Cell,
        no-of-broadcasts-compl  INTEGER
    }
    SEQUENCE OF SEQUENCE {
        cell-id          Cell,
        cause            Failure-Reason,
        diagnostic       Diagnostic-Info    OPTIONAL
    } OPTIONAL
}
```

```
Report-Failure ::= SEQUENCE {
    cause                Failure-Reason,
    diagnostic           Diagnostic-Info    OPTIONAL,
    message-Identifier  INTEGER (0 .. 65535)  OPTIONAL,
    serial-Number       Serial-Number    OPTIONAL
}
```

```
Failure-Reason ::= INTEGER {
    parameter-not-recognised (0),
    (1), -- not used
    parameter-value-invalid (2),
    valid-CBS-message-not-identified (3),
    cell-identity-not-valid (4),
    unrecognised-message (5),
    missing-mandatory-element (6),
    bss-capacity-exceeded (7),
    cell-memory-exceeded (8),
    bss-memory-exceeded (9),
    unspecified-error (10),
    incompatible-DRX-parameter (11)
}
```

Diagnostic-Info ::= OCTET STRING (SIZE (1..20))

```
BSC-Restart ::= SEQUENCE {
    cell-list            Cell-List,
    recovery-Indication  BOOLEAN          OPTIONAL
}
```

```
Reset ::= SEQUENCE {
    cell-list            Cell-List
}
```

```
Set-DRX ::= SEQUENCE {
    cell-list            Cell-List,
    schedule-Period     INTEGER (0 .. 48)    OPTIONAL
    reserved-Slots      INTEGER (0 .. 48)    OPTIONAL
}
```

```
Set-DRX-Resp ::= SEQUENCE {
    cell-list            Cell-List,
    SEQUENCE OF SEQUENCE {
        cell-id          Cell,
        cause            Failure-Reason}    OPTIONAL
}
```

Failure-Ind ::= cell-list Cell-List

END

2.3 Application Rules for Avoidance of Collision of CBSE Operations

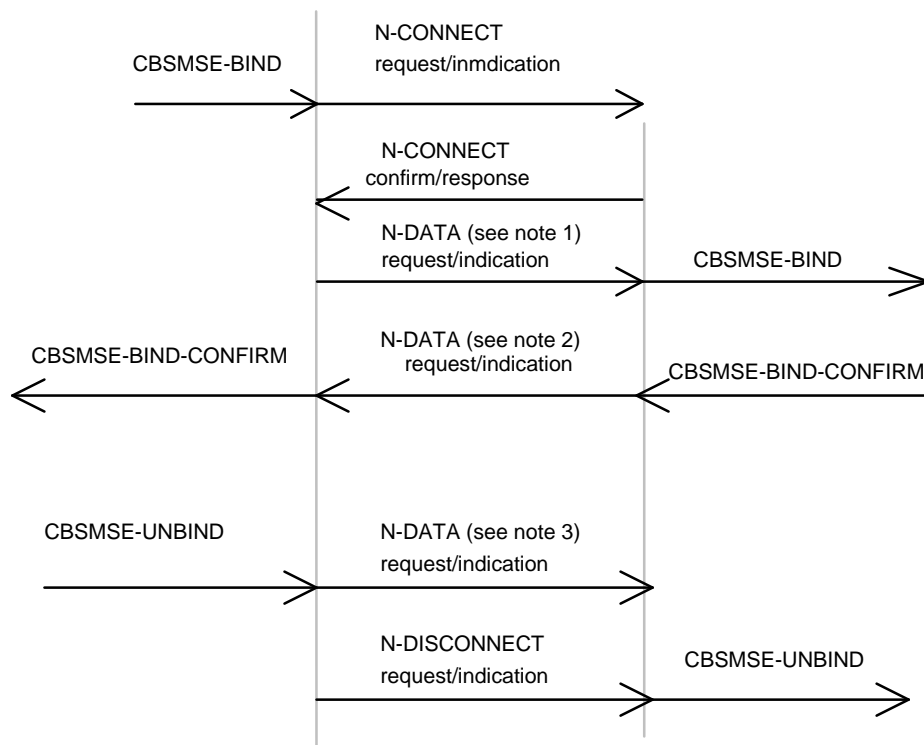
For the purpose of establishing the association between CBSEs in CBC and BSC then either the CBC or the BSC shall be designated as the entity responsible for initiating the association by the operation CBSE-BIND.

Following premature release of the association by N-DISCONNECT then either the CBC or the BSC shall be designated as the entity responsible for re-establishing the association.

Following receipt of N-RESET any command sent by the CBC, for which no corresponding response has been received by the CBC, will be re-sent to the BSC.

2.4 Non Support of 128 bytes of NS-user-data in Network Connection and Network Connection Release phases

It is generally intended to make the support of 128 bytes of NS user-data mandatory (see Sections 12.2.8 and 13.2.3 of X.213). CCITT Recommendation X.2 regards provision of Fast Select as essential, thereby ensuring support of 128 bytes of NS-user-data in network connection and network connection release phases. For an interim period support of 128 bytes of NS-user-data in network connection and network connection release phases will remain a provider option. The following figures are therefore provided in order to indicate how CBSE-BIND, CBSE-BIND-CONFIRM and CBSE-UNBIND should be mapped to/from an OSI Network Service definition which does not support 128 bytes of NS-user-data in network connection and network connection release phases.



NOTE 1: CBSE-BIND parameters are carried as NS-user-data

NOTE 2: CBSE-BIND-CONFIRM is carried as NS-user-data

NOTE 3: CBSE-UNBIND parameters are carried as NS-user-data

Figure 2

3 An OSI Protocol Stack For Interconnecting CBC and BSC

This section specifies a stack of communication protocols in terms of the OSI Reference Model (see X.200) and therefore makes use of all seven layers for the purpose of fulfilling the service requirements of the primitives specified for the CBC - BSC interface in GSM 03.41. The CBS application layer (layer 7) is mapped to the Presentation Layer via ACSE (see X.217 and X.227) and ROSE (see X.219 and X.229). Only the Kernel functional unit of the Presentation Layer is used. Only the Kernel and Duplex functional units are used in the Session Layer (see X.215 and X.225).

3.1 Service elements on the application layer

An association (class 3) between CBRSEs is formed via ACSE and ROSE operations (class 2 and 5) are used to implement the service requirements specified for the CBC - BSC interface in GSM 03.41.

This results in an asynchronous asymmetric situation where the application entity in the CBC or BSC can invoke a CBRSE operation at any time.

The new CBRSE service element is first defined in the following section, and then specified in ASN.1 notation in section 3.2.

CBRSE definition

This service element defines the following services:

CBRSE-BIND This operation will normally be invoked by the CBC to establish the application association, but in exceptional circumstances (e.g. following loss of data) the BSC may invoke the operation; only thereafter the remaining CBRSE services may be used. This operation reports either success or failure (result or error).

CBR-WRITE-REPLACE, CBR-KILL-MESSAGE, CBR-STATUS-CBCH-QUERY, CBR-STATUS-MESSAGE-QUERY, CBR-RESET, CBR-SET-DRX

These operations may be invoked by the application entity in the CBC; They are used to relay commands from the CBC to a given BSC. The operations report either success or failure.

CBR-RESTART, CBR-FAILURE

This operation may be invoked by the application entity in the BSC. The operation reports success or failure.

CBR-UNBIND This operation must be invoked by the CBC as the last CBRSE operation before releasing the application association. This operation reports success only.

Of the services defined above, CBR-WRITE-REPLACE semantically means the relay of cell broadcast messages across the CBC-BSC-connection in order to add them to the message list in the BSC, whereas CBR-KILL-MESSAGE is used to delete messages from the message list. The CBR-STATUS-CBCH-QUERY command inquires after the current loading of a specific cell broadcast channel, while the CBR-STATUS-MESSAGE-QUERY command requests status information concerning a specific message. The CBR-SET-DRX command sets the DRX related parameters. These five services combine the primitives defined in GSM 03.41, which can be invoked by the CBC.

The CBR-BIND service is used to exchange identifications, passwords, etc., and in order to negotiate the usage of the other services. The CBR-UNBIND service prepares for the release of the application association.

3.2 Detailed specification of the CBRSE services

On the following pages, the new CBRSE service element is specified with the ASN.1 notation, together with the entire protocol.

The Abstract Syntax Notation of
the Cell Broadcast Relay Service Element

CBRSE

1st module of 3:

CBS-UsefulDefinitions

```
CBS-UsefulDefinitions  {
    ccitt identified-organization (4) etsi (0) mobile-domain(0)
    gsm-messaging(4) gsm-sms4(13) usefulDefinitions(0) }
```

DEFINITIONS

IMPLICIT TAGS

```
::=
```

BEGIN

```
EXPORTS      id-cb-ot-CBC, id-cb-ot-BSC, id-cb-port,
              id-cb-ac-so, id-cb-CBRSE, id-cb-as-CBRSE;
```

```
ID ::= OBJECT IDENTIFIER
```

```
mobile-domain ID ::= { ccitt identified-organization (4) etsi (0) mobile-domain (0)}
```

```
-- root for all sms allocations
```

```
gsm-messaging ID ::= { mobile-domain gsm-messaging (4) }
```

```
gsm-sms4 ID ::= {gsm-messaging (13)}
```

```
-- categories
```

```
id-cb-mod ID ::= { gsm-messaging 1 } -- modules
id-cb-ot ID ::= { gsm-messaging 2 } -- object type
id-cb-pt ID ::= { gsm-messaging 3 } -- port types
id-cb-ac ID ::= { gsm-messaging 4 } -- appl. contexts
id-cb-ase ID ::= { gsm-messaging 5 } -- ASEs
id-cb-as ID ::= { gsm-messaging 6 } -- abstract syntaxes
```

```
-- modules
```

```
usefulDefinitions ID ::= { gsm-sms4 0 }
relayProtocol ID ::= { gsm-sms4 1 }
relayAbstractService ID ::= { gsm-sms4 2 }
```

```
-- object types
```

```
id-cb-ot-CBC ID ::= { id-cb-ot 0 }
id-cb-ot-BSC ID ::= { id-cb-ot 1 }
```

```
-- port types

id-cb-port      ID ::= { id-cb-pt 0 }

-- application contexts

id-cb-ac-so     ID ::= { id-cb-ac 0 }

-- application service elements

id-cb-CBRSE    ID ::= { id-cb-ase 0 }

-- abstract syntaxes

id-cb-as-CBRSE ID ::= { id-cb-as 0 }

END
```

2nd module of 3

RelayAbstractService

```
RelayAbstractService {
    ccitt identified-organization (4) etsi (0) mobile-domain(0)
    gsm-messaging(4) gsm-sms4(13) relayAbstractService(2) }
```

DEFINITIONS

IMPLICIT TAGS

::=

BEGIN

IMPORTS

BIND, UNBIND

```
FROM Remote-Operations-Notation {
    joint-iso-ccitt remote-operations(4) notation(0) }
```

OBJECT, PORT, ABSTRACT-BIND, ABSTRACT-UNBIND,
ABSTRACT-OPERATION, ABSTRACT-ERROR

```
FROM AbstractServiceNotation {
    joint-iso-ccitt mhs-motis(6) asdc(2) modules(0) notation(1) }
```

id-cb-ot-CBC, id-cb-ot-BSC, id-cb-port

```
FROM CBS-UsefulDefinitions{
    ccitt identified-organization (4) etsi (0) mobile-domain(0)
    gsm-messaging(4) gsm-sms4(13) usefulDefinitions(0) }
```

-- upper bound settings

ub-operator-name-length INTEGER ::= 20

ub-agreem-name-length INTEGER ::= 20

ub-X121Address-length INTEGER ::= 15

ub-password-length INTEGER ::= 20

-- Objects

-- The CBC and the BSC are modelled as atomic objects, cBC--Object and bSC-Object. Each
-- object has one port for the interconnection. ([S] and [C] indicate supply and consumption of
-- services, respectively).

```
cBC-Object OBJECT
    PORTS { cBR-port [S] }
    ::= id-cb-ot-CBC
```

```
bSC-Object OBJECT
    PORTS { cBR-port [C] }
    ::= id-cb-ot-BSC
```

```
-- Port
cBR-port  PORT
          CONSUMER INVOKES { CBR-Restart
                          CBR-Failure
                          }
          SUPPLIER INVOKES { CBR-Write-Replace
                          CBR-Kill-Message
                          CBR-Status-CBCH-Query
                          CBR-Status-Message-Query
                          CBR-Reset
                          CBR-Set-DRX
                          }
          ::= id-cb-port
```

-- The CBR-Bind operation

-- Both, BIND and UNBIND operations, are exclusively within the responsibility of the CBC. The
 -- BIND operation is therefore always requested by the CBC

```
CBR-Bind ::=
          ABSTRACT-BIND
          TO { cBR-port }
          BIND
          ARGUMENT  CBR-Bind-Parameters
          RESULT    CBR-Bind-confirm
          BIND-ERROR CBR-Bind-failure
```

-- The CBR-Unbind operation

-- The UNBIND is a harsh release of the association and all outstanding operations are aborted.
 -- UNBIND is always requested by the CBC. The CBC and the BSC should negotiate (during
 -- CBR-BIND) the use of services on the association (the operations parameter - list of operation
 -- types for the association) in such a way that no harmful losses of operations occur.

```
CBR-Unbind ::=
          ABSTRACT-UNBIND
          FROM { cBR-port }
          UNBIND
          ARGUMENT  Time-when-connected
          RESULT    Time-when-disconnected
```

- Association control parameters

```
CBR-Bind-Parameters ::= SEQUENCE {
                          initiatorID [0] Name,
                          password    [1] Password OPTIONAL,
                          pswNeeded   [2] BOOLEAN,
                          iniType     [3] Telecom-System-Type,
                          operations  [4] List-of-Operations,
                          transient    [5] BOOLEAN
                          }
}
```

-- Above and in SMR-Bind-confirm
 -- initiatorID/respID: identify the initiating/responding telecommunication subsystem
 -- password: may assist in authentication
 -- pswNeeded (BIND only): requests password into SMR-Bind, SMR-Bind-Confirm
 -- iniType/respType: identify the system entity
 -- operations: lists the SM relay operations requested and supported on the association:
 -- operations listed in both the BIND and the CONFIRM may be used (i.e. this is a negotiation
 -- between CBC and BSC)
 -- transient: forces the association (and the underlying connections), transient: it must be
 -- UNBOUND as soon as there are no operations to be performed

Name ::= SEQUENCE {

```

    operator          [0] Operator          OPTIONAL,
    bilateralAgreem   [1] BilateralAgreem   OPTIONAL,
    dataNetworkAddress [2] X121Address   OPTIONAL,
    iSDNAddress       [3] CBS-Address   OPTIONAL
}

```

-- operator is a text string containing the name of the CBC/PLMN operator. bilateralAgreem is a text string identifying the bilateral agreement between the CBC and the PLMN operators which allows for this association to be established.
 -- dataNetworkAddress is the PSPDN X.121 address of the CBC/BSC issuing the BIND or CONFIRM, occurring only if a PSPDN is used.
 -- iSDNAddress is the PLMN address of the CBC as seen by the MSs (same datum in both BIND and CONFIRM).

-- Any pair of subsets of these parameters may be used to identify the CBC and the BSC to one another.

Operator ::= PrintableString (SIZE(0..ub-operator-name-length))

BilateralAgreem ::= PrintableString (SIZE(0..ub-agreem-name-length))

X121Address ::= NumericString (SIZE(0..ub-X121Address-length))

-- CBS-Address is specified later in this module.

Password ::= PrintableString (SIZE(0..ub-password-length))

Telecom-System-Type ::= INTEGER {

```

    cell-Broadcast-Service-Centre (0),
    public-Land-Mobile-Network (1)

```

-- Extensions are possible: additional telecommunication subsystems might adopt this service element for their interconnection.

}

List-of-Operations ::= BIT STRING {

```

    cBR-From-CBC-Write-Replace (0),
    cBR-From-CBC-Kill-Message (1),
    cBR-From-CBC-Status-CBCH-Query (2),
    cBR-From-CBC-Status-Message-Query (3),
    cBR-From-BSC-Restart (4),
    cBR-From-CBC-Reset (5),
    cBR-From-BSC-Failure (6),
    cBR-From-CBC-Set-DRX (7)

```

-- Extensions are possible: additional operations may be defined within this service element. Existing systems should tolerate unknown values, but negotiate not to perform unknown operations.}

CBR-Bind-confirm ::= SEQUENCE {

```

    respId [0] Name,
    password [1] Password OPTIONAL,
    respType [3] Telecom-System-Type,
    operations [4] List-of-Operations,
    transient [5] BOOLEAN,
    connectTime [6] Time-when-connected
}

```

```
CBR-Bind-failure ::= SEQUENCE {
    connect-failure-reason
    [0] Connect-failure
}
```

-- connect-failure-reason contains one of the error indications given in the following table.

Table 2

Error indications	Reason
not-entitled	The responder is not entitled to accept a request for an association between itself and the initiator.
temporary-overload	The responder is not capable of establishing an association due to temporary overload.
temporary-failure	The responder is not capable of establishing an association due to a temporary failure (having impact on an entity at SM-RL or at layers above).
incorrect-ID-or-password	The responder will not accept the request to establish an association between itself and the initiator due to incorrect identity or password.
not-supported	The responder does not recognize the telecommunication subsystem type of the initiator, or cannot support any of the operations suggested on the association.

--

```
Connect-failure ::= INTEGER {
    not-entitled (0),
    temporary-overload (1),
    temporary-failure (2),
    incorrect-ID-or-password (3),
    not-supported (4)
}
```

```
Time-when-disconnected ::= UTCTime
Time-when-connected ::= UTCTime
```

-- The CBR-Write-Replace operation

```
CBR-Write-Replace ::=
    ABSTRACT-OPERATION
    ARGUMENT Write-Replace
    RESULT Report-Success
    ERRORS {Parameter-not-recognized,
    Parameter-value-invalid,
    Valid-CBS-message-not-identified,
    Cell-identity-not-valid,
    Unrecognized-message,
    Missing-mandatory-element,
    BSS-capacity-exceeded,
    Cell-memory-exceeded,
    BSS-memory-exceeded,
    Unspecified-error
    }
```

-- The CBR-Kill-Message operation

CBR-Kill-Message ::=

```

    ABSTRACT-OPERATION
    ARGUMENT      Kill-Message
    RESULT        Report-Success
    ERRORS        {Parameter-not-recognized,
                  Parameter-value-invalid,
                  Unrecognized-message,
                  Missing-mandatory-element,
                  Unspecified-error
                  }

```

-- The CBR-Status-CBCH-Query operation

CBR-Status-CBCH-Query ::=

```

    ABSTRACT-OPERATION
    ARGUMENT      Status-CBCH-Request
    RESULT        Status-CBCH-Response
    ERRORS        {Parameter-not-recognized,
                  Parameter-value-invalid,
                  Cell-identity-not-valid,
                  Unrecognized-message,
                  Missing-mandatory-element,
                  Unspecified-error
                  }

```

-- The CBR-Status-Message-Query operation

CBR-Status-Message-Query ::=

```

    ABSTRACT-OPERATION
    ARGUMENT      Status-Message-Request
    RESULT        Status-Message-Response
    ERRORS        {Parameter-not-recognized,
                  Parameter-value-invalid,
                  Cell-identity-not-valid,
                  Unrecognized-message,
                  Missing-mandatory-element,
                  Unspecified-error
                  }

```

-- The CBR-BSC-Restart operation

CBR-Restart ::=

```

    ABSTRACT-OPERATION
    ARGUMENT      Restart-Indication
    RESULT
    ERRORS        {Parameter-not-recognised,
                  Parameter-value-invalid,
                  Cell-identity-not-valid,
                  Unrecognized-message,
                  Missing-mandatory-element,
                  Unspecified-error
                  }

```

-- The CBR-Reset operation


```
CBR-Reset ::=
    ABSTRACT-OPERATION
    ARGUMENT      Reset-Request
    RESULT        Result-Request
    ERRORS        {Parameter-not-recognized,
                  Parameter-value-invalid,
                  Cell-identity-not-valid,
                  Unrecognized-message,
                  Missing-mandatory-element
                  Unspecified-error
                  }
```

-- The CBR-Failure operation

```
CBR-Failure ::=
    ABSTRACT-OPERATION
    ARGUMENT      Failure-Indication
    RESULT
    ERRORS        {Parameter-not-recognized,
                  Parameter-value-invalid,
                  Cell-identity-not-valid,
                  Unrecognized-message,
                  Missing-mandatory-element,
                  Unspecified-error
                  }
```

-- The CBR-Set-DRX operation

```
CBR-Set-DRX ::=
    ABSTRACT-OPERATION
    ARGUMENT      Set-DRX
    RESULT        Set-DRX-Resp
    ERRORS        {Parameter-not-recognized,
                  Parameter-value-invalid,
                  Valid-CBS-message-not-identified,
                  Cell-identity-not-valid,
                  Unrecognized-message,
                  Missing-mandatory-element,
                  BSS-capacity-exceeded,
                  Cell-memory-exceeded,
                  BSS-memory-exceeded,
                  Unspecified-error,
                  Incompatible-DRX-parameter
                  }
```

-- CBR operation ARGUMENT lists

```
Write-Replace ::= SEQUENCE {
    message-identifier      INTEGER (0 .. 65535),
    new-serial-number       Serial-Number,
    no-of-pages             INTEGER (1 .. 15),
    data-coding-scheme     INTEGER (0 .. 255),
    cell-list               Cell-List,
    repetition-rate        INTEGER (1 .. 7),
    no-of-broadcast-req    INTEGER (0 .. 2880),
    cBS-page-info          SEQUENCE OF Page-Inf,
    old-serial-number       [1] Serial-Number OPTIONAL,
    category                INTEGER (0..2)          OPTIONAL
}
```

```

Kill-Message ::= SEQUENCE {
    message-identifier INTEGER (0 .. 65535),
    old-serial-number  Serial-Number,
    cell-List          Cell-List
}

Status-CBCH-Request ::= SEQUENCE {
    cell-List          Cell-List
}

Status-Message-Request ::= SEQUENCE {
    message-identifier INTEGER (0 .. 65535),
    old-serial-no      Serial-Number
}

Restart-Indication ::= SEQUENCE {
    cell-list          Cell-List,
    recovery-Indication  BOOLEAN      OPTIONAL
}

Failure-Indication ::= Cell-List

Reset-Indication ::= Cell-List

Set-DRX ::= SEQUENCE {
    cell-list          Cell-List          OPTIONAL,
    schedule-Period    INTEGER (0 .. 48)  OPTIONAL,
    reserved-Slots     INTEGER (0 .. 48)  OPTIONAL
}

-- CBR operation RESULT list

Report-Success ::= SEQUENCE {
    message-identifier INTEGER (0 .. 65535),
    serial-number       Serial-Number,
    SEQUENCE OF SEQUENCE{
        cell-id          Cell,
        no-of-broadcasts-compl  INTEGER
    }OPTIONAL
    SEQUENCE OF SEQUENCE{
        cell-id          Cell,
        cause            Failure-Reason,
        diagnostic       Diagnostic-Info  OPTIONAL
    }OPTIONAL
}

Status-CBCH-Response ::= SEQUENCE
    SEQUENCE OF SEQUENCE {
        cell-id          Cell,
        cbch-loading     INTEGER (0..1019)
        -- indicates the total number of messages broadcast
        -- across the air interface within last 32
        -- minutes (min: 0, max: 1019)
    }
    SEQUENCE OF SEQUENCE{
        cell-id          Cell,
        cause            Failure-Reason,
        diagnostic       Diagnostic-Info  OPTIONAL
    }OPTIONAL
}

```

```
Status-Message-Response ::= SEQUENCE {
    message-identifier INTEGER (0 .. 65535),
    old-serial-number Serial-Number,
    SEQUENCE OF SEQUENCE {
        cell-id Cell,
        no-of-broadcasts-compl INTEGER
    }
    SEQUENCE OF SEQUENCE{
        cell-id Cell,
        cause Failure-Reason,
        diagnostic Diagnostic-Info OPTIONAL
    }OPTIONAL
}
```

```
Set-DRX-Response ::= SEQUENCE {
    cell-list Cell-List OPTIONAL,
    SEQUENCE OF SEQUENCE{
        cell-id Cell,
        cause Failure-Reason} OPTIONAL
}
```

-- CBR operation errors listed below

```
Parameter-not-recognized ::=
    ABSTRACT-ERROR
    PARAMETER Diagnostic-Info OPTIONAL
```

```
Parameter-value-invalid ::=
    ABSTRACT-ERROR
    PARAMETER Diagnostic-Info OPTIONAL
```

```
Valid-CBS-message-not-identified ::=
    ABSTRACT-ERROR
    PARAMETER Diagnostic-Info OPTIONAL
```

```
Cell-Identity-not-valid ::=
    ABSTRACT-ERROR
    PARAMETER Diagnostic-Info OPTIONAL
```

```
Unrecognized-message ::=
    ABSTRACT-ERROR
    PARAMETER Diagnostic-Info OPTIONAL
```

```
Missing-mandatory-element ::=
    ABSTRACT-ERROR
    PARAMETER Diagnostic-Info OPTIONAL
```

```
BSS-capacity-exceeded ::=
    ABSTRACT-ERROR
    PARAMETER Diagnostic-Info OPTIONAL
```

```
Cell-memory-exceeded ::=
    ABSTRACT-ERROR
    PARAMETER Diagnostic-Info OPTIONAL
```

```
BSS-memory-exceeded ::=
    ABSTRACT-ERROR
    PARAMETER Diagnostic-Info OPTIONAL
```

```
Unspecified-error ::=
    ABSTRACT-ERROR
    PARAMETER Diagnostic-Info OPTIONAL
```

incompatible-DRX-parameter ::=
 ABSTRACT-ERROR
 PARAMETER Diagnostic-Info OPTIONAL

Serial-Number ::= INTEGER (0 .. 65535)

Page-Inf ::= SEQUENCE{
 message-info-useful-octets INTEGER (0..82),
 message-info-page Message-Info-Page
}

Message-Info-Page ::= OCTET STRING (SIZE(82))

Cell-Id-Disc ::= OCTET-STRING (SIZE(1))
-- values from the following table

 lacAndCi ::= '00000001' --2 Octet lac followed by 2 Octet Cell Id
 ciOnly ::= '00000010' --Cell Id only

Cell-Id ::= OCTET-STRING (SIZE(4))

--Note: If Cell-Id-Disc equals ciOnly then only the last 2 octets of Cell-ID are to be considered --and
the first 2 octets are filler octets

Cell ::= SEQUENCE OF {
 disc Cell-Id-Disc,
 id Cell-Id
}

Cell-List ::= SEQUENCE {
 length INTEGER,
 disc Cell-Id-Disc,
 list SEQUENCE OF Cell-Id
}

Diagnostic-Info ::= OCTET STRING (SIZE (1 .. 20))

-- Definition of Cell Broadcast Relay Service address

CBS-Address ::= [APPLICATION 0] SEQUENCE {
 address-type INTEGER { unknown-type (0),
 international-number (1),
 national-number (2),
 network-specific-number (3),
 short-number (4) },
 numbering-plan INTEGER { unknown-numbering (0),
 iSDN-numbering (1),
 data-network-numbering (3),
 telex-numbering (4),
 national-numbering (8),
 private-numbering (9)}
 address-value CHOICE { octet-format
 SemiOctetString
 --other formats are for further study}
}

SemiOctetString ::= OCTET STRING (SIZE(1..10))
-- each octet contains two binary coded decimal digits

END

3rd module of 3

RelayProtocol

```
RelayProtocol {  
    ccitt identified-organization (4) etsi (0) mobile-domain(0)  
    gsm-messaging (4) gsm-sms4 (13) relayProtocol(1) }
```

DEFINITIONS

IMPLICIT TAGS

::=

BEGIN

IMPORTS

-- application service elements and application contexts

aCSE, APPLICATION-SERVICE-ELEMENT, APPLICATION-CONTEXT

```
FROM Remote-Operations-Notation-extension {  
    joint-iso-ccitt remote-operations(4) notation-extension(2) }
```

rOSE

```
FROM Remote-Operations-APDUs {  
    joint-iso-ccitt remote-operations(4) apdus(1) }
```

-- object identifiers

id-cb-ac-so, id-cb-CBRSE, id-cb-as-CBRSE,

```
FROM CBS-UsefulDefinitions{  
    ccitt identified-organization (4) etsi (0) mobile-domain(0)  
    gsm-messaging(4) gsm-sms4 (13) usefulDefinitions(0) } ;
```

aS-ACSE OBJECT IDENTIFIER ::=

```
{ joint-iso-ccitt association-control (2) abstractSyntax(1) apdus(0) version(1) }
```

-- abstract service parameters

CBR-Bind, CBR-Unbind, CBR-Write-Replace, CBR-Kill-Message,

CBR-Kill-Message, CBR-Status-CBCH-Query,

CBR-Status-Message-Query, CBR-Reset, CBR-Restart,

CBR-Failure, CBR-Set-DRX,

Parameter-not-recognized, Parameter-value-invalid,

Valid-CBS-message-not-identified, Cell-identity-not-valid,

Unrecognized-message,

Missing-mandatory-element, BSS-capacity-exceeded,

Cell-memory-exceeded, BSS-memory-exceeded, Unspecified-error, incompatible-DRX-parameter

```
FROM RelayAbstractService{  
    ccitt identified-organization (4) etsi (0) mobile-domain(0)  
    gsm-messaging(4) gsm-sms4(13) relayAbstractService(2) } ;
```

-- Application contexts

-- Only one application contexts is specified: the CBC is exclusively responsible for the BIND and

-- UNBIND operations.

cBC-BINDs-and-UNBINDs

```

APPLICATION-CONTEXT
APPLICATION-SERVICE-ELEMENTS { aCSE }
BIND CBR-Bind
UNBIND CBR-Unbind
REMOTE OPERATIONS { rOSE }
INITIATOR CONSUMER OF { cBRSE }
ABSTRACT SYNTAXES { id-cb-as-CBRSE , aS-ACSE }
::= id-cb-ac-so

```

-- Application service elements

```

cBRSE APPLICATION-SERVICE-ELEMENT
      CONSUMER INVOKES { CBR-Restart
                        CBR-Failure
                        }
      SUPPLIER INVOKES { CBR-Write-Replace
                        CBR-Kill-Message
                        CBR-Status-CBCH-Query
                        CBR-Status-Message-Query
                        CBR-Reset
                        CBR-Set-DRX
                        }
      ::= id-cb-SMRSE

```

-- Remote operations

```

cbr-write-replace CBR-Write-Replace
                  ::= 1
-- Note: localValue - words are omitted, since they are
-- typically not used, and likely to be removed from
-- the OPERATION and ERROR macros in ROSE.

cbr-kill-message CBR-Kill-Message
                 ::= 2

cbr-status-CBCH-query CBR-Status-CBCH-Query
                      ::= 3

cbr-status-message-query CBR-Status-Message-Query
                          ::= 4

cbr-restart CBR-Restart
            ::= 5

cbr-reset CBR-Reset
          ::= 6

cbr-failure CBR-Failure
            ::= 7

cbr-set-DRX CBR-Set-DRX
            ::= 8

-- Remote errors, the localValues are provisional

parameter-not-recognized Parameter-not-recognized
                          ::= 1

parameter-value-invalid Parameter-value-invalid
                        ::= 3

```

valid-CBS-message-not-identified	Valid-CBS-message-not-identified ::= 4
cell-identity-not-valid	Cell-identity-not-valid ::= 5
unrecognized-message	Unrecognized-message ::= 6
missing-mandatory-element	Missing-mandatory-element ::= 7
bss-capacity-exceeded	BSS-capacity-exceeded ::= 8
cell-memory-exceeded	Cell-memory-exceeded ::= 9
bss-memory-exceeded	BSS-memory-exceeded ::= 10
unspecified-error	Unspecified-error ::= 11
incompatible-DRX-parameter	Incompatible-DRX-Parameter ::= 12

END

3.3 Application rules

The following application rules specify the invocation of different operations on the association. Two alternative sets of application rules are given in 3.3.1 (for semi-permanent connections) and in 3.3.2 (for transient connections); additional sets are possible.

3.3.1 Application rule set 1 Semi-permanent symmetric connection

This set of application rules is to be used in situations where the connection (on all the protocol layers) between the CBC and the BSC is maintained for ever.

Within the CBR-BIND service, all operations are allowed on the association; semi-permanent connection is accepted (by not forcing the connection transient). This is negotiated within the CBR-BIND service as follows:

name of parameter value in request and report

operations	{cBR-From-CBC-Write-Replace, cBR-From-CBC-Kill-Message, cBR-From-CBC-Status-CBCH-Query, cBR-From-CBC-Status-Message-Query, cBR-From-BSC-Restart, cBR-From-BSC-Reset, cBR-From-BSC-Failure cBR-From-CBC-Set-DRX }
------------	--

transient	FALSE
-----------	-------

The CBC invokes cBR-From-CBC-Write-Replace, cBR-From-CBC-Kill-Message, cBR-From-CBC-Status-CBCH-Query, cBR-From-CBC-Status-Message-Query, cBR-From-CBC-Set-DRX operations as needed. The BSC invokes CBR-BSC-RESTART.

The CBR-UNBIND operation is not normally invoked on the association.

3.3.2 Application rule set 2 Transient asymmetric connection

This set of application rules is to be used e.g. in situations where a CBC has connections with many BSCs, and there is a switched data network connecting them. A data network connection (and the higher layer connections on top of it) is maintained for the duration of the relay or alert operations only.

Within the CBR-BIND service, only one type of operation is negotiated for use on the association. The operation of that type must be invoked by the CBC or by the BSC in exceptional circumstances (e.g. in order to invoke CBR-BSC-RESTART). The BSC or CBC accepts the one type of operation and forces the association transient.

The following is an example of a negotiation procedure within the CBR-BIND service, where the CBR-Write-Replace operation is initiated by the CBC.

name of parameter value

iniType	cell-Broadcast-Service-Centre
respType	public-Land-Mobile-Network
operations	{ cBR-From-CBC-Write-Replace }
transient	TRUE

The association for cBR-From-CBC-Kill-Message, cBR-From-CBC-Status-CBCH-Query or cBR-From-CBC-Status-Message-Query, cBR-From-CBC-Set-DRX are negotiated according to the same principle, the CBC always being the initiator of the CBR-BIND.

The association may be used for invoking operations of the negotiated type(s) as long as there are such operations to be invoked (in other words, until all commands have been relayed).

4 An SS7 Protocol Stack For Interconnecting CBC And BSC

Concepts described in Q.1400 (see CCITT Study Group XI - Report R219) are used. These concepts enable, with minor modifications, the protocol specified in Section 3 of GSM 03.49 to be supported via an SS7 protocol stack.

Q.1400 specifies the use of OSI concepts via SS7 for the development of signalling and operations & management protocols. The protocol specified in Section 3 of this report can be carried via an SS7 protocol stack consisting of TCAP, SCCP and MTP (see Q.700 series) with minor adaptations:

- ROSE operation classes 2 and 5 are replaced by TCAP operation classes 1 and 4 respectively.
- TCAP provides a connectionless service. The services provided by CBRSE-BIND, CBR-UNBIND, CBR-Bind-confirm and CBR-Bind-failure are therefore not required and Sections 3.3 is not applicable.

PLMN networks may provide interworking between either of the protocols specified by Sections 2 or 3 and the SS7 protocol stack for the purpose of fulfilling the service requirements of the primitives specified for the CBC - BSC interface in GSM 03.41.

History

Document history	
October 1995	Creation of Version 5.0.0 (Version 4.6.0 + AR001)
January 1996	Publication of Version 5.0.0
May 1996	Publication of Version 5.1.0