3GPP TR 30.802 V4.0.0 (2000-11)

Technical Report

3rd Generation Partnership Project;
Technical Specification Group Core Network;
Guidelines for the modification of the Mobile Application Part (MAP) in Phase 2+
(Release 4)

GLOBAL SYSTEM FOR
MOBILE COMMUNICATIONS

Keywords

3GPP, GSM, MAP, Guidelines

3GPP

Postal address

3GPP support office address

650 Route des Lucioles - Sophia Antipolis
Valbonne - FRANCE
Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Internet

http://www.3gpp.org

Contents

Foreword

This Technical Report has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

x the first digit:

1 presented to TSG for information;

2 presented to TSG for approval;

3 or greater indicates TSG approved document under change control.

y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.

z the third digit is incremented when editorial only changes have been incorporated in the document.

1 Scope

After completion of MAP version 2 the introduction of new or modification of existing features and services in phase 2+ and UMTS requires changes to the Mobile Application Part (MAP) specification (see [1]). This handbook gives some guidelines on how to introduce changes into the MAP. Protocol as well as application aspects.

1.2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies.

[1] 3GPP TS 29.002 "Mobile Application Part (MAP) specification".

[2] ETSI prETR 060 "Signalling Protocols and Switching (SPS); Guidelines for using Abstract Syntax Notation One (ASN.1) in telecommunication application protocols".

[3] ITU-T Recommendations Q.771 to Q.775 (Blue Book 1988/White Book 1993) "Specification of signalling system no. 7, transaction capabilities (TC)".

[4] Addendum to Recommendation ITU-T Recommendations Q.1400 (1994) "Architecture framework for the development of signalling and OAM protocols using OSI concepts. Add section 12.5".

[5] CCITT Recommendation X.208 (Blue Book 1988) "Specification of abstract syntax notation one (ASN.1)".

[6] CCITT Recommendation X.209 (Blue Book 1988) "Specification of basic encoding rules for ASN.1".

[7] ITU-T Recommendations X.680 to X.683 (1993) "Specification of abstract syntax notation one (ASN.1)" including Amendment 1 "Rules of extensibility" (1994).

[8] ITU-T Recommendations X.690 to X.691 (1993/94) "Specification of ASN.1 encoding rules".

1.3 Abbreviations

MAP	Mobile Application Part
AC	Application Context
SMS	Short Message Service
USSD	Unstructured Supplementary Service Data
ASN.1	Abstract Syntax Notation One
TCAP	Transaction Capability Application Part
SCCP	Signalling Connection Control Part
SS No.	Signalling System No. 7
ISUP	ISDN User Part
ISDN	Integrated Services Digital Network
ITU	International Telecommunication Union
CCITT	International Telegraph and Telephone Consultative Committee
ETSI	European Telecommunication Standard Institute
ETS	European Telecommunication Standard
ETR	European Technical Report
TC-TR	Technical Committee-Technical Report

2 MAP guidelines

2.1 General

Introduction of new or enhanced services and features should always be based on the requirements defined by stable stage 1 and stage 2 definitions. By this means the discussion of requirements will be separated from the development of protocols. However some feedback from the studies of protocol solutions may result in changes or additions to the stage 2 or even the stage 1 definitions.

Services and features generally comprise a number of functions requiring exchange of information between entities. Functions are defined by text, tables, SDL diagrams and information flows in the stage 2 definition, which does not necessarily define which kind of protocol is to be used.

For each function or protocol change compatibility needs careful consideration to ensure interworking with entities only supporting previous versions of that feature or not supporting the feature at all.

The decision on whether the MAP protocol is to be used for specific messages or parameters shall be based on one hand on the capabilities of the MAP protocol and on the other hand on the existing environment (interworking and compatibility).

The MAP protocol is normally used to request and control information in a remote data base (HLR, VLR, EIR, AuC). No traffic channel is associated in this case. However in some cases MAP signalling is also used in parallel with connection oriented signalling associated with a traffic channel (e.g. ISUP) to offer capabilities in addition to the connection oriented protocol (e.g. handover). MAP may also transport user information (e.g. SMS, USSD).

2.2 General aspects on MAP modifications

While MAP phase 2 contains text which is applicable to phase 1 and phase 2, the later versions of the MAP specification shall only include material which is applicable for the latest AC versions. As an exception to this pronciple, definitions of operation packages and application contexts for earlier versions shall be kept in section 17 of MAP.

Based on the stage 2 definition, MAP modifications should be done step by step to ensure a systematic approach. The following sections describe typical steps of analysis and implementation of changes in the MAP.

2.2.1 Analysis of the requirements

The requirements from the stage 2 definition need to be checked together with relevant stage 3 definitions of other interfaces (e.g. Radio- and ISUP-interface).

Resulting protocol changes may be:

- A new or modified parameter in a message;
- A new message;
- A new interface;
- A new or modified message sequence;
- A new or modified Application Context (AC).

Except for the Application Contexts it is easy to derive from the requirements a list of all protocol elements which have to be added or changed in the MAP. Whether a new Application Context is required needs further elaboration (see following sections).

2.2.2 Study of compatibility aspects

When an existing feature is changed or enhanced attention to the compatibility aspects should always be given. This refers not only to cases resulting in a protocol change but also where messages and parameters remain unchanged while the functionality associated with a feature is modified.

Normally a feature consists of a number of functions which are defined by the Stage 2 definition. To analyse the compatibility aspects it is necessary to look at these functions one by one.

Modification of features and services will result in changes of existing functions or introduction of new functions. Modifications can be classified as follows:

- Essential modifications always need support by the MAP entities involved. This implies that a MAP entity which receives a request for a modified (and therefore possibly unsupported) function must be informed of this modification in such a way that it can reject this request.
- Non-essential modifications do not always need support by the MAP entities involved. In this case the MAP entity which receives the request should not be forced to reject a modified (and therefore possibly unsupported) function. It should react as if the unmodified function was requested.

For some modifications the MAP protocol allows the specification of a non-essential modification. These are defined as follows:

- Association of a new or modified functionality with an existing message or parameter. Note that there should not be any change in the message flow.
- Modification of a function requiring addition of a parameter in an existing message.
- Changes on an internal interface (e.g. the MSC/VLR interface).

Other changes than the above can only be treated as essential modifications.

For compatibility reasons basic protocol mechanisms should not be changed (e.g. Service Provider, protocol error handling or version negotiation).

2.3 Introduction of changes in MAP

After all requirements have been classified changes can be introduced in the MAP document as described below. Normally only the following MAP sections are affected:

- MAP section 7.6: Definition of parameters
  Textual description of parameters.

- MAP sections 8 to 13: MAP Service User specific services
  Description of the MAP services offered by the MAP Service Provider to specific MAP Service Users (see MAP section 7.2).

- MAP section 16.2.2.4: Mapping between operations and services
  Definition of mapping of services to operations and vice versa.

- MAP section 17.2: Operation packages
  Definition of groups of operations for use within Application Contexts.

- MAP section 17.3: Application contexts
  Definition of which operations are available to the initiator of a dialogue and which are available to the responder when using the application context.

- MAP section 17.5: MAP operation and error codes
  ASN.1 definition of operation and error values.

- MAP section 17.6: MAP operation and error types
  ASN.1 definition of operation and error types.

- MAP section 17.7: MAP constants and data types
  ASN.1 definition of constants and data types.

- MAP sections 19 to 25: Procedure descriptions
  SDL description of the Service User procedures.

- MAP Annexes A and B: ASN.1 cross reference and expanded source
  cross reference and expanded source generated out of the ASN.1 sections.

2.3.1 Relationship between MAP service and MAP operation

As described above the additional protocol elements required can be directly derived from the Stage 2 definition.

A new MAP message requires the specification of a new MAP service (MAP sections 8 to 13). New MAP parameters can be defined in MAP section 7.6. They can be included directly or indirectly in the MAP Services (for details see below).

A MAP service is used by the Service User to instruct the Service Provider to send a message on an interface and to be able to receive a message via the Service Provider.

The following cases may exist for a MAP specific Service:

- The service is used for only one interface.

- The service is used for more than one interface.

In the second case a specific parameter included in the service may be used for one of these interfaces but must not be used on another. In this case the parameter should be defined as conditional in the service description so that it can be omitted on the interface where it is not required.

The interface between MSC and VLR is specified as an internal interface. It is only used as a descriptive interface which is defined by the service description (see MAP sections 8 to 13).

Only messages and parameters used on external interfaces are specified by operations in the ASN.1 definition (see MAP section 17). For external interfaces normally all parameters in the service description shall also be included in the ASN.1 definition of the corresponding operation.

Except for the Invoke Id and the Provider Error the following mapping between parameters in the service and the ASN.1 definition applies:

- Parameters defined in the request and indication columns map to and from parameters under the ASN.1 operation ARGUMENT.
- Parameters defined in the response and confirmation columns map to and from parameters under the ASN.1 operation RESULT.
- User Error Parameters defined in the response and confirmation columns map to and from the errors and parameters under ASN.1 operation ERRORS.

A parameter which is shown as conditional or optional in the service description maps to and from an OPTIONAL parameter in the ASN.1 definition.

A parameter which is shown as mandatory in the service description maps to and from a mandatory parameter in the ASN.1 definition.

Note that DEFAULT parameters are not used in MAP for the time being. They are shown as conditional or optional parameters in the service description.

However, when a MAP service is used for an external and an internal interface it may occur that a parameter is defined as conditional in the service description but is not included in the ASN.1 definition. This is because the parameter is used only internally and not externally.

2.3.2 Definition of a new parameter

A new parameter may be required for new or existing operations and errors. The parameter should be defined in MAP section 7.6. The text should unambiguously define the relationship of the parameter in the stage 2 and the protocol definition (Stage 3).

2.3.2.1 A new parameter in the service description

The use of a new parameter with specific services is given in the service descriptions in MAP sections 8 to 13. Operation and Error parameters need to be distinguished.

2.3.2.1.1 A new operation parameter

Not all parameters of a MAP service are explicitly shown in the service description table but may be included in a constructed parameter (hierarchical structure). A constructed parameter is shown in the service description if a common use of all sub-parameters can be assumed.

In the following cases the parameter should be explicitly listed in the service description:

- If the use of an operation is modified substantially by a new parameter. This may also need to be reflected under 'Definition' of the Service description.
- If a new parameter has no common use with other parameters.

Otherwise only the constructed parameter description in MAP section 7.6 will reference the sub parameter.

- If required, a new operation parameter name should be included in the table just before user error and provider error.
  - If the parameter is included in the invoke component, the Request and Indication column shall be marked with U, C or M;
  - If the parameter is included in the result component, the Response and Confirmation column shall be marked with U, C or M. The use of this marking is defined in MAP section 7.3. Note that although this definition is applied to the MAP common services, it is valid generally for all MAP services.
- 'U' is used if the support of the parameter is optional. Normally 'U' shall be used only in the request and response column. In this case the right neighbour column shall include 'C(=)' (see section 2.3.8.2). In special cases the indication and confirmation columns may be marked with 'U'.

- 'M' is used if the parameter is mandatory.
- 'C' is used if the parameter shall be included under specific conditions (application defined).
- '(=)' means that the parameter takes the same value as in the left neighbour column.
- 'O' shall not be used because it is not available for the MAP User.

Note that a parameter other than 'User Error' can be added only if the operation is of a class which can return a result (see section 2.3.4.1).

2.3.2.1.2 A new error parameter

Error parameters are not explicitly shown under the service primitives. If required some references can be given with the description of the relevant user error description.

2.3.2.2 A new parameter in the ASN.1 protocol definition

The coding of parameters is defined in the ASN.1 protocol in MAP section 17.7. Error parameters are defined in MAP subsection 17.7.7. Commonly used parameters are specified in MAP subsection 17.7.8. For all other parameters the subsection can be chosen corresponding to MAP section 7.6.

- The parameter name shall in principle be the same as in MAP section 7.6. In addition the syntax rules given in section 2.3.8.1 shall apply.
- If possible, the coding of a parameter shall be defined separately from the place where it is actually used. This means that an ASN.1 type is defined and identified by a name and this identifier is assigned to the parameter name where it is actually used.
- The new parameter must not be used in the old version of the protocol. Care must be taken to allow the new parameter to be unambigiously identified even against parameters which are only used in the old version of the protocol.

e.g.: **old version**

```
LocationInfo ::= CHOICE {
    roamingNumber                      [0] ISDN-AddressString,
    -- roamingNumber must not be used in version greater 1
    msc-Number                         [1] ISDN-AddressString}
```

new version

```
LocationInfo ::= CHOICE {
    msc-Number                         [1] ISDN-AddressString,
    newAlternative                     [2] ISDN-AddressString}
```

2.3.3 Definition of a new error

New user errors may be required by new or existing operations. User errors are defined in MAP section 7.6.1.4. The text should unambiguously define the relation of the error messages in the stage 2 and the return errors in the protocol definition (Stage 3).

Note that the introduction of new provider errors is not described here.

2.3.3.1 A new error in the service description

The use of a new error and its parameters with specific operations is described in the service descriptions in MAP sections 8 to 13. The name of the error shall be unique over the entire MAP protocol. In general the name expresses the problem to be indicated to the remote node e.g. 'Illegal Equipment'.

If required parameters shall be defined as described in the sections above.

Note that a user error may be added only if the operation is of a class which can return an error (see section 2.3.4.1)

2.3.3.2 A new error in the ASN.1 definition

In general the coding of errors shall be defined in the ASN.1 protocol in MAP section 17.6.6.

- The ASN.1 name is derived from the error name given in MAP section 7.6.1.4. The name is changed according to the syntax rules given in section 2.3.8.1.
- In MAP section 17.5 a local value must be assigned to the error under the appropriate headline. A value close to those already used shall be chosen. Note that the SS-Protocol defined in GSM 04.80 allocates error values beginning at the upper limit of 127 (decreasing values). To simplify interworking a specific value should not be assigned twice.
    - The new error must not be used in the old version of the protocol. Care must be taken to allow the new error being unambigiously identified even against errors which are only used in the old version of the protocol., i.e. the local value of the new error must be different from any local value of any error in any previous MAP version.

If required parameters shall be defined as described in the sections above.

In order to allow for future extensibility, extensible parameters (SEQUENCE with extension marker) shall be defined for all new errors.

2.3.4 Definition of a new operation

2.3.4.1 A new operation in the service description

The function of a new operation and its parameters and errors are defined in the service descriptions in MAP sections 8 to 13. A new operation requires a new subsection under the section to which the function of the operation relates. The text should unambiguously define the relationship of the message in the stage 2 and the MAP service (operation) in the protocol definition (Stage 3). In addition the entities using this service need to be defined, i.e. the applicable interfaces.

The name of the service shall be unique over the entire MAP protocol. In general the name expresses a command to the remote node, e.g. 'Update Location'.

The following relationship between the class of an operation and the table in the service description exists (see also section 2.3.1.2.1 and MAP section 17.1.2):

- Class 1 (result and error reported):
    - The table includes Request, Indication, Response and Confirmation columns.
    - The Invoke Id is a mandatory parameter (marked with 'M') in all the above columns.
    - The User Error is a conditional parameter (marked with 'C') in the Response and Confirmation columns.
    - The Provider Error is an optional parameter (marked with 'O') in the Confirmation column.
- Class 2 (only error reported)
    - The table includes Request, Indication, Response and Confirmation columns.
    - The Invoke Id is a mandatory parameter (marked with 'M') in all the above columns.
    - The User Error is a conditional parameter (marked with 'C') in the Response and Confirmation columns.
    - The Provider Error is an optional parameter (marked with 'O') in the Confirmation column.
- Class 3 (only result reported)
    - The table includes Request, Indication, Response and Confirmation columns.
    - The Invoke Id is a mandatory parameter (marked with 'M') in all the above columns.

  - There is no User Error in the Response and Confirmation columns.
  - The Provider Error is an optional parameter (marked with 'O') in the Confirmation column.
- Class 4 (neither result nor error reported)
  - The table includes only Request and Indication columns.
  - The Invoke Id is a mandatory parameter (marked with 'M') in both the above columns.

The mapping between service and operation in the table in MAP section 16.2.2.4 needs to be updated. The ASN.1 operation name is defined in MAP section 17 (see section 2.3.4.2).

If required new errors and parameters shall be defined as described in the sections above.

In order to allow for future extensiblity new operations should be defined as class 1 operations except for the case where it is foreseen to invoke the new operation in a TC-END message.

2.3.4.2 A new operation in the ASN.1 protocol definition

If a new operation is required the coding shall be defined in the ASN.1 protocol in MAP section 17.6. The subsection to be used shall be chosen corresponding to MAP section 8 to 13.

- In MAP section 17.5 a local value must be assigned to the operation under the appropriate headline. A value close to those already used shall be chosen. Note that the SS-Protocol defined in GSM 04.80 allocates operation values beginning at the upper limit of 127 (decreasing values). To simplify interworking a specific value should not be assigned twice.
- The ASN.1 name is derived from the MAP service name. The word 'MAP' is removed and the syntax is then changed according the description in section 2.3.8.1 (see below).
- The new operation must not be used in the old version of the protocol. Care must be taken to allow the new operation to be unambigiously identified even against operations which are only used in the old version of the protocol, i.e. the local value of the new operation must be different from the local value of any operation in any previous MAP version.
- A timer value to supervise the response to the operation shall be defined in the ASN.1 comment. One of the values defined in MAP section 17.1.2 may be chosen. Note that for class 4 operations a timer is also required to supervise possible rejection of the operation.
- The ASN.1 keyword ARGUMENT is included only if parameters are defined for the invoke component. If there is no option for the invoke component to be sent without parameters, the error DataMissing must be specified for the new operation, so that the responding entity can respond correctly to an invoke component with no parameters.
- The ASN.1 keywords RESULT and ERRORS must be included depending on the class of the operation (see above).
  - If parameters are defined after the keyword ARGUMENT or RESULT but the invoke or result component may be sent without parameters this shall be indicated by the ASN.1 comment '-- optional'. The keyword OPTIONAL must not be used in this case, e.g.

    NewOperation ::= OPERATION --Timer m

    RESULT

    operationRes OperationRes

    -- *optional*

    newOperation NewOperation ::= localValue 10

If required errors and parameters shall be defined as described in the sections above.

2.3.5 Addition of parameters to an existing operation or error

2.3.5.1 Addition of parameters in the service description

Before a parameter is added to an existing service description the class of the operation needs to be checked (see section 2.3.4.1).

- parameters can be added to existing services and their errors as described in section 2.3.2. However it is not allowed to add a parameter (other than 'User Error') for the result or confirm of a service which corresponds to an operation of a class which does not return a result. Such a change requires a new operation.

2.3.5.2 Addition of parameters in the ASN.1 definition

Addition of new parameters to existing operations or errors shall follow the extensibility rules defined in [4] and [7]. Additional information is given in [2]. The following text explains the most important items for MAP from the referenced documents and explains some further rules.

In the following cases a parameter may be added to an existing operation argument, result or error:

- It may be added as a new component in any SEQUENCE type.
- It may be added as a new alternative in any CHOICE type.
- It may be added as a new value in any ENUMERATED type.
- It may be added as a new assignment in any BIT STRING type.
- It may be added to an operation argument, result or error if this component was previously empty.

Whenever a parameter is added as a new alternative in a CHOICE type or to an operation argument, result or error which was previously empty, it should be embedded in a sequence parameter to allow future extension additions at this level. A new sequence parameter is not necessary if there is already an extensible sequence available at this level.

e.g. **OperationNameArg** ::= SEQUENCE {

firstParameter FirstParameterType,

...}

2.3.6 Addition of errors to operations

2.3.6.1 Addition of errors in the service description

- User errors can simply be added to the error list in the service description.

Note that this is only allowed when the user error is already marked as present in the response column of the table i.e. the TCAP class of the operation must not be changed. Otherwise a new operation may be required (see section 2.3.4.1).

2.3.6.2 Addition of errors in the ASN.1 definition

- Errors can simply be added to the ASN.1 definition of the operation in MAP section 14.6.

2.3.7 Deletion of parameters, errors and operations

Whenever an existing protocol element is no longer used in the new version of the protocol it can simply be deleted.

2.3.8 ASN.1 guide-lines and compatibility

2.3.8.1 ASN.1 names

The following syntax rules for MAP ASN.1 names shall apply (see also MAP section 17.1.5):

- If the parameter name consists of more than one word the character following a space shall be converted to a capital letter and afterwards spaces are removed.
- The name of an ASN.1 derived data type starts with a capital letter; the name of a parameter or a value starts with a lower case letter.
- Abbreviations or acronyms are normally in capital letters and are separated by '-' from the following word. Abbreviations or acronyms at the beginning of ASN.1 names are always in lower case letters and are separated by '-' from the following word.
- The length of a name should not exceed 25 characters;

Note that the above rules are not valid in MAP sections 17.2 and 17.3.

2.3.8.2 Essential and non-essential modifications and the use of Application Contexts

As shown above, new functionality, possibly together with the addition of new parameters to existing operations, may be specified as an essential or non-essential modification.

Non-essential modifications:

Support of this type of protocol change is optional on the sending side as well as on the responding side. Therefore this kind of parameter is marked as optional ('U') in the request and/or response columns.

> This class of modifications will use an existing AC and AC version and will therefore not have an impact on the version negotiation. Only Extension Additions after an Extension Marker may be added in this case (see MAP section 17.1.4).

This kind of modification may be added as an optional Extension Addition to an already published protocol version without the necessity to upgrade the AC version. This will prevent unnecessary version negotiation. However non-essential information may be discarded by a receiving node which does not support this protocol extension without any notice to the sender, although it is part of the actual AC version.

There are only limited possibilities for the sending entity to be aware of whether the receiving entity supports a non-essential modification and to react to that information.

If entity X sends a request message with a non-essential modification to entity Y, then entity Y can inform entity X that it supports the new function which uses the non-essential modification only by including a non-essential modification in the response message. Entity X can deduce from the presence or absence of the non-essential modification in the response message whether or not entity Y supports the new function; however the possibilities for entity X to react to this information are very limited if the dialogue between entities X and Y does not continue after the response from entity Y. The dialogue structure is part of the application context definition, so if a change to the dialogue structure is needed it will be necessary to raise the application context version. For most dialogue types, therefore, it will be necessary to raise the application context version if entity X needs to be aware of, and react to, whether entity Y supports the function which uses a protocol modification.

The possibilities for the sender of a reponse including a non-essential modification to determine whether the receiver supports the non-essential modification are also limited. The most practical solution is for the sender of the request (which will be the receiver of the response) to include in the request an indication of the capabilities (implied by the possible non-essential modifications which can be included in the response) which it supports. If a new version of the protocol specification includes several non-essential modifications in the response, a capability indication would be required for each function which uses a non-essential modification. The only alternative exists if the dialogue continues after the response message containing the non-essential modification, which allows a process of negotiation between the two entities concerned.

Note that non-essential extension of a sequence is allowed only at the end of already defined extension additions i.e. after an extension marker and just before '}' (see 2.3.8.4.1). If no extension marker is available the protocol only allows an essential modification (see below).

Essential modifications:

Support of this type of protocol change is mandatory at least at the receiving side. Therefore parameters of this kind are either marked as conditional ('C') or mandatory ('M') in the indication and/or in the confirmation column.

This class of modifications requires application context (AC) version negotiation, i.e. it needs a new AC or AC version.

- If fallback is required a new version of an existing AC should be chosen.
- If fallback is not required a new AC should be chosen.

New operations and errors may be introduced without any restrictions. New parameters in existing operations or errors may be introduced; the only restriction is that the creation of a superset protocol covering all older versions and the new version of the operation or error must be possible. This can be achieved by tagging the outermost sequence of the argument, result and error parameter of an existing operation in a new version with a new context specific tag.

2.3.8.3 Order of information in the ASN.1 definition

Although the following guide-lines are not based on ASN.1 rules, they will help the human reader to check and maintain the ASN.1 modules defined in MAP:

- In MAP section 17.5 a new operation or error shall be added at the end of the list under the appropriate headline. Operations and errors are not sorted according to their value.
- In MAP sections 17.6.1 to 17.6.5 and 17.6.7 onwards, new operations shall be added at the end of the appropriate ASN.1 module. In the list of errors following the keyword ERRORS the order is defined by the order in MAP section 17.5.
- In MAP section 17.6.6 the order of errors shall be identical to the order in section 17.5
- In MAP section 17.7 a new parameter of an operation should be inserted after already existing parameters of that operation.
- Parameters used by more than one ASN.1 module may be moved to the common data types in MAP section 17.7.8.

2.3.8.4 Future Extensions

In order to cope with future extension requirements the following is recommended:

2.3.8.4.1 ENUMERATED types

If applicable, ENUMERATED types should be enhanced with ellipsis notation. An exception handling has to be defined: Reasonable ranges of unused values should be defined for a specific exception handling.
e.g.:

```
RegionalSubscriptionResponse ::= ENUMERATED {
    msc-AreaRestricted              (0),
    tooManyZoneCodes                (1),
    zoneCodesConflict               (2),
    regionalSubscNotSupported       (3),
    ...}
    -- exception handling:
    -- reception of values in the range 4 - 13 shall be treated like
    -- regionalSubscriptionResponse not present
    -- reception of values in the range 14 - 23 shall be treated like
    -- reception of value 0
    -- reception of values in the range 24 - 33 shall be treated like
    -- reception of value 1
    -- reception of values in the range 34 - 43 shall be treated like
    -- reception of value 2
    -- reception of values in the range 44 - 53 shall be treated like
    -- reception of value 3
    -- other values shall be rejected (unexpected data value)
```

NOTE: If the ENUMERATED type is a component of an extensible SEQUENCE, then the ENUMERATED type does not need to be enhanced with ellipsis notation. The implicit extensibility mechanism may be used as described below.

2.3.8.4.2 CHOICE types

If applicable, CHOICE types should be defined with one extra alternative which is reserved for future extension. An exception handling has to be defined.

e.g.:

old version:

```
SM-RP-OA ::= CHOICE {
    msisdn                          [2] ISDN-AddressString,
    serviceCentreAddressOA          [4] AddressString,
    noSM-RP-OA                      [5] NULL}
```

new version:

```
SM-RP-OA ::= CHOICE {
    msisdn                          [2] ISDN-AddressString,
    serviceCentreAddressOA          [4] AddressString,
    noSM-RP-OA                      [5] NULL,
    extraAlternative                [6] ExtraAlternative}
    -- exception handling: reception of extraAlternative shall be treated like    --
reception of noSM-RP-OA
```

```
ExtraAlternative ::= SEQUENCE {
    extensionContainer             ExtensionContainer,
    ...}
```

NOTE: If the CHOICE type is a component of an extensible SEQUENCE, then the CHOICE type does not need to be defined with an extra alternative. The implicit extensibility mechanism may be used as described below.

2.3.8.4.3 BIT STRING types

If applicable, BIT STRING types should be defined with reasonable SIZE rages. An exception handling should be defined.
e.g.:

```
ODB-GeneralData ::= BIT STRING {
    allOG-CallsBarred  (0),
    internationalOGCallsBarred  (1),
    internationalOGCallsNotToHPLMN-CountryBarred  (2),
    premiumRateInformationOGCallsBarred  (3),
    premiumRateEntertainementOGCallsBarred  (4),
    ss-AccessBarred  (5)} (SIZE (6..32))
    -- exception handling: reception of unknown bit assignments shall be treated
    -- like unsupported ODBs
```

2.3.8.4.4 OCTET STRING types

If applicable OCTET STRING types should be defined with reasonable SIZE ranges. An exception handling has to be defined.

If a received OCTET STRING conforms to the length constraints in the ASN.1 definition, it shall be treated as syntactically correct. If the component is syntactically optional but not required in the context in which the operation is used, the receiving entity shall treat the OCTET STRING as having an unexpected data value, as defined in 3GPP TS 29.002 section 7.6.1.4. If ASN.1 comments are used to define the internal structure of the OCTET STRING and the internal structure of the received OCTET STRING does not conform to the definition in the ASN.1 comment, the receiving entity shall treat the OCTET STRING as having an unexpected data value.

Note that the response by the receiving entity to a component with an unexpected data value depends on the individual application. If the component is part of the ARGUMENT of an operation and the receiving entity is to return an error when it receives a component with an unexpected data value, it shall use the error UnexpectedDataValue. The designer of an application which requires a new operation should therefore specify the error UnexpectedDataValue for the new operation if the ARGUMENT includes a component of type OCTET STRING for which the internal structure is not completely unrestricted and an error is to be returned if the component has an unexpected data value.

2.3.8.4.5 INTEGER types

If applicable, INTEGER types should be defined with reasonable value ranges. An exception handling has to be defined.

2.3.8.4.6 NON-SEQUENTIAL DATA STRUCTURES

Non-Sequential Operation Arguments or Results should be replaced with a SEQUENCE type containing the original parameter as component. Empty Operation Arguments or Results should be replaced with an empty optional SEQUENCE.

e.g.:

```
old version:
```

```
ProvideRoamingNumber ::= OPERATION                                  --Timer m
    ARGUMENT
        provideRoamingNumberArg      ProvideRoamingNumberArg
    RESULT
        roamingNumber                ISDN-AddressString
    ERRORS {
        SystemFailure,
        DataMissing,
        UnexpectedDataValue,
        FacilityNotSupported,
        AbsentSubscriber,
        NoRoamingNumberAvailable}
```

```
new version:
```

```
ProvideRoamingNumber ::= OPERATION                                  --Timer m
    ARGUMENT
        provideRoamingNumberArg      ProvideRoamingNumberArg
    RESULT
        provideRoamingNumberRes      ProvideRoamingNumberRes
    ERRORS {
        SystemFailure,
        DataMissing,
        UnexpectedDataValue,
        FacilityNotSupported,
        AbsentSubscriber,
        NoRoamingNumberAvailable}
```

```
ProvideRoamingNumberRes ::= SEQUENCE {
    roamingNumber                    ISDN-AddressString,
    extensionContainer               ExtensionContainer             OPTIONAL,
    ...}
```

Non-extensible ERROR parameters and ERRORS without parameters shall be enhanced with extensible parameters.

The new extensible parameter shall be OPTIONAL if the old parameter was OPTIONAL or the ERROR was defined without a parameter.

The new extensible parameter shall be defined as a CHOICE type with the alternatives being the old parameter and an extensible SEQUENCE including the old parameter as a component, if the ERROR is defined for a phase 2 or higher operation.

2.3.8.4.7 SEQUENCE types

Every SEQUENCE type should include an extension container which can be used for private extensions, PCS extensions and other non-ETSI extensions.
Every SEQUENCE type should include an extension marker.

e.g.:

```
ProvideRoamingNumberRes ::= SEQUENCE {
    roamingNumber                    ISDN-AddressString,
    extensionContainer               ExtensionContainer            OPTIONAL,
    ...}
```

2.3.8.4.8 Implicit Extensibility

ASN.1 types which are components of an extensible SEQUENCE type can be extended implicitly by adding a new component to the SEQUENCE type after the extension marker. For this type of extension an exception handling does not need to be defined.
e.g:
new version:

```
O-BcsmCamelTDP-Data ::= SEQUENCE {
    o-BcsmTriggerDetectionPoint      O-BcsmTriggerDetectionPoint,
    serviceKey                           ServiceKey,
    gsmSCF-Address                   [0] ISDN-AddressString,
    defaultCallHandling              [1] DefaultCallHandling,
    extensionContainer               ExtensionContainer            OPTIONAL,
    ...}
```

```
DefaultCallHandling ::= ENUMERATED {
    continueCall (0) ,
    releaseCall (1) }
```

extended new version:

```
O-BcsmCamelTDP-Data ::= SEQUENCE {
    o-BcsmTriggerDetectionPoint      O-BcsmTriggerDetectionPoint,
    serviceKey                       ServiceKey,
    gsmSCF-Address                   [0] ISDN-AddressString,
    defaultCallHandling              [1] DefaultCallHandling, -- fallback value
                                     -- to be used if ext-DefaultCallHandling is
                                     -- not supported by the receiving entity
    extensionContainer               ExtensionContainer            OPTIONAL,
    ...,
    ext-DefaultCallHandling          [2] Ext-DefaultCallHandling   OPTIONAL}
```

```
DefaultCallHandling ::= ENUMERATED {
    continueCall (0) ,
    releaseCall (1) }
```

```
Ext-DefaultCallHandling ::= ENUMERATED {
    continueCallWithSpecialTreatment1 (2),
    continueCallWithSpecialTreatment2 (3),
    releaseCallWithSpecialTreatment1  (4),
    releaseCallWithSpecialTreatment2  (5),
    connectToAnnouncement             (6),
    connectToOperator                 (7)}
```

In the example above the new version has an extensible SEQUENCE type (O-BcsmCamelTDP-Data) with one of its components being a non extensible ENUMERATED type (DefaultCallHandling). The extended new version extends the ENUMERATED type by making use of the Implicit Extensibility mechanism: A new ENUMERATED type with new values (Ext-DefaultCallHandling) is defined and added as a new component to the SEQUENCE type after the extension marker. The non-extensible ENUMERATED type (DefaultCallHandling) is not removed; it is used to define the fallback value for cases where the receiving entity does not support the extension.

2.3.8.4.9 Private Extensions

The privateExtensionList which is a component of the extensionContainer defined in every SEQUENCE type shall be used to define private extensions of the protocol. An example is given below:

Example for private extension

In the following example the ASN.1 modules MAP-CH-DataTypes and MAP-ExtensionDataTypes are modified in order to define three private extensions. The modifications are revision marked.

The private extensions are identified by OBJECT IDENTIFIER.

The three private extensions are:

1. The first private extension defined by protocol desiner1 which is a Category and which is identified by the OBJECT IDENTIFIER {? ? ? ? 1 1}.

2. The third private extension defined by protocol desiner1 which is an ISDN-AddressString and which is identified by the OBJECT IDENTIFIER {? ? ? ? 1 3}.

3. The first private extension defined by protocol designer5 which is a TeleserviceList and which is identified by OBJECT IDENTIFIER {? ? ? ? 5 1}.

In order to allow the third private extension defined by protocol designer1 to be an extension to ProvideRoamingNumberRes only, the ExtensionContainer in the definition of ProvideRoamingNumberRes has been replaced by PRN-ResContainer which makes use of the PRN-ResExtensionSet.

In order to allow the first private extension defined by protocol designer1 and the first private extension defined by protocol designer5 to be extensions to SendRoutingInfoArg only, the ExtensionContainer in the definition of SendRoutingInfoArg has been replaced by SRI-ArgContainer which makes use of the SRI-ArgExtensionSet.

```
MAP-CH-DataTypes {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-CH-DataTypes (13) version3 (3)}

DEFINITIONS

IMPLICIT TAGS

::=

BEGIN

EXPORTS
    SendRoutingInfoArg,
    SendRoutingInfoRes,
    ProvideRoamingNumberArg,
    ProvideRoamingNumberRes,
    NumberOfForwarding
;

IMPORTS
    CUG-Interlock,
    ForwardingData
FROM MAP-SS-DataTypes {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-SS-DataTypes (14) version3 (3)}

    ISDN-AddressString,
    ExternalSignalInfo,
    IMSI,
    LMSI
FROM MAP-CommonDataTypes {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-CommonDataTypes (18) version3 (3)}

    ExtensionContainer,
    PRN-ResContainer,
    SRI-ArgContainer
FROM MAP-ExtensionDataTypes {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-ExtensionDataTypes (21) version3 (3)}
;
```

```
CUG-CheckInfo ::= SEQUENCE {
    cug-Interlock                 CUG-Interlock,
    cug-OutgoingAccess            NULL                           OPTIONAL,
    extensionContainer            ExtensionContainer             OPTIONAL,
    ...}
```

```
NumberOfForwarding ::= INTEGER (1..5)
```

```
SendRoutingInfoArg ::= SEQUENCE {
    msisdn                        [0] ISDN-AddressString,
    cug-CheckInfo                 [1] CUG-CheckInfo              OPTIONAL,
    numberOfForwarding            [2] NumberOfForwarding         OPTIONAL,
    networkSignalInfo             [10] ExternalSignalInfo        OPTIONAL,
~~    extensionContainer            [11] ExtensionContainer        OPTIONAL,~~
    sri-ArgExtension              [11] SRI-ArgContainer          OPTIONAL,
    ...}
```

```
SendRoutingInfoRes ::= SEQUENCE {
    imsi                          IMSI,
    routingInfo                   RoutingInfo,
    cug-CheckInfo                 CUG-CheckInfo                  OPTIONAL,
    extensionContainer            [0] ExtensionContainer         OPTIONAL,
    ...}
```

```
RoutingInfo ::= CHOICE {
    roamingNumber                 ISDN-AddressString,
    forwardingData                ForwardingData}
```

```
ProvideRoamingNumberArg ::= SEQUENCE {
    imsi                          [0] IMSI,
    msc-Number                    [1] ISDN-AddressString,
    msisdn                        [2] ISDN-AddressString         OPTIONAL,
    lmsi                          [4] LMSI                       OPTIONAL,
    gsm-BearerCapability          [5] ExternalSignalInfo         OPTIONAL,
    networkSignalInfo             [6] ExternalSignalInfo         OPTIONAL,
    extensionContainer            [7] ExtensionContainer         OPTIONAL,
    ...}
```

```
ProvideRoamingNumberRes ::= SEQUENCE {
    roamingNumber                 ISDN-AddressString,
~~    extensionContainer            ExtensionContainer             OPTIONAL,~~
    prn-ResExtension              PRN-ResContainer               OPTIONAL,
    ...}
```

```
END
MAP-ExtensionDataTypes {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-ExtensionDataTypes (21) version3 (3)}

DEFINITIONS

IMPLICIT TAGS

::=

BEGIN

EXPORTS

    PrivateExtension,
    ExtensionContainer,
    PRN-ResContainer,
    SRI-ArgContainer;
IMPORTS
    ISDN-AddressString,
    TeleserviceList,
    Category
FROM MAP-CommonDataTypes {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-CommonDataTypes (18) version3 (3)}
;

-- IOC for private MAP extensions
```

```
MAP-EXTENSION  ::= CLASS {
    &ExtensionType                     OPTIONAL,
    &extensionId                       OBJECT IDENTIFIER }
```

```
-- data types
```

```
ExtensionContainer ::= SEQUENCE {
    privateExtensionList               [0] PrivateExtensionList       OPTIONAL,
    pcs-Extensions                     [1] PCS-Extensions             OPTIONAL,
    ...}
```

```
PRN-ResContainer ::= SEQUENCE {
    prn-ResExtensionList               [0] PRN-ResExtensionList       OPTIONAL,
    pcs-Extensions                     [1] PCS-Extensions             OPTIONAL,
    ...}
```

```
SRI-ArgContainer ::= SEQUENCE {
    sri-ArgExtensionList               [0] SRI-ArgExtensionList       OPTIONAL,
    pcs-Extensions                     [1] PCS-Extensions             OPTIONAL,
    ...}
```

```
PrivateExtensionList ::= SEQUENCE SIZE (1..maxNumOfPrivateExtensions) OF
                                   PrivateExtension
```

```
PRN-ResExtensionList ::= SEQUENCE SIZE (1..maxNumOfPrivateExtensions) OF
                                   PRN-ResExtension
```

```
SRI-ArgExtensionList ::= SEQUENCE SIZE (1..maxNumOfPrivateExtensions) OF
                                   SRI-ArgExtension
```

```
PrivateExtension ::= SEQUENCE {
    extId                              MAP-EXTENSION.&extensionId
                                       ({ExtensionSet}),
    extType                            MAP-EXTENSION.&ExtensionType
                                       ({ExtensionSet}{@extId})       OPTIONAL
    }
```

```
PRN-ResExtension ::= SEQUENCE {
    extId                              MAP-EXTENSION.&extensionId
                                       ({PRN-ResExtensionSet}),
    extType                            MAP-EXTENSION.&ExtensionType
                                       ({PRN-ResExtensionSet}{@extId})
        OPTIONAL
    }
```

```
SRI-ArgExtension ::= SEQUENCE {
    extId                              MAP-EXTENSION.&extensionId
                                       ({SRI-ArgExtensionSet}),
    extType                            MAP-EXTENSION.&ExtensionType
                                       ({SRI-ArgExtensionSet}{@extId})
        OPTIONAL
    }
```

```
maxNumOfPrivateExtensions  INTEGER ::= 10
```

```
ExtensionSet                           MAP-EXTENSION ::=
        {...
         -- ExtensionSet is the set of all defined private extensions
    }
```

```
PRN-ResExtensionSet                        MAP-EXTENSION ::=
        {thirdDesigner1Extension
        -- PRN-ResExtensionSet is the set of all defined private extensions
        -- for ProvideRoamingNumberRes
    }
```

```
SRI-ArgExtensionSet                        MAP-EXTENSION ::=
        {firstDesigner1Extension |
        firstDesigner5Extension
        -- SRI-ArgExtensionSet is the set of all defined private extensions
        -- for SendRoutingInfoArg
    }
```

```
firstDesigner1Extension                 MAP-EXTENSION::= {
        &ExtensionType                  Category,
        &extensionId                    {? ? ? ? 1 1}
    }
```

```
thirdDesigner1Extension                 MAP-EXTENSION::= {
        &ExtensionType                  ISDN-AddressString,
        &extensionId                    {? ? ? ? 1 3}
    }
```

```
firstDesigner5Extension                 MAP-EXTENSION::= {
        &ExtensionType                  TeleserviceList,
        &extensionId                    {? ? ? ? 5 1}
    }
```

```
PCS-Extensions ::= SEQUENCE {
    ...}
```

```
END
```

2.3.9 Definition of a new Application Context

If a new application context is required, a new operation package defining the group of operations for use within the new application context shall be defined in MAP section 17.2.
In MAP section 17.3.2 the new application context shall be defined using the new operation packages. This definition shall indicate, which operations are available to the initiator of the dialogue and which are available to the responder when using the new application context. An application context name must be assigned to that definition. A value close to those already used shall be chosen.
The new application context has to be added to MAP section 17.3.3.

2.3.10 Definition of a new interface

If a new MAP interface is introduced or a new MAP entity is required this is a major modification of MAP and therefore no general guide-lines can be given here. Such a case needs careful checking of protocol architecture and interworking, as well as addressing and routing, by SS NO.7 experts.

Note also that the introduction of new interfaces which do not use the MAP protocol may have a major impact on the MAP specification.

2.3.11 Update of procedural descriptions

If existing procedural descriptions in MAP sections 19 to 25 are affected by new or modified services and features, the procedural descriptions shall be updated according to the appropriate stage 2 specification. Procedural descriptions which describe phase 1 or phase 2 functionality only and which are not applicable to the latest AC versions shall be deleted.

2.3.12 Error handling

The principles in this section must be followed when a new operation (and hence a new operation procedure) is added or when the procedure for an existing operation is modified.

3GPP TS 29.002 specifies that the parameter of an invoke component is syntactically optional, but semantically mandatory unless the ASN.1 definition shows that the parameter is semantically optional (see section 2.3.4.2). If a responding entity receives an invoke component with no parameters and the ASN.1 definition shows that the ARGUMENT of the operation is semantically mandatory, the responding entity shall return an error component with the user error DataMissing.

If the dialogue structure allows a responding entity to return a result component in a TC-CONTINUE message, the procedure definition shall specify the action to be taken by the requesting entity if a result component which is specified to include a parameter is empty.

If the procedure for a new operation uses a parameter of the operation as the key to access a database and the key value does not correspond to a record in the database in the responding entity, the procedure in the responding entity shall return an appropriate error, which has to be defined for the new operation.

Even if there is no functional requirement for the application at a responding entity to return a user error such as data missing or unexpected data value, if the definition of the operation allows the condition which would trigger the error to be detected then the corresponding error should be defined for the operation. This will allow the O & M subsystems in both peer entities to compile statistics on badly constructed requests. The handling of the conditions which would trigger the error at the responding entity and the handling of the user errors at the requesting entity shall be described either in the procedure in MAP sections 18 to 25 or in the application procedures in the stage 2 definition.

Annex A:
Change History

SMG#	Spec	CR	Phase	Ver	New Vers	Subject
S28	10.02	A001r2	R96	5.0.0	5.1.0	Definition of procedures for MAP error handling
S28	10.02	A002	R97	5.0.0 (note)	6.0.0	Reflection of new MAP chapters in the MAP handbook NOTE: Version 5.1.0 was used as a basis for implementation of this CR.

TSG#	Spec	CR	Phase	Ver	New Vers	Subject
N#04	10.02		R97		6.0.0	Transferred to 3GPP TSG CN responsibility
N#05	10.02	002r1	R97	6.0.0	6.1.0	Use of parameters after the ellipsis marker
N#09	10.02	004r1	Rel4	6.1.0	30.802 4.0.0	The handling of application layer errors in MAP Also converted to 3GPP Rel4 document format (MCC)