3GPP TSG_CN **Tdoc NP-000327**
Plenary Meeting #8, Dusseldorf, Germany
21st – 23rd June 2000.

Source: **TSG_N WG4**

Title: **CR to 3G Work Item "Security" for Release 00**

Agenda item: **6.3.4**

Document for: **APPROVAL**

Introduction:

This document contains **"1"** CR on **Work Item "Security" for Release 00,** that has been agreed by **TSG_N WG4,** and is forwarded to **TSG_N Plenary** meeting #8 for approval.

TDoc	SPEC	CR	REV	PHAS	VERS	SUBJECT	CAT	NEW_VERS
N4-000407	29.002	148	4	R00	3.5.0	Changes to the MAP protocol machine for secure MAP	B	4.0.0s

3GPP TSG CN WG4
Rotenburg, Germany, 22-26 May 2000

Document N4-000407
e.g. for 3GPP use the format TP-99xxx
or for SMG, use the format P-99-xxx

CHANGE REQUEST

Please see embedded help file at the bottom of this page for instructions on how to fill in this form correctly.

29.002	**CR**	**148r4**	Current Version:	3.4.0

GSM (AA.BB) or 3G (AA.BBB) specification number ↑ *↑ CR number as allocated by MCC support team*

For submission to:	CN#08	for approval	X	strategic		(for SMG use only)
list expected approval meeting # here ↑		for information		non-strategic	X	

Form: CR cover sheet, version 2 for 3GPP and SMG *The latest version of this form is available from: ftp://ftp.3gpp.org/Information/CR-Form-v2.doc*

Proposed change affects: *(at least one should be marked with an X)*	(U)SIM		ME		UTRAN / Radio		Core Network	X

Source: N4 **Date:** 13th June, 2000

Subject: Changes to MAP for secure transport of MAP messages

Work item: Security

Category: *(only one category shall be marked with an X)*

F	Correction		**Release:**	Phase 2	
A	Corresponds to a correction in an earlier release			Release 96	
B	Addition of feature	X		Release 97	
C	Functional modification of feature			Release 98	
D	Editorial modification			Release 99	
				Release 00	X

Reason for change: To define the changes to the MAP services, MAP protocol machine and ASN.1 protocol which are necessary to support secure transport of MAP messages

Clauses affected: 5.1.2, 7.3, 7.3.1, 7.3.7 (new), 7.3.8 (new), 7.3.9 (new), 7.3.10 (new), 7.6.1.4, 7.6.12 (new), 15 (replacement clause), 16.2.2.4, 17.1.5, 17.1.6, 17.2.2.8 (new), 17.3.2.8 (new), 17.3.3, 17.4, 17.5, 17.6.6, 17.6.9 (new), 17.7.7, 17.7.14 (new)

Other specs affected:

Other 3G core specifications	X	→ List of CRs:
Other GSM core specifications		→ List of CRs:
MS test specifications		→ List of CRs:
BSS test specifications		→ List of CRs:
O&M specifications		→ List of CRs:

Other comments: The entire text and diagrams of clause 15 & subclause 16.3 have been deleted. To save space, the deleted text and diagrams have not been shown struck out.

5.1.2 Overload control for MAP entities

For all MAP entities, especially the HLR, the following overload control method is applied:

If overload of a MAP entity is detected requests for certain MAP operations (see tables 5.1/1, 5.1/2, 5.1/3 and 5.1/4) may be ignored by the responder. The decision as to which MAP Operations may be ignored is made by the MAP service provider and is based upon the priority of the application context.

Since most of the affected MAP operations are supervised in the originating entity by TC timers (medium) an additional delay effect is achieved for the incoming traffic.

If overload levels are applicable in the Location Registers the MAP operations should be discarded taking into account the priority of their application context (see table 5.1/1 for HLR, table 5.1/2 for MSC/VLR, table 5.1/3 for the SGSN and table 5.1/4 for the SMLC; the lowest priority is discarded first).

The ranking of priorities given in the tables 5.1/1, 5.1/2, 5.1/3 and 5.1/4 is not normative. The tables can only be seen as a proposal which might be changed due to network operator/implementation matters.

If secure transport is used, the encapsulated application context for the requested dialogue determines the priority for discarding the received MAP operation.

****** Next modified section ******

7.3 Common MAP services

All MAP service-users require access to services for performing basic application layer functions:

- for establishing and clearing MAP dialogues between peer MAP service-users;
- for accessing functions supported by layers below the applications layer;
- for reporting abnormal situations;
- for handling of different MAP versions;
- for testing whether or not a persistent MAP dialogue is still active at each side.

For these purposes the following common services are defined:

- MAP-OPEN service;
- MAP-CLOSE service;
- MAP-DELIMITER service;
- MAP-U-ABORT service;
- MAP-P-ABORT service;
- MAP-NOTICE service;
- MAP-SECURE-TRANSPORT-CLASS-1 service;
- MAP-SECURE-TRANSPORT-CLASS-2 service;
- MAP-SECURE-TRANSPORT-CLASS-3 service;
- MAP-SECURE-TRANSPORT-CLASS-4 service;

In defining the service-primitives the following convention is used for categorising parameters:

M the inclusion of the parameter is mandatory. The M category can be used for any primitive type and specifies that the corresponding parameter must be present in the indicated primitive type;

O the inclusion of the parameter is a service-provider option. The O category can be used in indication and confirm type primitives and is used for parameters that may optionally be included by the service-provider;

U the inclusion of the parameter is a service-user option. The U category can be used in request and response type primitives. The inclusion of the corresponding parameter is the choice of the service-user;

C the inclusion of the parameter is conditional. The C category can be used for the following purposes:

- to indicate that if the parameter is received from another entity it must be included for the service being considered;
- to indicate that the service user must decide whether to include the parameter, based on the context on which the service is used;
- to indicate that one of a number of mutually exclusive parameters must be included (e.g. parameters indicating a positive result versus parameters indicating a negative result);
- to indicate that a service user optional parameter (marked "U") or a conditional parameter (marked "C") presented by the service user in a request or response type primitive is to be presented to the service user in the corresponding indication or confirm type primitive;

(=) when appended to one of the above, this symbol means that the parameter takes the same value as the parameter appearing immediately to its left;

blank the parameter is not present.

A primitive type may also be without parameters, i.e. no parameter is required with the primitive type; in this case the corresponding column of the table is empty.

7.3.1 MAP-OPEN service

This service is used for establishing a MAP dialogue between two MAP service-users. The service is a confirmed service with service primitives as shown in table 7.3/1.

Table 7.3/1: Service-primitives for the MAP-OPEN service

Parameters	Request	Indication	Response	Confirm
Application context name	M	M(=)	U	C(=)
Destination address	M	M(=)		
Destination reference	U	C(=)		
Originating address	U	O		
Originating reference	U	C(=)		
Specific information	U	C(=)	U	C(=)
Responding address			U	C(=)
Result			M	M(=)
Refuse-reason			C	C(=)
Provider error				O

Application context name:

This parameter identifies the type of application context being established. If the dialogue is accepted the received application context name shall be echoed. In case of refusal of dialogue this parameter shall indicate the highest version supported.

Destination address:

A valid SCCP address identifying the destination peer entity (see also clause 6). As an implementation option, this parameter may also, in the indication, be implicitly associated with the service access point at which the primitive is issued.

Destination-reference:

This parameter is a reference which refines the identification of the called process. It may be identical to Destination address but its value is to be carried at MAP level. Table 7.3/2 describes the MAP services using this parameter. Only these services are allowed to use it.

Table 7.3/2: Use of the destination reference

MAP service	Reference type	Use of the parameter
MAP-REGISTER-SS	IMSI	Subscriber identity
MAP-ERASE-SS	IMSI	Subscriber identity
MAP-ACTIVATE-SS	IMSI	Subscriber identity
MAP-DEACTIVATE-SS	IMSI	Subscriber identity
MAP-INTERROGATE-SS	IMSI	Subscriber identity
MAP-REGISTER-PASSWORD	IMSI	Subscriber identity
MAP-PROCESS-UNSTRUCTURED-SS-REQUEST	IMSI	Subscriber identity
MAP-UNSTRUCTURED-SS-REQUEST	IMSI	Subscriber identity
MAP-UNSTRUCTURED-SS-NOTIFY	IMSI	Subscriber identity
MAP-FORWARD-SHORT-MESSAGE	IMSI (note)	Subscriber identity
MAP-REGISTER-CC-ENTRY	IMSI	Subscriber identity
MAP-ERASE-CC-ENTRY	IMSI	Subscriber identity

NOTE: Only when the IMSI and the LMSI are received together from the HLR in the mobile terminated short message transfer.

Originating address:

A valid SCCP address identifying the requestor of a MAP dialogue (see also clause 6). As an implementation option, this parameter may also, in the request, be implicitly associated with the service access point at which the primitive is issued.

Originating-reference:

This parameter is a reference which refines the identification of the calling process. It may be identical to the Originating address but its value is to be carried at MAP level. Table 7.3/3 describes the MAP services using the parameter. Only these services are allowed to use it. Processing of the Originating-reference shall be performed according to the supplementary service descriptions and other service descriptions, e.g. operator determined barring.

Table 7.3/3: Use of the originating reference

MAP service	Reference type	Use of the parameter
MAP-REGISTER-SS	ISDN-Address-String	Originated entity address
MAP-ERASE-SS	ISDN-Address-String	Originated entity address
MAP-ACTIVATE-SS	ISDN-Address-String	Originated entity address
MAP-DEACTIVATE-SS	ISDN-Address-String	Originated entity address
MAP-INTERROGATE-SS	ISDN-Address-String	Originated entity address
MAP-REGISTER-PASSWORD	ISDN-Address-String	Originated entity address
MAP-PROCESS-UNSTRUCTURED-SS-REQUEST	ISDN-Address-String	Originated entity address
MAP-REGISTER-CC-ENTRY	ISDN-Address-String	Originated entity address
MAP-ERASE-CC-ENTRY	ISDN-Address-String	Originated entity address

Specific information:

This parameter may be used for passing any user specific information. Establishment and processing of the Specific information is not specified by GSM and shall be performed according to operator specific requirements.

Responding address:

An address identifying the responding entity. The responding address is included if required by the context (e.g. if it is different from the destination address).

Result:

This parameter indicates whether the dialogue is accepted by the peer.

Refuse reason:

This parameter is only present if the Result parameter indicates that the dialogue is refused. It takes one of the following values:

- Application-context-not-supported;
- Invalid-destination-reference;
- Invalid-originating-reference;
- No-reason-given;
- Remote node not reachable;
- Potential version incompatibility;

  Secured transport not possible.

****** Next modified section ******

7.3.7 MAP-SECURE-TRANSPORT-CLASS-1 service

This service is used for secure transport of a specific confirmed MAP service which is mapped on to a TCAP class 1 operation (i.e. one which can return a result or an error). The service is a confirmed service with service primitives as shown in table 7.3/11.

Table 7.3/11: Service-primitives for the MAP-SECURE-TRANSPORT-CLASS-1 service

Parameters	Request	Indication	Response	Confirm
Security header	M	M(=)	M	M(=)
Protected payload	C	C(=)	U	C(=)
User error			U	C(=)
Provider error				O

Security header:

This parameter carries the security header information required for secure transport of MAP messages. The details of this parameter are given in subclause 7.6.12.

Protected payload:

This parameter represents in protected mode the complete Request, Indication, Response or Confirm primitive of the service which makes use of the MAP-SECURE-TRANSPORT-CLASS-1 service.

User error:

If the application at the responding entity returns an error to be carried in the secure transport envelope, this parameter contains the Secure transport error defined in subclause 7.6.1.

Provider error

For the definition of provider errors see subclause 7.6.1.

7.3.8 MAP-SECURE-TRANSPORT-CLASS-2 service

This service is used for secure transport of a specific confirmed MAP service which is mapped on to a TCAP class 2 operation (i.e. one which can return an error but no result). The service is a confirmed service with service primitives as shown in table 7.3/12.

Table 7.3/12: Service-primitives for the MAP-SECURE-TRANSPORT-CLASS-2 service

Parameters	Request	Indication	Response	Confirm
Security header	M	M(=)	M	M(=)
Protected payload	C	C(=)		
User error			U	C(=)
Provider error				O

Security header:

This parameter carries the security header information required for secure transport of MAP messages. The details of this parameter are given in subclause 7.6.12.

Protected payload:

This parameter represents in protected mode the complete Request, Indication, Response or Confirm primitive of the service which makes use of the MAP-SECURE-TRANSPORT-CLASS-2 service.

User error:

If the application at the responding entity returns an error to be carried in the secure transport envelope, this parameter contains the Secure transport error defined in subclause 7.6.1.

Provider error

For the definition of provider errors see subclause 7.6.1.

7.3.9 MAP-SECURE-TRANSPORT-CLASS-3 service

This service is used for secure transport of a specific confirmed MAP service which is mapped on to a TCAP class 3 operation (i.e. one which can return a result but no error). The service is a confirmed service with service primitives as shown in table 7.3/13.

Table 7.3/13: Service-primitives for the MAP-SECURE-TRANSPORT-CLASS-3 service

Parameters	Request	Indication	Response	Confirm
Security header	M	M(=)	M	M(=)
Protected payload	C	C(=)	U	C(=)
Provider error				O

Security header:

This parameter carries the security header information required for secure transport of MAP messages. The details of this parameter are given in subclause 7.6.12.

Protected payload:

This parameter represents in protected mode the complete Request, Indication, Response or Confirm primitive of the service which makes use of the MAP-SECURE-TRANSPORT-CLASS-3 service.

Provider error

For the definition of provider errors see subclause 7.6.1.

7.3.10 MAP-SECURE-TRANSPORT-CLASS-4 service

This service is used for secure transport of a specific unconfirmed MAP service which is mapped on to a TCAP class 4 operation (i.e. one which can return neither a result nor an error). The service is an unconfirmed service with service primitives as shown in table 7.3/14.

Table 7.3/14: Service-primitives for the MAP-SECURE-TRANSPORT-CLASS-4 service

Parameters	Request	Indication
Security header	M	M(=)
Protected payload	C	C(=)

Security header:

This parameter carries the security header information required for secure transport of MAP messages. The details of this parameter are given in subclause 7.6.12.

Protected payload:

This parameter represents in protected mode the complete Request or Indication primitive of the service which makes use of the MAP-SECURE-TRANSPORT-CLASS-4 service.

****** Next modified section ******

7.6.1.4 User error

This parameter can take values as follows:

NOTE: The values are grouped in order to improve readability; the grouping has no other significance.

a) Generic error:

- system failure, i.e. a task cannot be performed because of a problem in another entity. The type of entity or network resource may be indicated by use of the network resource parameter;
- data missing, i.e. an optional parameter required by the context is missing;
- unexpected data value, i.e. the data type is formally correct but its value or presence is unexpected in the current context;
- resource limitation;
- initiating release, i.e. the receiving entity has started the release procedure;
- facility not supported, i.e. the requested facility is not supported by the PLMN;
- incompatible terminal, i.e. the requested facility is not supported by the terminal.

b) Identification or numbering problem:

- unknown subscriber, i.e. no such subscription exists;
- number changed, i.e. the subscription does not exist for that number any more;
- unknown MSC;
- unidentified subscriber, i.e. if the subscriber is not contained in the database and it has not or cannot be established whether or not a subscription exists;
- unallocated roaming number;
- unknown equipment;
- unknown location area.

c) Subscription problem:

- roaming not allowed, i.e. a location updating attempt is made in an area not covered by the subscription;
- illegal subscriber, i.e. illegality of the access has been established by use of authentication procedure;
- bearer service not provisioned;
- teleservice not provisioned;
- illegal equipment, i.e. the IMEI check procedure has shown that the IMEI is blacklisted or not whitelisted.

d) Handover problem:

- no handover number available;
- subsequent handover failure, i.e. handover to a third MSC failed for some reason.

e) Operation and maintenance problem:

- tracing buffer full, i.e. tracing cannot be performed because the tracing capacity is exceeded.

f) Call set-up problem:

- no roaming number available, i.e. a roaming number cannot be allocated because all available numbers are in use;
- absent subscriber, i.e. the subscriber has activated the detach service or the system detects the absence condition. This error may be qualified to indicate whether the subscriber was IMSI detached, in a restricted area or did not respond to paging;
- busy subscriber. This error may be qualified to indicate that the subscriber was busy due to CCBS and that CCBS is possible;
- no subscriber reply;
- forwarding violation, i.e. the call has already been forwarded the maximum number of times that is allowed;
- CUG reject, i.e. the call does not pass a CUG check; additional information may also be given in order to indicate rejection due to e.g. incoming call barred or non-CUG membership.
- call barred. Optionally, additional information may be included for indicating either that the call meets a barring condition set by the subscriber or that the call is barred for operator reasons. In the case of barring of Mobile Terminating Short Message, the additional information may indicate a barring condition due to «~~"~~"Unauthorised Message Originator"~~"~~».
- optimal routeing not allowed, i.e. the entity which sends the error does not support optimal routeing, or the HLR will not accept an optimal routeing interrogation from the GMSC, or the call cannot be optimally routed because it would contravene optimal routeing constraints.
- forwarding failed, i.e. the GMSC interrogated the HLR for forwarding information but the HLR returned an error.

g) Supplementary services problem:

- call barred;
- illegal SS operation;
- SS error status;
- SS not available;
- SS subscription violation;
- SS incompatibility;
- negative password check;
- password registration failure;
- Number of Password Attempts;
- USSD Busy;
- Unknown Alphabet.
- short term denial;
- long term denial.

For definition of these errors see GSM 04.80.

h) Short message problem:

- SM delivery failure with detailed reason as follows:
  - memory capacity exceeded;
  - MS protocol error;
  - MS not equipped;
  - unknown service centre (SC);
  - SC congestion;
  - invalid SME address;
  - subscriber is not an SC subscriber;
  - and possibly detailed diagnostic information, coded as specified in TS GSM 03.40, under SMS-SUBMIT-REPORT and SMS-DELIVERY-REPORT. If the SM entity which returns the SM Delivery Failure error includes detailed diagnostic information, it shall be forwarded in the MAP_MO_FORWARD_SHORT_MESSAGE and in the MAP_MT_FORWARD_SHORT_MESSAGE response.
- message waiting list full, i.e. no further SC address can be added to the message waiting list;
- Subscriber busy for MT SMS, i.e. the mobile terminated short message transfer cannot be completed because:
  - another mobile terminated short message transfer is going on and the delivery node does not support message buffering; or
  - another mobile terminated short message transfer is going on and it is not possible to buffer the message for later delivery; or
  - the message was buffered but it is not possible to deliver the message before the expiry of the buffering time defined in GSM 03.40;
- Absent Subscriber SM, i.e. the mobile terminated short message transfer cannot be completed because the network cannot contact the subscriber. Diagnostic information regarding the reason for the subscriber's absence may be included with this error.

i) Location services problem:

- Unauthorized Requesting Network
- Unauthorized LCS Client with detailed reason as follows
- Unauthorzied Privacy Class
- Unauthoized Call Unrelated External Client
- Unauthorized Call Related External Client
- Privacy override not applicable
- Position method failure with detailed reason as follows:
    - Congestion
    - Insufficient resources
    - Insufficient Measurement Data
    - Inconsistent Measurement Data
    - Location procedure not completed
    - Location procedure not supported by target MS
    - QoS not attainable
- Unknown or unreachable LCS Client

j) Problem detected by an application using secure transport:

- Secure transport error. This error indicates that the application using secure transport returned an error. The parameters of the error indicate:
    - The security header (see subclause 7.6.12);
    - The protected payload, which carries the result of applying the protection function specified in TS 33.102 to the encoding of the parameter of the original error.

****** Next modified section ******

7.6.12 Secure Transport Parameters

7.6.12.1 Security Header

This parameter carries the security header information which is required by a receiving entity in order to extract the protected information from a securely transported MAP message. The components of the security header are shown in table 7.6.12/1.

See TS 33.102 for the use of these parameters.

Table 7.6.12/1: Components of the Security Header

Component name	Presence requirement	Description
Sending PLMN identity	M	The Mobile Country Code and the Mobile Network Code of the PLMN which sent the secure MAP message.
Protection mode	M	The protection mode required for the message – one of: - No protection; - Integrity & Authenticity; - Integrity, Authenticity & Confidentiality.
Encryption algorithm identifier	C~~M~~	Identifies the encryption algorithm to be used for confidentiality-protection. Shall be present if Protection mode indicates 'Integrity, Authenticity & Confidentiality''; otherwise shall be absent.
Mode of operation	C~~M~~	The mode of operation for confidentiality protection – one of: - ECB; - CBC; - CFB; - OFB. Modes of operation are defined in ISO/IEC 10116 (1991). Shall be present if Encryption algorithm identifier is present; otherwise shall be absent.
Key version number for Encryption algorithm key	C~~M~~	The version number of the protection key to be used. Shall be present if Encryption algorithm identifier is present; otherwise shall be absent.
Hash algorithm identifier	C	Identifies the hash algorithm to be used for integrity protection. Shall be present if Protection mode is not 'No protection'; otherwise shall be absent.
Key version number for Hash algorithm key	C	The version number for the key used for the Hash algorithm. Shall be present if Hash algorithm identifier is present; otherwise shall be absent.
Initialisation vector	C~~U~~	An initialisation vector for the message protection function. Shall be present if the Mode of operation is CBC, CFB or OFB, otherwise shall be absent.
Original component identifier	M	Identifies the type of component to be securely transported – one of: - Operation, identified by the operation code; - Error, defined by the error code; - User information.

****** Next modified section ******

15 Elements of procedure

15.1 Handling of unknown operations

Unknown operations (i.e. a standard operation introduced in a later version of the MAP specification, or a private operation) can be introduced into MAP in a backwards compatible way. This means that the receiver of an unknown operation shall, if the dialogue state allows it, send a TC-REJECT component to the sender of the operation indicating 'unrecognised operation' and continue with the processing of further components or messages exchanged within the dialogue as if the unknown operation had not been received.

The standardised structure of a MAP dialogue shall not be affected by the invocation of unknown operations, i.e. if a dialogue uses only a TC-BEGIN message which is acknowledged by a TC-END message, a TC-CONTINUE

message shall not be used to invoke an unknown operation. However the standardised structure of a MAP dialogue may be affected by the rejection of unknown operations, i.e. if a dialogue uses only a TC-BEGIN message which is acknowledged by a TC-END message, a TC-CONTINUE message followed by a TC-END message may be used to carry the rejection of an unknown operation and the response to the standardised operation. The entity which initiated a dialogue whose standardised structure is a TC-BEGIN message which is acknowledged by a TC-END message shall not send any messages in that dialogue after the TC-BEGIN. Note that if the dialogue structure is affected as described in this paragraph the TC-CONTINUE shall include the dialogue portion required to confirm the acceptance of the dialogue.

Unknown operations may be invoked in the following types of message (there is no restriction as to how many unknown operations can be invoked in a message):

- TC-BEGIN: the component to invoke the unknown operation shall follow the component of the standard operation which is included in this message.

- TC-CONTINUE: the component to invoke the unknown operation may be transported as the only component in a stand-alone message or may be grouped with existing operations. In the latter case a specific sequencing of components is not required.

- TC-END: if the component to invoke the unknown operation is grouped with an existing operation a specific sequencing of components is not required

The TC-REJECT component may be sent in the following messages:

- TC-CONTINUE or TC-END: either as the only component of the message or grouped with an existing component. The choice is up to the MAP-Service User.

  If the received message contains only unknown operations the MAP-Service User shall send the TC-REJECT components in a TC-CONTINUE message to the peer entity, if the dialogue state allows it.

  If the received message contains unknown operations and standard operations and the standardised structure of the dialogue requires the response to the standard operation to be sent within a TC-END message, then the MAP-Service User may send the response to the standard operations and the TC-REJECT components for the unknown operations in a TC-CONTINUE message followed by a TC-END message. Neither a specific distribution of the components to the TC messages nor a specific sequencing of components is required.

Note that the SDL diagrams of clauses 19 - 25 do not show the report to the MAP-Service User about the reception of the unknown operation. This has been done for simplicity of description; the MAP PM may inform the MAP-Service User.

The sender of the unknown operation shall ensure that there is enough room in the used message for the unknown operation.

15.2 Dialogue establishment

The establishment of a MAP dialogue involves two MAP-service-users: the dialogue-initiator and the dialogue-responder.

This procedure is driven by the following signals:

- a MAP-OPEN request primitive from the dialogue-initiator;

- a TC-BEGIN indication primitive occurring at the responding side;

- a MAP-OPEN response primitive from the dialogue-responder;

- the first TC-CONTINUE indication primitive occurring at the initiating side;

and under specific conditions:

- a TC-END indication primitive occurring at the initiating side;

- a TC-U-ABORT indication primitive occurring at the initiating side;

- a TC-P-ABORT indication primitive occurring at the initiating side.

One instance of the MAP dialogue state machine runs at the initiating side, and one at the responding side.

15.2.1 Behaviour at the initiating side

The behaviour of the MAP dialogue state machine at the initiating side is defined in sheets 1 – 9 of the process Secure_MAP_DSM.

Sheet 1: The MAP protocol machine decides according to the application context name received in the MAP-OPEN request and the identity of the responder whether secure transport of the MAP dialogue is required. If secure transport is required, the MAP protocol machine builds a protected dialogue portion (including the AC name and any user information received in the MAP-OPEN request, encoded as user information for the TC-BEGIN) for the TC-BEGIN; otherwise it builds a normal dialogue portion using the application context name and any user data included in the MAP-OPEN request.

Sheet 2: If secure transport is used, each MAP specific service request is stored in case drop-back to unsecured transport is to be invoked.

Sheet 2: If secure transport is required, each MAP specific service request triggers the creation of an instance of the Secure_Requesting_MAP_SSM to handle the secure transport of the request. If secure transport is not required, each MAP specific service request triggers the creation of an instance of the Requesting_MAP_SSM to handle the transport of the request.

Sheet 3: When the the MAP dialogue state machine at the initiating side is waiting for a response from the responding side, a TC-END indication which echoes the AC name which was sent in the TC-BEGIN indicates acceptance of the dialogue. If secure transport is required, acceptance of the dialogue opening request which was transported in the secure dialogue opening request is indicated by the encapsulated AC name transported in the user information of the TC-END being equal to the encapsulated AC which was included in the user information of the TC-BEGIN. Mismatch of either the AC name or the encapsulated AC name indicates failure of the dialogue opening.

Sheet 3: If the dialogue opening is accepted, any components included in the TC-END are processed and passed to the MAP-Service User. The dialogue is closed by sending a MAP-CLOSE to the MAP-Service User.

Sheet 3, sheet 4, sheet 5, sheet 6, sheet 7, sheet 8, sheet 9: when a dialogue is terminated, the MAP dialogue state machine terminates all instances of the Requesting_MAP_SSM or Secure_Requesting_MAP_SSM which are active for this dialogue.

Sheet 4: A TC-P-ABORT with an abort parameter incorrect transaction portion indicates that the responding side does not support a MAP version higher than 1. If secure transport is not required, this triggers a MAP-OPEN confirm indicating that the dialogue is refused, with a refuse reason potential version incompatibility. The MAP-Service User may then decide to retry the dialogue at MAP version 1. If secure transport is required and fallback to unsecured transport is acceptable, the dialogue machine retries the dialogue with unsecured transport. If secure transport is required and fallback to unsecured transport is not acceptable, this triggers a MAP-OPEN confirm indicating that the dialogue is refused, with a refuse reason secured transport not possible. No retry of the dialogue with a lower version is allowed.

Sheet 5: If the intiating side receives a TC-U-ABORT with an abort reason AC not supported and secure transport is required, then secured transport is not possible. If fallback to unsecured transport is acceptable, the dialogue machine retries the dialogue with unsecured transport. If fallback to unsecured transport is not acceptable, this triggers a MAP-OPEN confirm indicating that the dialogue is refused, with a refuse reason secured transport not possible. No retry of the dialogue with a lower version is allowed.

Sheet 7: A TC-U-ABORT with a user-specific abort reason leads to a check of the user information. User information carrying a MAP-Refuse PDU with a refuse reason encapsulated AC not supported means that the responding entity supports the secure transport AC, but not the AC required for the protected request. This triggers a MAP-OPEN confirm indicating that the dialogue is refused, with a refuse reason AC not supported. The MAP-Service User may then decide to retry the dialogue with a lower AC version; this will again use secure transport.

Sheet 9: When the the MAP dialogue state machine at the initiating side is waiting for a response from the responding side, a TC-CONTINUE indication which echoes the AC name which was sent in the TC-BEGIN indicates acceptance of the dialogue. If secure transport is required, acceptance of the dialogue opening request which was transported in the secure dialogue opening request is indicated by the encapsulated AC name transported in the user information of the TC- CONTINUE being equal to the encapsulated AC which was included in the user information of the TC-BEGIN. Mismatch of either the AC name or the encapsulated AC name indicates failure of the dialogue opening.

Sheet 9: If the dialogue opening is accepted, any components included in the TC-CONTINUE are processed and passed to the MAP-Service User. The dialogue has then reached the established state.

15.2.2 Behaviour at the responding side

The behaviour of the MAP dialogue state machine at the responding side is defined in sheets 10 – 14 of the process Secure_MAP_DSM.

Sheet 10: If no application context information is included in the TC-BEGIN indication, this implies a MAP version 1 dialogue. An explicit application context indicating version 1 is treated as abnormal behaviour.

Sheet 10: The task "Extract_User_Information" includes decryption of the protected user information if confidentiality protection has been applied.

Sheet 11: The v1 application context name which corresponds to a v1 operation is derived using the information in table 15.2/1.

Table 15.2/1: Mapping of V1 operation codes on to application-context-names

Operation	Application-context-name (note 1)
updateLocation	networkLocUpContext-v1
cancelLocation	locationCancellationContext-v1
provideRoamingNumber	roamingNumberEnquiryContext-v1
insertSubscriberData	subscriberDataMngtContext-v1
deleteSubscriberData	subscriberDataMngtContext-v1
sendParameters	infoRetrievalContext-v1
	networkLocUpContext-v1 (note 2)
beginSubscriberActivity	networkFunctionalSsContext-v1
sendRoutingInfo	locationInfoRetrievalContext-v1
performHandover	handoverControlContext-v1
reset	resetContext-v1
activateTraceMode	tracingContext-v1
deactivateTraceMode	tracingContext-v1
sendRoutingInfoForSM	shortMsgGatewayContext-v1
forwardSM	shortMsgRelayContext-v1
reportSM-deliveryStatus	shortMsgGatewayContext-v1
noteSubscriberPresent	mwdMngtContext-v1
alertServiceCentreWithoutResult	shortMsgAlertContext-v1
checkIMEI	EquipmentMngtContext-v1

NOTE 1: These symbolic names refer to the object identifier value defined in clause 17 and allocated to each application-context used for the MAP.

NOTE 2: The choice between the application contexts is based on the parameters received in the operation.

Sheet 12: If the AC name received in the TC-BEGIN indicated that secure transport is required, the MAP dialogue state machine checks whether the encapsulated application context name is supported. If it is supported, the dialogue can be accepted. If the encapsulated AC name is not supported, the MAP dialogue machine indicates this by sending a TC-U-ABORT with a user-specific abort reason and user information indicating that the encapsulated AC name is not supported.

Sheet 12: If the dialogue is accepted, each component present in the TC-BEGIN is forwarded to an instance of a Performing_MAP_SSM or Secure_Performing MAP_SSM, by executing the procedure Process_Components.

Sheet 13: If the MAP dialogue state machine receives a MAP-OPEN confirm with a result accepted, it waits for any MAP specific service request or response primitives or a MAP-DELIMITER request.

Sheet 14: A MAP-DELIMITER request triggers a TC-CONTINUE request to accept the dialogue. The dialogue has then reached the established state.

Sheet 13, sheet 14: When a dialogue is terminated, the MAP dialogue state machine terminates all instances of the Requesting_MAP_SSM, Secure_Requesting_MAP_SSM, Performing_MAP_SSM or Secure_Performing_MAP_SSM which are active for this dialogue.

15.3 Dialogue continuation

Once established the dialogue is said to be in a continuation phase. The behaviour of the MAP dialogue state machine in this phase is defined in sheets 15 – 17 of the process Secure_MAP_DSM.

Both MAP users can request the transfer of MAP APDUs until one of them requests the termination of the dialogue.

Normal closure of an established dialogue is shown on sheet 16; abnormal termination is shown on sheet 17.

15.4 Load control

If an entity which should respond to a MAP dialogue opening request is overloaded, it uses the AC of the request to determine whether to discard the request. If the AC of the request is secure transport, the encapsulated AC (i.e. the AC of the dialogue for which secure transport is required) is used to determine whether the request is discarded.

The priority level allocated to each application-context is described in clause 5, tables 5.1/1 and 5.1/2.

15.5 Procedures for MAP specific services

This subclause describes the MAP procedures for MAP specific services. These procedures are driven by the following types of event:

- a MAP specific request or a MAP specific response primitive;
- a component handling primitive from TC.

A Service State Machine is activated when of one of the following signals is received:

- a MAP request primitive, which activates a requesting SSM;
- a TC-INVOKE indication primitive without a linked identifier, which activates a performing SSM.

For component handling primitives there are two types of event:

- events which activate a Service State Machine or which can be related to an existing one;
- events which cannot be related to a Service State Machine.

15.5.1 Service invocation for unsecured dialogues

The behaviour of the requesting SSM which handles a service for an unsecured dialogue is defined by the SDL for the process Requesting_MAP_SSM. The requesting SSM receives a MAP service request from the MAP-Service User via the MAP dialogue state machine and sends a TC-INVOKE request to TCAP. When a confirm is received from TCAP via the MAP dialogue state machine, the requesting SSM forwards a MAP service confirm to the MAP-Service User.

The response to a MAP service invocation may come in the form of a linked request. If the linked request corresponds to a class 4 operation, this is handled by the requesting SSM. If the linked request corresponds to a

class 1, 2 or 3 operation, the MAP dialogue state machine sends a notification to the requesting SSM and creates an instance of a performing SSM to handle the linked request.

The mapping of MAP specific services on to remote operations is given in table 16.2/1.

15.5.2 Service invocation for secured dialogues

The behaviour of the requesting SSMs which handle a service for a secured dialogue is defined by the SDL for the processes Secure_Requesting_MAP_SSM and Requesting_MAP_SSM. The secure requesting SSM receives a MAP service request from the MAP-Service User via the MAP dialogue state machine and constructs the corresponding MAP secure transport service request. It then creates an instance of the requesting SSM and sends the MAP secure transport service request to it. The requesting SSM sends a TC-INVOKE request to TCAP. When the MAP dialogue state machine receives a confirm from TCAP, it forwards it to the secure requesting SSM, which unpacks the MAP service confirm from the MAP secure transport service confirm and sends it to the requesting SSM. The requesting SSM forwards the MAP service confirm to the MAP-Service User.

The response to a MAP service invocation which was carried in a secure dialogue may come in the form of a linked request. This linked request is carried in a MAP secure transport service request of the class corresponding to the operation; however the MAP secure transport service request is not linked to another MAP secure transport service request. If the linked request which is carried in the MAP secure transport service corresponds to a class 4 operation, this is handled by the secure requesting service state machine, which unpacks the linked request and sends it to the requesting SSM. If the linked request which is carried in the MAP secure transport service corresponds to a class 1, 2 or 3 operation, the MAP dialogue state machine sends a notification to the secure requesting SSM (which passes the notification to the requesting SSM) and creates an instance of a secure performing SSM to handle the linked request.

15.5.3 Service invocation receipt for unsecured dialogues

The behaviour of the performing SSM which handles a service for an unsecured dialogue is defined by the SDL for the process Performing_MAP_SSM. The performing SSM receives a TC-INVOKE component from TCAP via the MAP dialogue state machine and sends a MAP service indication to the MAP-Service User. When a MAP service response is received from the MAP-Service User via the MAP dialogue state machine, the performing SSM forwards a TC-RESULT or TC-U-ERROR component to TCAP.

15.5.4 Service invocation receipt for unsecured dialogues

The behaviour of the performing SSMs which handle a service for a secured dialogue is defined by the SDL for the processes Secure_Performing_MAP_SSM and Performing_MAP_SSM. The secure performing SSM receives a TC-INVOKE component containing a secure MAP transport service from TCAP via the MAP dialogue state machine and unpacks the MAP service indication from it. It then creates an instance of the performing SSM and sends the MAP service indication to it. The performing SSM forwards the MAP service indication to the MAP-Service User. When the MAP dialogue state machine receives a MAP service response from the MAP-Service User it forwards it to the secure performing SSM. The secure performing SSM constructs a MAP secure transport service response and sends it to the performing SSM, which forwards a TC-RESULT or TC-U-ERROR component to TCAP.

15.5.5 Handling of components received from TC

The procedure Process_Components shows the handling of components received in a TC-BEGIN, TC-CONTINUE or TC-END message.

Sheet 1: If a linked invoke component is transported securely, the linked invoke ID is carried as part of the security header, so that it can be checked without the need to unpack the protected component.

Sheet 2: If a linked invoke component corresponds to a class 4 operation, the MAP dialogue state machine sends it to the requesting SSM instance identified by the linked invoke ID. If a linked invoke component corresponds to any other class of operation, the MAP dialogue state machine sends a notification to the requesting SSM instance identified by the linked invoke ID, creates an instance of a performing SSM and sends the invoke component to it.

15.6 SDL descriptions

The following SDL specification describes a system which includes three blocks: MAP-user, MAP-provider and TC.

Such a system resides in each network component supporting MAP and communicates with its peers via the lower layers of the signalling network which are part of the environment.

Only the MAP-provider is fully described in this subclause. The various types of processes which form the MAP-User block and the TC block are described respectively in clauses 18 to 25 of the present document and in CCITT Recommendation Q.774.

The MAP-Provider block communicates with the MAP_USER via two channels U1 and U2. Via U1 the MAP-provider receives the MAP request and response primitives. Via U2 it sends the MAP indication and confirm primitives.

The MAP-Provider block communicates with TC via two channels P1 and P2. Via P1 the MAP-Provider sends all the TC request primitives. Via P2 it receives all the TC indication primitives.

The MAP-Provider block is composed of the six following types of process:

a) Secure_MAP_DSM: This type of process handles a dialogue for both secured and unsecured transport of MAP messages. There exists one process instance per MAP dialogue.

b) Load_Ctrl: This type of process is in charge of load control. There is only one instance of this process in each system.

c) Requesting_MAP_SSM: This type of process handles a MAP service requested during a dialogue. For unsecured transport of MAP messages, an instance of this process is created by the instance of the Secure_MAP_DSM process for each requested MAP-service. For secured transport of MAP messages, an instance of this process is created by the instance of the Secure_Requesting_MAP_SSM process for each requested MAP-Secure-Transport-service.

d) Secure_Requesting_MAP_SSM: This type of process handles a MAP service requested during a dialogue for secured transport of MAP messages. An instance of this process is created by the Secure_MAP_DSM process for each requested MAP-service.

e) Performing_MAP_SSM: This type of process handles a MAP service performed during a dialogue. For unsecured transport of MAP messages, an instance of this process is created by the instance of the Secure_MAP_DSM process for each MAP-service to be performed. For secured transport of MAP messages, an instance of this process is created by the instance of the Secure_Performing_MAP_SSM process for each MAP-Secure-Transport-service to be performed.

f) Secure_Performing_MAP_SSM: This type of process handles a MAP service performed during a dialogue for secured transport of MAP messages. An instance of this process is created by the Secure_MAP_DSM process for each MAP-service to be performed.

A process Secure_MAP_DSM exchanges external signals with other blocks as well as internal signals with the other processes of the MAP-Provider block. The external signals are either MAP service primitives or TC service primitives.

The signal routes used by the various processes are organized as follows:

a) A process Secure_MAP_DSM receives and sends events from/to the MAP_user via signal route User1/User2. These routes use channels U1 and U2 respectively.

b) A process Secure_MAP_DSM receives and sends events from/to the TCAP via signal route TC1/TC2. These routes use channels P1 and P2 respectively.

c) A process Secure_MAP_DSM receives and sends events from/to the LOAD_CTRL process via signal route Load1/Load2. These routes are internal.

d) A process Secure_MAP_DSM sends events to the Performing_MAP_SSM processes via signal route Intern1. This route is internal.

e) A process Secure_MAP_DSM sends events to the Requesting_MAP_SSM processes via signal route Intern2. This route is internal.

f) A process Secure_MAP_DSM sends events to the Secure_Performing_MAP_SSM processes via signal route Intern3. This route is internal.

g) A process Secure_MAP_DSM sends events to the Secure_Requesting_MAP_SSM processes via signal route Intern4. This route is internal.

h) A process Performing_MAP_SSM sends events to the MAP_USER via signal route User3. This route uses channel U2.

i) A process Performing_MAP_SSM sends events to the TCAP via signal route TC3. This route uses channel P1.

j) A process Requesting_MAP_SSM sends events to the MAP_USER via signal route User4. This route uses channel U2.

k) A process Requesting_MAP_SSM sends events to the TCAP via signal route TC4. This route uses channel P1.

l) A process Secure_Performing_MAP_SSM sends events to the MAP_USER via signal route User5. This route uses channel U2.

m) A process Secure_Performing_MAP_SSM sends events to the TCAP via signal route TC5. This route uses channel P1.

n) A process Secure_Performing_MAP_SSM sends events to the corresponding Performing_MAP_SSM process via signal route Intern5. This route is internal.

o) A process Secure_Requesting_MAP_SSM sends events to the MAP_USER via signal route User6. This route uses channel U2.

p) A process Secure_Requesting_MAP_SSM sends events to the TCAP via signal route TC6. This route uses channel P1.

q) A process Secure_Requesting_MAP_SSM sends events to the corresponding Requesting_MAP_SSM process via signal route Intern6. This route is internal.

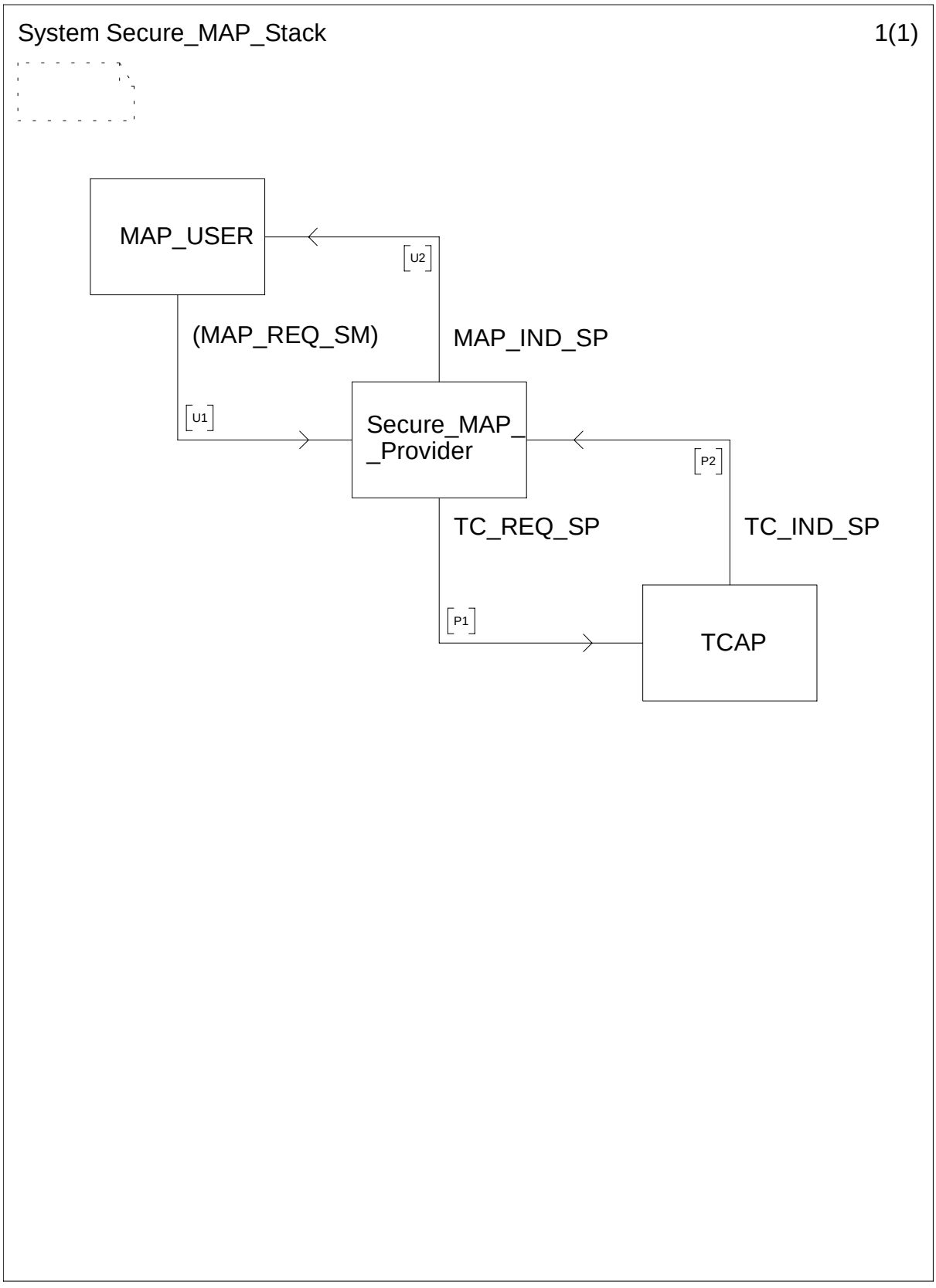

Figure 15.6/1: System Secure_MAP_Stack

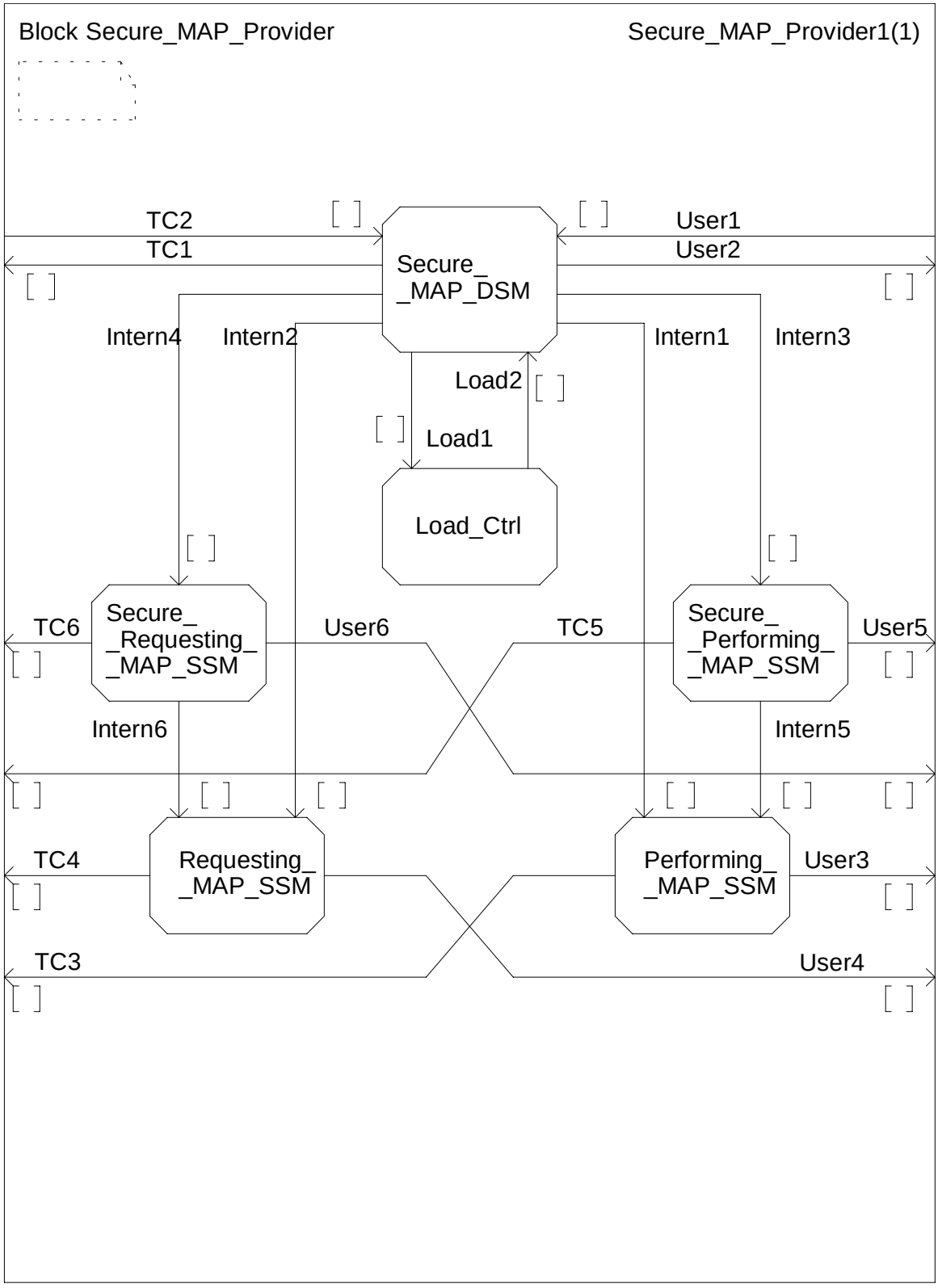

Figure 15.6/2: Block Secure_MAP_Provider

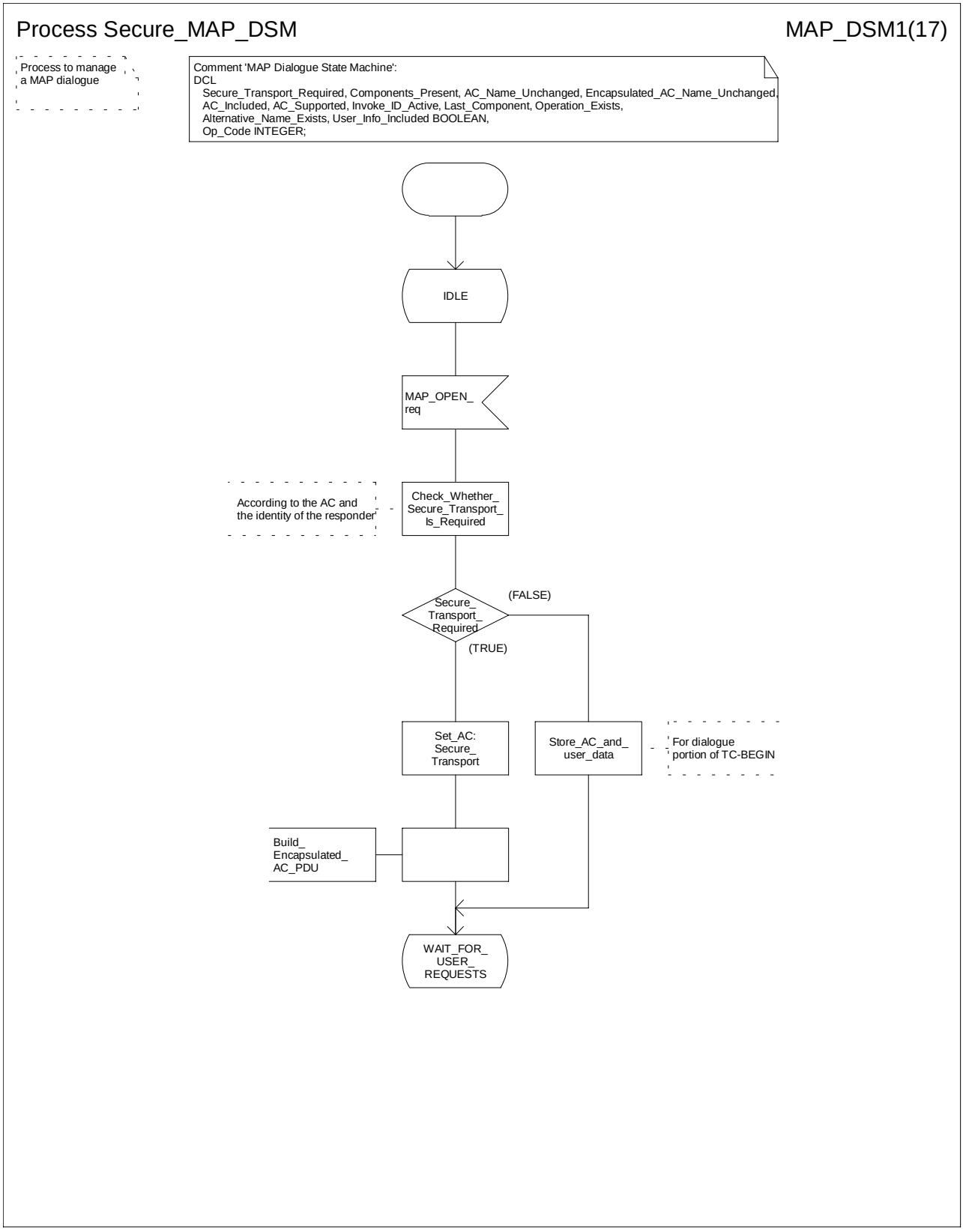

Figure 15.6/3a: Process Secure_MAP_DSM (sheet 1)

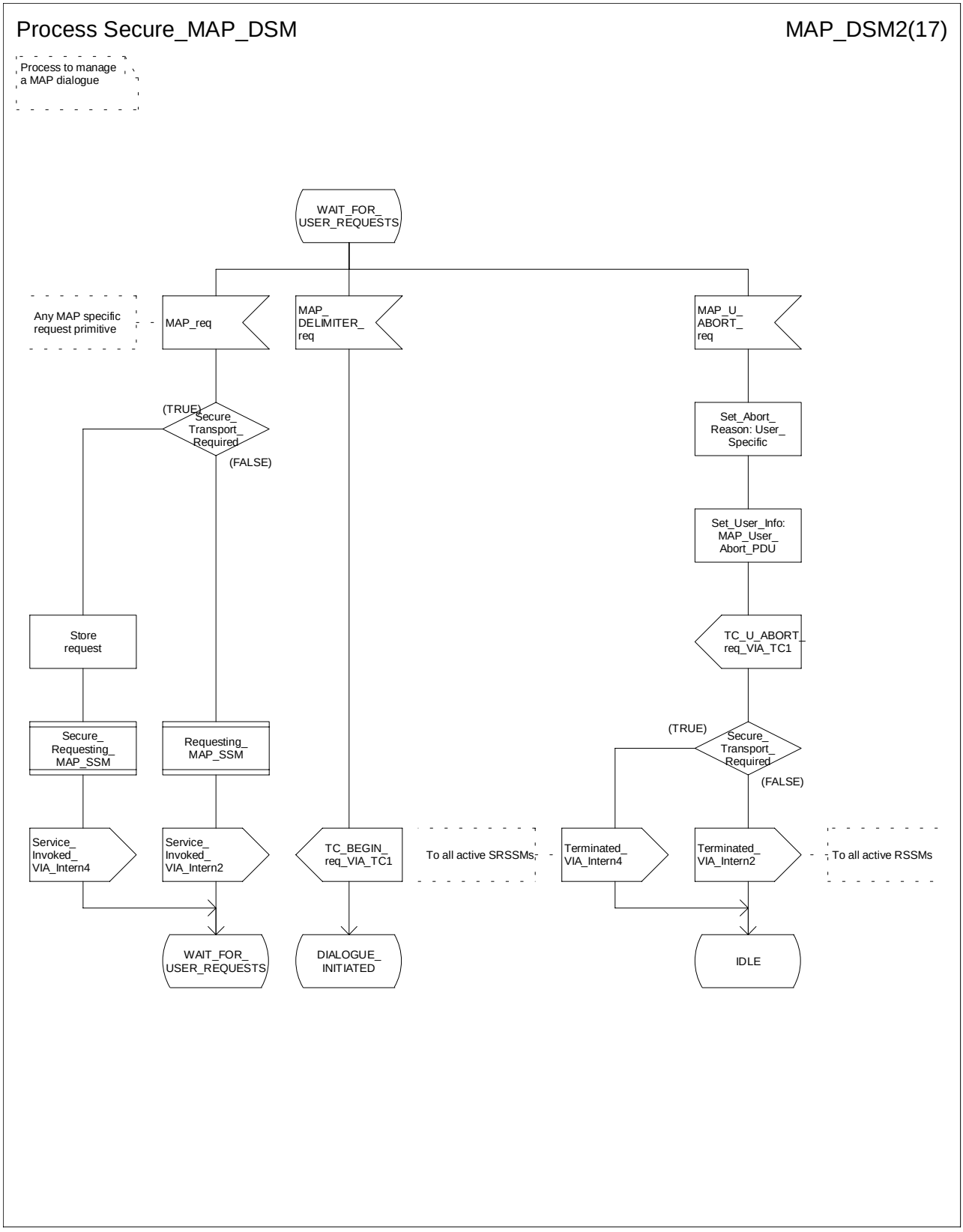

Figure 15.6/3b: Process Secure_MAP_DSM (sheet 2)

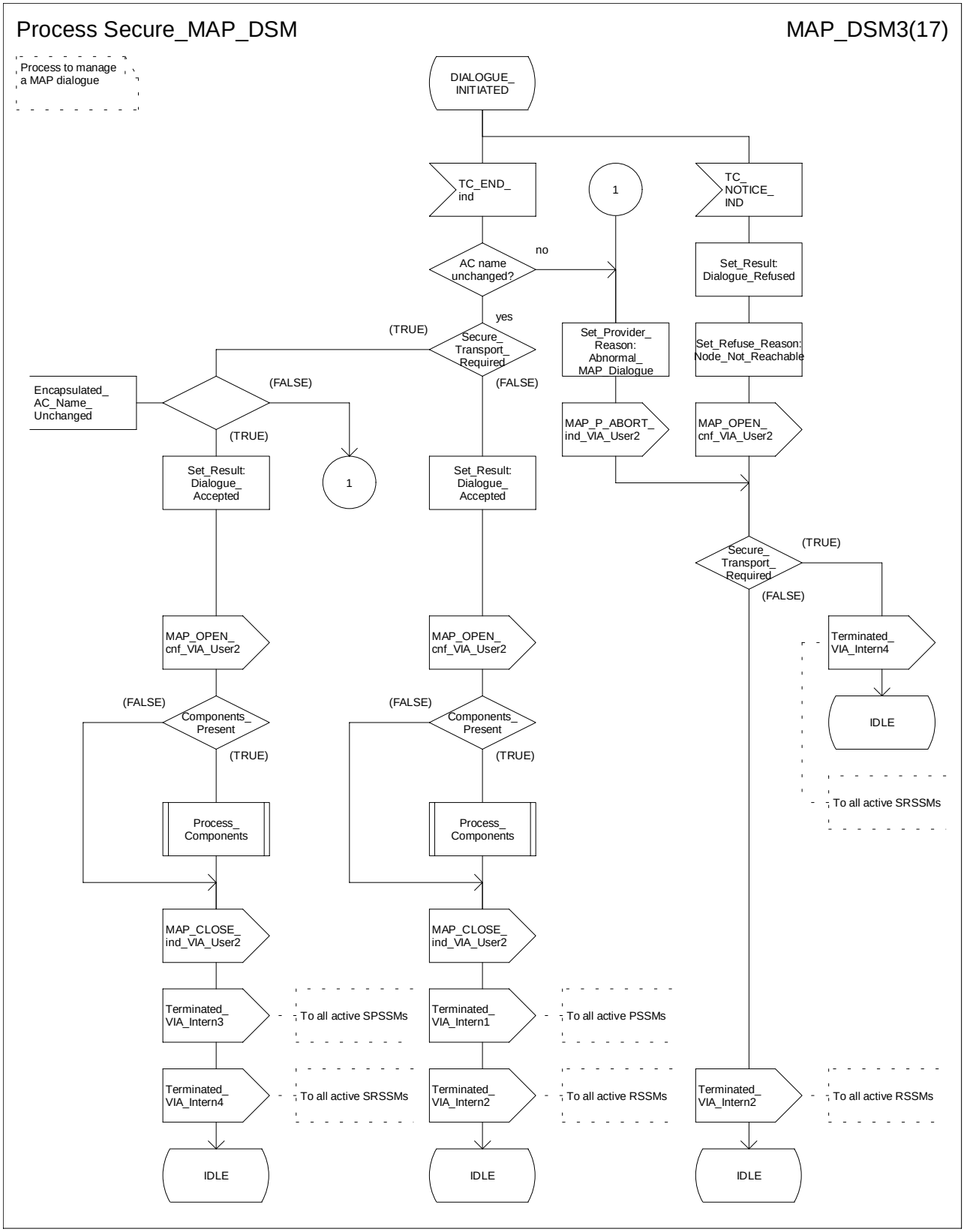

Figure 15.6/3c: Process Secure_MAP_DSM (sheet 3)

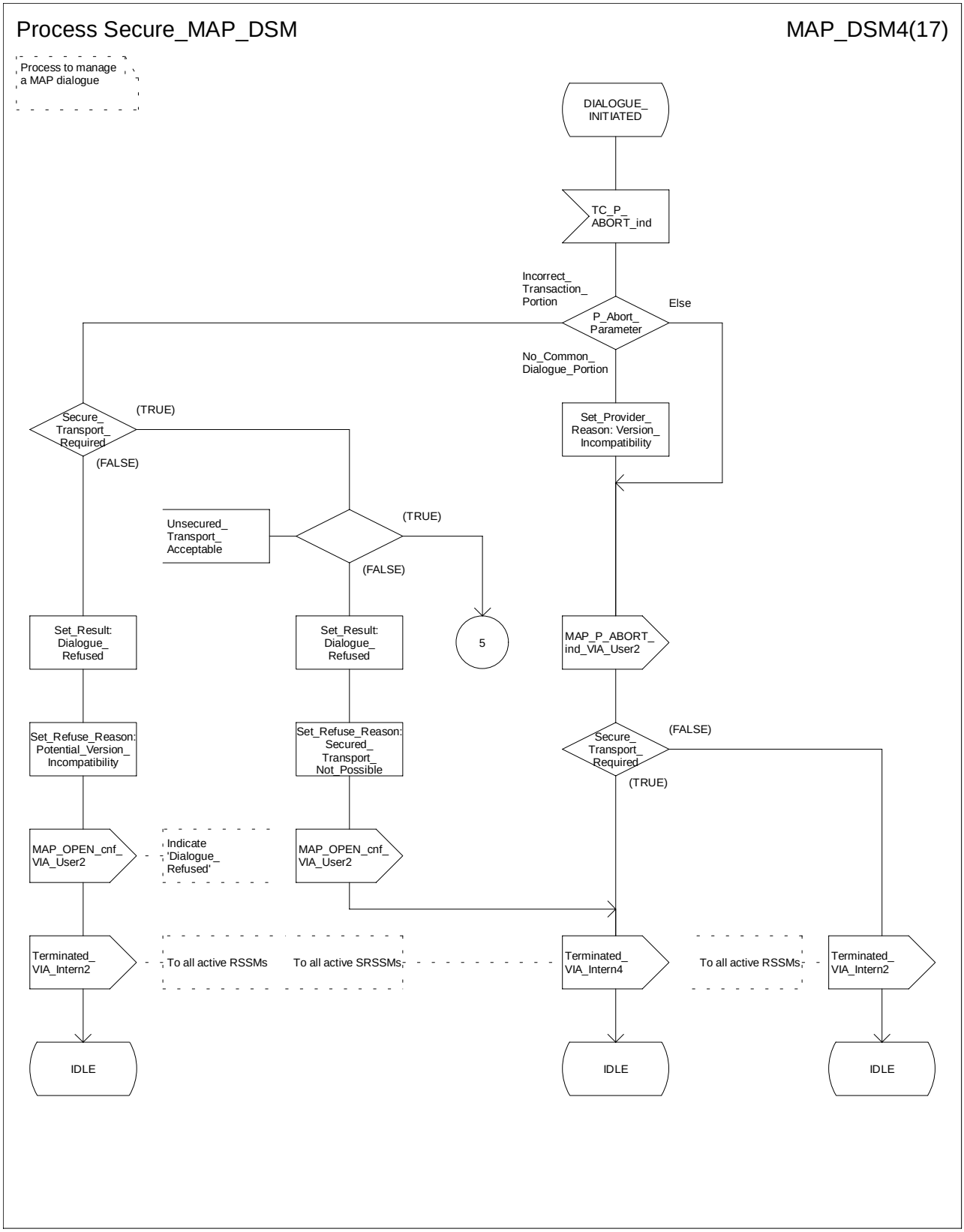

Figure 15.6/3d: Process Secure_MAP_DSM (sheet 4)

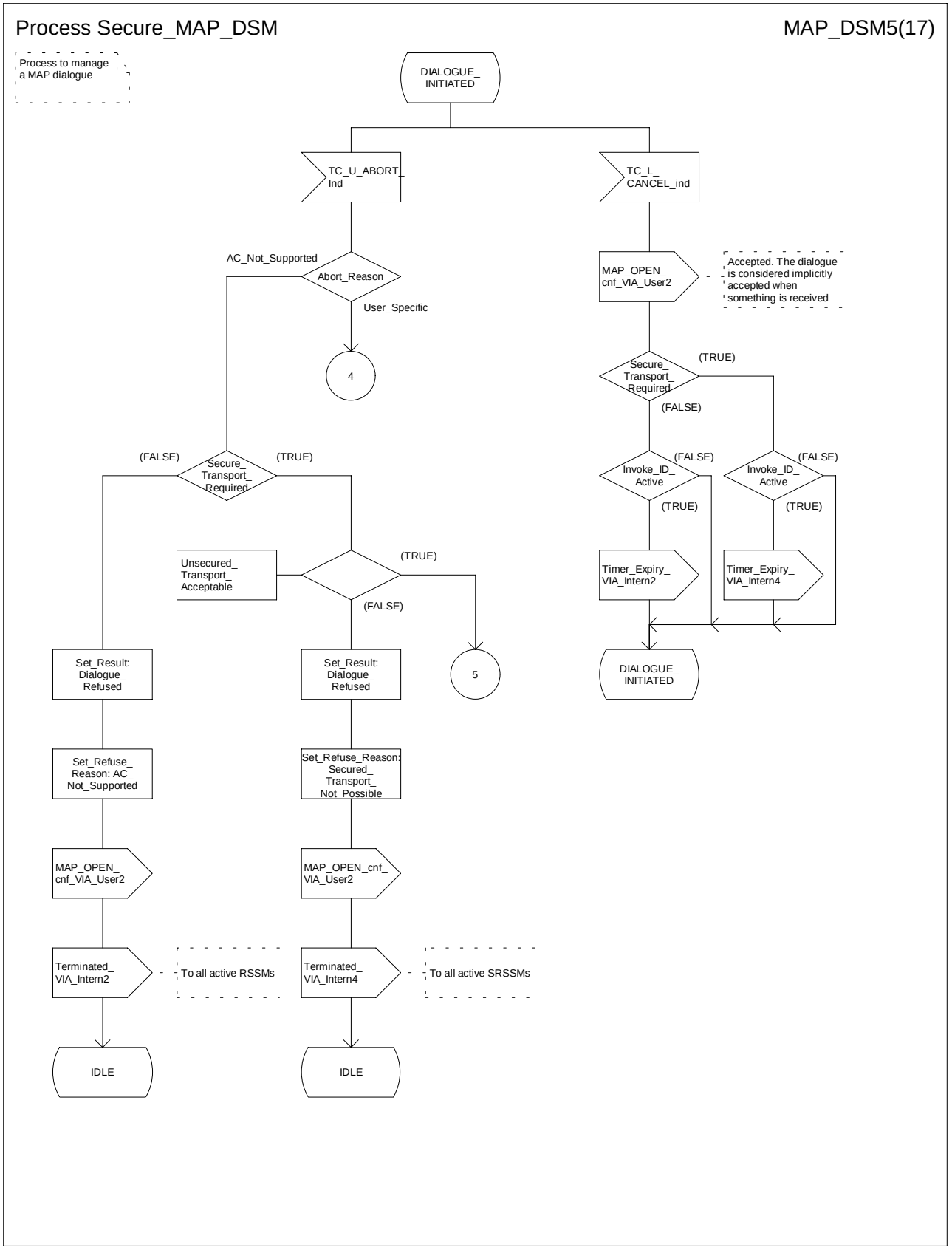

Figure 15.6/3e: Process Secure_MAP_DSM (sheet 5)

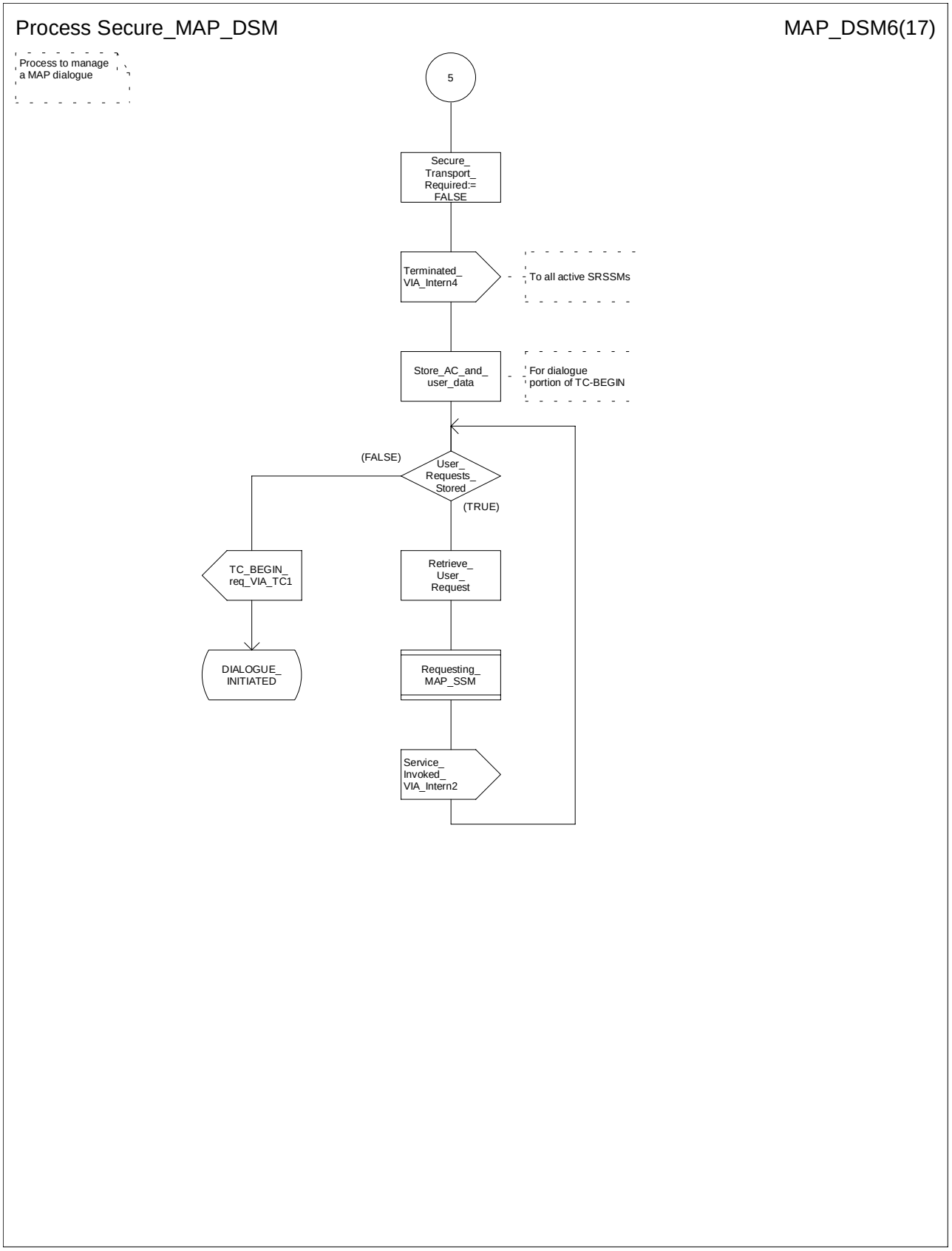

Figure 15.6/3f: Process Secure_MAP_DSM (sheet 6)

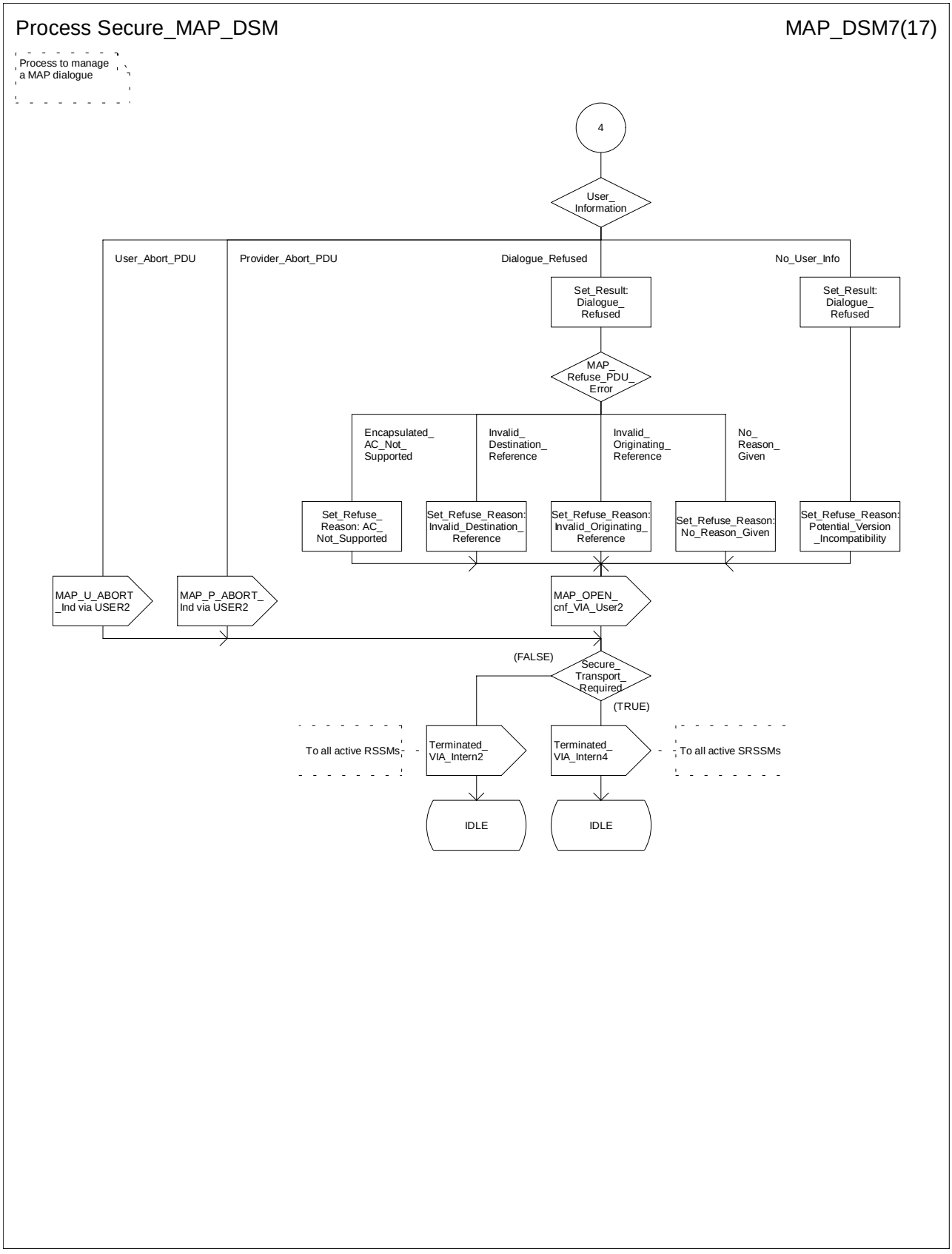

Figure 15.6/3g: Process Secure_MAP_DSM (sheet 7)

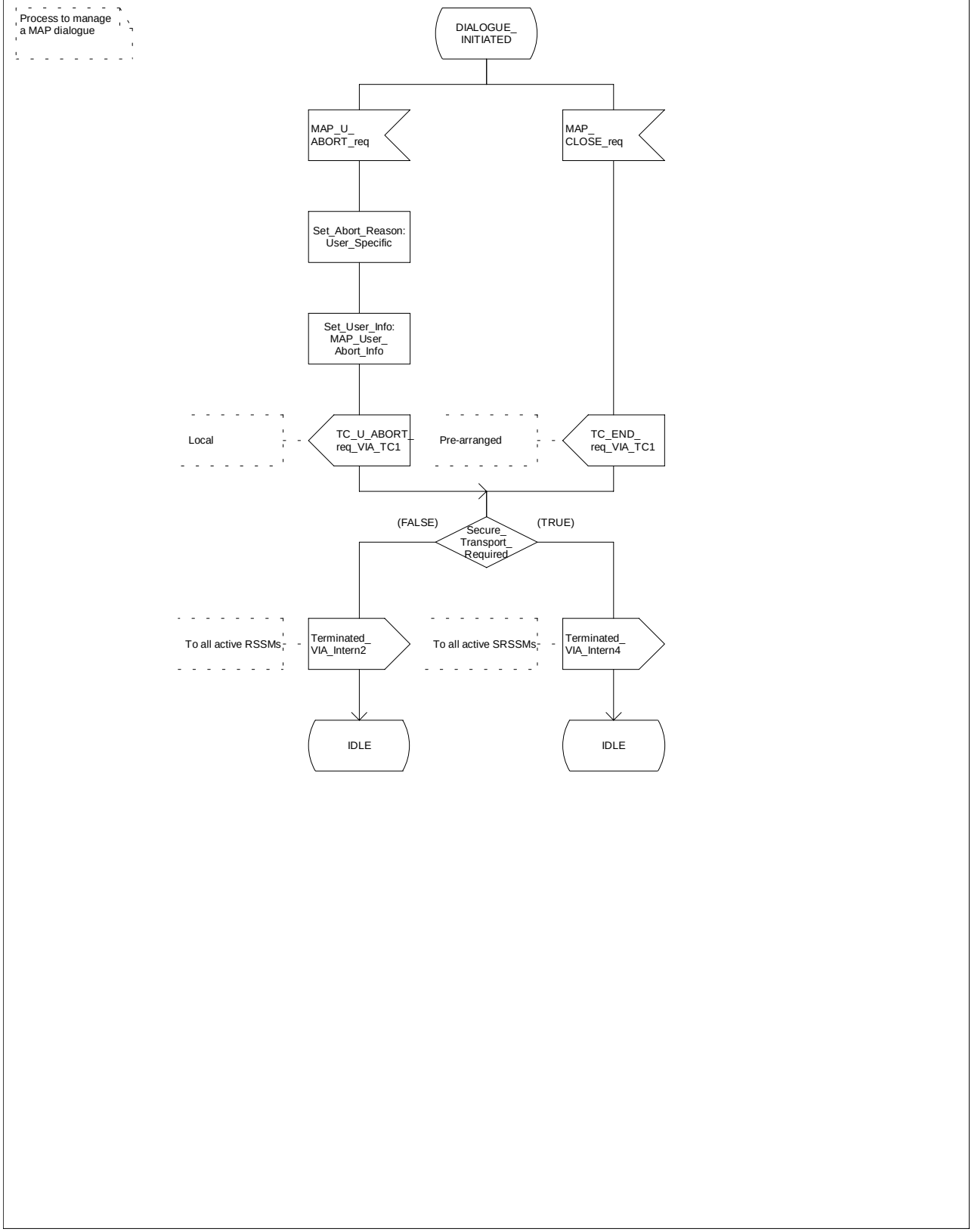

Figure 15.6/3h: Process Secure_MAP_DSM (sheet 8)

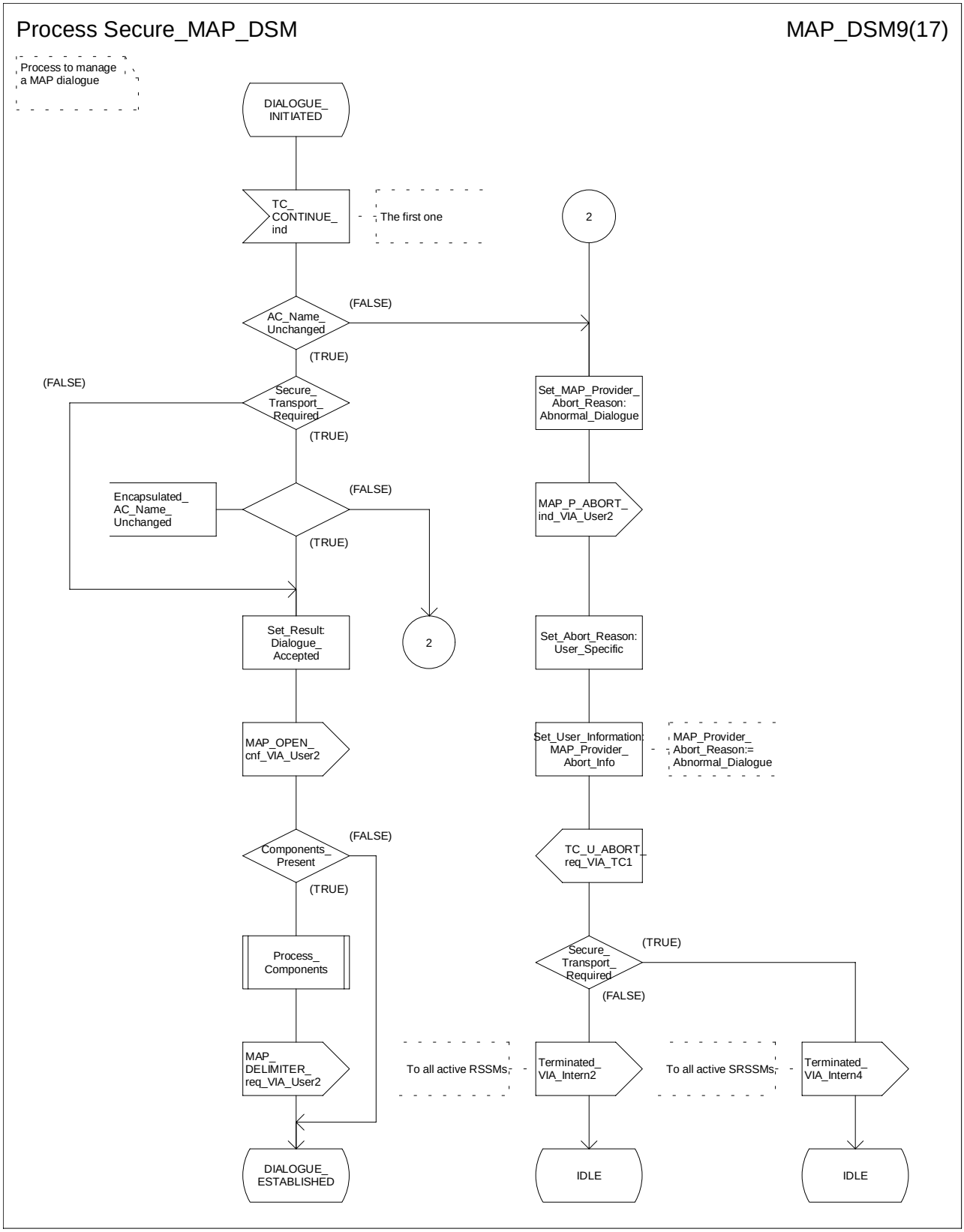

Figure 15.6/3i: Process Secure_MAP_DSM (sheet 9)

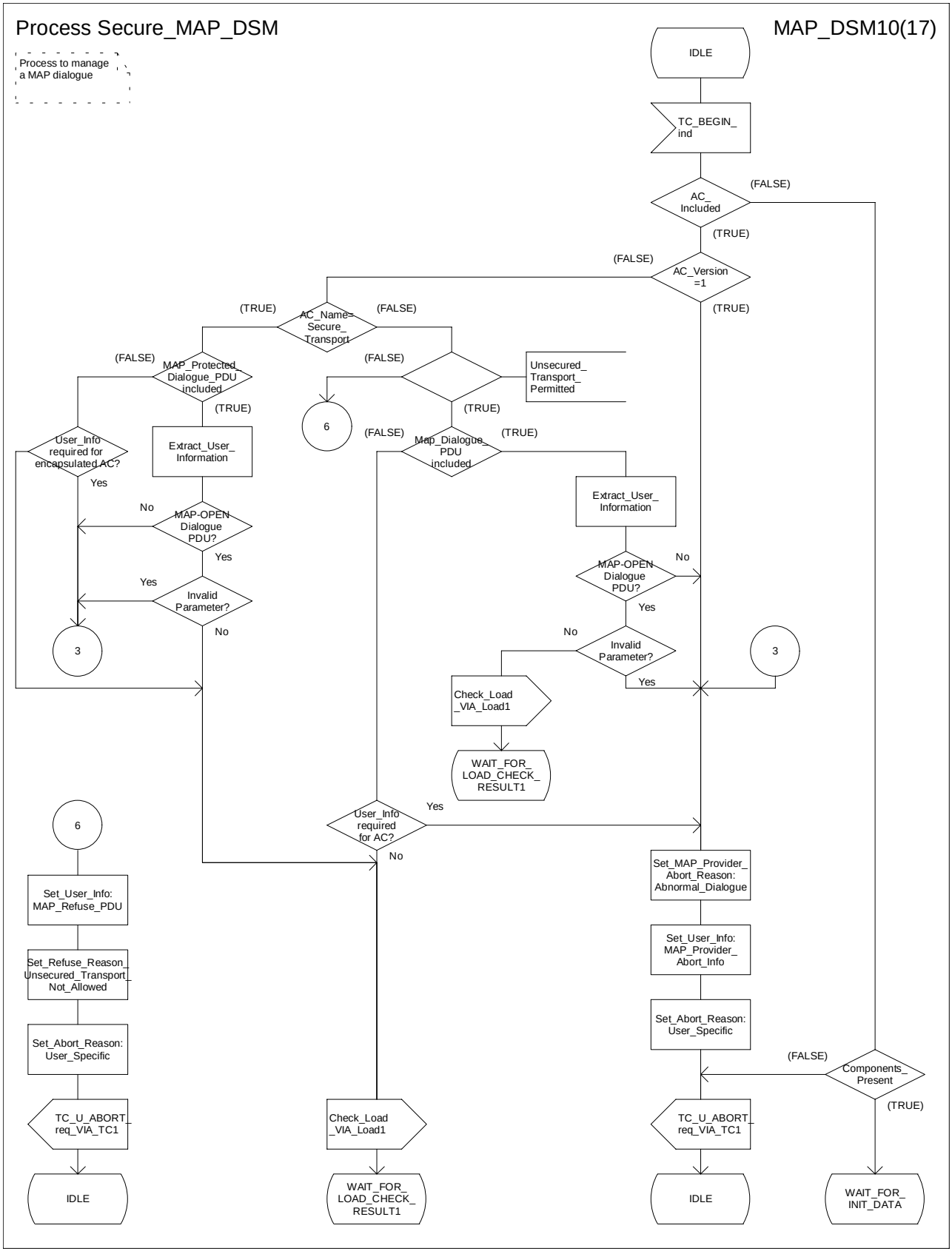

Figure 15.6/3j: Process Secure_MAP_DSM (sheet 10)

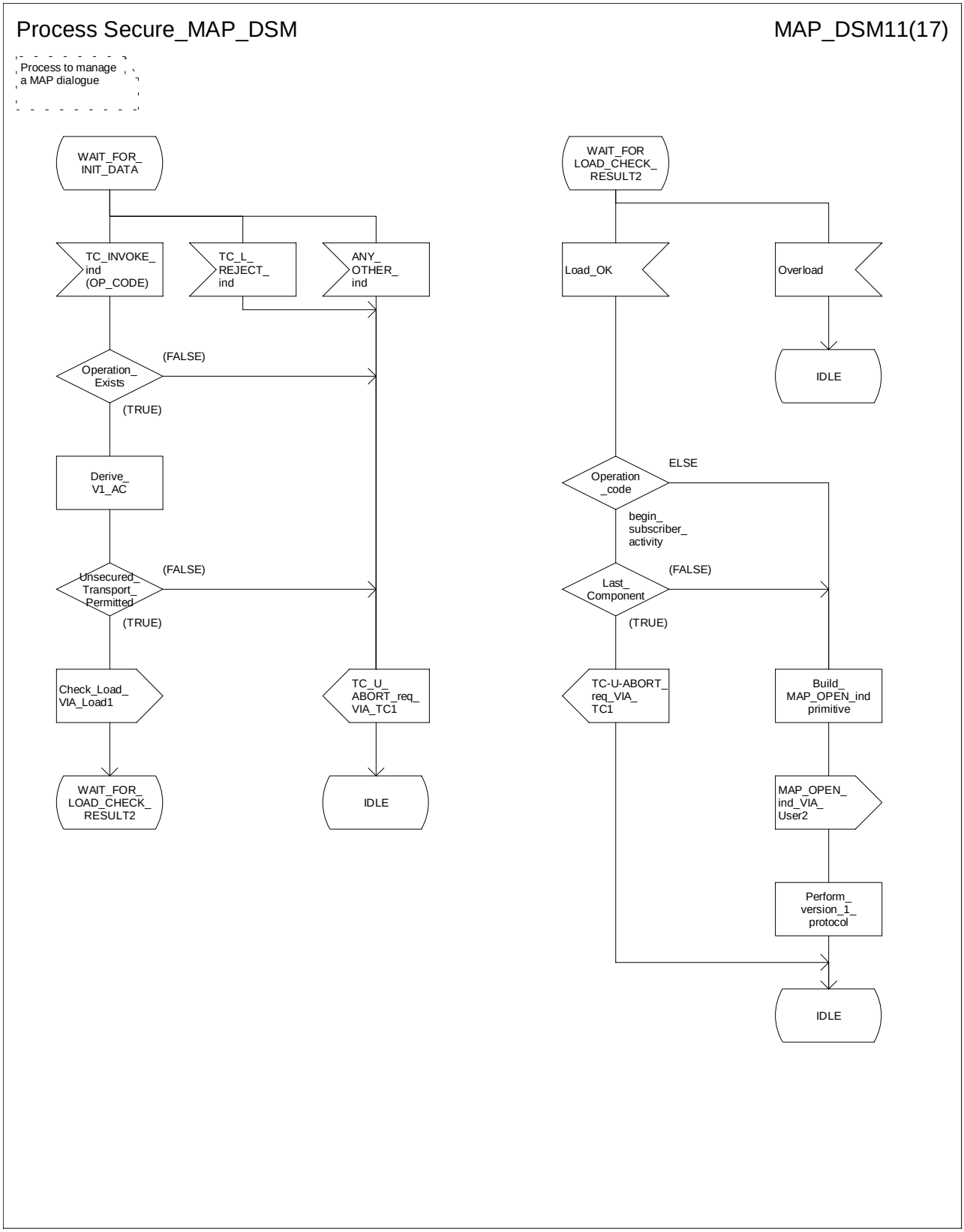

Figure 15.6/3k: Process Secure_MAP_DSM (sheet 11)

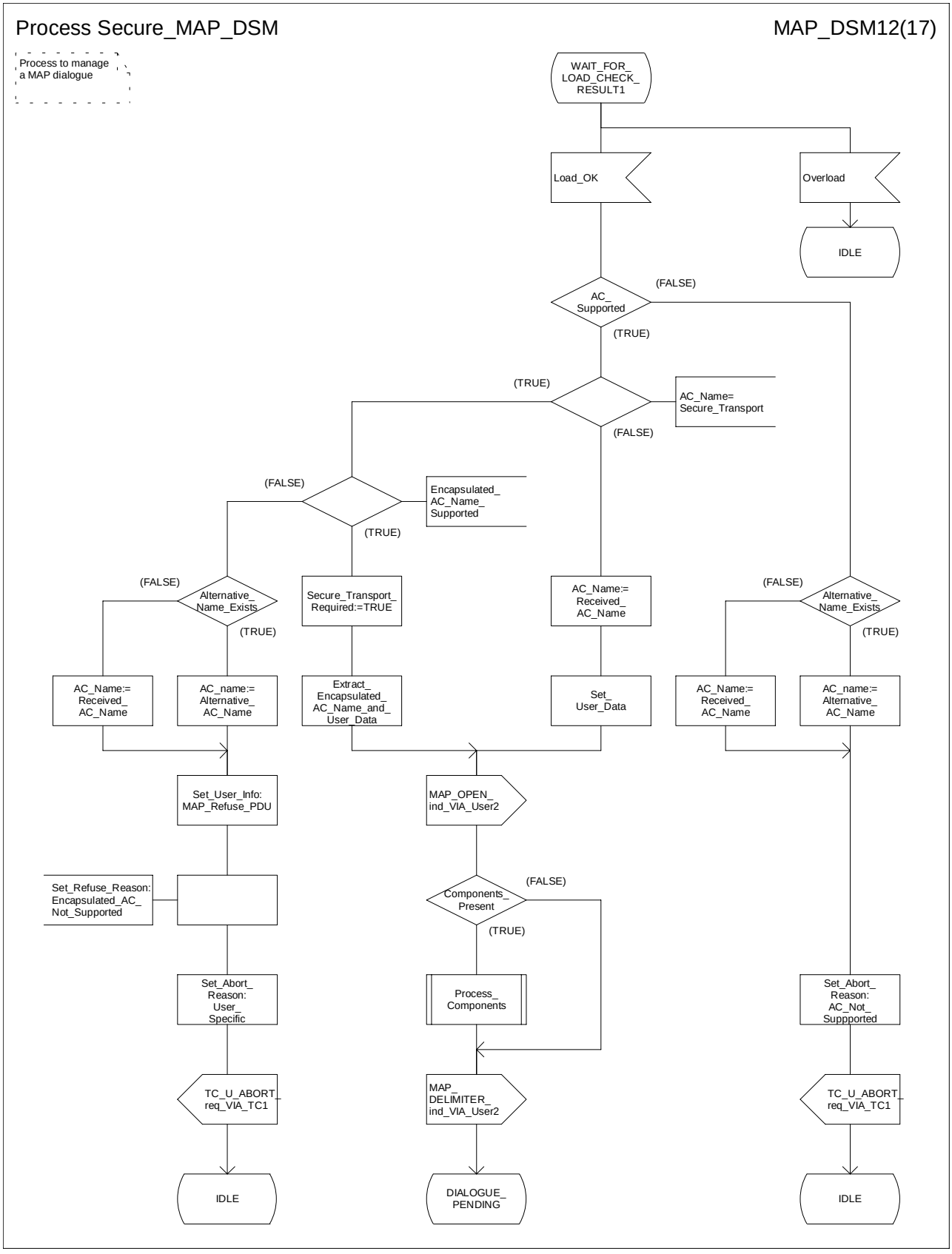

Figure 15.6/3l: Process Secure_MAP_DSM (sheet 12)

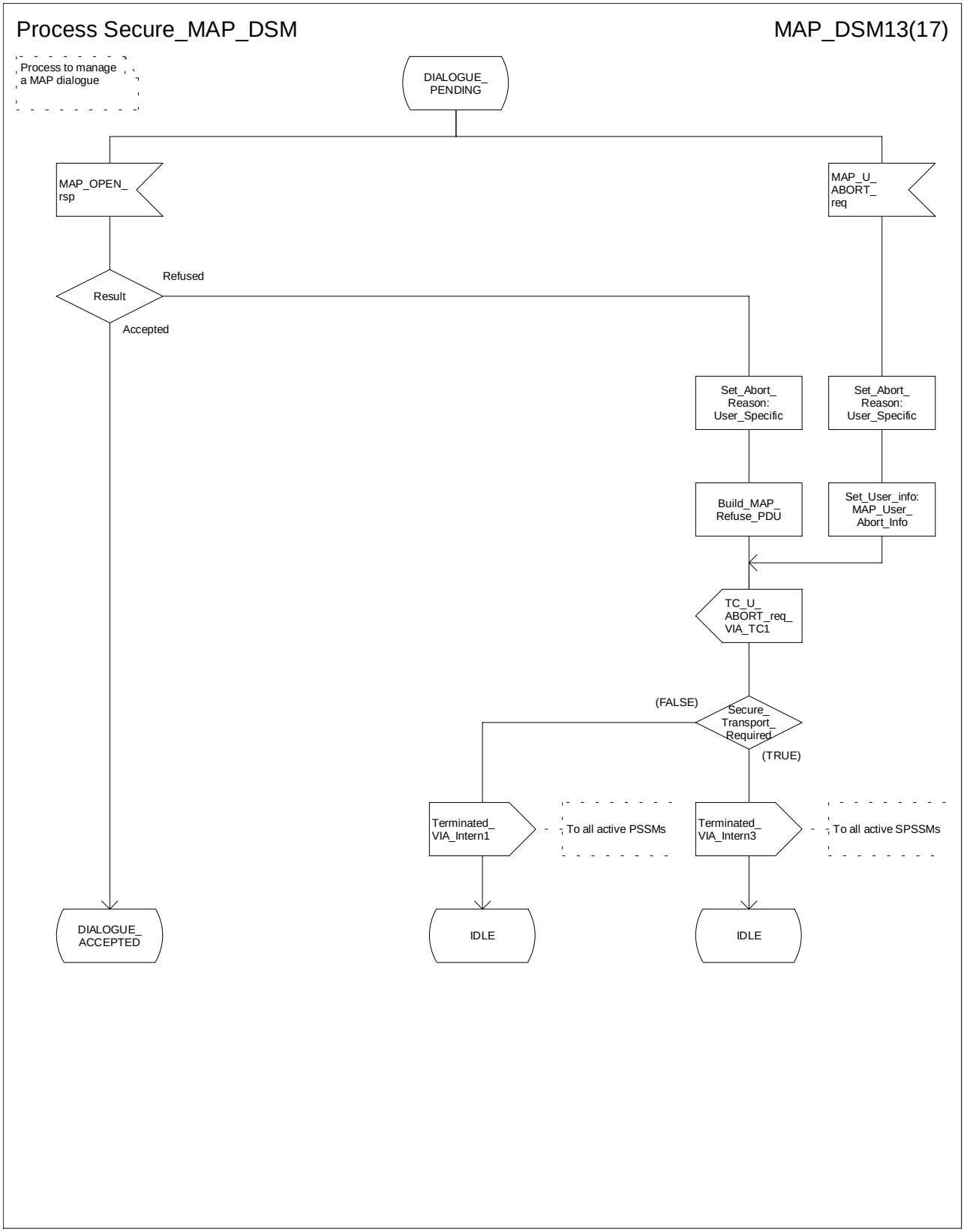

Figure 15.6/3m: Process Secure_MAP_DSM (sheet 13)

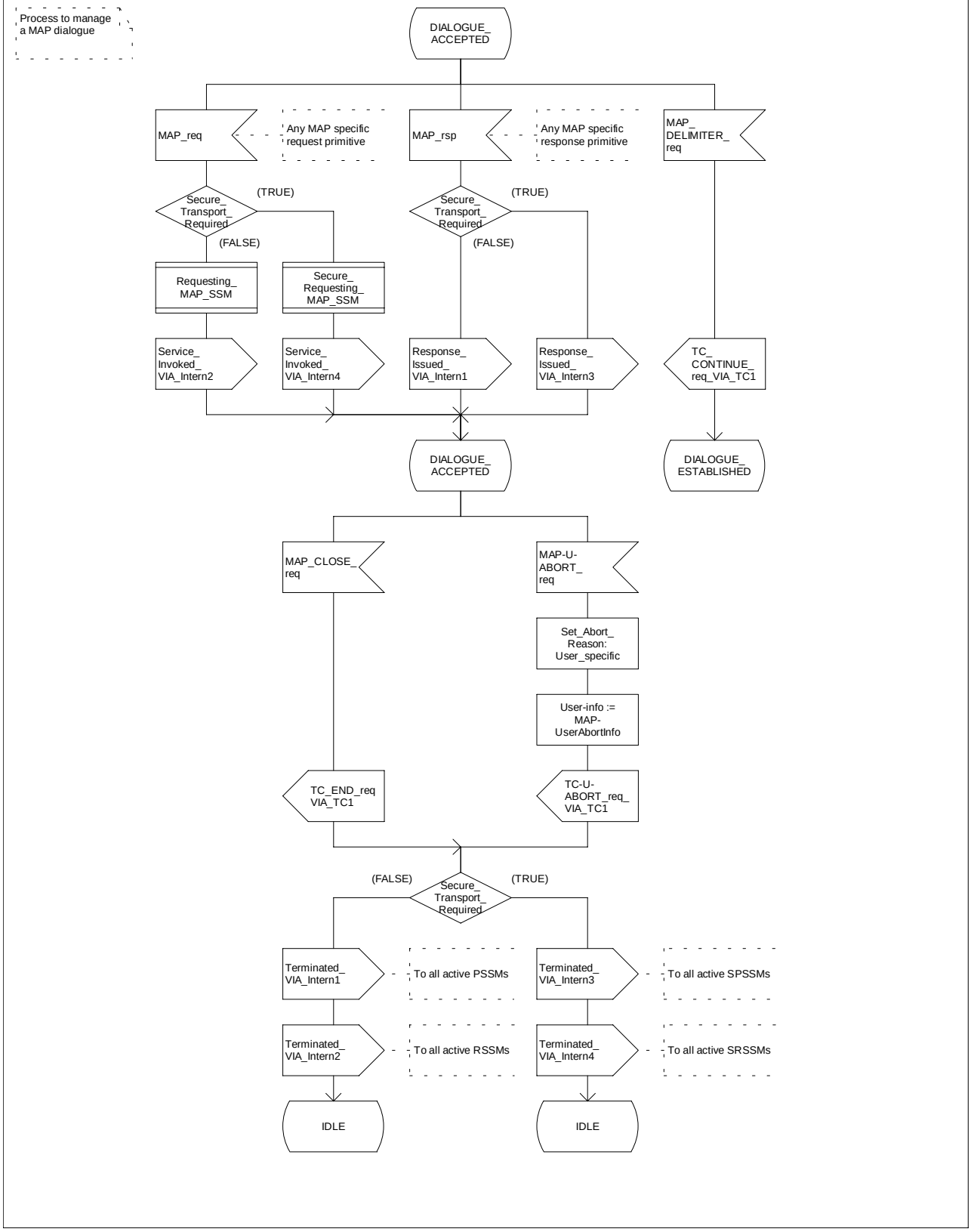

Figure 15.6/3n: Process Secure_MAP_DSM (sheet 14)

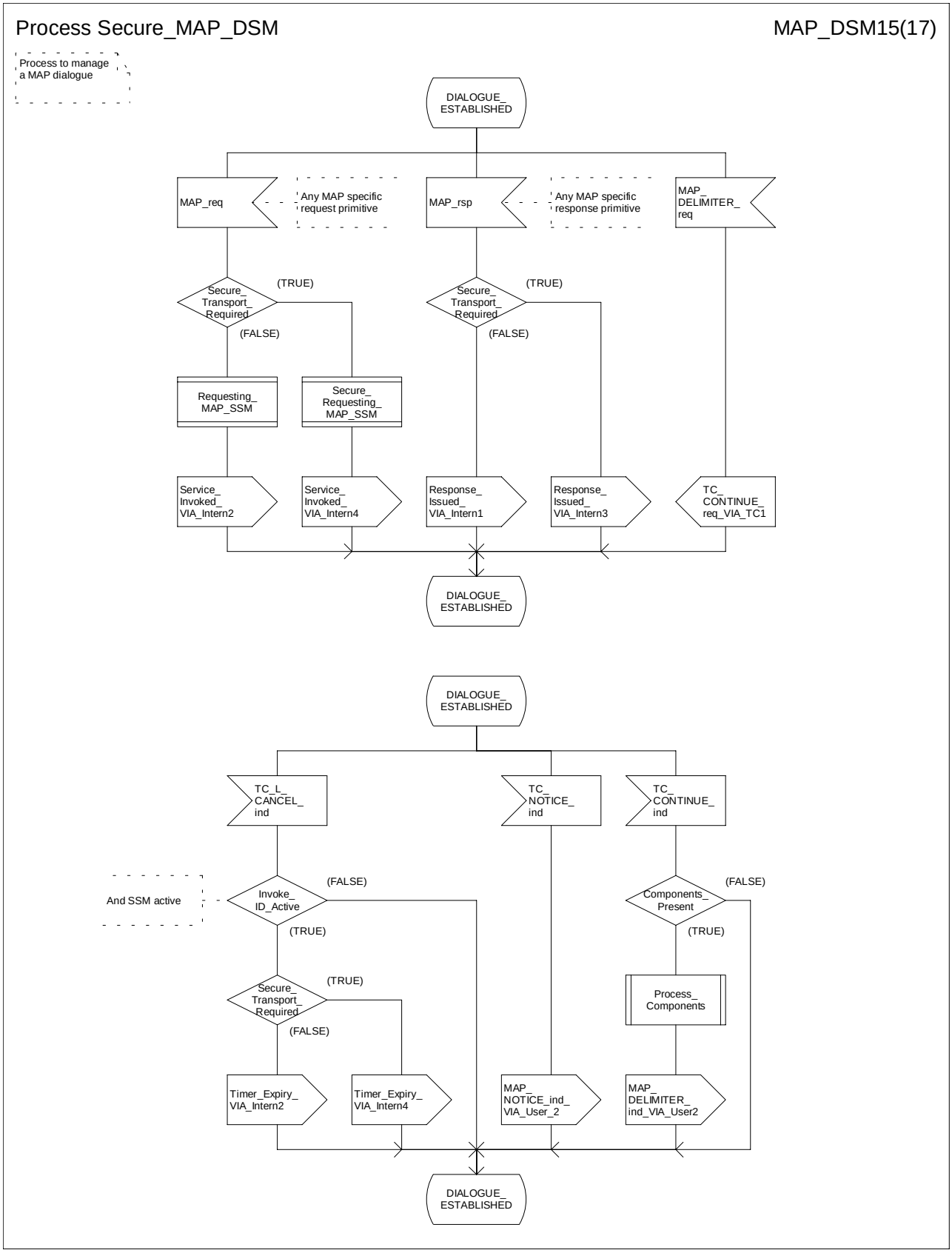

Figure 15.6/3o: Process Secure_MAP_DSM (sheet 15)

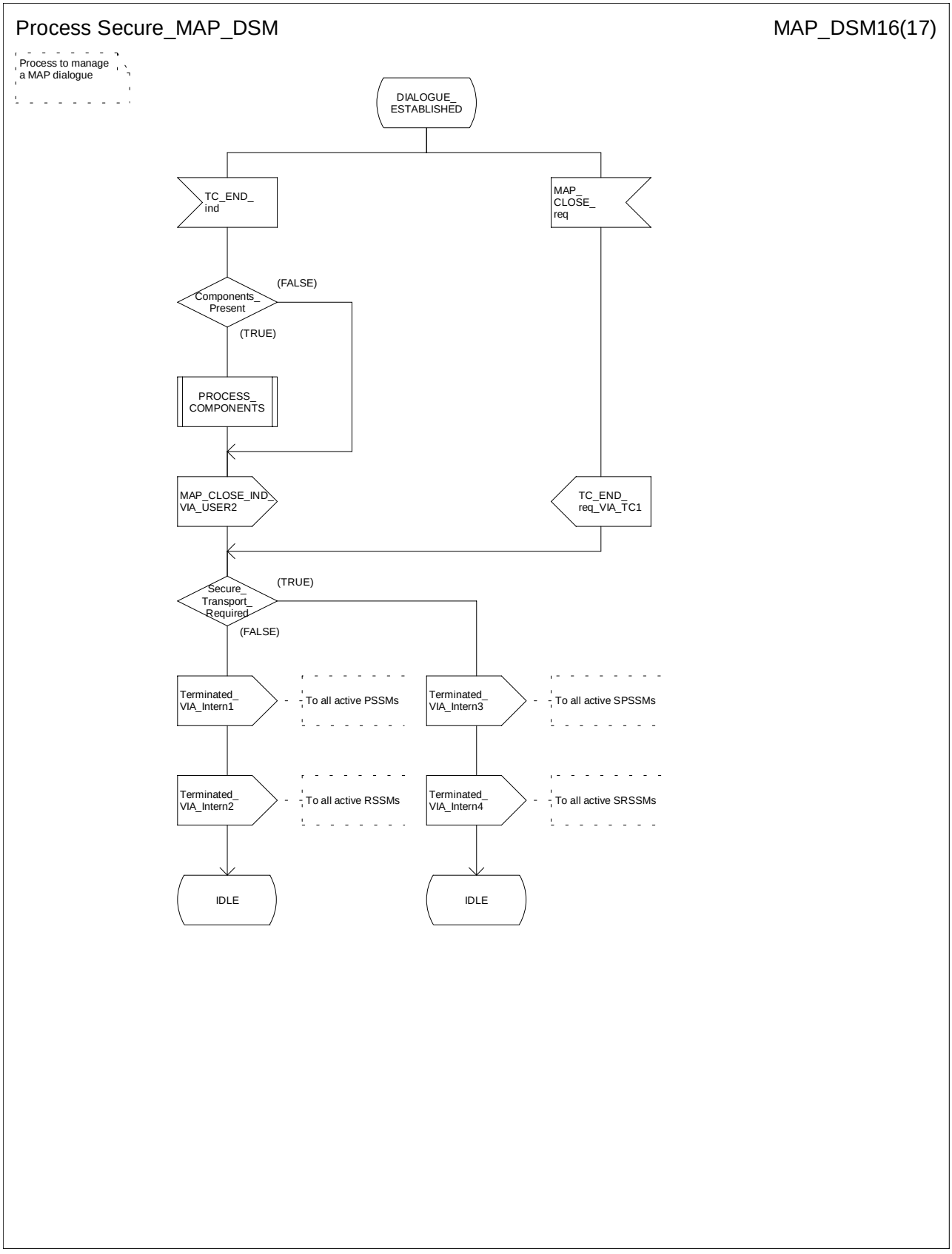

Figure 15.6/3p: Process Secure_MAP_DSM (sheet 16)

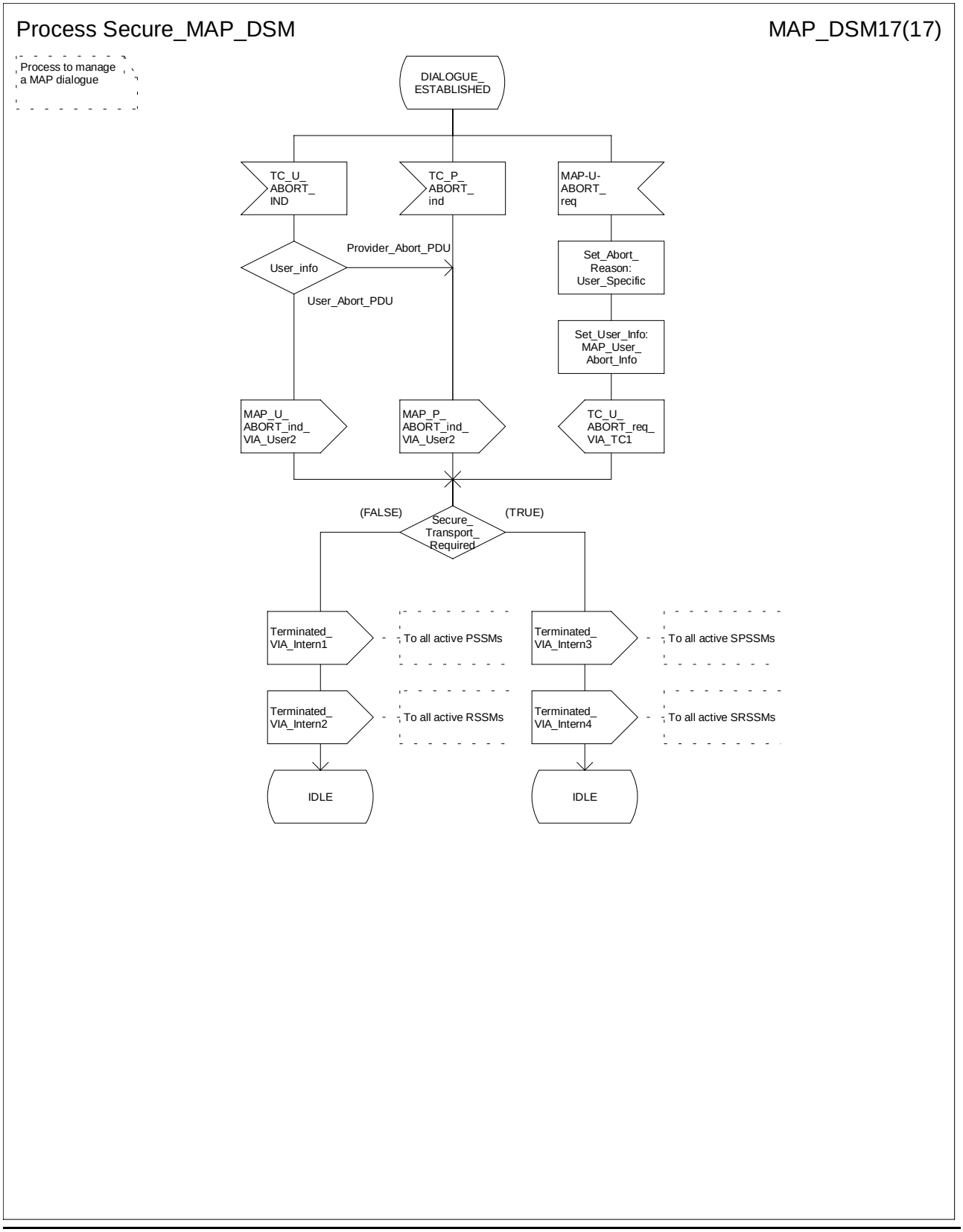

Figure 15.6/3q: Process Secure_MAP_DSM (sheet 17)

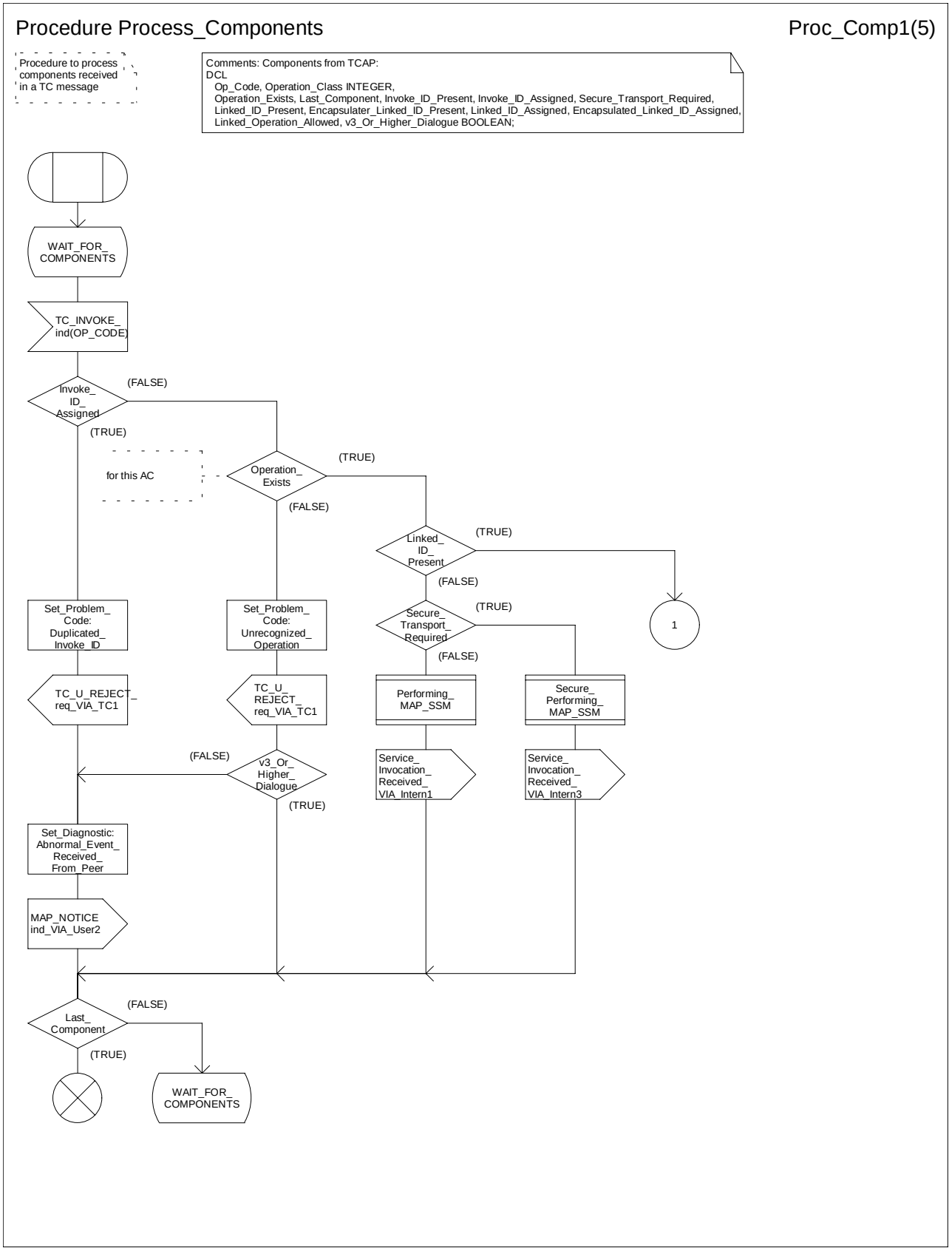

Figure 15.6/4a: Procedure Process_Components (sheet 1)

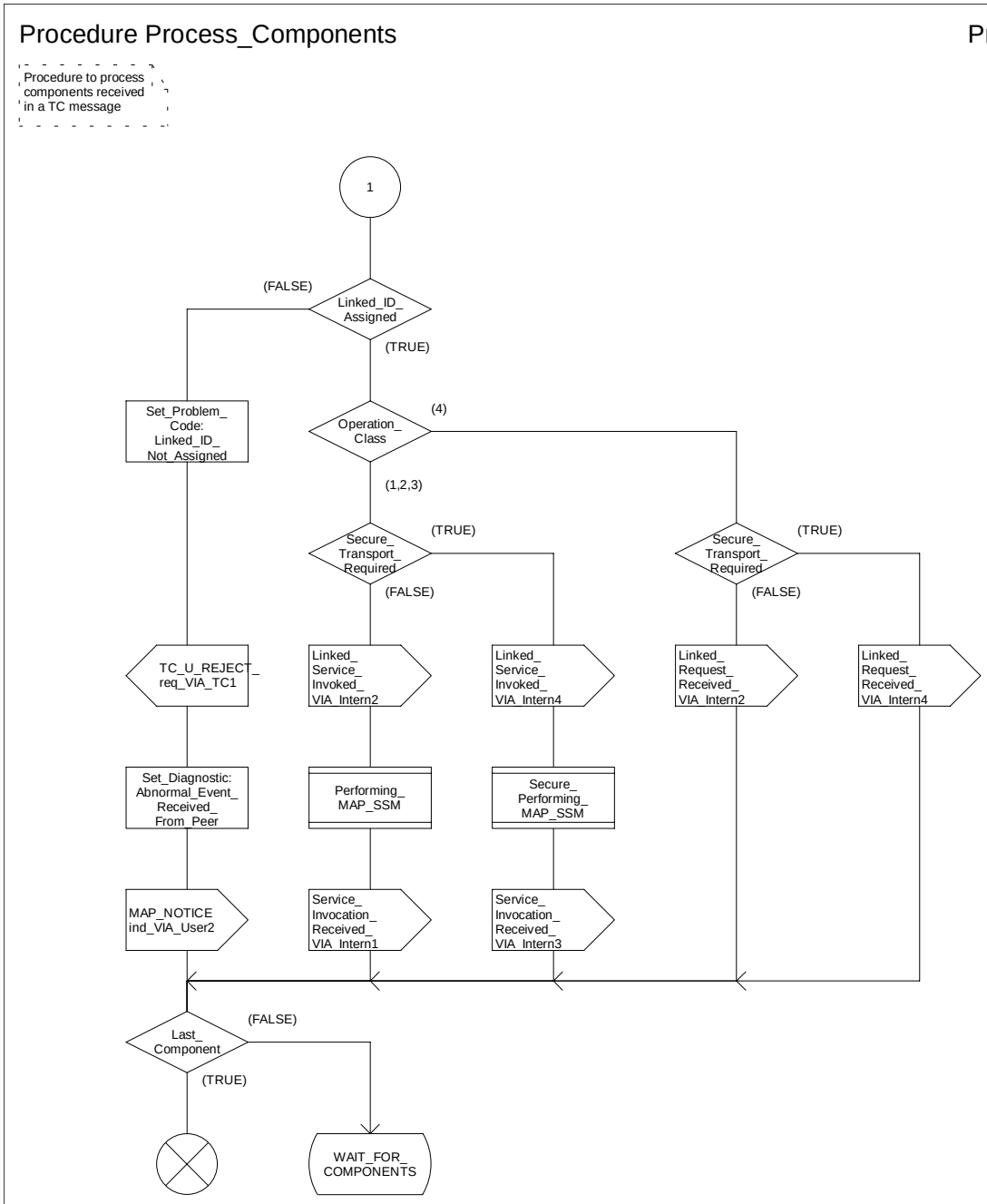

Figure 15.6/4b: Procedure Process_Components (sheet 2)

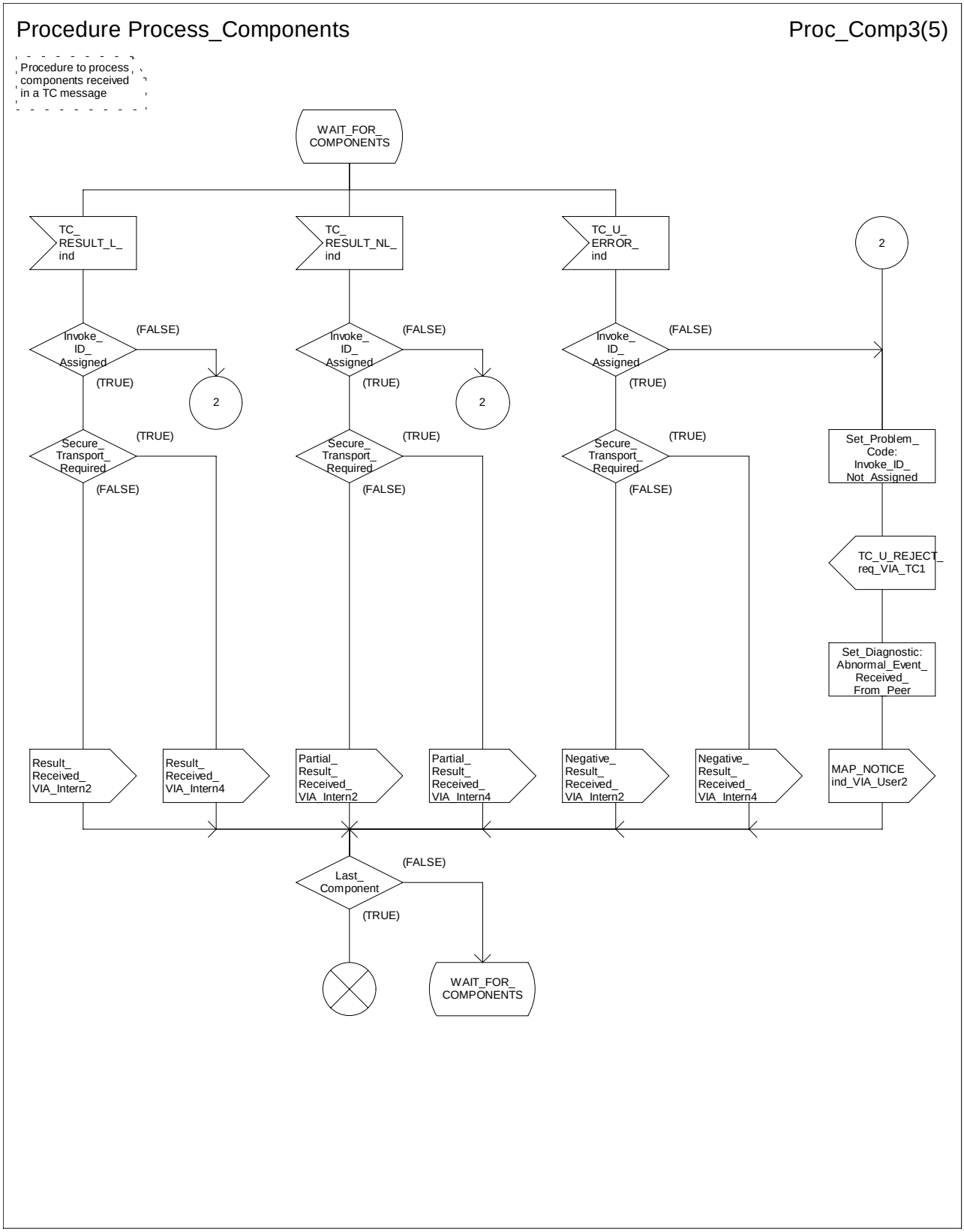

Figure 15.6/4c: Procedure Process_Components (sheet 3)

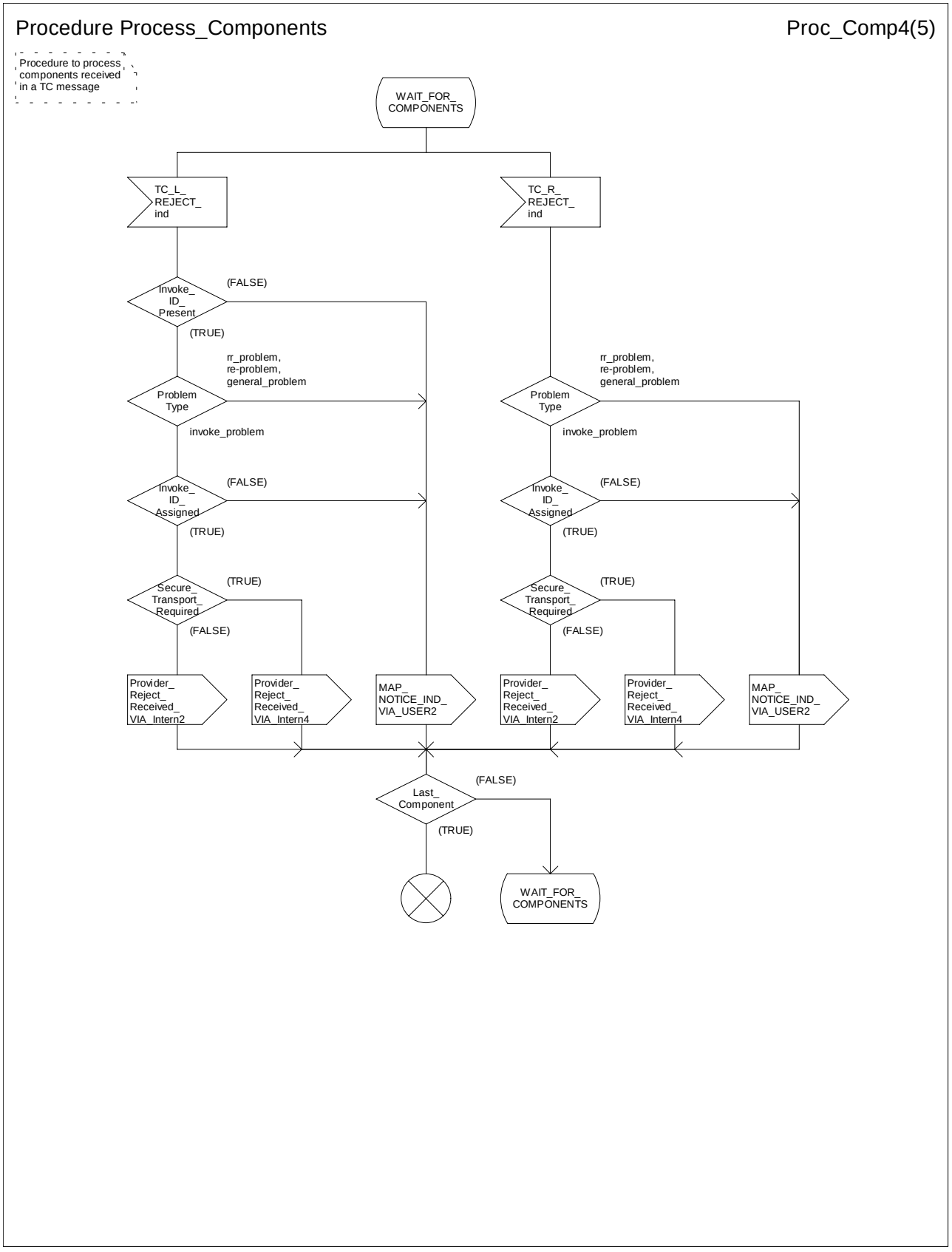

Figure 15.6/4d: Procedure Process_Components (sheet 4)

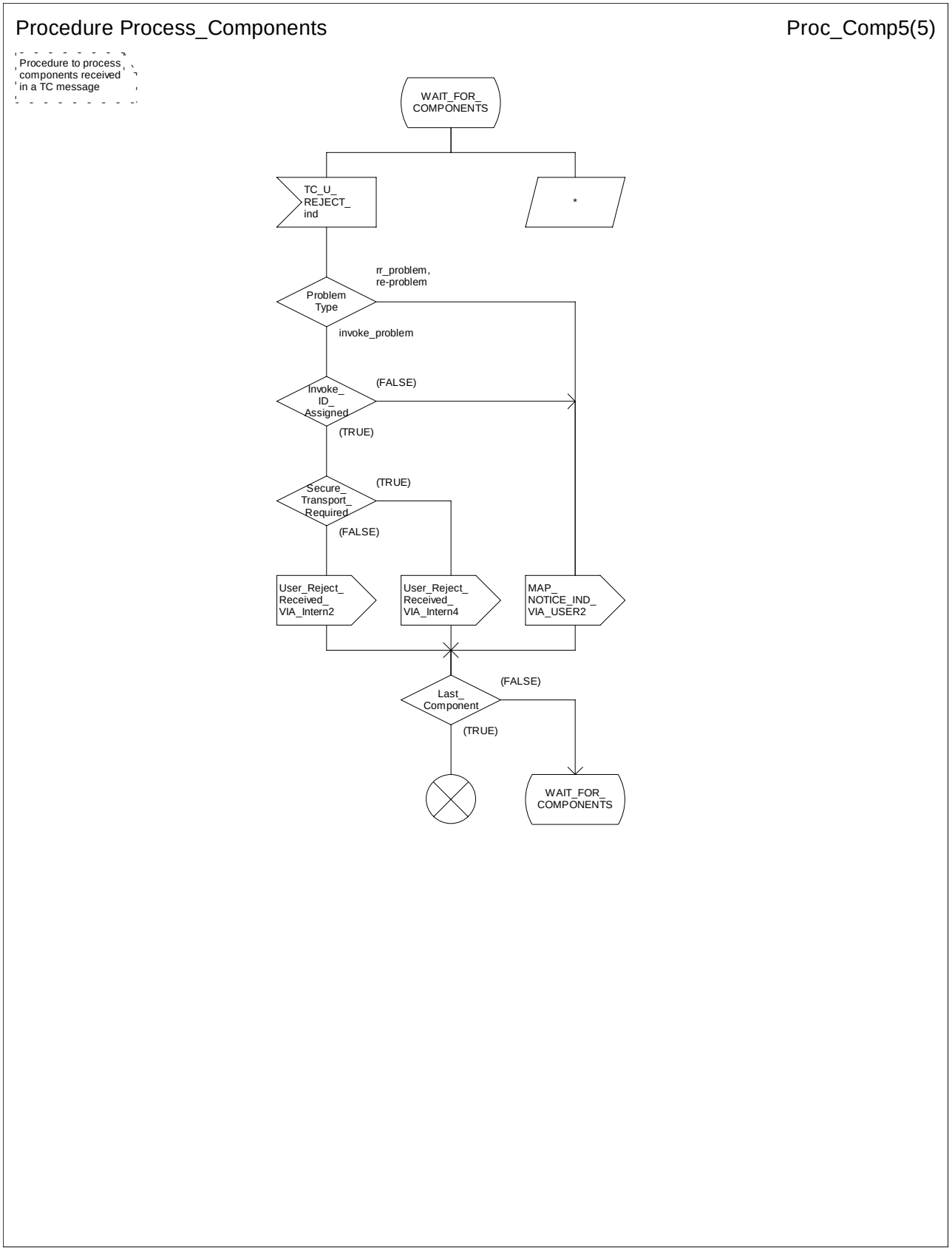

Figure 15.6/4e: Procedure Process_Components (sheet 5)

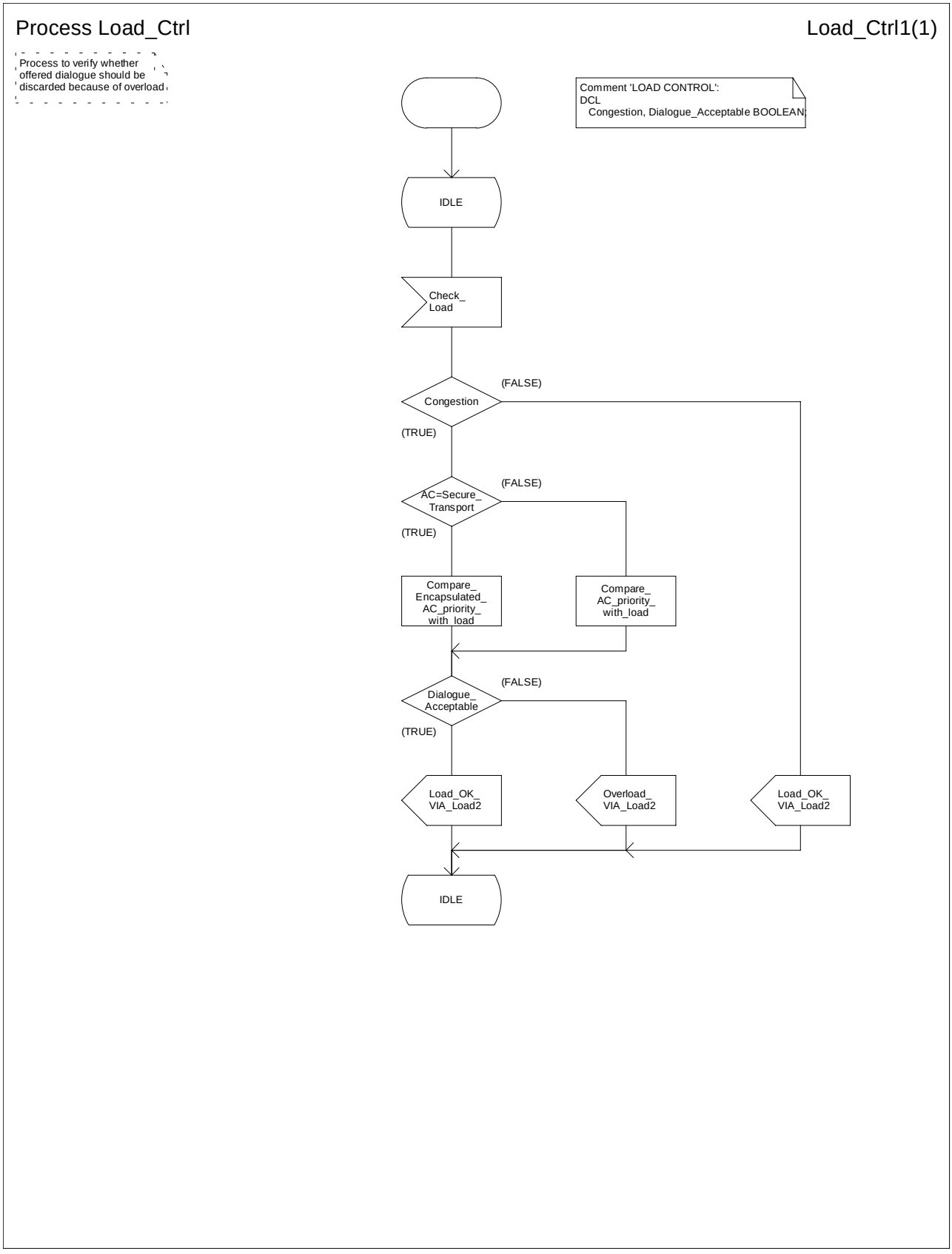

Figure 15.6/5: Process Load_Ctrl

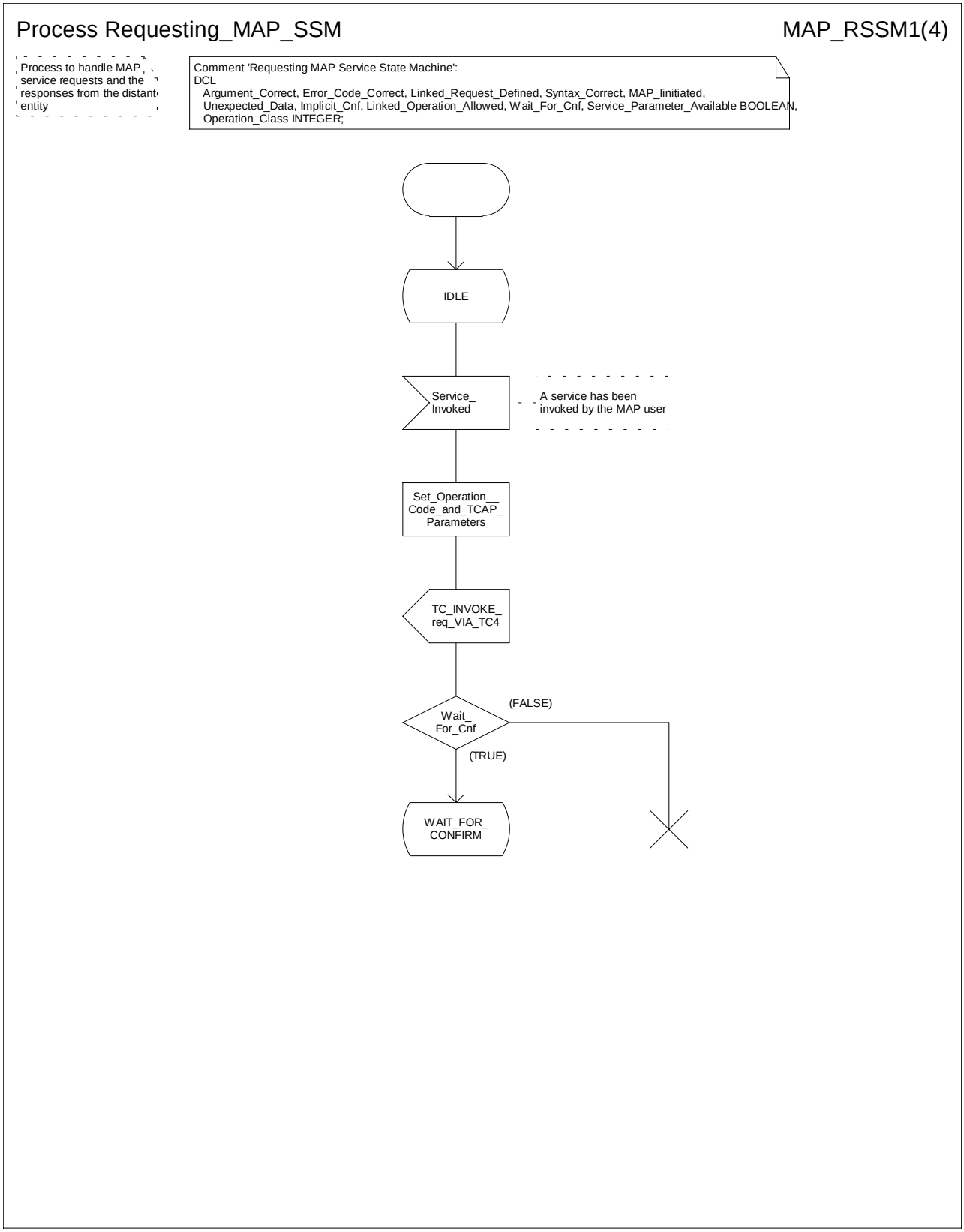

Figure 15.6/6a: Process Requesting_MAP_SSM (sheet 1)

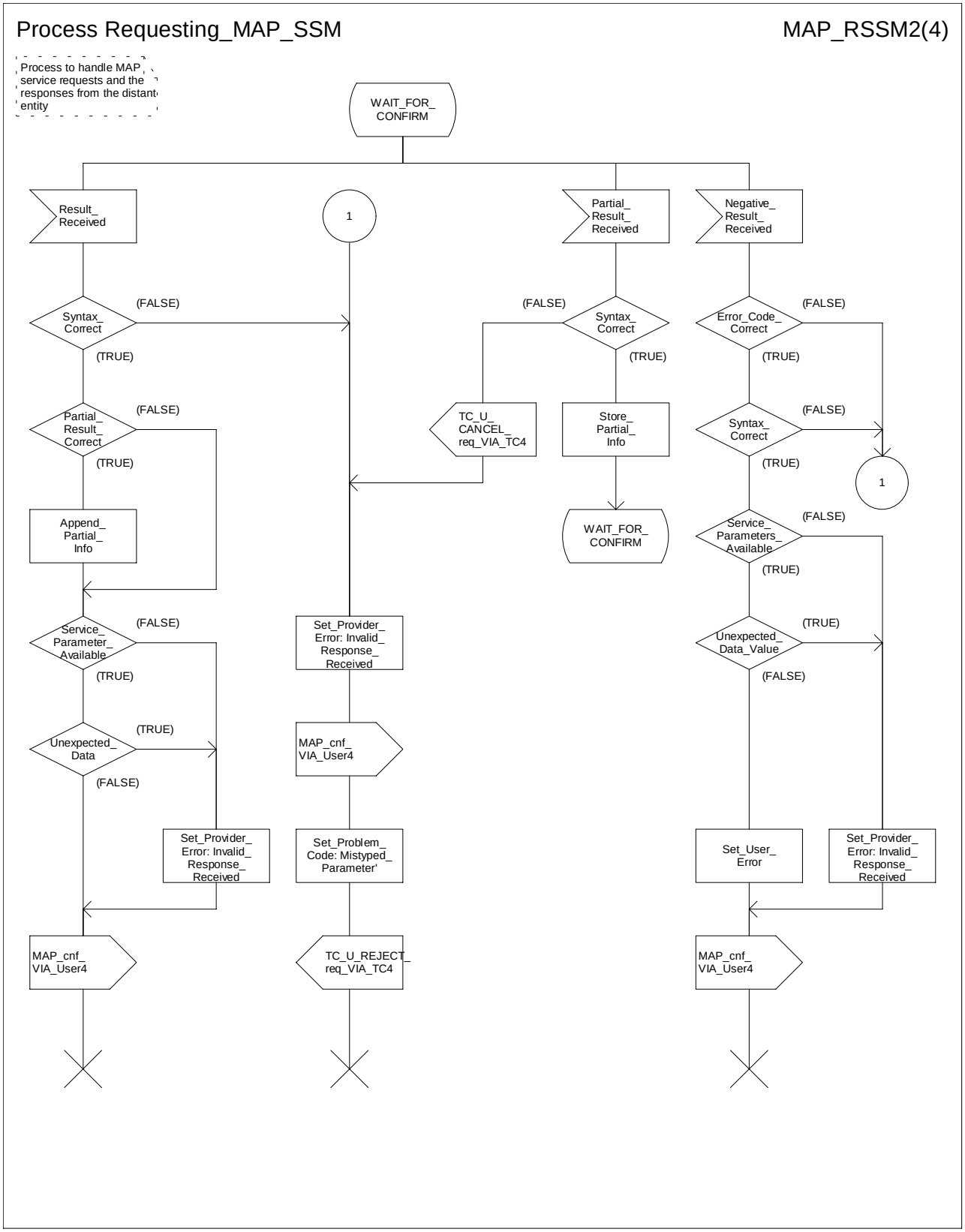

Figure 15.6/6b: Process Requesting_MAP_SSM (sheet 2)

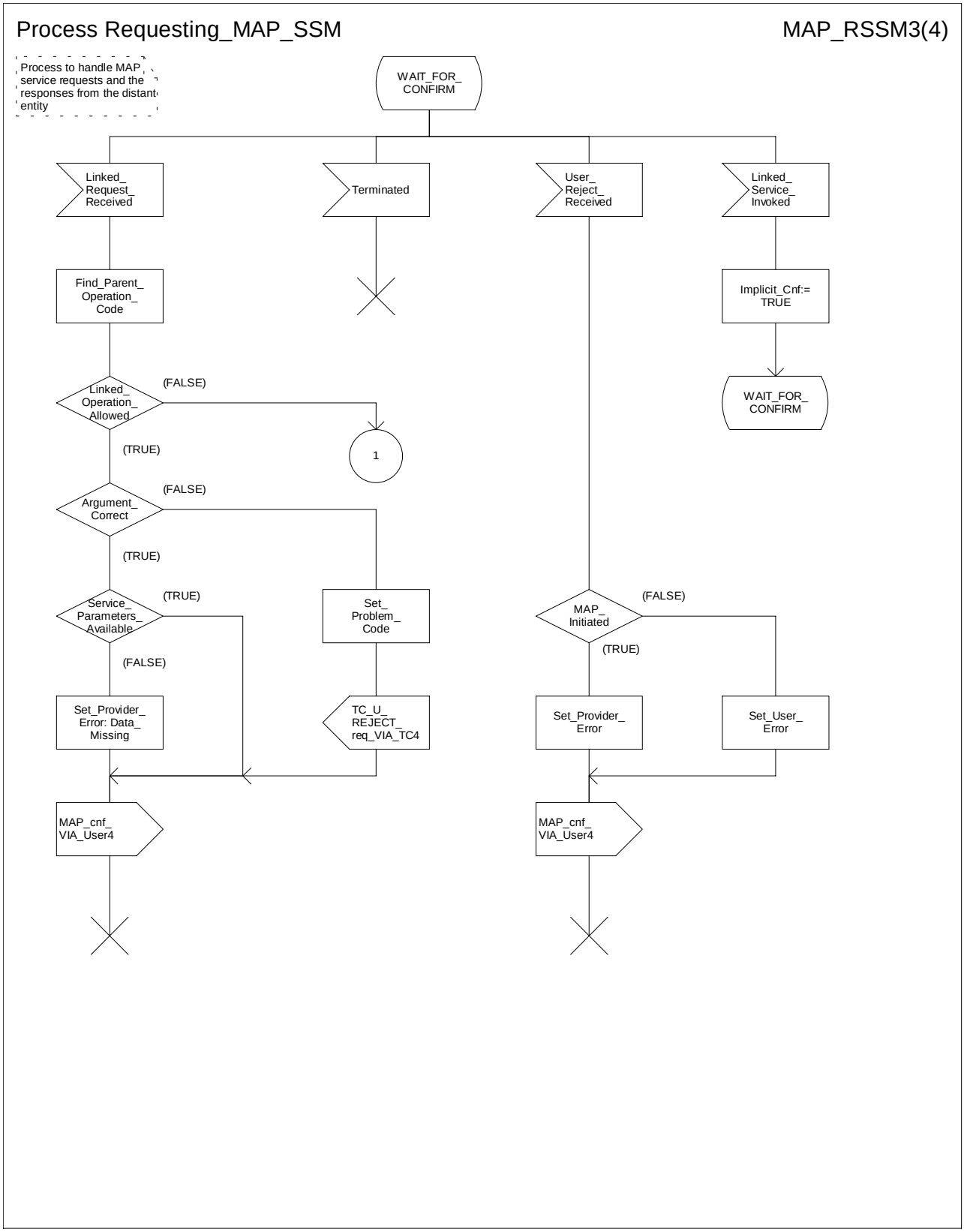

Figure 15.6/6c: Process Requesting_MAP_SSM (sheet 3)

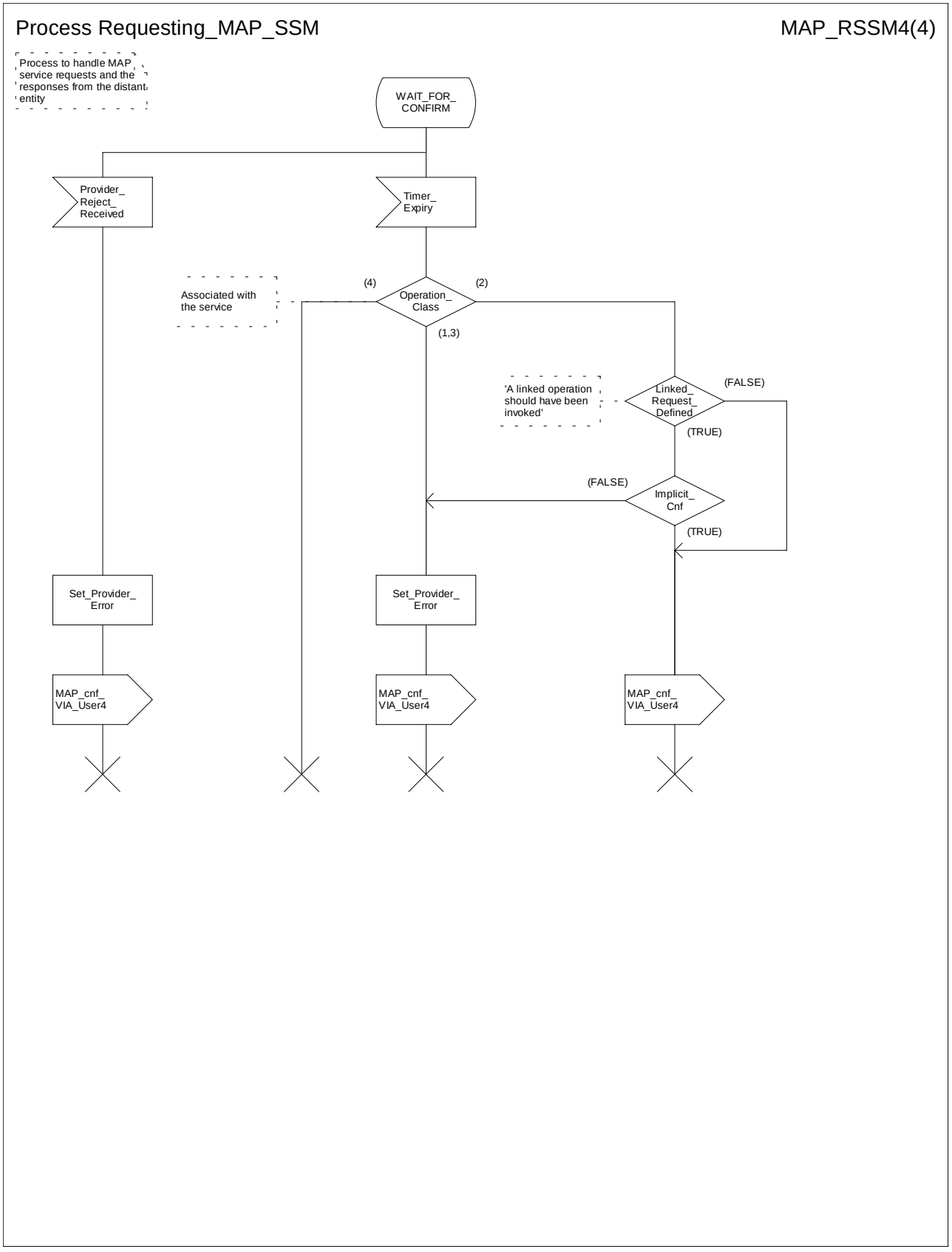

Figure 15.6/6d: Process Requesting_MAP_SSM (sheet 4)

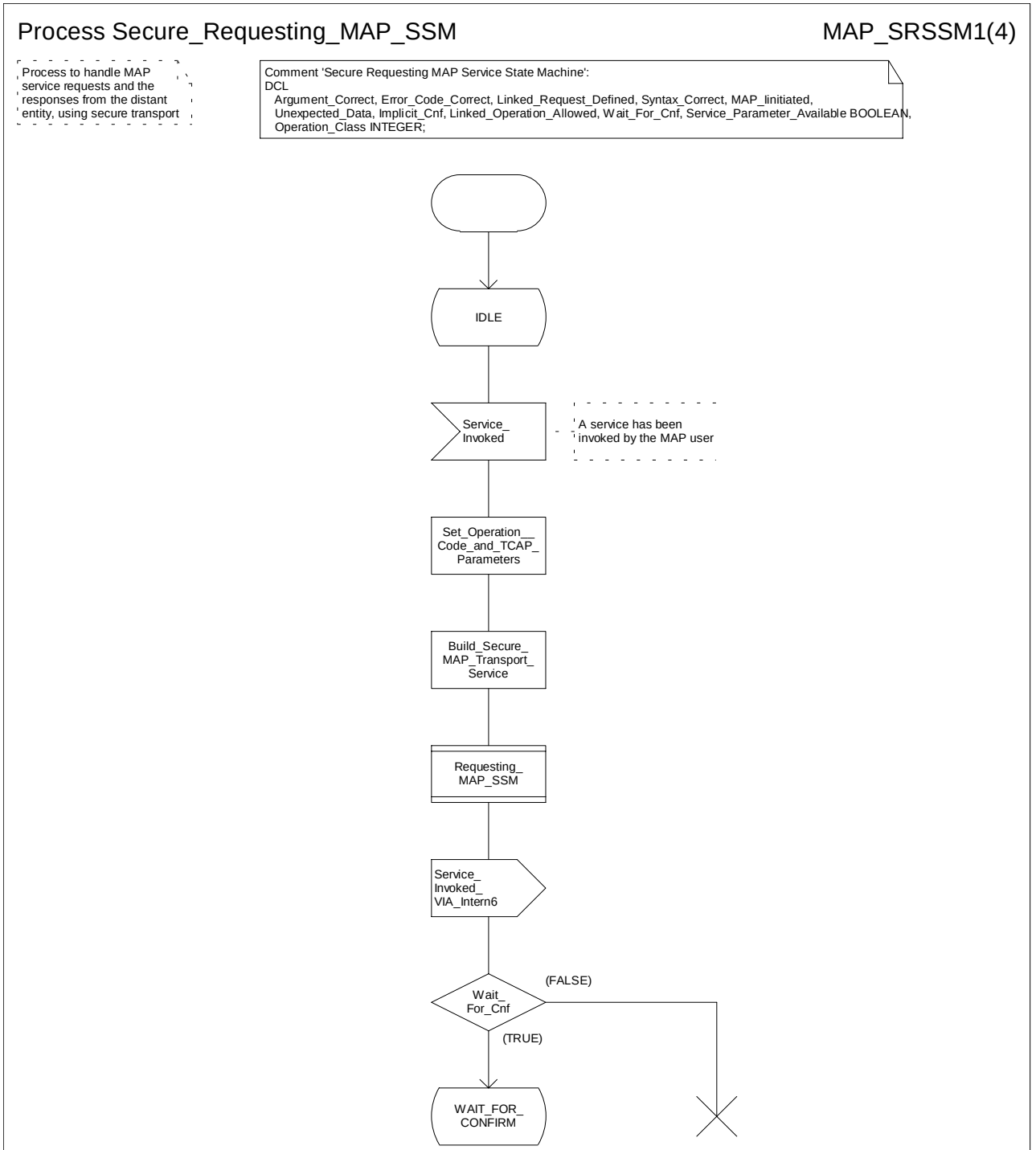

Figure 15.6/7a: Process Secure_Requesting_MAP_SSM (sheet 1)

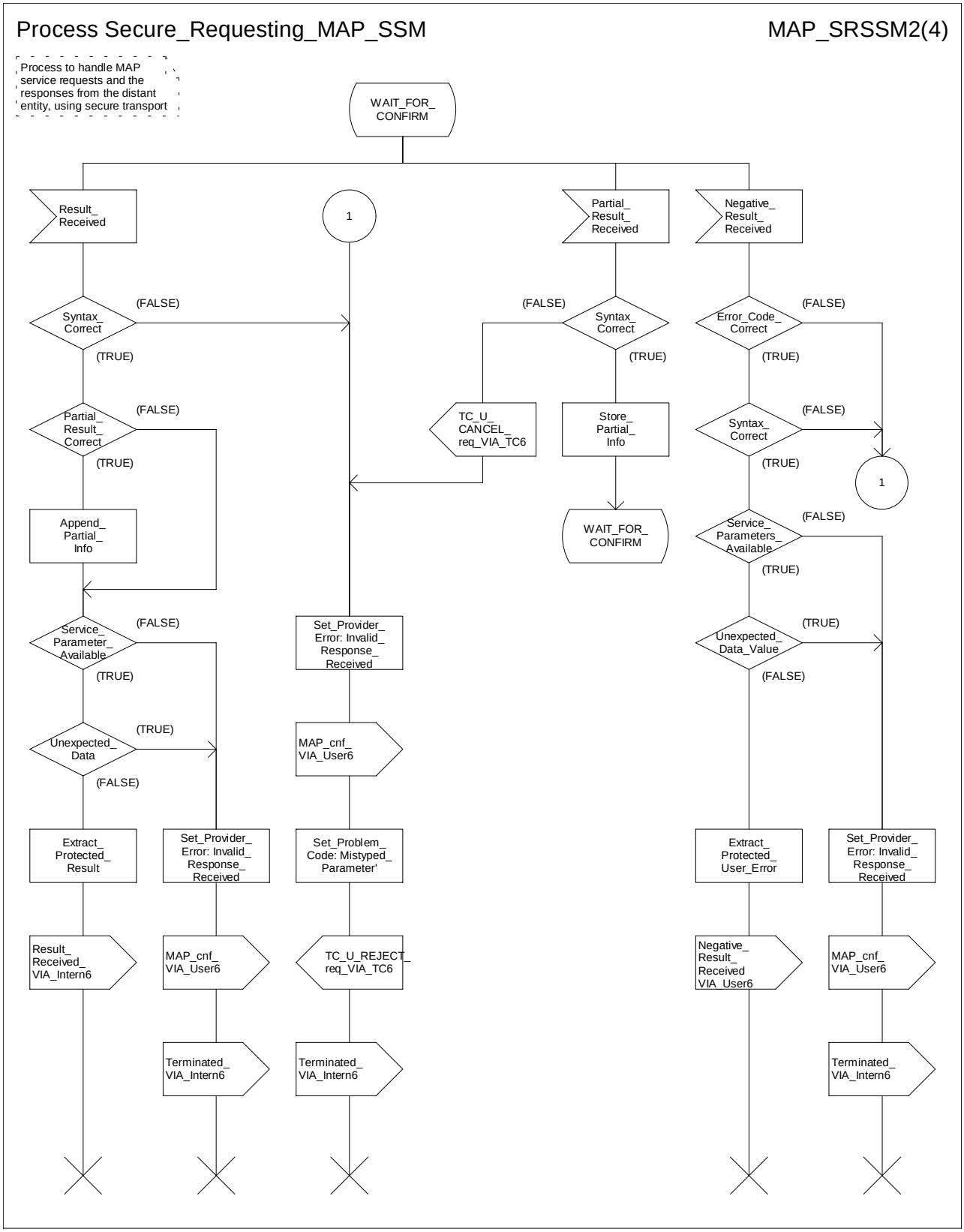

Figure 15.6/7b: Process Secure_Requesting_MAP_SSM (sheet 2)

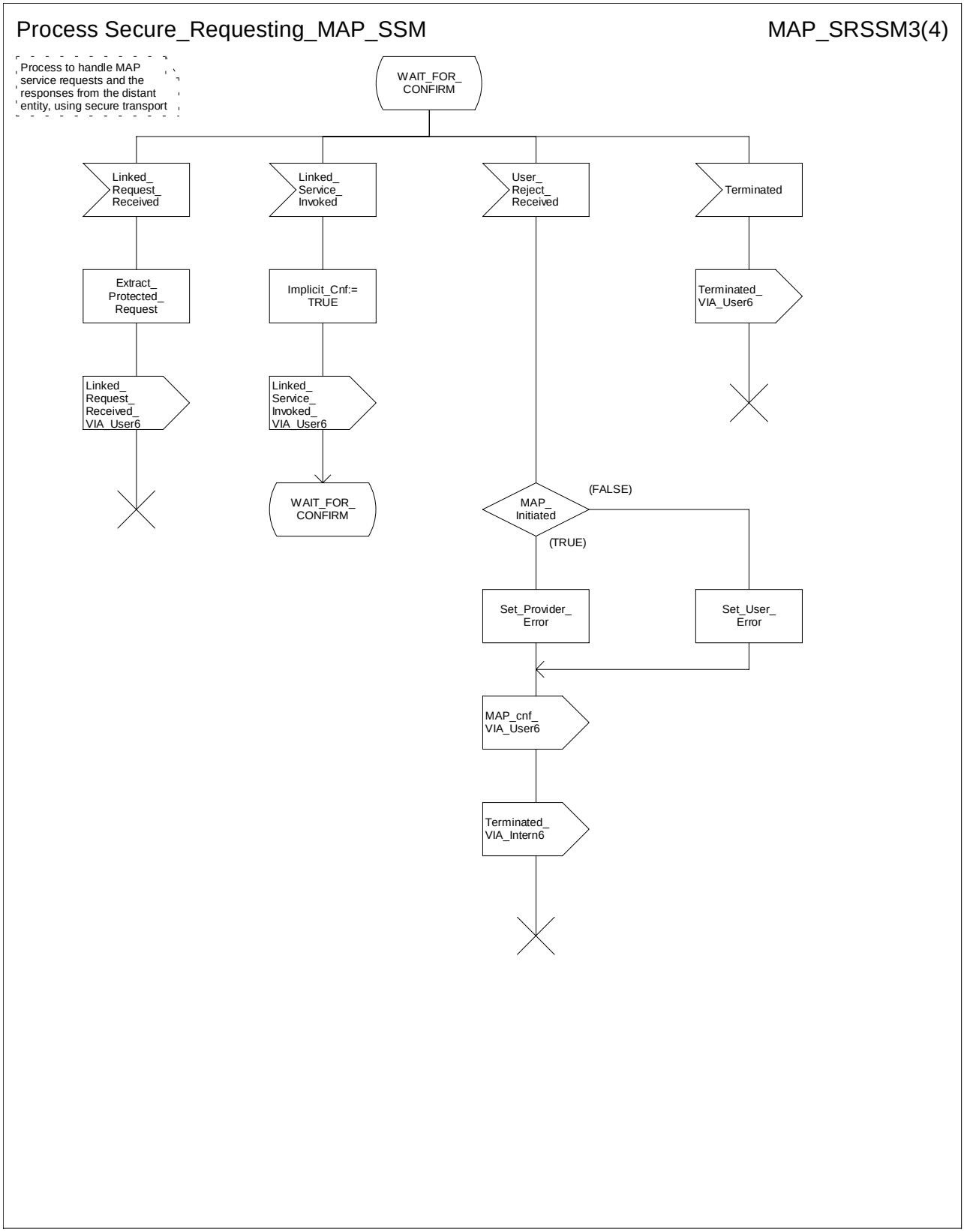

Figure 15.6/7c: Process Secure_Requesting_MAP_SSM (sheet 3)

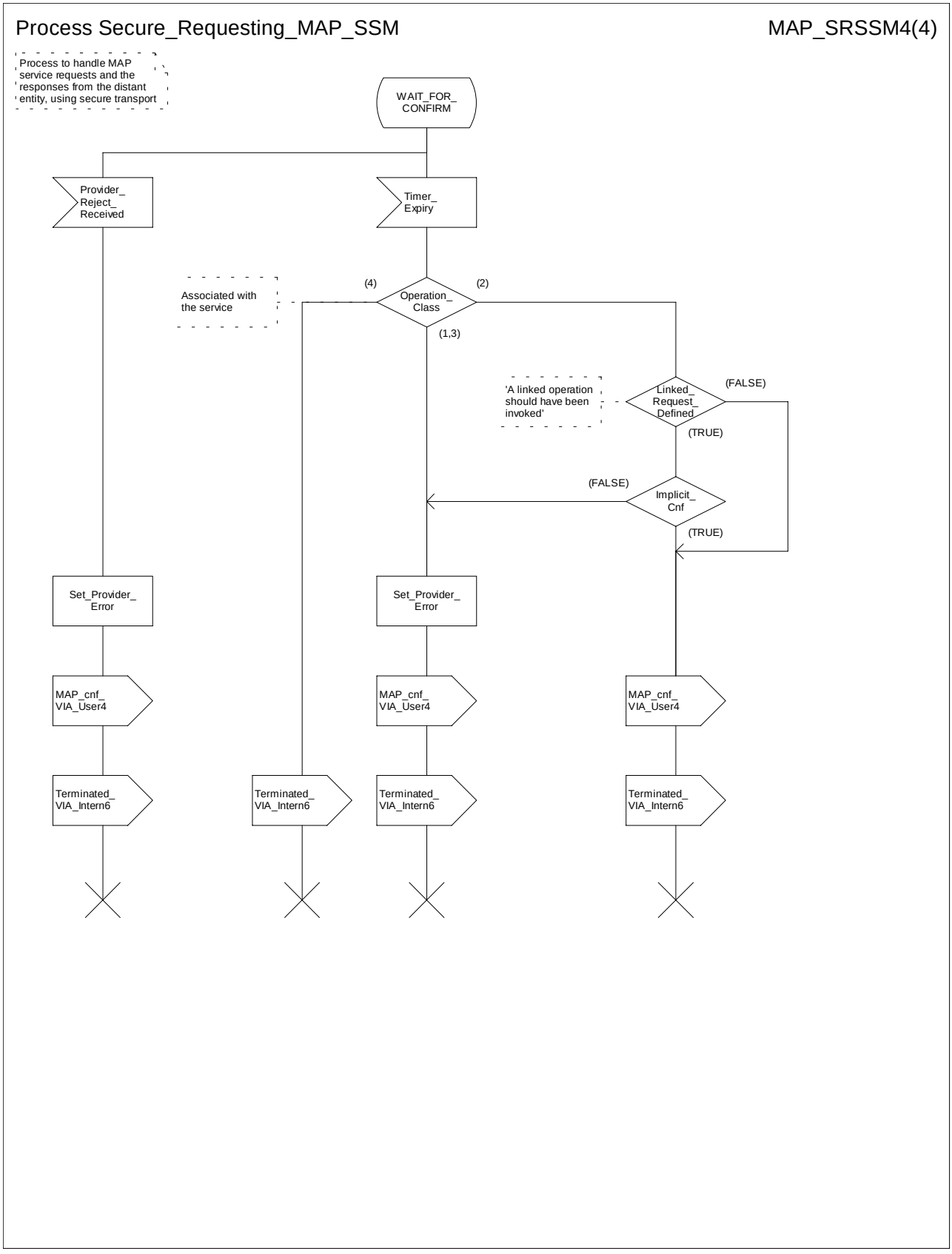

Figure 15.6/7d: Process Secure_Requesting_MAP_SSM (sheet 4)

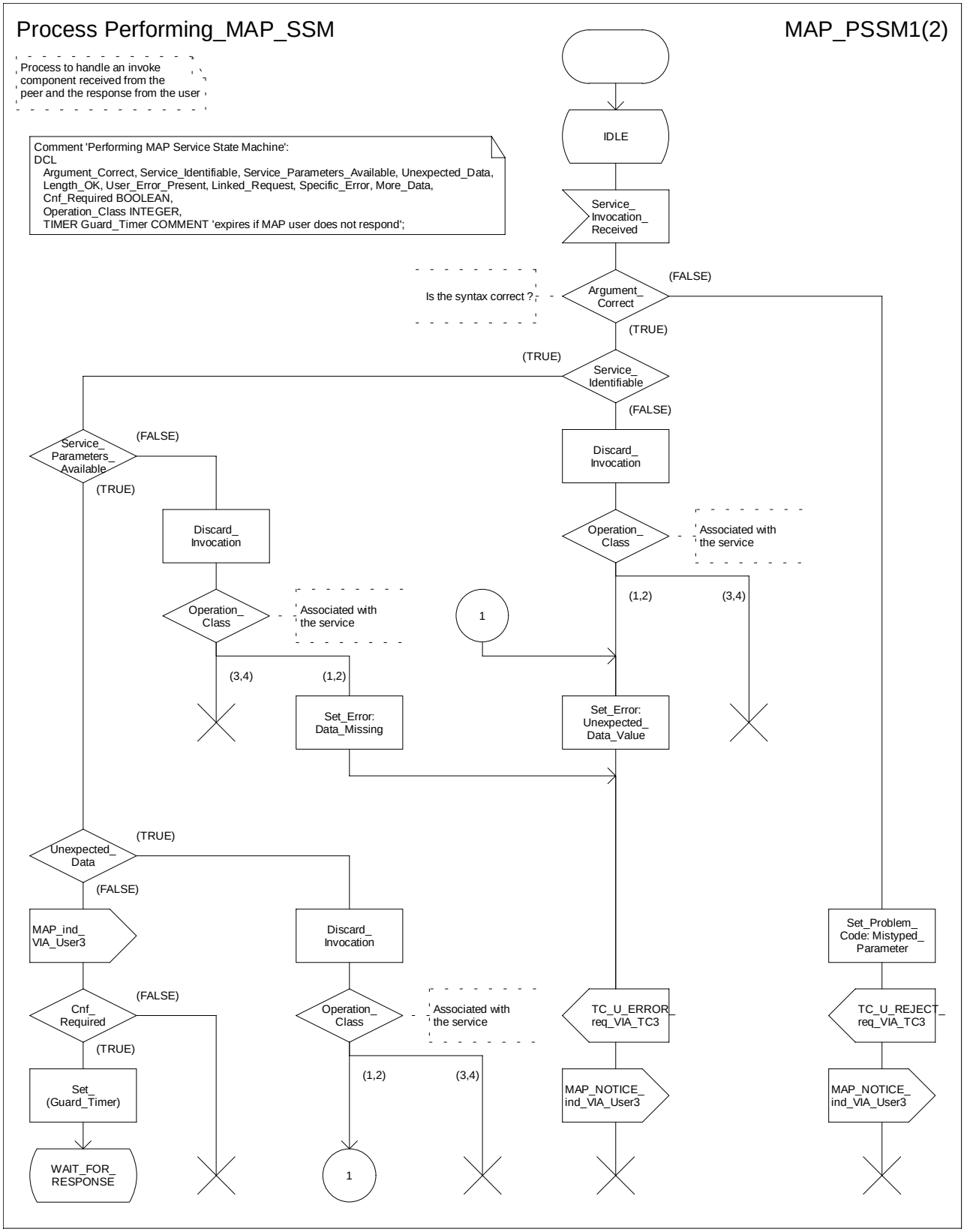

Figure 15.6/8a: Process Performing_MAP_SSM (sheet 1)

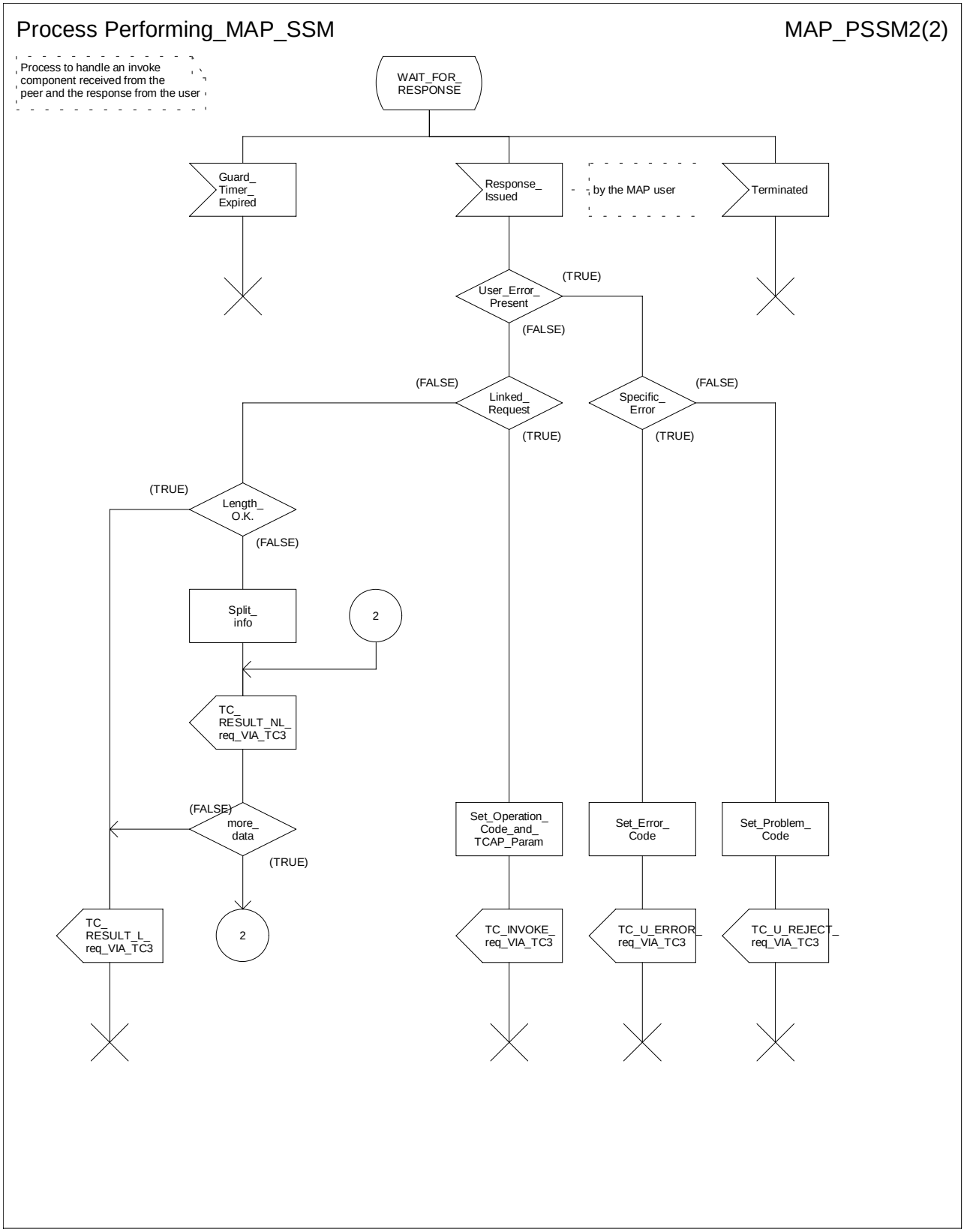

Figure 15.6/8b: Process Performing_MAP_SSM (sheet 2)

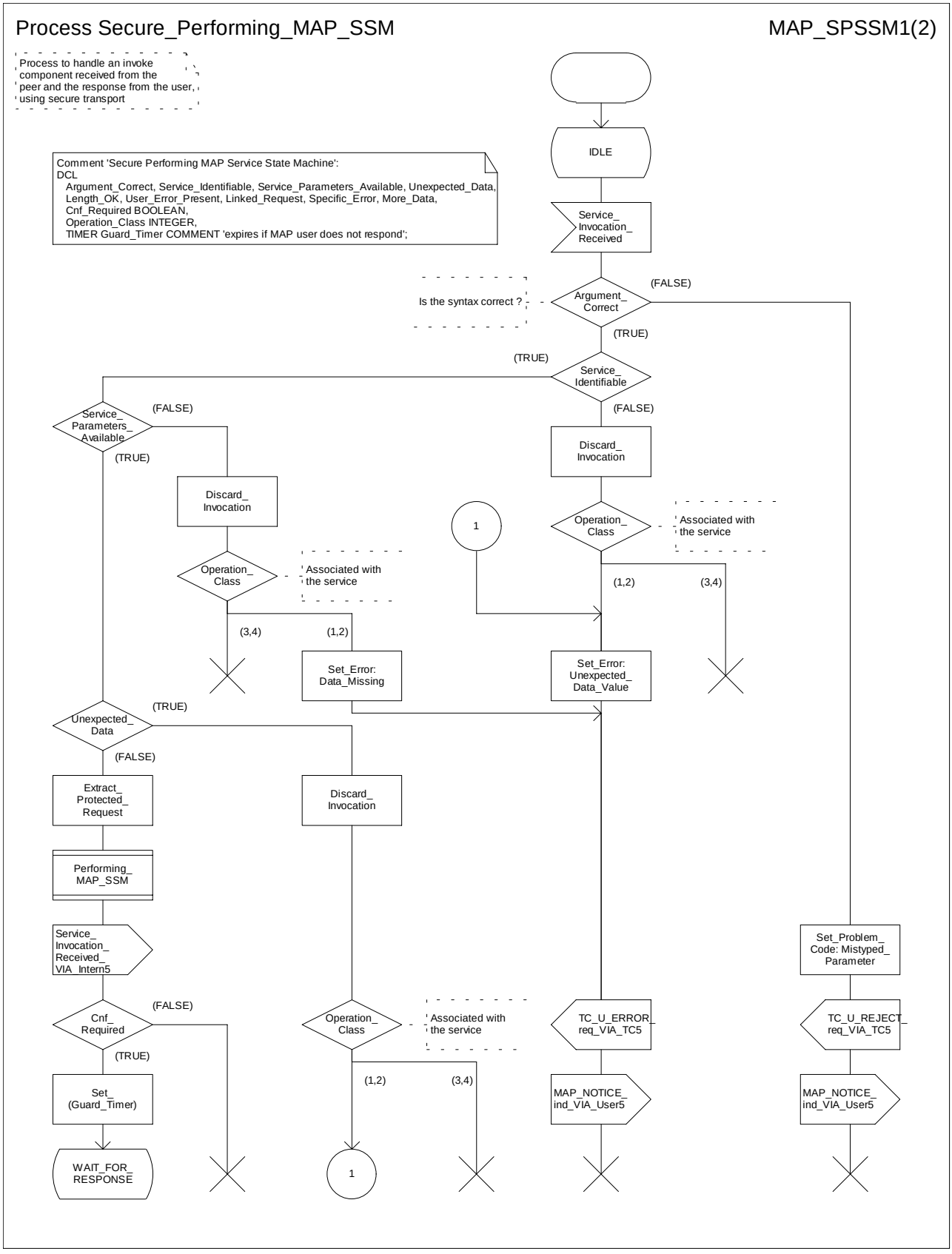

Figure 15.6/9a: Process Secure_Performing_MAP_SSM (sheet 1)

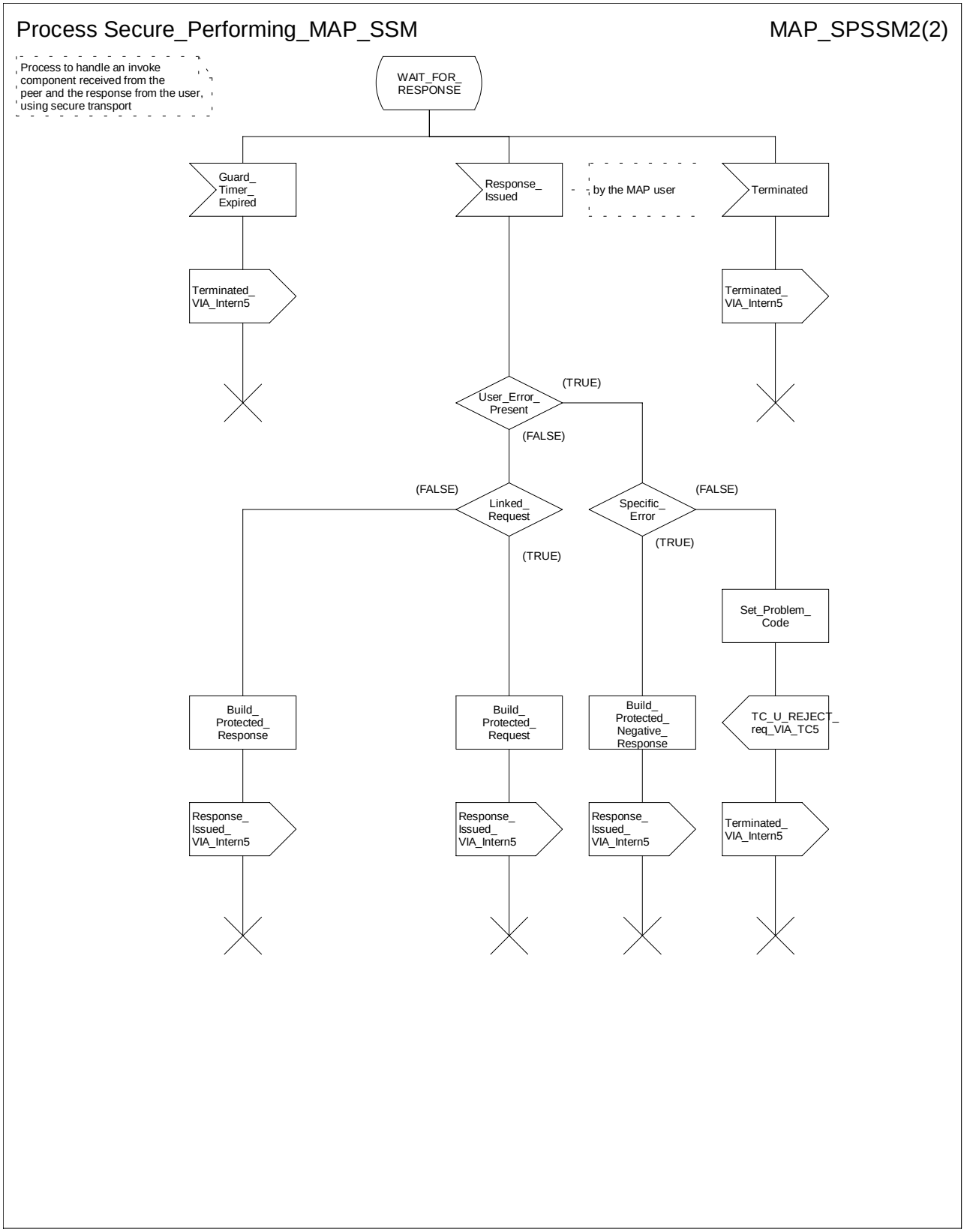

Figure 15.6/9b: Process Secure_Performing_MAP_SSM (sheet 2)

****** Next modified section ******

16.2.2.4 Operation

When mapping a request primitive on to a Remote Operations PDU (invoke), the MAP PM shall set the operation code according to the mapping described in table 16.2/1.

When mapping a response primitive on to a Remote Operations service, the MAP PM shall set the operation code of the TC-RESULT-L/NL primitive (if required) to the same value as the one received at invocation time.

Table 16.2/1: Mapping of MAP specific services on to MAP operations

MAP-SERVICE	operation
MAP-ACTIVATE-SS	activateSS
MAP-ACTIVATE-TRACE-MODE	activateTraceMode
MAP-ALERT-SERVICE-CENTRE	alertServiceCentre
MAP-ANY-TIME-INTERROGATION	anyTimeInterrogaton
MAP-ANY-TIME-MODIFICATION	anyTimeModification
MAP-ANY-TIME-SUBSCRIPTION-INTERROGATION	anyTimeSubscriptionInterrogaton
MAP-CANCEL-LOCATION	cancelLocation
MAP-CHECK-IMEI	checkIMEI
MAP-DEACTIVATE-SS	deactivateSS
MAP-DEACTIVATE-TRACE-MODE	deactivateTraceMode
MAP-DELETE-SUBSCRIBER-DATA	deleteSubscriberData
MAP-ERASE-CC-ENTRY	eraseCC-Entry
MAP-ERASE-SS	eraseSS
MAP-FAILURE-REPORT	failureReport
MAP-FORWARD-ACCESS-SIGNALLING	forwardAccessSignalling
MAP-FORWARD-CHECK-SS-INDICATION	forwardCheckSsIndication
MAP-FORWARD-GROUP-CALL-SIGNALLING	forwardGroupCallSignalling
MAP-MT-FORWARD-SHORT-MESSAGE	mt-forwardSM
MAP-MO-FORWARD-SHORT-MESSAGE	mo-forwardSM
MAP-GET-PASSWORD	getPassword
MAP-INFORM-SERVICE-CENTRE	informServiceCentre
MAP-INSERT-SUBSCRIBER-DATA	insertSubscriberData
MAP-INTERROGATE-SS	interrogateSs
MAP-IST-ALERT	istAlert
MAP-IST-COMMAND	istCommand
MAP-NOTE-MS-PRESENT-FOR-GPRS	noteMsPresentForGprs
MAP-NOTE-SUBSCRIBER-DATA-MODIFIED	noteSubscriberDataModified
MAP-PREPARE-GROUP-CALL	prepareGroupCall
MAP-PREPARE-HANDOVER	prepareHandover
MAP-PREPARE-SUBSEQUENT-HANDOVER	prepareSubsequentHandover
MAP-PROCESS-ACCESS-SIGNALLING	processAccessSignalling
MAP-PROCESS-GROUP-CALL-SIGNALLING	processGroupCallSignalling
MAP-PROCESS-UNSTRUCTURED-SS-REQUEST	processUnstructuredSS-Request
MAP-PROVIDE-ROAMING-NUMBER	provideRoamingNumber
MAP-PROVIDE-SIWFS-NUMBER	provideSIWFSNumber
MAP-PROVIDE-SUBSCRIBER-LOCATION	provideSubscriberLocation
MAP-PROVIDE-SUBSCRIBER-INFO	provideSubscriberInfo
MAP-PURGE-MS	purgeMS
MAP-READY-FOR-SM	readyForSM
MAP-REGISTER-CC-ENTRY	registerCC-Entry
MAP-REGISTER-PASSWORD	registerPassword
MAP-REGISTER-SS	registerSS
MAP-REMOTE-USER-FREE	remoteUserFree
MAP-REPORT-SM-DELIVERY-STATUS	reportSmDeliveryStatus
MAP-RESET	reset
MAP-RESTORE-DATA	restoreData
MAP-SECURE-TRANSPORT-CLASS-1	secureTransportClass1
MAP-SECURE-TRANSPORT-CLASS-2	secureTransportClass2
MAP-SECURE-TRANSPORT-CLASS-3	secureTransportClass3
MAP-SECURE-TRANSPORT-CLASS-4	secureTransportClass4
MAP-SEND_GROUP-CALL_END_SIGNAL	sendGroupCallEndSignal
MAP-SEND-END-SIGNAL	sendEndSignal
MAP-SEND-AUTHENTICATION-INFO	sendAuthenticationInfo
MAP-SEND-IMSI	sendIMSI
MAP-SEND-IDENTIFICATION	sendIdentification
MAP-SEND-ROUTING-INFO-FOR-SM	sendRoutingInfoForSM
MAP-SEND-ROUTING-INFO-FOR-GPRS	sendRoutingInfoForGprs
MAP-SEND-ROUTING-INFO-FOR-LCS	sendRoutingInfoForLCS
MAP-SEND-ROUTING-INFORMATION	sendRoutingInfo
MAP-SET-REPORTING-STATE	setReportingState

MAP-SIWFS-SIGNALLING-MODIFY	SIWFSSignallingModify
MAP-STATUS-REPORT	statusReport
MAP-SUBSCRIBER-LOCATION-REPORT	subscriberLocationReport
MAP-SUPPLEMENTARY-SERVICE-INVOCATION-NOTIFICATION	ss-Invocation-Notification
MAP-UNSTRUCTURED-SS-NOTIFY	unstructuredSS-Notify
MAP-UNSTRUCTURED-SS-REQUEST	unstructuredSS-Request
MAP-UPDATE-GPRS-LOCATION	updateGprsLocation
MAP-UPDATE-LOCATION	updateLocation
MAP-NOTE-MM-EVENT	NoteMM-Event

****** Next modified section ******

17.1.5 Structure of the Abstract Syntax of MAP

For each MAP parameter which has to be transferred by a MAP Protocol Data Unit (MAP message), there is a PDU field (an ASN.1 NamedType) whose ASN.1 identifier has the same name as the corresponding parameter, except for the differences required by the ASN.1 notation (blanks between words are removed or replaced by hyphen, the first letter of the first word is lower-case and the first letter of the following words are capitalized, e.g. "no reply condition time" is mapped to "noReplyConditionTime"). Additionally some words may be abbreviated as follows:

bs basic service

ch call handling

cug closed user group

ho handover

ic incoming call

id identity

info information

mm mobility management

lcs location services

ms mobile service

oc outgoing call

om operation & maintenance

pw Password

sm short message service

ss supplementary service

st secure transport

The MAP protocol is composed of several ASN.1 modules dealing with either operations, errors, data types, and, if applicable, split into those dealing with mobile services, call handling services, supplementary services and short message services. For operations and errors no values are assigned, but only the operation and error types in order to allow use of the defined types also by other protocols (e.g. TS GSM 04.80). The values (operation codes and error codes) are defined in a separate module. The ASN.1 source lines are preceded by line-numbers at the left margin in order to enable the usage of the cross-reference in annex A.

The module containing the definition of the operation packages for MAP is:

1. MAP-OperationPackages.

The module containing the definition of the application contexts for MAP is:

2. MAP-ApplicationContexts.

The module containing the data types for the Abstract Syntax to be used for TCAPMessages.DialoguePortion for MAP is:

3. MAP-DialogueInformation.

The module containing the operation codes and error codes for MAP is:

4. MAP-Protocol.

The modules containing all operation type definitions for MAP are:

5. MAP-MobileServiceOperations;
6. MAP-OperationAndMaintenanceOperations;
7. MAP-CallHandlingOperations;
8. MAP-SupplementaryServiceOperations;
9. MAP-ShortMessageServiceOperations;
10. MAP-Group-Call-Operations;~~.~~
11. MAP-LocationServiceOperations;
12. MAP-SecureTransportOperations.

The module containing all error type definitions for MAP is:

13~~2~~. MAP-Errors.

Modules containing all data type definitions for MAP are:

14~~3~~. MAP-MS-DataTypes;
15~~4~~. MAP-OM-DataTypes;
16~~5~~. MAP-CH-DataTypes;
17~~6~~. MAP-SS-DataTypes;
18~~7~~. MAP-SS-Code;
19~~8~~. MAP-SM-DataTypes;
20~~19~~. MAP-ER-DataTypes;
21~~0~~. MAP-CommonDataTypes;
22~~1~~. MAP-TS-Code;
23~~2~~. MAP-BS-Code;
24~~3~~. MAP-ExtensionDataTypes;
25~~4~~. MAP-GR-DataTypes;
26~~5~~. MAP-LCS-DataTypes;
27. MAP-ST-DataTypes.

References are made also to modules defined outside of the present document. They are defined in the technical specification Mobile Services Domain and technical specification Transaction Capability respectively:

MobileDomainDefinitions;

TCAPMessages;

DialoguePDUs.

17.1.6 Application Contexts

The following informative table lists the latest versions of the Application Contexts used in this specification, with the operations used by them and, where applicable, whether or not the operation description is exactly the same as for previous versions. Information in sections 17.6 & 17.7 relates only to the ACs in this table.

AC Name	AC Version	Operations Used	Comments *
locationCancellationContext	v3	cancelLocation	
equipmentMngtContext	v2	checkIMEI	
imsiRetrievalContext	v2	sendIMSI	
infoRetrievalContext	v3	sendAuthenticationInfo	
interVlrInfoRetrievalContext	v3	sendIdentification	
handoverControlContext	v3	prepareHandover forwardAccessSignalling sendEndSignal processAccessSignalling prepareSubsequentHandover	the syntax of this operation has been extended in comparison with release 98 version
mwdMngtContext	v3	readyForSM	
msPurgingContext	v3	purgeMS	
shortMsgAlertContext	v2	alertServiceCentre	
resetContext	v2	reset	
networkUnstructuredSsContext	v2	processUnstructuredSS-Request unstructuredSS-Request unstructuredSS-Notify	
tracingContext	v3	activateTraceMode deactivateTraceMode	
networkFunctionalSsContext	v2	registerSS eraseSS activateSS deactivateSS registerPassword interrogateSS getPassword	
shortMsgMO-RelayContext	v3	mo-forwardSM	
shortMsgMT-RelayContext	v3	mt-forwardSM	
shortMsgGatewayContext	v3	sendRoutingInfoForSM reportSM-DeliveryStatus InformServiceCentre	the syntax of this operation has been extended in comparison with release 96 version
networkLocUpContext	v3	updateLocation forwardCheckSs-Indication restoreData insertSubscriberData activateTraceMode	the syntax is the same in v1 & v2
gprsLocationUpdateContext	v3	updateGprsLocation insertSubscriberData activateTraceMode	
subscriberDataMngtContext	v3	insertSubscriberData deleteSubscriberData	
roamingNumberEnquiryContext	v3	provideRoamingNumber	
locationInfoRetrievalContext	v3	sendRoutingInfo	
gprsNotifyContext	v3	noteMsPresentForGprs	
gprsLocationInfoRetrievalContext	v3	sendRoutingInfoForGprs	
failureReportContext	v3	failureReport	
callControlTransferContext	v4	resumeCallHandling	
subscriberInfoEnquiryContext	v3	provideSubscriberInfo	
anyTimeEnquiryContext	v3	anyTimeInterrogation	
anyTimeInfoHandlingContext	v3	anyTimeSubscriptionInterrogation anyTimeModification	
ss-InvocationNotificationContext	v3	ss-InvocationNotification	
sIWFSAllocationContext	v3	provideSIWFSNumber sIWFSSignallingModify	
groupCallControlContext	v3	prepareGroupCall processGroupCallSignalling forwardGroupCallSignalling sendGroupCallEndSignal	

reportingContext	v3	setReportingState statusReport remoteUserFree	
callCompletionContext	v3	registerCC-Entry eraseCC-Entry	
istAlertingContext	v3	istAlert	
ImmediateTerminationContext	v3	istCommand	
locationSvcEnquiryContext	v3	provideSubscriberLocation subscriberLocationReport	
locationSvcGatewayContext	v3	sendRoutingInfoForLCS	
mm-EventReportingContext	v3	noteMM-Event	
subscriberDataModificationNotificationContext	v3	noteSubscriberDataModified	
authenticationFailureReportContext	v3	authenticationFailureReport	
secureTransportHandlingContext	v3	secureTransportClass1 secureTransportClass2 secureTransportClass3 secureTransportClass4	

NOTE (*): The syntax of the operations is not the same as in previous versions unless explicitly stated

****** Next modified section ******

17.2.2.8 Secure transport

This operation package includes the operations required for the secure transport of MAP messages between any MAP entities.

```
SecureTransportHandlingPackage-v3 ::= OPERATION-PACKAGE
     CONSUMER INVOKES {
          SecureTransportClass1,          -- to be used if the original operation is a
                                          -- TCAP class 1 operation
          SecureTransportClass2,          -- to be used if the original operation is a
                                          -- TCAP class 2 operation
          SecureTransportClass3,          -- to be used if the original operation is a
                                          -- TCAP class 3 operation
          SecureTransportClass4}          -- to be used if the original operation is a
                                          -- TCAP class 4 operation
```

This package is v3 only.

17.2.2.9 Void

****** Next modified section ******

17.3.2.8 Secure transport

This application context is used for the secure transport of MAP messages between any MAP entities.

```
SecureTransportHandlingContext-v3 APPLICATION-CONTEXT
     INITIATOR CONSUMER OF {
          SecureTransportHandlingPackage-v3}
::= {map-ac secureTransportHandling(40) version3(3)}
```

This application-context is v3 only.

17.3.2.9 - 17.3.2.10 Void

****** Next modified section ******

17.3.3 ASN.1 Module for application-context-names

```
.
.
subscriberDataModificationNotificationContext-v3  OBJECT IDENTIFIER ::=
     {map-ac subscriberDataModificationNotification(22) version3(3)}

secureTransportHandlingContext-v3  OBJECT IDENTIFIER ::=
     {map-ac secureTransportHandling(40) version3(3)}

.
.
```

****** Next modified section ******

17.4 MAP Dialogue Information

```
MAP-DialogueInformation {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-DialogueInformation (3) version6 (6)}

DEFINITIONS

IMPLICIT TAGS

::=

BEGIN

EXPORTS
   map-DialogueAS,
   MAP-DialoguePDU,
   map-ProtectedDialogueAS,
   MAP-ProtectedDialoguePDU

;

IMPORTS
   gsm-NetworkId,
   as-Id
FROM MobileDomainDefinitions {
   ccitt (0) identified-organization (4) etsi (0) mobileDomain (0)
   mobileDomainDefinitions (0) version1 (1)}

   AddressString
FROM MAP-CommonDataTypes {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network(1) modules (3) map-CommonDataTypes (18) version6 (6)}

   ExtensionContainer
FROM MAP-ExtensionDataTypes {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-ExtensionDataTypes (21) version6 (6)}

   SecurityHeader,
   ProtectedPayload
FROM MAP-ST-DataTypes {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-ST-DataTypes (27) version6 (6)}

;


-- abstract syntax name for MAP-DialoguePDU
```

```
map-DialogueAS  OBJECT IDENTIFIER ::=
    {gsm-NetworkId as-Id map-DialoguePDU (1) version1 (1)}
```

```
MAP-DialoguePDU ::= CHOICE {
    map-open                        [0] MAP-OpenInfo,
    map-accept                      [1] MAP-AcceptInfo,
    map-close [2] MAP-CloseInfo,
    map-refuse                      [3] MAP-RefuseInfo,
    map-userAbort                   [4] MAP-UserAbortInfo,
    map-providerAbort               [5] MAP-ProviderAbortInfo}
```

```
MAP-OpenInfo ::= SEQUENCE {
    destinationReference            [0] AddressString           OPTIONAL,
    originationReference            [1] AddressString           OPTIONAL,
    ...,
    extensionContainer              ExtensionContainer          OPTIONAL
    -- extensionContainer must not be used in version 2
    }
```

```
MAP-AcceptInfo ::= SEQUENCE {
    ...,
    extensionContainer              ExtensionContainer          OPTIONAL
    -- extensionContainer must not be used in version 2
    }
```

```
MAP-CloseInfo ::= SEQUENCE {
    ...,
    extensionContainer              ExtensionContainer          OPTIONAL
    -- extensionContainer must not be used in version 2
    }
```

```
MAP-RefuseInfo ::= SEQUENCE {
    reason    Reason,
    ...,
    extensionContainer              ExtensionContainer          OPTIONAL,
    -- extensionContainer must not be used in version 2
    alternativeApplicationContext   OBJECT IDENTIFIER           OPTIONAL
    -- alternativeApplicationContext must not be used in version 2
    }
```

```
Reason ::= ENUMERATED {
    noReasonGiven  (0),
    invalidDestinationReference    (1),
    invalidOriginatingReference    (2),
    encapsulatedAC-NotSupported    (3)}
    -- encapsulatedAC-NotSupported must not be used in dialogues with an AC
    -- different from secureTransportHandling
```

```
MAP-UserAbortInfo ::= SEQUENCE {
    map-UserAbortChoice             MAP-UserAbortChoice,
    ...,
    extensionContainer              ExtensionContainer          OPTIONAL
    -- extensionContainer must not be used in version 2
    }
```

```
MAP-UserAbortChoice ::= CHOICE {
    userSpecificReason              [0] NULL,
    userResourceLimitation          [1] NULL,
    resourceUnavailable             [2] ResourceUnavailableReason,
    applicationProcedureCancellation    [3] ProcedureCancellationReason}
```

```
ResourceUnavailableReason ::= ENUMERATED {
    shortTermResourceLimitation  (0),
    longTermResourceLimitation  (1)}
```

```
ProcedureCancellationReason ::= ENUMERATED {
     handoverCancellation  (0),
     radioChannelRelease  (1),
     networkPathRelease  (2),
     callRelease  (3),
     associatedProcedureFailure  (4),
     tandemDialogueRelease  (5),
     remoteOperationsFailure  (6)}
```

```
MAP-ProviderAbortInfo ::= SEQUENCE {
     map-ProviderAbortReason             MAP-ProviderAbortReason,
     ...,
     extensionContainer                  ExtensionContainer                OPTIONAL
     -- extensionContainer must not be used in version 2
     }
```

```
MAP-ProviderAbortReason ::= ENUMERATED {
     abnormalDialogue  (0),
     invalidPDU  (1)}
```

```
-- abstract syntax name for MAP-ProtectedDialoguePDU

map-ProtectedDialogueAS  OBJECT IDENTIFIER ::=
     {gsm-NetworkId as-Id map-ProtectedDialoguePDU (3) version1 (1)}

MAP-ProtectedDialoguePDU ::= SEQUENCE {
     encapsulatedAC                      OBJECT IDENTIFIER,
     securityHeader                      SecurityHeader                    OPTIONAL,
     protectedPayload                    ProtectedPayload                  OPTIONAL,
     ...}
     -- The protectedPayload carries the result of applying the security function
     -- defined in 3G TS 33.102 to the encoding of the securely transported
     -- MAP-DialoguePDU

END
```

****** Next modified section ******

17.5 MAP operation and error codes

```
MAP-Protocol {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-Protocol (4) version6 (6)}

DEFINITIONS

::=

BEGIN

IMPORTS
.
.
FROM MAP-LocationServiceOperations {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-LocationServiceOperations (24)
   version6 (6)}

SecureTransportClass1,
SecureTransportClass2,
SecureTransportClass3,
SecureTransportClass4

FROM MAP-SecureTransportOperations {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-SecureTransportOperations (26)
   version6 (6)}
.
.
   MM-EventNotSupported,
   SecureTransportError
```

```
FROM MAP-Errors {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-Errors (10) version6 (6)}
;
.
.
-- Mobility Management operation codes

noteMM-Event  NoteMM-Event ::= localValue 89


-- Secure transport operation codes

secureTransportClass1 SecureTransportClass1 ::= localValue 78
secureTransportClass2 SecureTransportClass2 ::= localValue 79
secureTransportClass3 SecureTransportClass3 ::= localValue 80
secureTransportClass4 SecureTransportClass4 ::= localValue 81


.
.
-- Mobility Management error codes

mm-EventNotSupported  MM-EventNotSupported ::= localValue 59


-- Secure transport error codes

secureTransportError secureTransportError ::= localValue 4


.
.
```

****** Next modified section ******

17.6.6 Errors

```
MAP-Errors {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-Errors (10) version6 (6)}

DEFINITIONS

::=

BEGIN

EXPORTS

.
.
   -- Mobility Management errors
   MM-EventNotSupported,

   -- Secure transport errors
   SecureTransportError

;

IMPORTS
.
.
   InformationNotAvailableParam,
   SecureTransportErrorParam
.
.
MM-EventNotSupported ::= ERROR
    PARAMETER
         mm-EventNotSupported-Param      MM-EventNotSupported-Param
         -- optional

   -- Secure transport errors

```

```
SecureTransportError ::= ERROR
    PARAMETER
        secureTransportErrorParam       SecureTransportErrorParam
```

.
.

****** Next modified section ******

17.6.9 Secure transport operations

```
MAP-SecureTransportOperations {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-SecureTransportOperations (26)
   version6 (6)}

DEFINITIONS

::=

BEGIN

EXPORTS
   SecureTransportClass1,
   SecureTransportClass2,
   SecureTransportClass3,
   SecureTransportClass4
;

IMPORTS
   OPERATION
FROM TCAPMessages {
   ccitt recommendation q 773 modules (2) messages (1) version2 (2)}

   DataMissing,
   SecureTransportError,
   UnexpectedDataValue

FROM MAP-Errors {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-Errors (10) version6 (6)}

   SecureTransportArg,
   SecureTransportRes

FROM MAP-ST-DataTypes {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-ST-DataTypes (27) version6 (6)}
;
```

```
SecureTransportClass1 ::= OPERATION                --Timer shall be the same as for the
                                                   --securely transported operation
    ARGUMENT
        secureTransportArg              SecureTransportArg
    RESULT
        secureTransportRes              SecureTransportRes
    ERRORS {
        SecureTransportError,
        DataMissing,
        UnexpectedDataValue}
```

```
SecureTransportClass2 ::= OPERATION                --Timer shall be the same as for the
                                                   --securely transported operation
    ARGUMENT
        secureTransportArg              SecureTransportArg
    ERRORS {
        SecureTransportError,
        DataMissing,
        UnexpectedDataValue}
```

```
SecureTransportClass3 ::= OPERATION                --Timer shall be the same as for the
                                                   --securely transported operation
    ARGUMENT
        secureTransportArg              SecureTransportArg
    RESULT
        secureTransportRes              SecureTransportRes
```

```
SecureTransportClass4 ::= OPERATION                --Timer shall be the same as for the
                                                   --securely transported operation
    ARGUMENT
        secureTransportArg              SecureTransportArg
```

```
END
```

****** Next modified section ******

17.7.7 Error data types

```
MAP-ER-DataTypes {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-ER-DataTypes (17) version6 (6)}

.
.
EXPORTS
.
.
   MM-EventNotSupported-Param,
   SecureTransportErrorParam,
.
.
IMPORTS
.
.
   SignalInfo,
   BasicServiceCode,
   NetworkResource
FROM MAP-CommonDataTypes {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-CommonDataTypes (18) version6 (6)}

   SecurityHeader,
   ProtectedPayload
FROM MAP-ST-DataTypes {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-ST-DataTypes (27) version6 (6)}

.
.
MM-EventNotSupported-Param ::= SEQUENCE {
    extensionContainer                  ExtensionContainer              OPTIONAL,
    ...}

SecureTransportErrorParam ::= SEQUENCE {
    securityHeader                      SecurityHeader,
    protectedPayload                    ProtectedPayload                OPTIONAL
    }
    -- The protectedPayload carries the result of applying the security function
    -- defined in 3G TS 33.102 to the encoding of the securely transported error
    -- parameter

END
```

****** Next modified section ******

17.7.14 Secure transport data types

```
MAP-ST-DataTypes {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-ST-DataTypes (27) version6 (6)}

DEFINITIONS
IMPLICIT TAGS
::=
BEGIN

EXPORTS
   SecureTransportArg,
   SecureTransportRes,
   SecurityHeader,
   ProtectedPayload
;

IMPORTS
   IMSI

FROM MAP-CommonDataTypes {
   ccitt identified-organization (4) etsi (0) mobileDomain (0)
   gsm-Network (1) modules (3) map-CommonDataTypes (18) version6 (6)}
;
```

```
SecureTransportArg ::= SEQUENCE {
    securityHeader                  SecurityHeader,
    protectedPayload                ProtectedPayload            OPTIONAL
    }
    -- The protectedPayload carries the result of applying the security function
    -- defined in 3G TS 33.102 to the encoding of the argument of the securely
    -- transported operation
```

```
SecureTransportRes ::= SEQUENCE {
    securityHeader                  SecurityHeader,
    protectedPayload                ProtectedPayload            OPTIONAL
    }
    -- The protectedPayload carries the result of applying the security function
    -- defined in 3G TS 33.102 to the encoding of the result of the securely
    -- transported operation
```

```
SecurityHeader ::= SEQUENCE {
    originalComponentIdentifier     OriginalComponentIdentifier,
    sendingPLMN-Id                  PLMN-Id,
    protectionMode                  [0] ProtectionMode              OPTIONAL,
    encryptionAlgorithmIdentifier   [1] EncryptionAlgorithmIdentifier OPTIONAL,
    modeOfOperation                 [2] ModeOfOperation             OPTIONAL,
    encryptionKkeyVersionNumber     [3] EncryptionKeyVersionNumber  OPTIONAL,
    initialisationVector            [40] InitialisationVector       OPTIONAL,
    hashAlgorithmIdentifier         [5] HashAlgorithmIdentifier     OPTIONAL,
    hashKeyVersionNumber            [6] HashKeyVersionNumber        OPTIONAL,
    ...}
```

```
PLMN-Id ::= TBCD-STRING (SIZE (3))
    -- digits of MCC, MNC, are concatenated in this order.
```

```
ProtectedPayload ::= OCTET STRING(SIZE(1..1000)
    -- In protection mode 0 (noProtection) the ProtectedPayload carries the transfer
        -- syntax value of the component parameter identified by the
        -- originalComponentIdentifier.
    -- In protection mode 1 (integrityAuthenticity) the protectedPayload carries 4
        -- octets TVP, followed by the transfer syntax value of the component
        -- parameter identified by the originalComponentIdentifier, followed by
        -- the integrity check value.
        -- The integrity check value is the result of applying the hash algorithm
        -- to the concatenation of TVP, transfer syntax value of the SecurityHeader,
        -- transfer syntax value of the component parameter.
    -- In protection mode 2 (confidentialityIntegrityAuthenticity) the protected
        -- payload carries 4 octets TVP, followed by the encrypted transfer syntax
        -- value of the component parameter identified by the
        -- originalComponentIdentifier, followed by the integrity check value.
        -- The integrity check value is the result of applying the hash algorithm
        -- to the concatenation of TVP, transfer syntax value of the SecurityHeader,
        -- encrypted transfer syntax value of the component parameter.
    -- See 33.102.
    -- The length of the protectedPayload is adjusted according to the capabilities of
    -- the lower protocol layers
```

```
ProtectionMode ::= ENUMERATED {
    noProtection                        (0),
    integrityAuthenticity               (1),
    confidentialityIntegrityAuthenticity (2)}

EncryptionAlgorithmIdentifier ::= INTEGER (1..127)
    -- The encryption algorithm corresponding to each value of the Encryption
    -- Algorithm Identifier type is defined in TS 33.102

HashAlgorithmIdentifier ::= INTEGER (1..127)
    -- The encryption algorithm corresponding to each value of the Hash Algorithm
    -- Identifier type is defined in TS 33.102

ModeOfOperation ::= ENUMERATED {
    ecb                                 (0),
    cbc                                 (1),
    cfb                                 (2),
    ofb                                 (3),
    ...}
    -- Modes of operation are defined in ISO/IEC 10116 (1991)

EncryptionKeyVersionNumber ::= INTEGER (0..127)

HashKeyVersionNumber ::= INTEGER (0..127)

InitialisationVector ::= OCTET STRING (SIZE(8))

OriginalComponentIdentifier ::= CHOICE {
    operationCode                       [0] OperationCode,
    errorCode                           [1] ErrorCode,
    userInfo                            [2] NULL}

OperationCode ::= CHOICE {
    localValue                          INTEGER,
    globalValue                         OBJECT IDENTIFIER}

ErrorCode ::= CHOICE {
    localValue                          INTEGER,
    globalValue                         OBJECT IDENTIFIER}

END
```