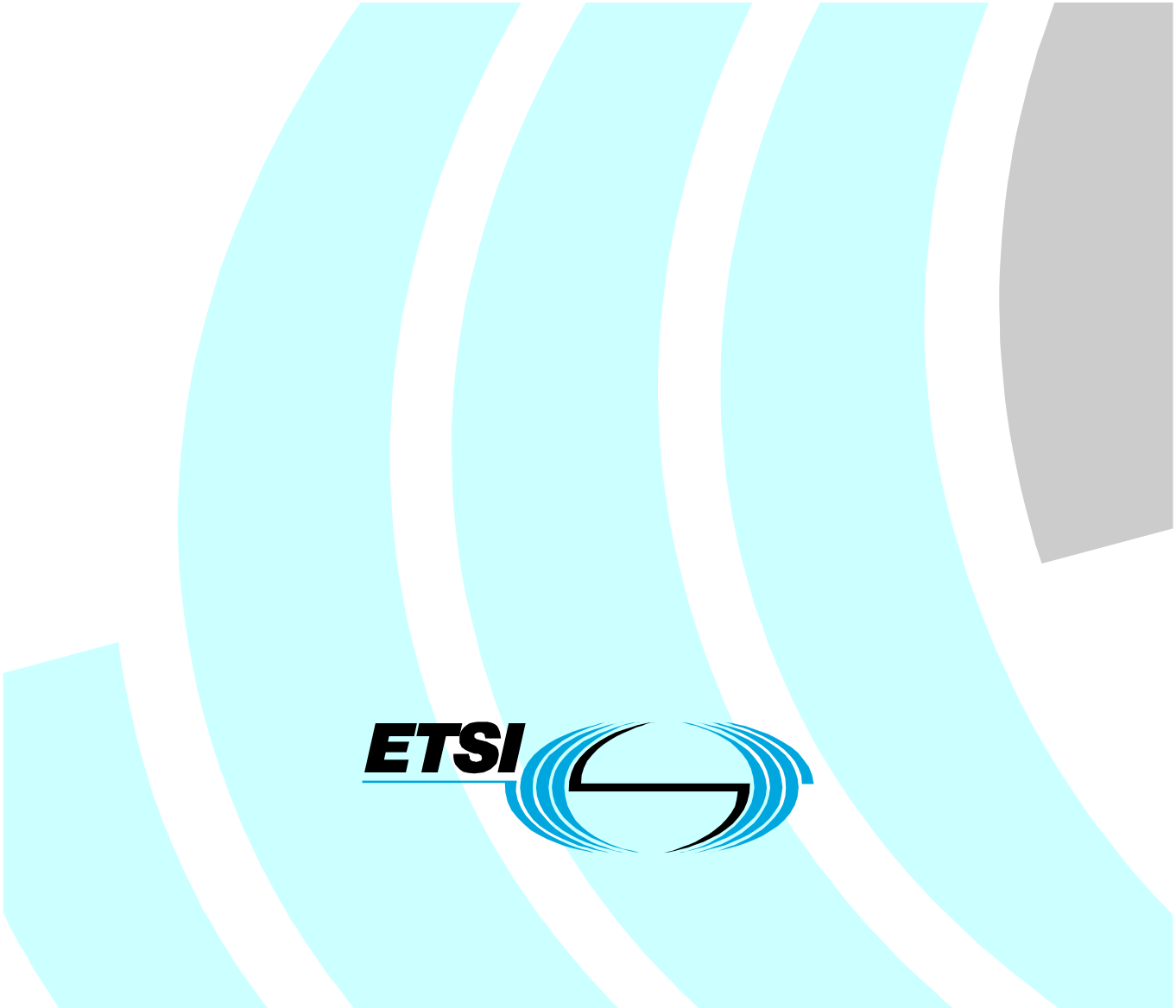
ETSI TS 101 882-1 V4.1.1 (2003-09)

Technical Specification

Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 4; Protocol Framework Definition; Part 1: Meta-protocol design rules, development method, and mapping guideline



Reference
RTS/TIPHON-03016-1R4

Keywords
interface, IP, meta-protocol

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

Individual copies of the present document can be downloaded from:
http://www.etsi.org

The present document may be made available in more than one electronic version or in print. In any case of existing or perceived difference in contents between such versions, the reference version is the Portable Document Format (PDF). In case of dispute, the reference shall be the printing on ETSI printers of the PDF version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at
http://portal.etsi.org/tb/status/status.asp

If you find errors in the present document, send your comment to:
editor@etsi.org

Copyright Notification

Contents

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://webapp.etsi.org/IPR/home.asp).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This Technical Specification (TS) has been produced by ETSI Project Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON).

The present document is part 1 of a multi-part deliverable covering the TIPHON meta-protocol, as identified below:

Part 1: Meta-protocol design rules, development method, and mapping guideline;

Part 2: Registration and Service Attachment service meta-protocol definition;

Part 3: TIPHON Simple Call;

Part 4: Media control and meta-protocol;

Part 5: Transport control service meta-protocol definition.

1 Scope

The present document defines by means of an information model, a functional entity behavioural model, and by validated SDL a model of the abstract behaviour of each service and service capability identified as being essential in TIPHON R4.

The present document defines the method of defining a meta-protocol using a worked example for clarification. In addition the present document defines the approach to be used to identify the mapping of any candidate protocol to the meta-protocol.

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 and/or edition number or version number) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies.

Referenced documents which are not found to be publicly available in the expected location might be found at http://docbox.etsi.org/Reference.

[1] ETSI TS 101 314: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 4; Abstract Architecture & Reference Points Definition; Network Architecture and Reference Points".

[2] Void.

[3] ETSI TR 101 877: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON); Requirements Definition Study; Scope and Requirements for a Simple call".

[4] ETSI TS 101 878 V4.1.1: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 4; Service Capability Definition; Service Capabilities for a simple call".

[5] Void.

[6] ETSI TS 102 024-2: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 4; End-to-end Quality of Service in TIPHON Systems; Part 2: Definition of Speech Quality of Service (QoS) Classes".

[7] ITU-T Recommendation Z.100: "Specification and Description Language (SDL)".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in TR 101 877 [3] and TS 101 878 [4] apply.

3.2 Abbreviations

For the purposes of the present document, the abbreviations defined in TR 101 877 [3] and TS 101 878 [4] and the following apply:

FEA	Functional Entities Actions
OMG	Object Management Group
PDU	Protocol Data Units
PE	Protocol Entities
QFE	Qos Functional Entities
SCN	Switched Circuit Network
SDL	Specification and Description Language
UML	Unified Modelling Language

4 What is the meta-protocol?

4.1 Introduction

The meta-protocol serves as the platform for candidate protocols to map to and in which the mapping identifies exactly how the requirements are met.

The description of the meta-protocol consists of 2 parts:

- requirements of the service or service capability; and
- functional behaviour of the service or service capability.

4.2 Content of the requirements section

The requirements section of the meta-protocol for any service (or service capability) is an overall description from the user's standpoint and consists of a definition and description of the service (or service capability) in prose, followed by an overall summary of the dynamic behaviour in either an SDL process diagram, or an UML activity diagram.

4.2.1 Definition

This clause provides a short description of the service (or service capability) in terms of the perceptions of the user receiving the service (or service capability) and any other users involved in the service (or service capability).

4.2.2 Description

This clause expands on the definition and summarizes the operation of the service (or service capability) in a generic form which does not constrain terminal or network design. It is intended to allow an understanding of the service (or service capability) without regard to implementation. It also includes any specific terminology used within the prose definition and description, and any qualifications. For basic services this clause details the applications which could utilize the service whilst for supplementary services this clause details their applicability to particular telecommunication services.

4.2.3 Procedures

The overall operation of the service (or service capability) in its various states is described in this clause. These procedures relate to all actions between the user(s) and the network during the period that the service (or service capability) is available.

4.2.3.1 Provision/withdrawal

This clause describes the means by which the service (or service capability) is made available by the service provider, e.g. it may be generally available to all customers, or only be available to those customers who have made a prior arrangement.

4.2.3.2 Normal procedures

The clauses under this heading describe the normal procedures for activation, deactivation, registration, invocation and operation for the service (or service capability) as appropriate. This clause describes only the successful outcome of each procedure, and the procedures which are executed as a result of such successful outcomes. The procedures are described in a time-based sequence of events. They describe the interactions of the users involved in the service (or service capability) with the service provider and with each other which lead to, and are elements of, the successful operation of the service (or service capability).

4.2.3.2.1 Activation/deactivation/registration

The procedures for activation, which is the operation of bringing the service into the "ready for invocation" state, and deactivation, which is the complementary action, are described in this clause. For some services (or service capability) there may be a specific user procedure to allow activation and deactivation as necessary, whilst for others the service (or service capability) is permanently activated on provision and thus no procedure is provided.

Registration describes the procedures by which any specific information, necessary for the successful operation of the supplementary service, is given to the network. The need to register information with the network, e.g. a forwarding number, only applies to certain services (or service capabilities).

4.2.3.2.2 Invocation and operation

This clause describes the procedures for invocation, which is the action and conditions under which the service (or service capability) is brought into operation; in the case of some services (or service capabilities) this may only be on a particular call. It should be noted that although a service (or service capability) may be activated, it may not necessarily be invoked on all calls. (Invocation may take place either subsequent to or simultaneously with activation.)

In the case of basic services this clause shall describe the events, perceived at the service access point, during the establishment, information transfer and clearing phases.

Operation is the procedure which occurs once a service (or service capability) has been invoked.

This description gives details of the significant actions of the network, treated in principle as a single entity, and the perception of the users involved in the service (or service capability). It includes details of the information exchanged between the network and relevant users and the indications given to each user, by the network, concerning the states of the service (or service capability).

4.2.3.2.3 Interrogation/editing

Interrogation is the facility which enables a served user to determine, from the service provider, the current status of a particular service. Whether this facility is provided for the service (or service capability) being described, and if so, the procedures that accompany it, are detailed in this clause.

4.2.3.3 Exceptional procedures

The clauses under this heading describe the exceptional procedures which result in an unsuccessful outcome of the service (or service capability). Included within this description are the details for such situations as invalid user action and the handling of certain network and interface conditions. For the case of basic services this includes the handling of such network conditions as congestion.

4.2.4 Interaction with other services and service capabilities

This clause shall describe all interactions of the service or service capability with other services or service capabilities insofar as they have been identified.

For example, for some service or service capability pairs there is no interaction as the two are not permitted to be both in operation at the same time. For other pairs, one or both services (or service capabilities) may be modified whilst the pair are in operation simultaneously.

4.2.5 Content of the dynamic definition

The dynamic description of a service (or service capability) contains all the information that is sent and received by the user from activation/invocation of the service (or service capability) to completion of the service (or service capability). The information shall be presented in the form of an overall specification and description language (SDL) diagram an UML activity diagram.

4.3 Content of the behavioural section

4.3.1 Derivation of a functional model

A functional model shall be derived for each service (or service capability) for which a meta-protocol is required. The functions required to provide the service (or service capability) are grouped into functional entities. The functional model is the aggregate of the functional entities and their relationships.

4.3.2 Information flow diagrams

The distribution of the functions needed to provide a service (or service capability) as defined by the functional model requires that interactions be defined between functional entities. Such an interaction is referred to as an "information flow" and shall have a name descriptive of the intent of the information flow. Information flow diagrams shall be created for normal operation and should be created as appropriate for exceptional operation.

The semantic meaning and information content of each information flow shall be identified. The information content of each information flow shall be described using ASN.1.

4.3.3 SDL/UML diagrams for functional entities

The functions performed within a functional entity shall be identified and represented in the form of either a Specification and Description Language (SDL) diagram or a Unified Modelling Language (UML) diagram.

The inputs and outputs of the SDL (UML) diagram are to and from the users as described in the requirements part and are information flows to and from other functional entities.

SDL (UML) diagrams shall be defined for each functional entity based on the information flows defined for the normal behaviour of the service. The SDL diagrams shall also covers the exceptional behaviour.

4.3.4 Functional Entity Actions

The actions performed within a functional entity shall be represented as a list, or sequence, of functional entity actions (FEAs) in prose form. These form the basis for understanding the meaning of the information flows and provide a basis for the mapping activity.

5 Method of mapping to the meta-protocol

5.1 Introduction

As identified in clause 4.1 the purpose of a meta-protocol is to serve as "the platform for candidate protocols to map to and in which the mapping identifies exactly how the requirements are met".

5.2 Pre-analysis of the candidate

In the meta-protocol development the resulting description is made without prejudice to the protocol used to implement the meta-protocol. This approach allows many protocols to be used to implement the service or service capability, however the ability of any candidate to meet the requirements of any meta-protocol has to be determined in advance. If a candidate cannot meet the requirements of any meta-protocol no mapping should be performed, however, the shortcomings in the candidate that prevent it from making a successful mapping should be identified in a separate report as the basis of a request to extend the functionality of the candidate sufficiently to allow the mapping to be made.

5.2.1 Identify Protocol Entities (PE) of candidate

In order to perform the mapping exercise the candidate has to explicitly identify the protocol entities of the candidate protocol.

EXAMPLE: In SIP the protocol entities are described as the UAC, UAS, proxy server, registrar and redirect server.

5.2.2 Identify Protocol Data Units of candidate

In order to perform the mapping exercise the candidate has to explicitly identify the protocol data units of the candidate protocol.

EXAMPLE 1: In SIP the PDUs are not easy to identify. However, each method could be considered as a PDU type (although it is hard to evaluate the impact of the session description) and the header elements as the PDU data elements.

EXAMPLE 2: A SIP INVITE PDU request has mandatory content of Call-id, Contact, Cseq, From, Max-forwards, To, Via.

5.3 Mapping of PE to FE

If many FEs map to a single PE the information flows between the encompassed FEs do not need to be mapped.

5.3.1 Allocation of functional entities to protocol entities

In this step, the functional entities and information flows identified in the meta-protocol are allocated to candidate specific types of physical locations, e.g. a PABX, an exchange, a gatekeeper, a server. Each allocation is called a scenario and is specific to the candidate protocol's capabilities. The relationship supported between two functional entities located in different physical locations must be realized within protocol(s) supported between those locations.

The physical location applicable to the candidate shall also be mapped to the reference architecture described in TS 101 314 [1].

EXAMPLE: The QoS functional entity model is given in annex A. If we consider SIP we can map QFE1 to the UAC and identify it as belonging to the originating terminal functional grouping.

Annex A (informative): Example

The following example is drawn from the description of the End-to-End QoS Control model for TIPHON.

A.1 Description

A.1.1 General description

End-to-End QoS Signalling is used within a TIPHON network to ensure that a caller is provided with an end-to-end connection having at least the QoS class subscribed to or a lower QoS class if this is acceptable to the user. A QoS level may either be requested explicitly by the user on a call-by-call basis or may be predefined as part of the user's subscription. Additionally, the caller may be able to take specific actions if the QoS moves outside the accepted level during an established call.

The user may use any of the following methods to request a specific end-to-end QoS at call establishment:

- By subscription:

  The agreement between the user and the user's service provider identifies the QoS level to be requested for any call. The QoS level may be fixed or variable based upon such parameters as time-of-day and call destination. This method requires no signalling between the user and the service provider at call setup time.

- By the use of a standardized TIPHON QoS Class:

NOTE: And so forth outlining the essential considerations of the service from the viewpoint of the service user.

A.1.2 Procedures

A.1.2.1 Provision and withdrawal

End-to-End QoS Signalling shall be provided on a per-name, per application basis to all subscribers to the simple call within a TIPHON system.

When establishing a call, a user shall be able to select at least one of the options identified in table A.1.

Table A.1: QoS option

Option	Value
QoS class	- Predefined by user and service provider (TIPHON or non-TIPHON class) - TIPHON Best - QoS Class 3 (see note 1) - TIPHON High - QoS Class 2H (see note 1) - TIPHON Medium - QoS Class 2M (see note 1) - TIPHON Acceptable - QoS Class 2A (see note 1) - TIPHON Best effort - QoS Class 1 (see note 1) - Non-TIPHON QoS Class (see note 2)
NOTE 1: This value shall be as defined in TS 102 024-2 [6]. NOTE 2: This may be any value agreed between the user and the service provider to indicate a specific QoS	

A.1.2.2 Normal procedures

A.1.2.2.1 Activation, deactivation and interrogation

QoS Signalling shall be permanently activated.

A.1.2.2.2 Invocation and operation

When establishing a simple call, the calling user (or an agent within the user's network) may request a TIPHON standardized QoS class or a non-standardized QoS class, to be applied to the call in order to achieve a required end-to-end QoS.

If the end-to-end QoS requested by the calling user is available, communication using that QoS shall be established following the simple call procedures specified in TS 101 878 [4].

A.1.2.3 Exceptional procedures

A.1.2.3.1 Invocation and operation

If it is not possible to offer the requested end-to-end QoS at call establishment, the calling user shall be informed and may take one of the following actions:

- terminate the call attempt;
- request a lower QoS;
- proceed with the call at the QoS available between the caller and the called user.

If, during an established call, the end-to-end QoS perceived by the calling user falls below an acceptable level the following practical options are available:

- terminate the call;
- continue with the call at the inferior QoS level.

NOTE: These sections clearly describe the behaviour expected without worrying about the kind of entity that implements the behaviour.

A.1.3 Interactions with other TIPHON service capabilities

This clause specifies interactions with other TIPHON services for which standards were available at the time of publication of the present document.

A.1.3.1 Registration service capabilities

A.1.3.1.1 Terminal transport service registration

No interaction.

A.1.3.1.2 User service registration

No interaction.

NOTE: The QOS to be used for subsequent calls by the registered user may form part of the information supplied at registration.

A.1.3.2 Call connectivity service capabilities

A.1.3.2.1 Simple call establishment

No interactions.

A.1.3.2.2 Calling user identity generation

No interactions.

NOTE: This carries on to describe the expected interactions for every service or service capability already described in some (published) form. In doing this additional requirements are set for the behavioural modelling which will follow.

A.1.4 Interworking considerations

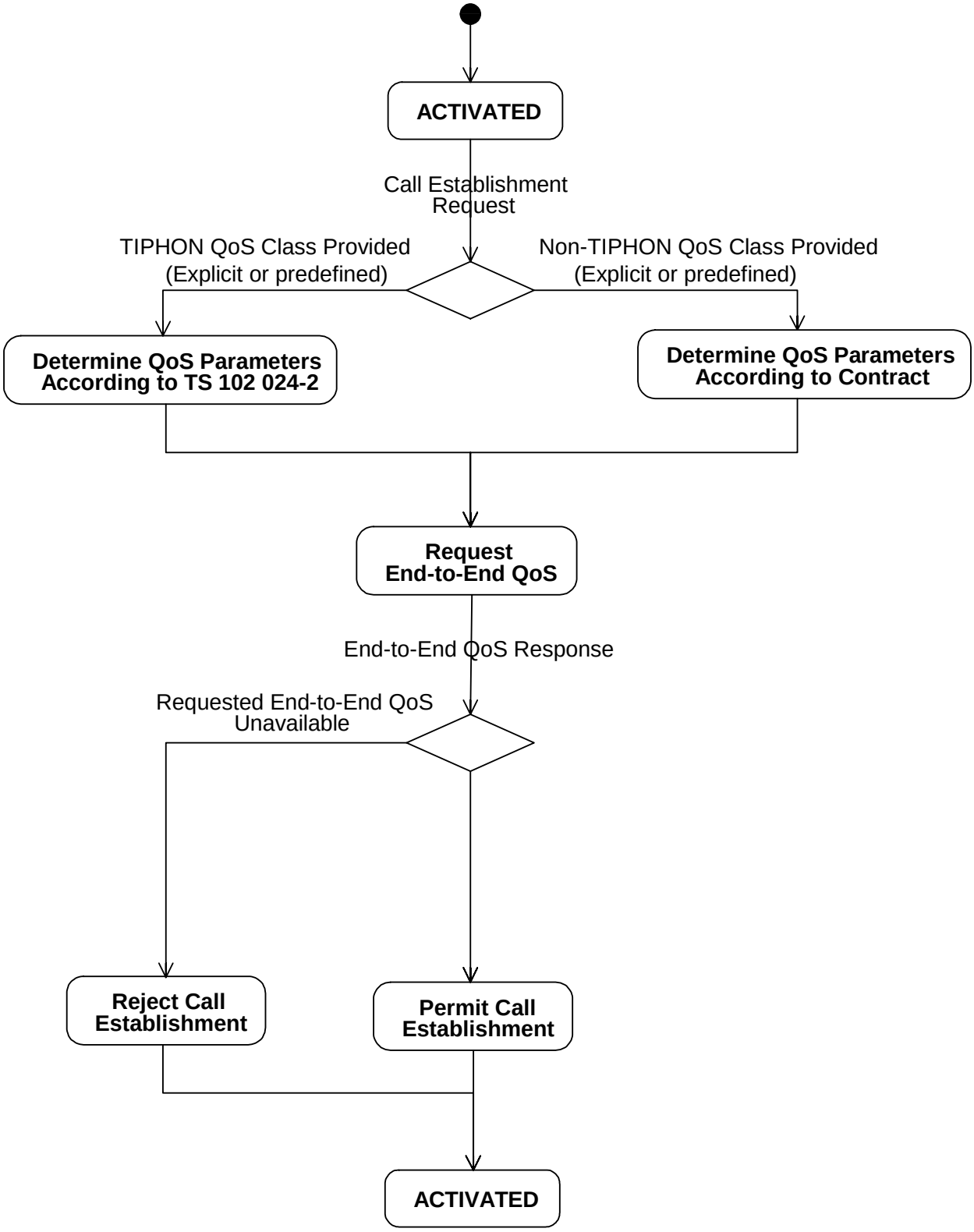
When interworking with a Switched Circuit Network (SCN) where the only variable affecting QoS is the choice of bearer service, fixed QoS parameters shall be assumed based on the bearer service selected.

NOTE: This section gives guidance to the developer. In this case as SCNs have QoS implicit in the bearer-service signalling may not be required. Clearly describe the behaviour expected without worrying about the kind of entity that implements the behaviour.

A.1.5 Overall behaviour

Table A.1contains the dynamic description of End-to-End QoS Signalling using a Unified Modelling Language (UML) activity diagram. The activity diagram represents the behaviour of a TIPHON system in providing End-to-End QoS Signalling.

NOTE: The syntax and semantics of UML diagrams are defined by the Object Management Group (OMG).



NOTE: The form of this diagram is up to the user but UML and SDL are the most appropriate. It should summarize the normal and exceptional behaviour as simply as possible (i.e. without considering data too much).

Figure A.1: Overall behaviour of End-to-End QoS Signalling at call establishment

A.2 Functional entity model and information flows

A.2.1 Functional entity model

A.2.1.1 Description of model

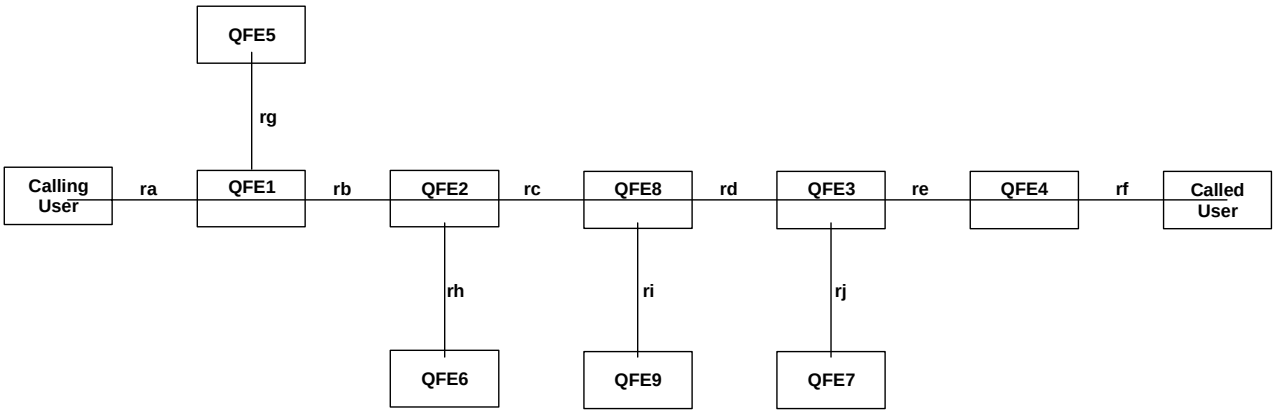
The functional model shall comprise the following QoS Functional Entities (QFE):

- Calling User — The application at the calling user's terminal which instigates the service request;
- QFE1 — The service agent that processes the calling user's request for end-to-end QoS signalling;
- QFE2 — The originating QoS coordination function. This QFE is responsible for negotiating and establishing a particular QoS on behalf of the calling user;
- QFE3 — The terminating QoS coordination function. This QFE is responsible for establishing a particular QoS on behalf of the called user;
- QFE4 — The service agent that processes an incoming call to the called user;
- QFE5 — The QoS policy control function associated with the calling user's service provider;
- QFE6 — The transport coordination function serving the calling user;
- QFE7 — The transport coordination function serving the called user;
- QFE8 — An intervening QoS coordination function. This QFE is responsible for establishing a particular QoS within an intervening domain;
- QFE9 — An intervening transport coordination function;
- Called User — The application in the called user's terminal at which the service request is terminated.

The following functional relationships shall exist between these QFEs:

- **ra** — between the Calling User and the Calling User's service agent (QFE1);
- **rb** — between the Calling User's service agent (QFE1) and the Calling User's QoS coordination function (QFE2);
- **rc** — between QoS coordination functions in the originating side (QFE2) and an intervening QoS coordination function (QFE8);
- **rd** — between an intervening QoS coordination function (QFE8) and the Called User's QoS coordination function (QFE3);
- **re** — between the Called User's QoS coordination function (QFE3) and the Called User's service agent (QFE4);
- **rf** — between the Called User's service agent (QFE4) and the Called User;
- **rg** — between the Calling User's service agent (QFE1) and the QoS Policy Control function associated with the Calling User (QFE5);
- **rh** — between the originating QoS coordination function (QFE2) and the originating transport coordination function (QFE6);
- **ri** — between an intervening QoS coordination function (QFE8) and the transport coordination function associated with it (QFE9);
- **rj** — between the terminating QoS coordination function (QFE3) and the terminating transport coordination function (QFE7).

Figure A.2 shows the Functional Entities (QFE) and the relationships between them.



NOTE: This model represents a simple view of the essential functional elements required to implement the service.

Figure A.2: End-to-end QoS signalling functional entity model

A.2.1.2 Description of functional entities

A.2.1.2.1 Calling user

The Calling User functional entity acts on behalf of the human user to request the establishment of an end-to-end call with a specific QoS.

A.2.1.2.2 Calling user's service agent, QFE1

On receipt of a QoS establishment request from the Calling User, QFE1 determines whether current policy permits the requested QoS to be used with the call and either initiates a simple call establishment request towards the destination address or rejects the call request back to the calling user.

NOTE: and so on for the remainder of the FEs. This description should act as justification for why the FE appears in the model.

A.2.2 Information flows

A.2.2.1 Definition of information flows

A.2.2.1.1 Relationship ra

A.2.2.1.1.1 OrigQoSEstab

`OrigQoSEstab` is a confirmed information flow that shall be sent across relationship ra from the calling user to QFE1 to indicate a request for a new call establishment with a specific end-to-end QoS. Table A.2 lists the elements within the `OrigQoSEstab` information flow.

Table A.2: Content of OrigQoSEstab

OrigQoSEstab			
Information element	**Value**	**Request**	**Response**
QoS Service Class	- Predefined - TIPHON QoS class - 3 Best - 2H High - 2M Medium - 2A Acceptable - 1 Best effort - Non-standardized QoS class	M	
Called user ID	TIPHON user name	M	
Codec	- List of possible codecs - Codec type - Frames per packet	M	O (see notes 1 and 2)
Result	- End-to-End QoS Established - with requested QoS - Rejection cause - Requested QoS not available - Called user unknown - No compatible codec available - Policy Rejection		M
NOTE 1: The list of codecs shall be limited to a single entry in the response. NOTE 2: This element shall be included if the Result element is set to "End-to-End QoS Established".			

A.2.2.1.2 Relationships rb, rc, rd and re

A.2.2.1.2.1 QoSEstab

`QoSEstab` is a confirmed information flow that shall be sent across relationships rb, rc, rd and re to indicate a request for the provision of a guaranteed end-to-end QoS for the associated TIPHON simple call.

NOTE: The information flows belong to information relationships between FEs and should be grouped accordingly. A short text description of the purpose of the flow followed by a tabular grouping of the elements is appropriate.

A.2.2.2 Information flow sequences

A standard specifying TIPHON meta-protocols for end-to-end QoS Signalling shall provide signalling procedures in support of the information flow sequences specified below. In addition, signalling procedures should be provided to cover other sequences arising from error situations, interactions with simple call and interactions with other service capabilities.

In the figures, end-to-end QoS Signalling information flows are represented by solid arrows. Within a column representing a QoS Signalling functional entity, the numbers refer to functional entity actions listed in the MSC diagrams.

The following abbreviations are used:

- req request;
- resp response.

A.2.2.2.1 Normal operation

Figure A.3 shows the information flows for the successful establishment of End-to-End QoS.

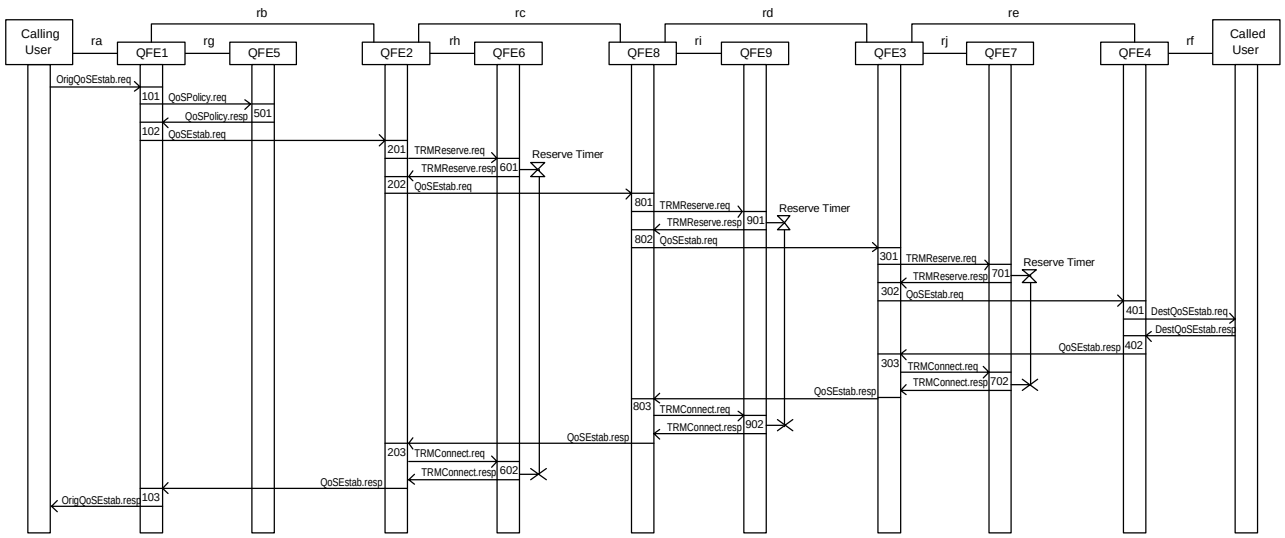


Figure A.3: Information flows for successful QoS establishment

The information flows in figure A.4 for successful QoS establishment can also be expressed in the form of a UML collaboration diagram as shown in figure A.1.

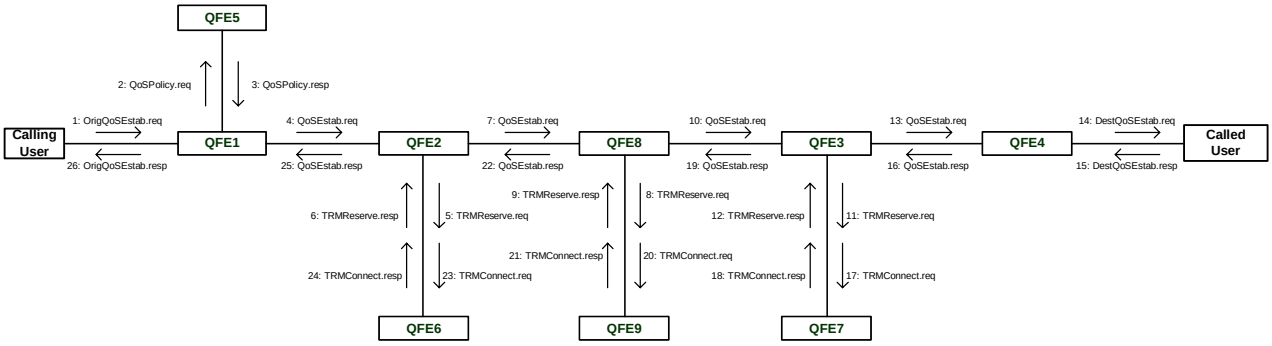


Figure A.4: Successful QoS establishment shown in a collaboration diagram

NOTE: The intention here is to place all of the information in the preceding sections into context. The normal approach is the MSC format although the UML collaboration diagram looks a bit more like the FE model presented earlier. Every attempt should be made to get the diagram onto a single page.

A.2.3 QoS Functional entity actions

Throughout the descriptions of QFE actions, the following conventions are used to identify information flows:

- an information flow is referred to as a "request" at the QFE that sends it and as an "indication" at the QFE that receives it;
- the corresponding confirmation is referred to as a "response" at the QFE that sends it and as a "confirmation" at the QFE that receives it.

The following QFE actions shall occur at the points indicated in the information flow diagrams.

A.2.3.1 Actions of QFE1

101: On request from the Calling User (`OrigQoSEstab` indication), determine transport parameters to be used, formulate a `QoSPolicy` request and send it to QFE5;

102: Receive a positive `QoSPolicy` confirmation indicating that the call is permitted with the requested QoS, formulate a `QoSEstab` request and send it to QFE2;

103: Receive a positive `QoSEstab` confirmation, formulate a positive `OrigQoSEstab` response and send it to the Calling User;

104: Receive a negative `QoSEstab` confirmation, formulate a corresponding negative `OrigQoSEstab` response and send it to the Calling User;

105: Receive a negative `QoSPolicy` confirmation, formulate a corresponding negative `OrigQoSEstab` response and send it to the Calling User;

106: Receive a positive `QoSPolicy` confirmation indicating that the call is permitted but with an alternative QoS to that requested by the Calling User, formulate a `QoSEstab` request and send it to QFE2;

107: Receive a positive `QoSEstab` confirmation, formulate a positive `OrigQoSEstab` response indicating that an alternative QoS to that requested has been used and send it to the Calling User.

A.2.3.2 Actions of QFE2

201: Receive a `QoSEstab` indication, formulate a `TRMReserve` request and send it to QFE6;

202: Receive a positive `TRMReserve` confirmation, formulate a `QoSEstab` request and send it to QFE8;

203: Receive a positive `QoSEstab` confirmation, formulate a `TRMConnect` request and send it to QFE6. On receipt of a positive `TRMConnect` confirmation, send a positive `QoSEstab` response to QFE1;

204: Receive a negative `QoSEstab` confirmation, send a `TRMRelease` request to QFE6 and send a negative `QoSEstab` response to QFE1;

205 (not shown): Receive a positive `QoSEstab` confirmation, formulate a `TRMConnect` request and send it to QFE6. On receipt of a negative `TRMConnect` confirmation, formulate a negative `QoSEstab` response, send it to QFE1 and stimulate the release of the TIPHON Simple Call towards the Called User.

NOTE: And so on for each FE and each point on the MSCs where some FE action is identified.

A.2.4 QoS Functional Entity behaviour

The behaviour specified in this subclause is intended to illustrate typical QFE behaviour in terms of information flows sent and received.

The behaviour of each QFE is shown using the Specification and Description Language (SDL) defined in ITU-T Recommendation Z.100 [7].

A.2.4.1 Behaviour of QFE1

The behaviour of QFE1 is shown in the SDL process diagram in A.5, A.6, A.7 and A.8.

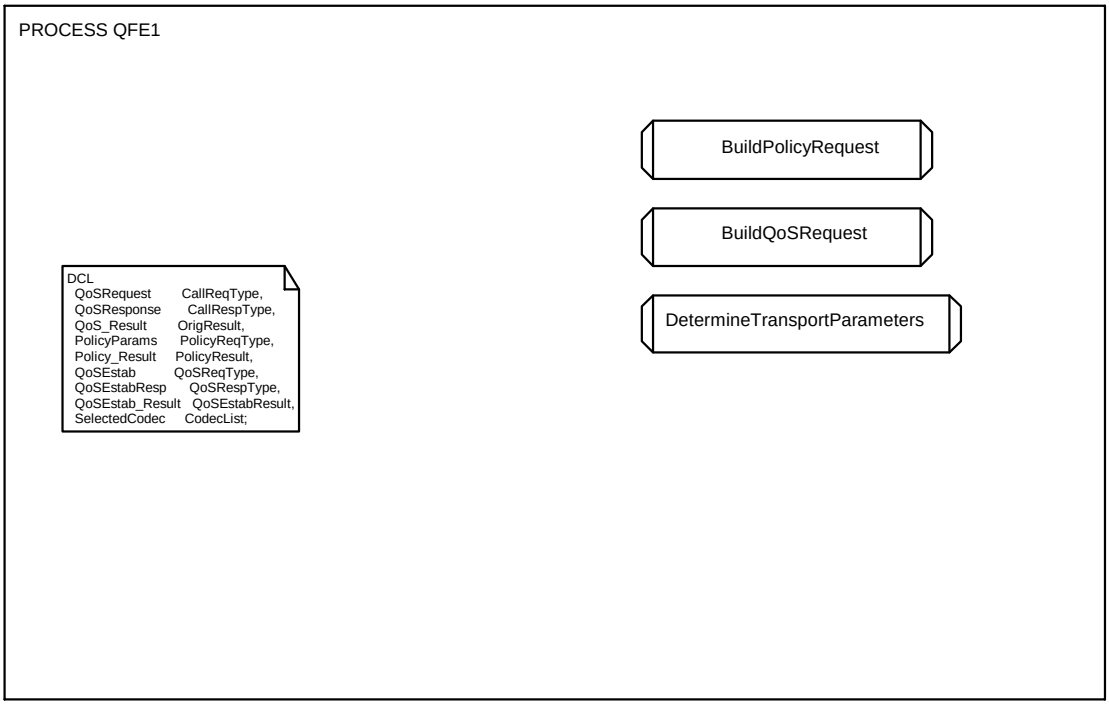


Figure A.5: SDL process diagram for functional entity QFE1 (1 of 4)

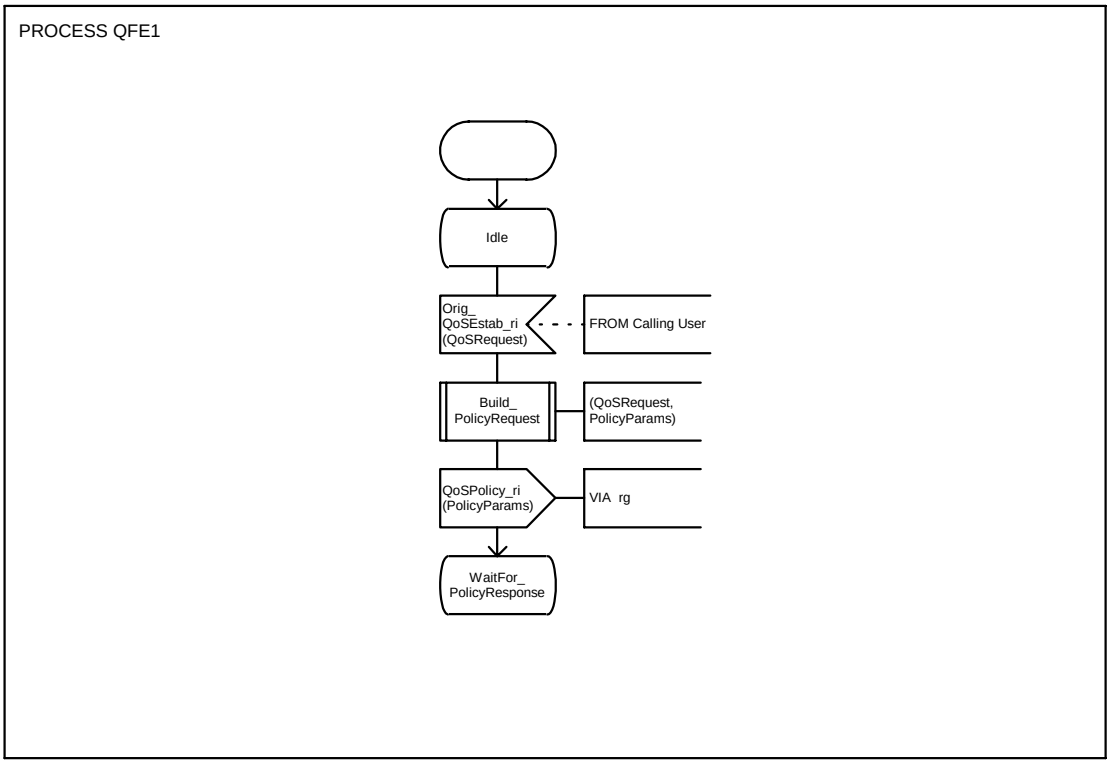


Figure A.6: SDL process diagram for functional entity QFE1 (2 of 4)

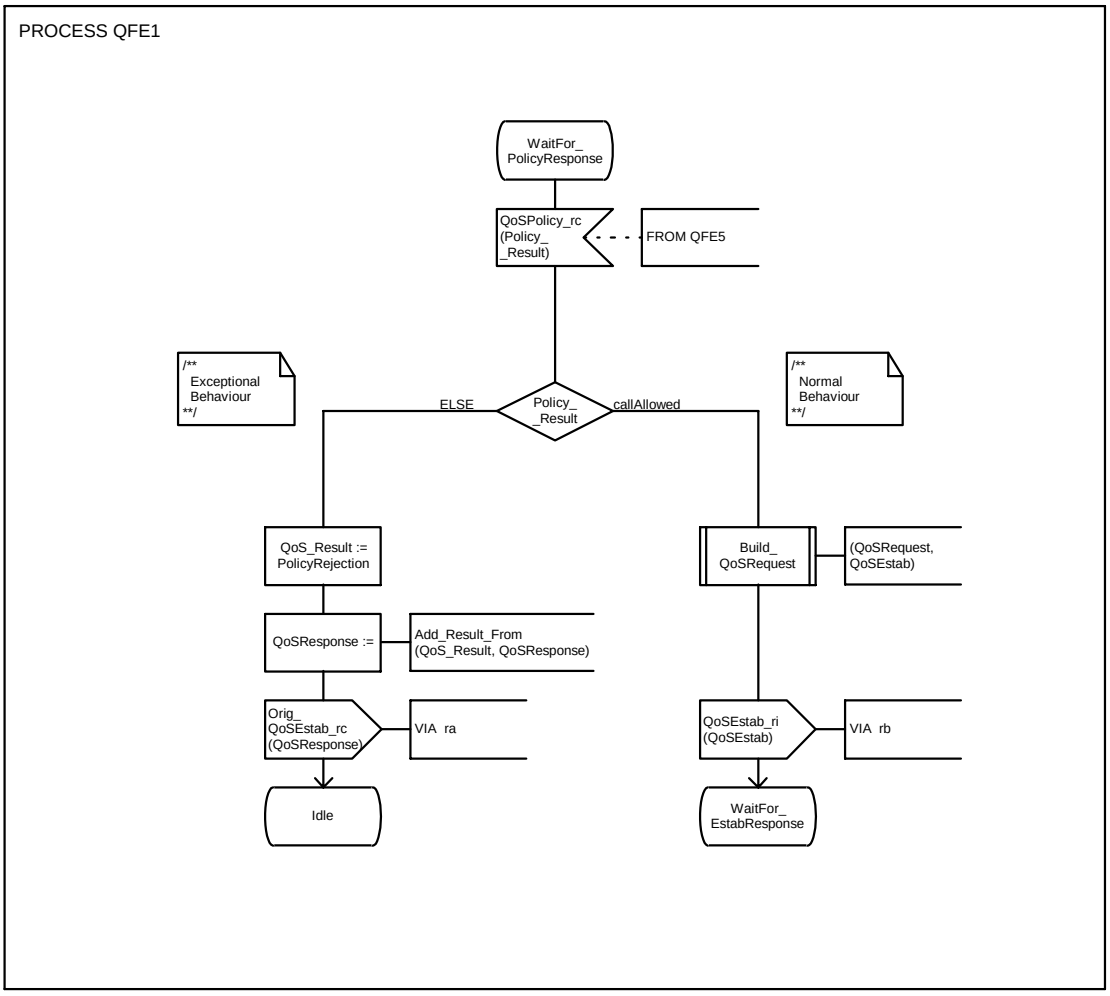


Figure A.7: SDL process diagram for functional entity QFE1 (3 of 4)

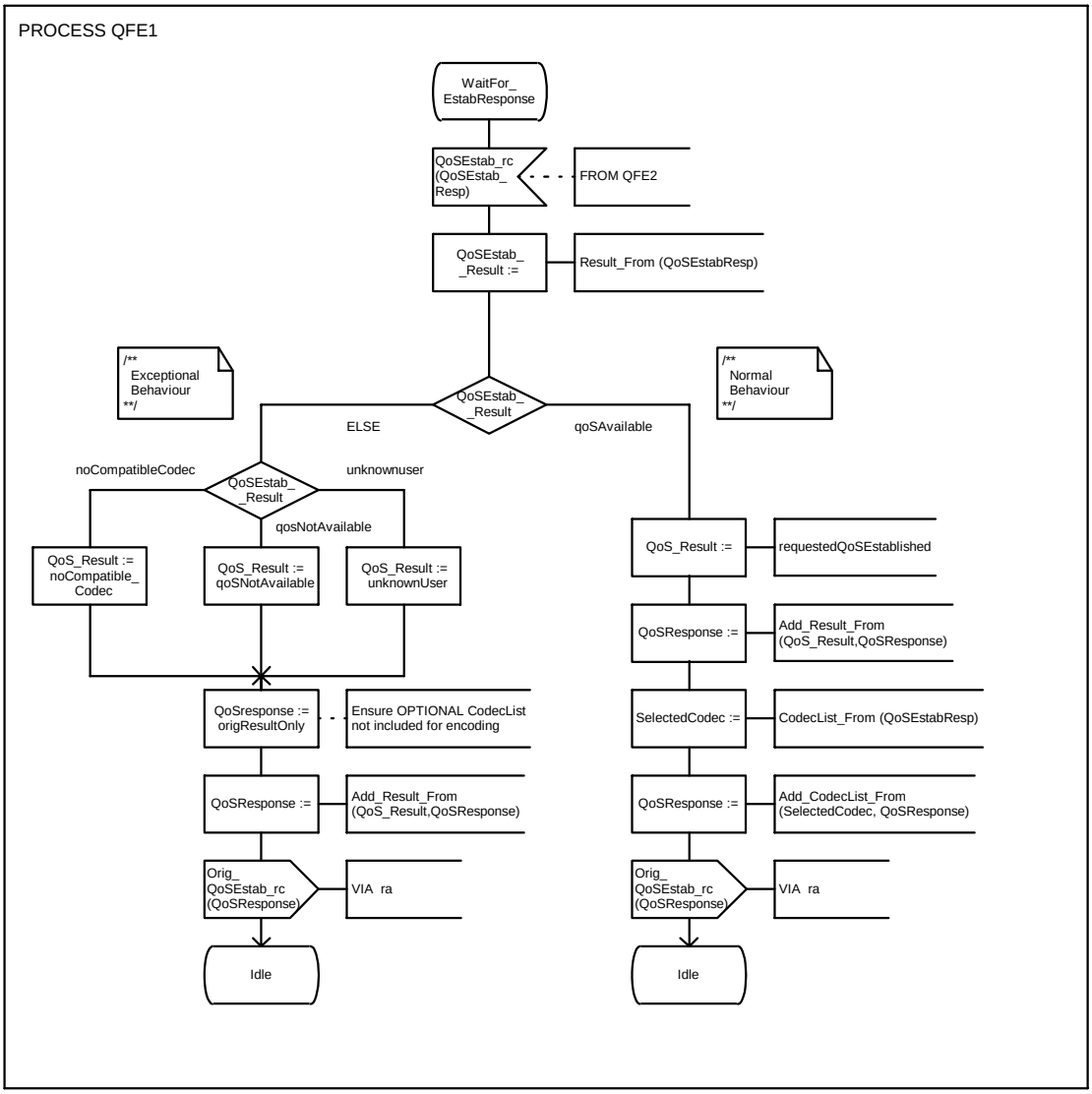


Figure A.8: SDL process diagram for functional entity QFE1 (4 of 4)

NOTE: And for each FE. The model should be built, validated and simulated.

Annex B (informative): Messaging capability meta-protocol

B.1 Derivation from service capability

The following example is derived from the service capabilities in the message class defined in TS 101 878 [4] and is used to illustrate the relationship between the service capabilities defined in TS 101 878 [4].

The class diagram from TS 101 878 [4] is replicated below (note that the normative source is of course TS 101 878 [4]).

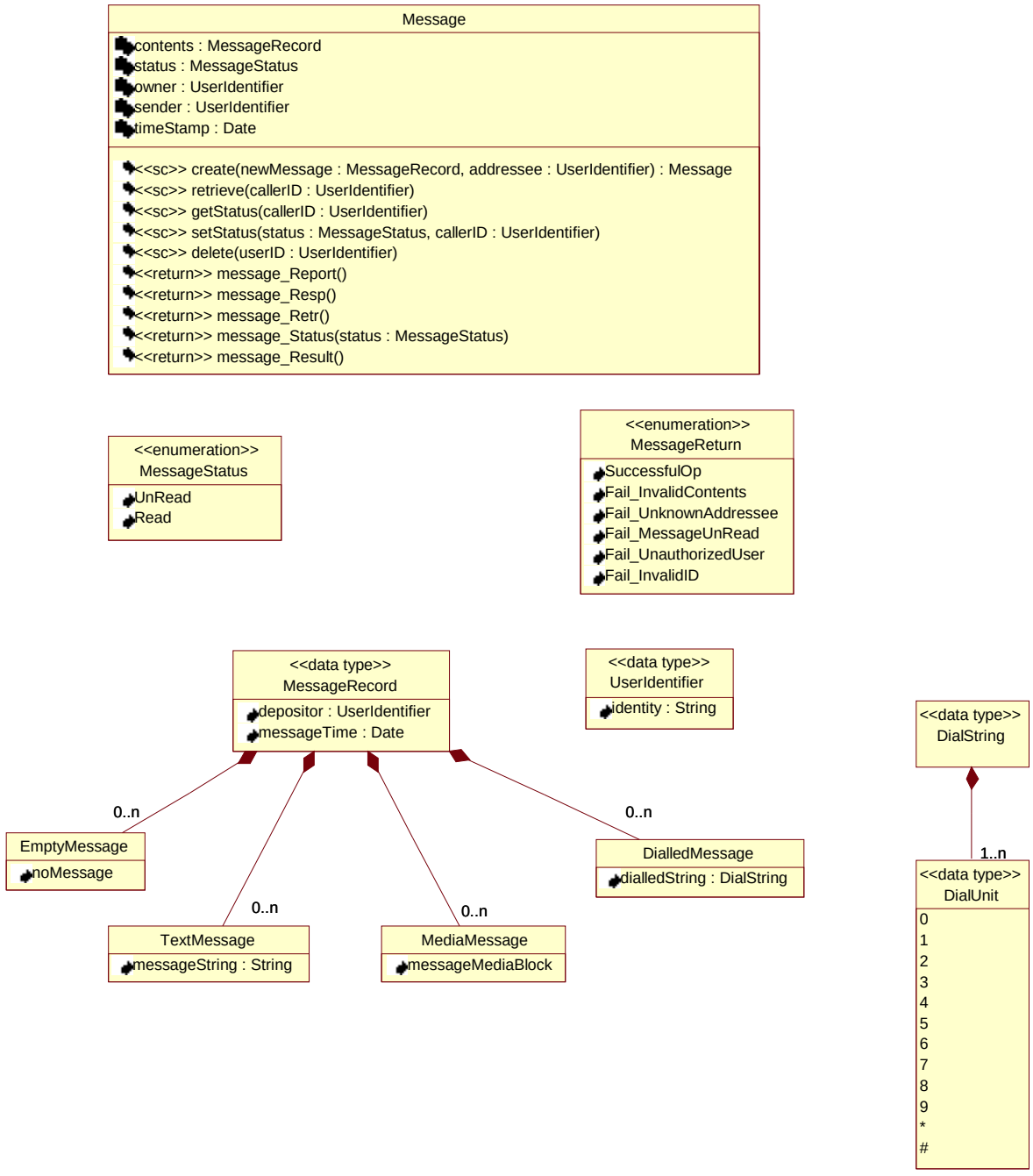


Figure B.1: Class diagram of message

B.2 SDL model

From the message class in figure B.1 a system in SDL is drawn as shown in figure B.2. This demonstrates a simple system containing a single block and two signal channels to the environment for each of the address and depositor (who are not modelled).

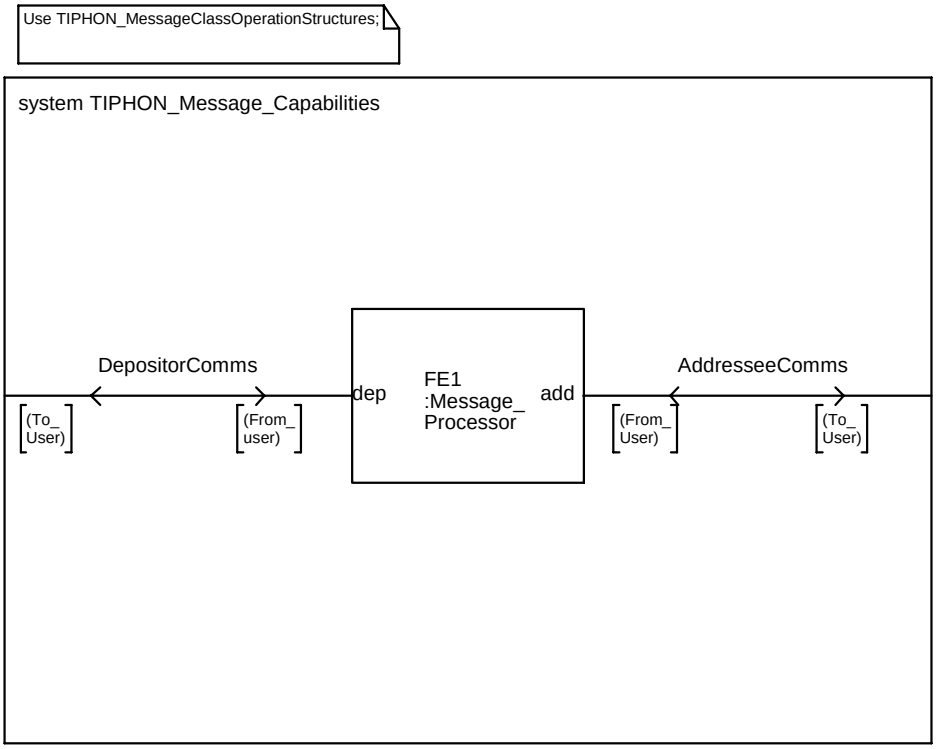


Figure B.2: System diagram of message (page 1 of 3)

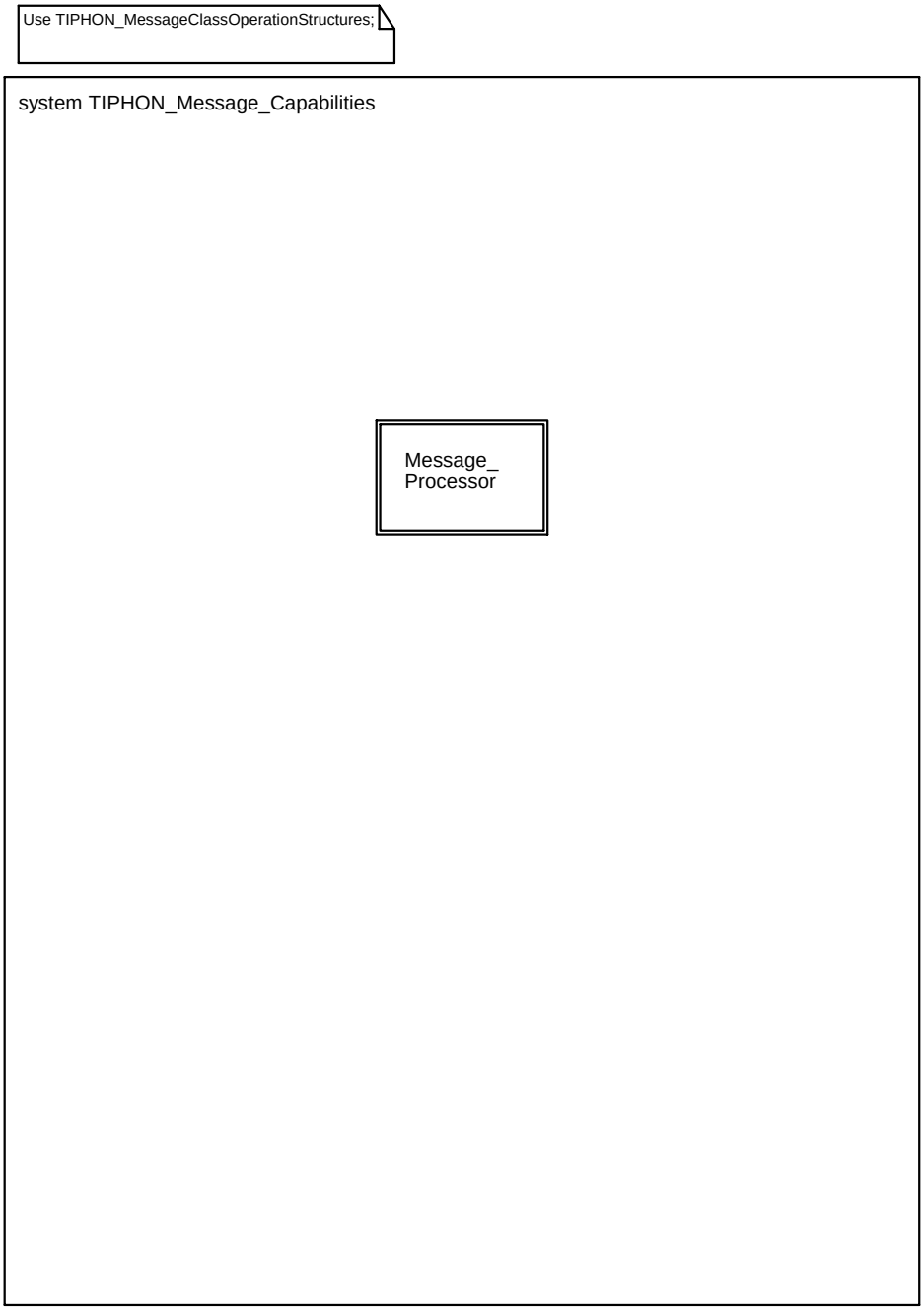


Figure B.3: System diagram of message showing one block type (page 2 of 3)

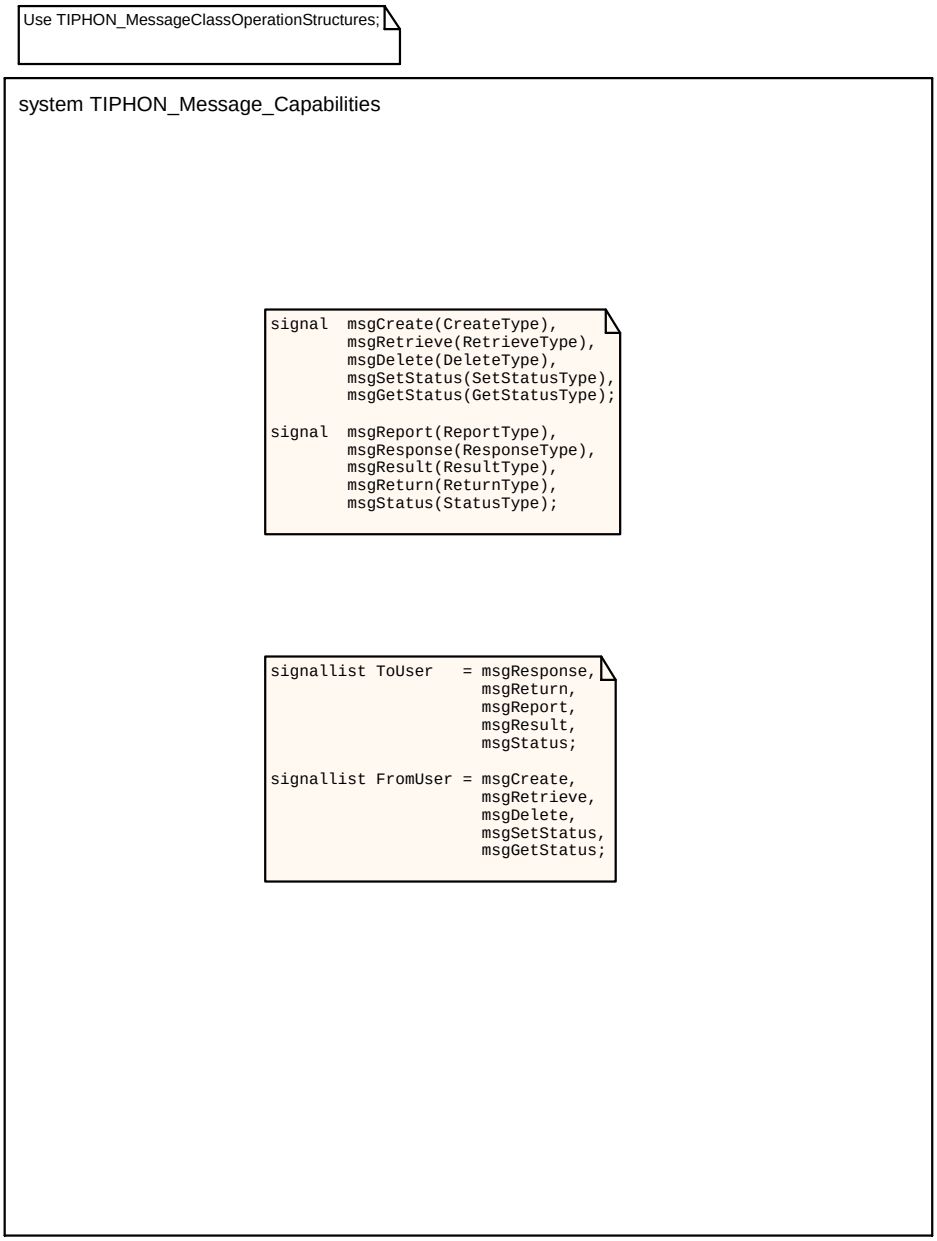


Figure B.4: System diagram of message showing signal lists (page 3 of 3)

From the system diagram a single block type is defined containing, in this case, a single process. Using block types as opposed to blocks allows some degree of object orientation in the design.

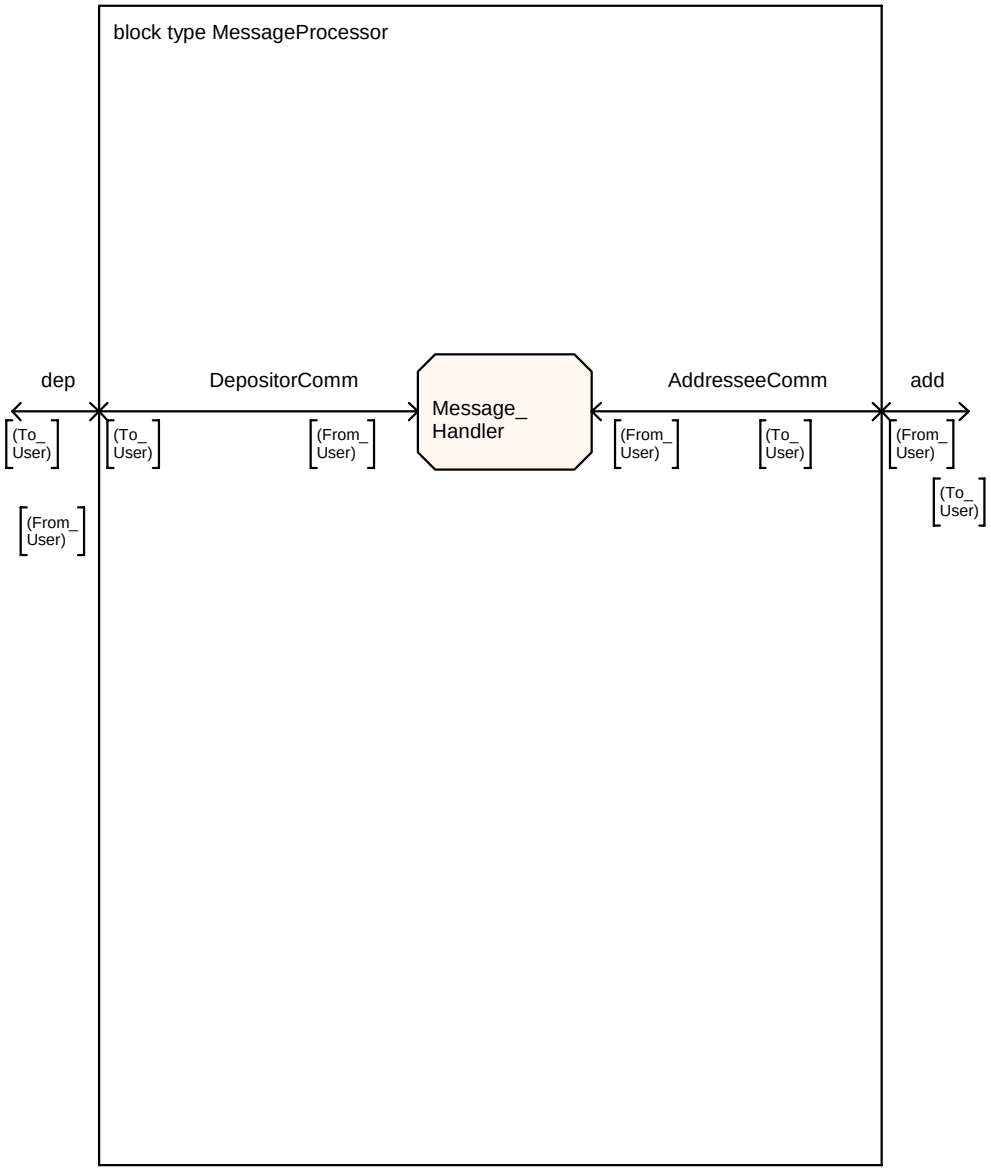


Figure B.5: Block type model of messaging

As we delve deeper into the model we now define the process itself. In this model we also show a number of procedures required although for the purpose of standardization these procedures are not fully defined other than by their boundary conditions.

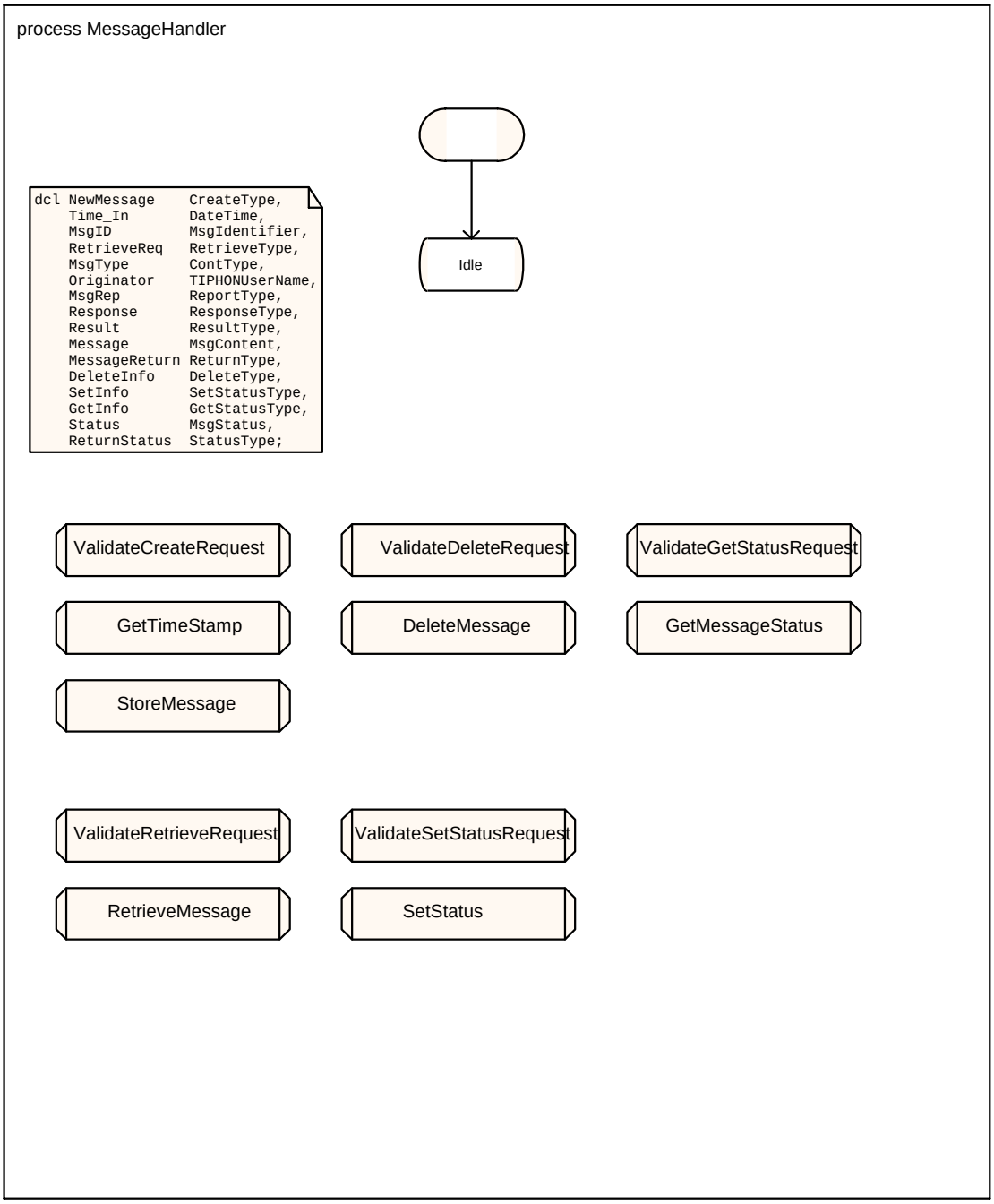


Figure B.6: Process model of messaging (page 1 of 6)

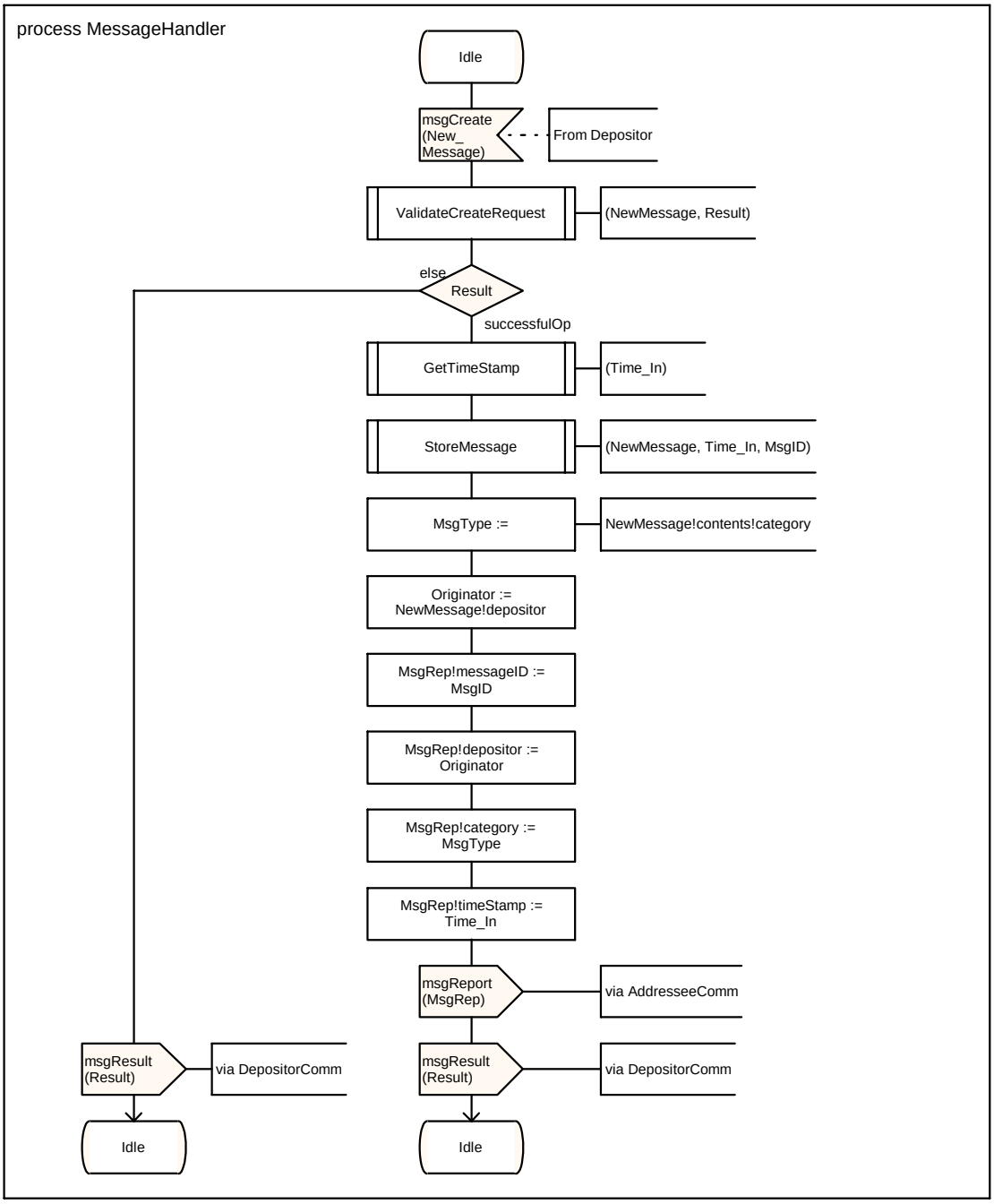


Figure B.7: Process model of messaging (page 2 of 6)

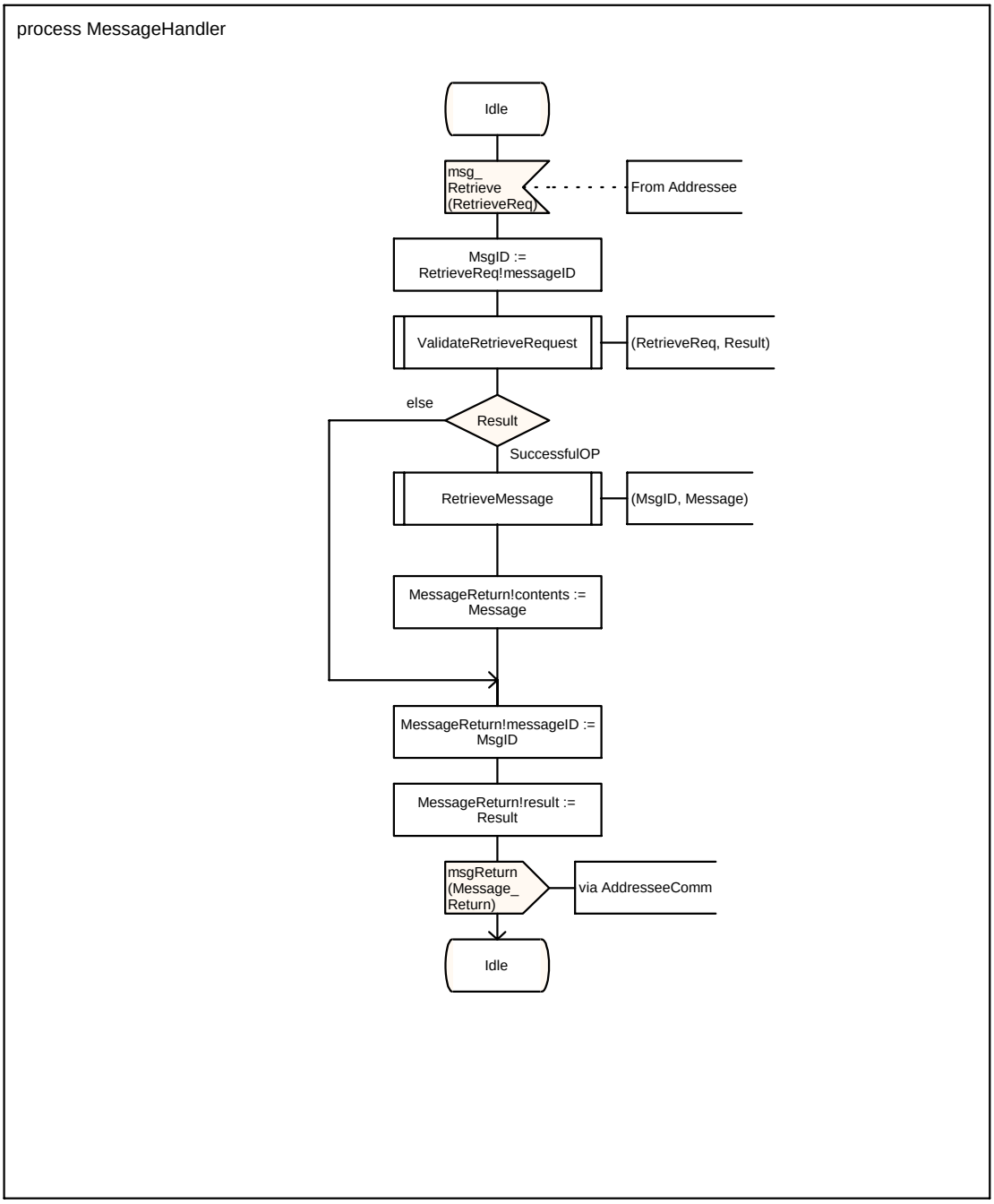


Figure B.8: Process model of messaging (page 3 of 6)

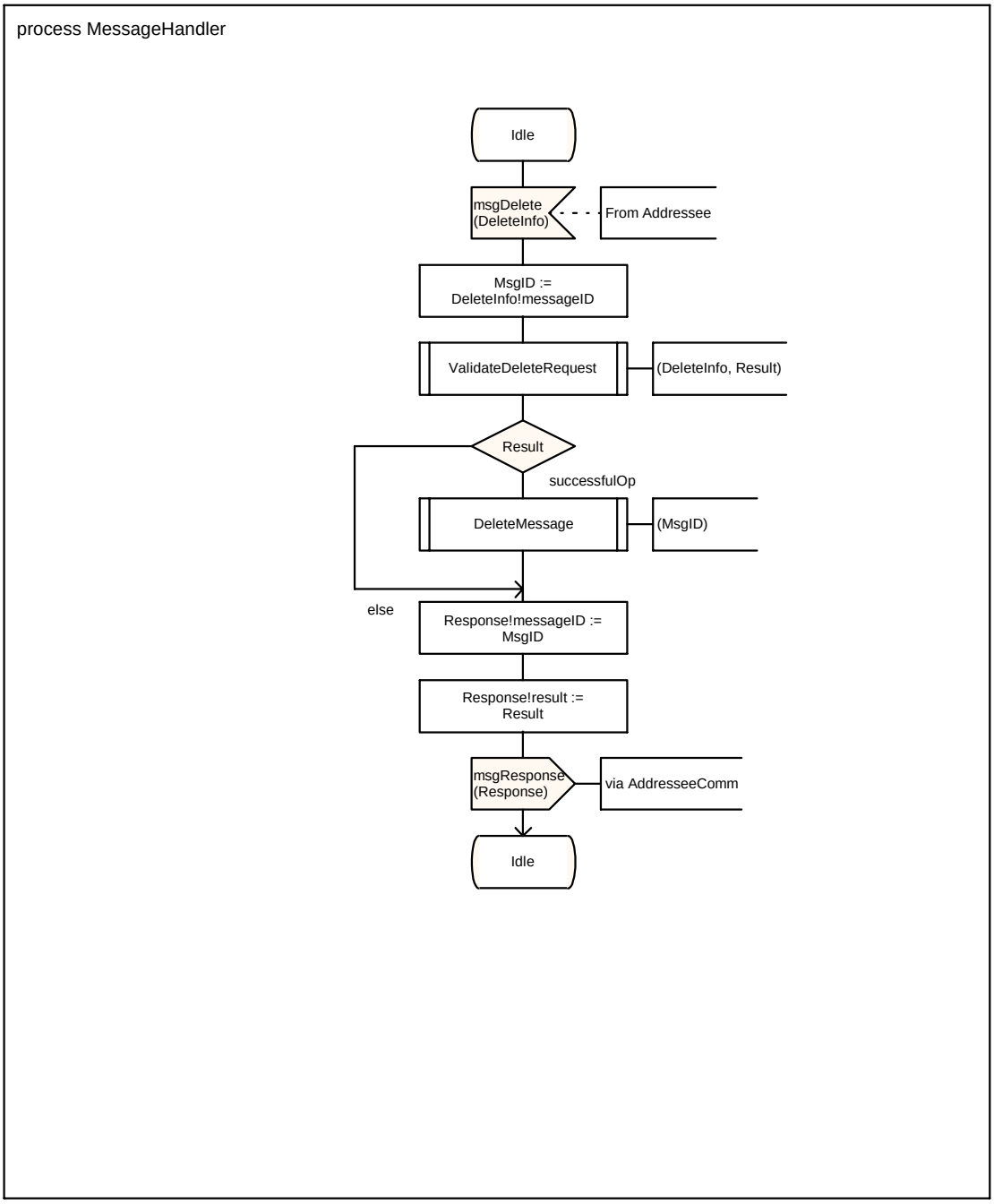


Figure B.9: Process model of messaging (page 4 of 6)

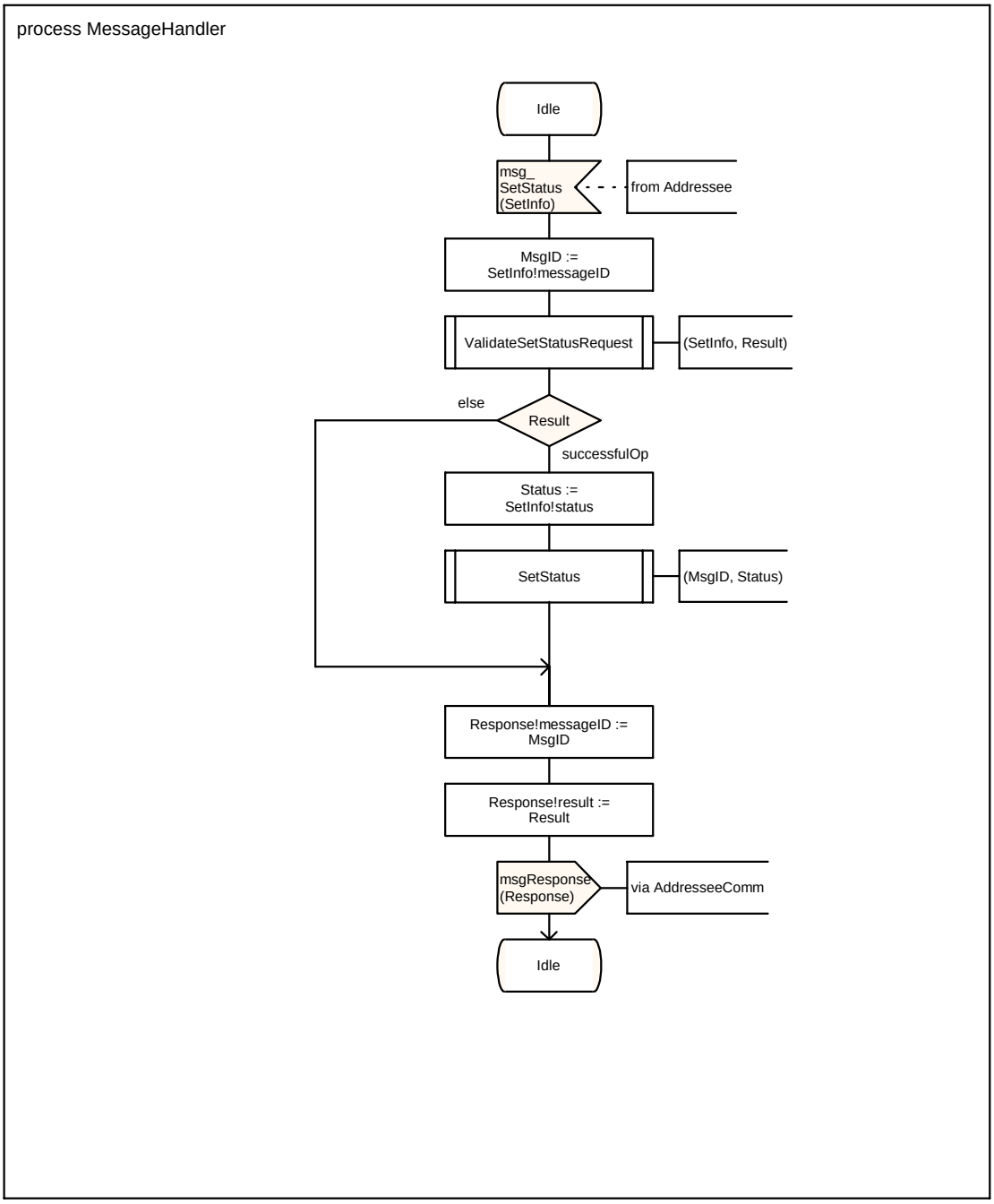


Figure B.10: Process model of messaging (page 5 of 6)

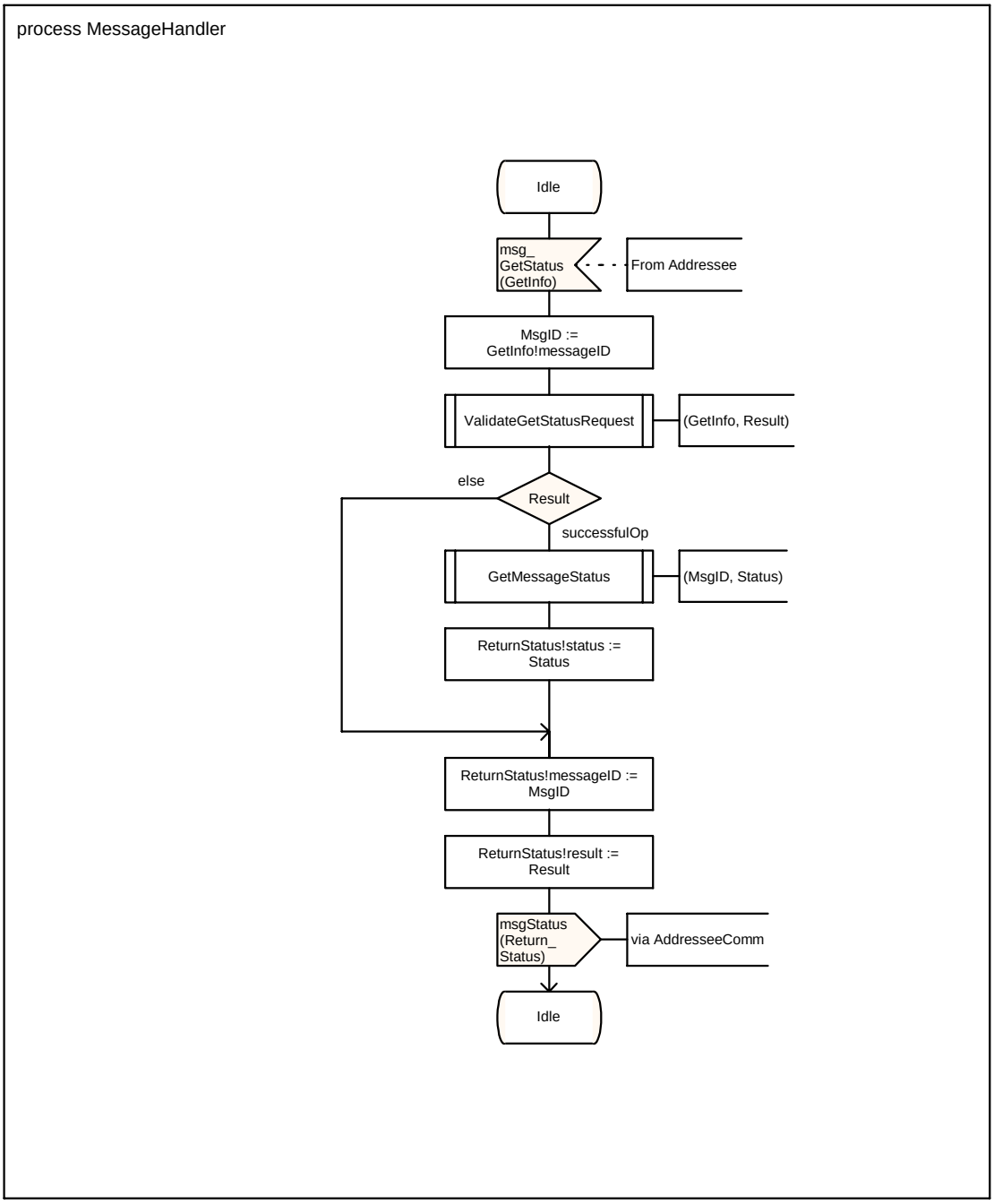


Figure B.11: Process model of messaging (page 6 of 6)

B.3 ASN.1 data definitions

```
TIPHON-MessageClassOperationStructures DEFINITIONS ::=

  BEGIN

-- Type definitions for the contents of Service Capability Requests associated with the Message
Class --

    -- Message Create --
    CreateType    ::= SEQUENCE    { owner     TIPHONUserName,
                                    depositor TIPHONUserName,
                                    contents  MsgContent                }

    -- Message Retrieve --
    RetrieveType  ::= SEQUENCE    { messageID MsgIdentifier,
                                    owner     TIPHONUserName            }

    -- Message Delete --
    DeleteType    ::= SEQUENCE    { messageID MsgIdentifier,
                                    owner     TIPHONUserName            }

    -- Set Message Status --
    SetStatusType ::= SEQUENCE    { messageID MsgIdentifier,
                                    owner     TIPHONUserName,
                                    status    MsgStatus                 }

    -- Get Message Status --
    GetStatusType ::= SEQUENCE    { messageID MsgIdentifier,
                                    owner     TIPHONUserName            }

-- Type definitions for the contents of responses associated with Message Service Capabilities --

    -- Report Incoming Message --
    ReportType    ::= SEQUENCE    { messageID MsgIdentifier,
                                    depositor TIPHONUserName,
                                    category  ContType,
                                    timeStamp DateTime                  }

    -- Signal Response --
    ResponseType  ::= SEQUENCE    { messageID MsgIdentifier,
                                    result    ResultType                }

    -- Retrieved Message --
    ReturnType    ::= SEQUENCE    { result      ResultType,
                                    messageID   MsgIdentifier,
                                    contents    MsgContent      OPTIONAL  }

    -- Retrieved Message Status --
    StatusType    ::= SEQUENCE    { result      ResultType,
                                    messageID   MsgIdentifier,
                                    status      MsgStatus       OPTIONAL  }

-- Miscellaneous data type definitions --

    ResultType    ::= ENUMERATED  { successfulOp,
                                    failInvalidID,
                                    failInvalidContents,
                                    failUnknownAddressee,
                                    failMessageUnread,
                                    failUnauthorizedUser                }

    TIPHONUserName::= E164Number

    E164Number    ::= NumericString (SIZE (1..15))

    MsgContent    ::= SEQUENCE    { category    ContType,
                                    content     CHOICE    { null    EmptyContent,
                                                            dialled DialledContents,
                                                            text    TextContent,
                                                            media   MediaContent   }
                                                          }
```

```
    ContType      ::= ENUMERATED  { noMessage,
                                    dialledMessage,
                                    textMessage,
                                    mediaMessage                        }

    EmptyContent  ::= NULL

    DialledContents ::= IA5String (FROM ("0".."9"|"*"|"#"))(SIZE (1..100))

    TextContent   ::= VisibleString (SIZE (1..1024))

    MediaContent  ::= SEQUENCE    { codec       Codec,
                                    media       BIT STRING                }

    Codec         ::= ENUMERATED  { a1016,
                                    celB,
                                    dVI4,
                                    g721,
                                    g722,
                                    g722-1,
                                    g723-1,
                                    g728,
                                    g729,
                                    gSM,
                                    h261,
                                    h263,
                                    iSO1449,
                                    jPEG,
                                    l16,
                                    lPC,
                                    mP2T,
                                    mPA,
                                    mPAA,
                                    mPV,
                                    nV,
                                    pCMA,
                                    pCMU,
                                    vDVI                                }

    MsgStatus     ::= ENUMERATED  { unRead,
                                    read                                 }

    DateTime      ::= SEQUENCE    { year        INTEGER (1..3000),
                                    month       INTEGER (1..12),
                                    day         INTEGER (1..31),
                                    hour        INTEGER (0..23),
                                    minute      INTEGER (0..59),
                                    second      INTEGER (0..59),
                                    milliSecond INTEGER (0..999)         }

    MsgIdentifier ::= VisibleString (SIZE (1..24))

END
```

Annex C (informative): Bibliography

- ETSI TS 101 315: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 4; Information Flow and Reference Point Definitions; Guidelines for application of TIPHON functional architecture to inter-domain services"
- ITU-T Recommendation I.130: " Method for the characterization of telecommunication services supported by an ISDN and network capabilities of an ISDN".

History

Document history		
V1.1.1	May 2002	Publication as TS 101 882
V4.1.1	September 2003	Publication