INTERNATIONAL TELECOMMUNICATION UNION

ITU-T M.3020

(02/2000)

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

SERIES M: TMN AND NETWORK MAINTENANCE: INTERNATIONAL TRANSMISSION SYSTEMS, TELEPHONE CIRCUITS, TELEGRAPHY, FACSIMILE AND LEASED CIRCUITS

Telecommunications management network

TMN interface specification methodology

ITU-T Recommendation M.3020

(Formerly CCITT Recommendation)

ITU-T M-SERIES RECOMMENDATIONS

TMN AND NETWORK MAINTENANCE: INTERNATIONAL TRANSMISSION SYSTEMS, TELEPHONE CIRCUITS, TELEGRAPHY, FACSIMILE AND LEASED CIRCUITS

Introduction and general principles of maintenance and maintenance organization	M.10–M.299
International transmission systems	M.300–M.559
International telephone circuits	M.560–M.759
Common channel signalling systems	M.760–M.799
International telegraph systems and phototelegraph transmission	M.800–M.899
International leased group and supergroup links	M.900–M.999
International leased circuits	M.1000–M.1099
Mobile telecommunication systems and services	M.1100–M.1199
International public telephone network	M.1200–M.1299
International data transmission systems	M.1300–M.1399
Designations and information exchange	M.1400–M.1999
International transport network	M.2000–M.2999
Telecommunications management network	**M.3000–M.3599**
Integrated services digital networks	M.3600–M.3999
Common channel signalling systems	M.4000–M.4999

For further details, please refer to the list of ITU-T Recommendations.

ITU-T Recommendation M.3020

TMN interface specification methodology

Summary

This ITU-T Recommendation describes the TMN interface specification methodology UTRAD (Unified TMN Requirements, Analysis and Design). It describes the process to derive interface specifications based on user requirements, analysis and design (RAD). Guidelines are given to describe RAD using Unified Modelling Language (UML) notation; however, other interface specification techniques are not precluded. The guidelines for using UML are described at a high level in this ITU-T Recommendation.

Source

ITU-T Recommendation M.3020 was prepared by ITU-T Study Group 4 (1997-2000) and approved under the WTSC Resolution 1 procedure on 4 February 2000.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications. The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Conference (WTSC), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSC Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementors are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database.

CONTENTS

ITU-T Recommendation M.3020

TMN interface specification methodology

1 Introduction

1.1 Scope

This ITU-T Recommendation describes the TMN interface specification methodology UTRAD (Unified TMN Requirements, Analysis and Design). It describes the process to derive interface specifications based on user requirements, analysis and design (RAD). Guidelines are given to describe RAD using Unified Modelling Language (UML) notation; however, other interface specification techniques are not precluded. The guidelines for using UML are described at a high level in this ITU-T Recommendation. Further ITU-T Recommendations in this series will provide a more detailed definition of the specific use of UML notation within the TMN.

An interface specification addresses management service(s) defined in ITU-T Recommendation M.3200. Such a specification may support part of or one or more management services. The management services comprise of management functions. These functions may reference those defined in ITU-T Recommendation M.3400, specialize to suit a specific managed area or new functions may be identified as appropriate.

1.2 Related Recommendations

The following ITU-T Recommendations should be referred to in connection with this ITU-T Recommendation:

[1] ITU-T Recommendation M.3010 (2000), *Principles for a telecommunications management network.*

[2] ITU-T Recommendation M.3200 (1997), *TMN management services and telecommunications managed areas: Overview.*

[3] ITU-T Recommendation M.3400 (2000), *TMN management functions.*

[4] Unified Modelling Language, Section 1 of *OMG Modelling*, OMG Doc. No. Formal/ 99-06-01.

[5] ITU-T Recommendation M.3208.1 (1997), *TMN management services for dedicated and reconfigurable circuits network: Leased circuit services.*

[6] ITU-T Recommendation X.680 (1997) | ISO/IEC 8824-1:1998, *Information technology – Abstract syntax Notation Om (ASN.1): Specification of basic notation*.

[7] ITU-T Recommendation Z.100 (1999), *Specification and Description Language*.

1.3 Abbreviations

This ITU-T Recommendation the uses the following abbreviations:

ASN.1 Abstract Syntax Notation One

CMIP Common Management Information Protocol

CNM Customer Network Management

CORBA Common Object Request Broker Architecture

GDMI Guidelines for the Definition of Management Interface

GDMO Guidelines for the Definition of Managed Objects

GRM General relationship Model

IDL Interface Definition Language

LCS Leased Circuit Service

NE Network Element

OAM&P Operations, Administration, Maintenance and Provisioning

OMG Object Management Group

OO Object Oriented

OS Operations System

OSI Open Systems Interconnection

SC Service Customer

SDL Specification and Description Language

SLA Service level Agreement

SP Service Provider

TMN Telecommunications Management Network

UML Unified Modelling Language

UTRAD Unified TMN Requirements, Analysis and Design

1.4 Definitions

This ITU-T Recommendation defines the following definitions terms:

1.4.1 TMN management goals: High-level objectives of a user in performing management activities.

1.4.2 TMN management roles: TMN management roles define the activities that are expected of the staff or system that perform telecommunications management. TMN management roles are defined independent of other components, i.e. telecommunications resources and TMN management functions.

1.4.3 telecommunications resources: Telecommunications resources are physical or logical entities requiring management, using TMN management services.

1.4.4 TMN management scenario: A TMN management scenario is an example of management interactions from a management service.

1.4.5 Terms imported from M.3010

The following terms from M.3010 [1] are used in this ITU-T Recommendation.

– User.

– TMN management service.

– TMN management function set.

1.4.6 Terms imported from UML

The following terms from UML [4] are used in this ITU-T Recommendation.

– Activity Diagram.

– Actor.

– Class.
– Class Diagram.
– Collaboration Diagram.
– Sequence Diagram.
– State Diagram.
– Stereotype.
– Use Case.

1.5 Requirements for methodology and notational support

In developing the methodology and choosing a notation, the following goals were used.

1) The notation and methodology shall support the capture of all the relevant requirements of the problem space, namely telecommunications management.
2) The notation shall facilitate unambiguous generation of the specification in the target NM paradigms specified in ITU-T Recommendation Q.812.
3) Non-optional conformance points shall be specified in all three phases. If optional features are required to support the telecommunications problem space these will be specified. For the Requirements and Analysis phases, the allowable optional features will be specified in this series of ITU-T Recommendations. Allowable optional features for the design phase will be specified in the Q.81x-series Recommendations.
4) It shall be possible to generate, from the design specification, interoperable language specific definitions (for example UML to IDL, UML to GDMO/ASN.1).

The current chosen notation, as noted later does not meet all the above requirements. However, it is expected that these requirements should be met as the notations are applied widely in the industry.

The optional features to be supported include features of the notation for the three phases as well as for target NM paradigm specific capabilities (e.g. selection of CORBA facilities and optional features within a given facility). These features are not included in the methodology but should be found in other ITU-T Recommendations.

1.6 Use of UML Notation

Table 1 identifies the correspondence between TMN concepts and UML notation. This ITU-T Recommendation specifies the high-level concepts and notations to be used in the different phases. Further recommendations in this series will describe the guidelines for using specific aspects of the notations, required extensions such as new stereotypes appropriate for use within TMN.

Stereotypes are used to extend UML notation. The approved stereotypes for use within the TMN environment are included in this ITU-T Recommendation (see Annex C).

Table 1/M.3020 – Requirements concepts

TMN concept	UML notation	Comment
user	Actor	A user is modelled as an actor.
management role	Actor	An actor plays a role. It is normally advisable to only model a single role for each actor.
management function	use case	A management function is modelled by one or more use cases.
management function set	use case	A management function set is a composite use case with each management function (potentially) modelled as a separate use case.
management service	use case	A management service is modelled as a high-level use case.
management scenario	sequence diagram	Sequence diagrams are preferred over collaboration diagrams.
telecommunication resource type	Class	The class diagrams depict the property details of the telecommunications resource type, at the level of detail appropriate to the phase of the methodology.
management goals	–	Management goals are captured as textual descriptions as there is no applicable UML notation.

2 Methodology

2.1 General considerations

The purpose of this methodology is to provide a description of the processes leading towards the definition of TMN interfaces.

2.2 Application and structure of the methodology

The Unified TMN Requirements, Analysis and Design (UTRAD) methodology specifies an iterative three-phase process with features that allow traceability across the three phases. The three phases apply industry-accepted techniques using object oriented analysis and design principles. The three phases are requirements, analysis and design. The techniques should allow the use or development of commercially available support tools. Different techniques may be used for the phases depending on the nature of the problem.

2.3 Detailed methodology

The requirements and analysis phases produce UML specifications. The Design phase uses Network Management Paradigm specific notation. The outputs of the 3 phases are:

- Requirements phase – Requirements.
- Analysis phase – Implementation independent specification.
- Design phase – Technology specific specification.

Initially, the design phase will be developed using a manual or customized approach. When interoperable protocol specific definition can be generated by tools, then UML notation can be applied to the design phase. However some protocol specific definitions, such as class hierarchy, can be depicted using UML notation.

The subclauses below describe the three phases.

2.3.1 Requirements

The requirements for the problem being solved fall into two classes. The first class of requirements is referenced here as business requirements. A subject matter expert on the topic shall be able to determine that the requirements adequately represent the needs of the management problem being solved. The second class is referred to as specification requirements. These requirements shall provide sufficient details so that the interface definition in the analysis and design phase can be developed. As final interface definitions must be traceable to the requirements, it may be necessary to have an iterative process among the three phases. Any ambiguity in the requirements will have to be resolved by this iterative process to assure that an implementable specification can be developed.

Different techniques may be used to specify the two classes of requirement. Irrespective of the technique, the readability of the requirements is critical. The requirements themselves are not required to be in a machine-readable notation as long as readability and traceability are possible. Enumerating requirements is one possible approach to delineate the different requirements for traceability.

The requirements phase include identifying aspects such as security policy, scope of the problem domain in terms of the applications, resources, and roles assumed by the resources. An example of the requirements is available in Appendix I. The requirements specify roles, responsibilities, and the relationships between the constituent entities for the problem space. Different techniques including textual representation may be used to specify the business level requirements. In order to facilitate traceability of these requirements to the design and implementation phases, enumerating requirements is recommended.

The problem must be bounded with a specific scope. One way to determine the scope is by using the management services identified in ITU-T Recommendation M.3200 and function sets identified in M.3400. Requirements are specified using the resources being managed and TMN management functions. Augmenting ITU-T Recommendation M.3400 may be required in order to meet the business requirements of the problem.

UML use cases and scenarios should be used to interact with subject matter experts in capturing the business level requirements. The requirements should also identify the failure conditions visible to the business process.

The requirements produced must be complete and detailed. The recursive nature of the UTRAD methodology is used to achieve this completeness. The completeness of the requirements (clear and well-documented) drives the analysis and design phases.

2.3.2 Analysis

In the analysis phase, the requirements are used to identify the interacting entities, their properties and the relationships among them. This allows the interfaces offered by the entities to be defined. In the UML notation, these entities become classes. The class descriptions along with the interfaces exposed should be traceable to the requirements. The relationship among the classes, defined in the analysis specification, and the classes in the design specification is not necessarily one to one.

This ITU-T Recommendation gives high-level guidance on the use of UML notation to support TMN interface specification; however SDL [7], an industry accepted technique might be used to augment the UML definitions.

The analysis phase should be independent of design constraints. For example the analysis may be documented using OO principles even though the design may use a non-object oriented technology. The information specified in the analysis phase includes class descriptions, data definitions, class relationships, interaction diagrams (sequence diagrams and/or collaboration diagrams), state transition diagrams and activity diagrams. The class definitions include specification of operations, signal (asynchronous stimulus such as receipt of operations, events and exceptions), attributes and behaviour captured as notes or textual description.

The generic definitions (operations and stereotypes) contained in Annex C are provided for use in the analysis phase. They define retrieving and setting multiple attributes and issuing notifications. These can be included in the class definitions and the interaction diagrams.

2.3.3 Design

In the design phase an implementable interoperable interface specification is produced. This will involve the selection of a target specification language. The design phase specifications are dependent on the specific TMN network management paradigm.

The selection of specific TMN paradigm is addressed in other TMN Recommendations.

In the context of the TMN paradigm based on OSI Systems Management, the design specification is the information model specification using GDMO templates for managed object classes, attributes, behaviour, notifications, actions, naming instances of the class, and error/exception specifications. The syntax of the information is specified using ASN.1 notation.

In GDMO, the object class hierarchy specifies the properties of the object classes that are needed for management. Extensive use of inheritance (super and subclasses) is needed to benefit the most from the reuse of specifications. The object classes are specified using the templates from ITU-T Recommendation X.722, structure of management information – Guidelines for the definition of managed objects. The templates defining the information model should be registered (according to the rules of ITU-T Recommendation X.722) with a value for the ASN.1 object identifier. Annex B describes the procedure for assigning the registration values. For those object classes that are already specified in other ITU-T Recommendations and ISO standards, only a reference to the particular Recommendation and object class is needed. Naming is not a part, nor the purpose, of the object class hierarchy.

In the context of CORBA based TMN, the information model is defined using IDL.

As additional paradigms are added to the TMN, the notations/languages defined by these paradigms will be used.

In the design phase, it is recommended that the UML descriptions from the requirements and analysis phases be referenced to augment behavioral specification. For example, behaviour definition of GDMO can reference state charts, sequence diagrams and class definition in the analysis phase. If required additional UML diagrams describing interactions between entities, corresponding to specific protocol paradigms, may be included.

2.4 TMN interface specifications

A TMN interface specification includes the Requirements, Analysis and Design specifications discussed in 2.3. A structure for specifying these specifications is provided in Annex A and is called Guidelines for the Definition of Management Interface (GDMI).

These techniques and supporting notations are also applicable when designing a system to the TMN interface specifications, even though system design is not considered as part of TMN Recommendations. They assist in describing how the interface specifications are applied in managing the resources within a system such as an NE.

2.5 Traceability in UTRAD Process

In order to achieve traceability between requirements, analysis and design , it is necessary that appropriate identification and pointers are provided by each model element. For example the requirements can be identified by numbers or references to the functions in ITU-T Recommendation M.3400 list of functions. Numbering other requirements (new functions not in ITU-T Recommendation M.3400) or security policy and any performance requirements is also recommended because a design specification may meet these requirements differently based on the

underlying protocols. Another approach used in the example in Appendix I (I.2.2.3) is to reference the use cases and the associated textual description. The analysis phase output specifies for the various use cases further detailed information requirements. The design phase should point to the various diagrams and text in the analysis phase output. The pointer may be in terms of a reference to the appropriate sections.

An iterative process may be required to trace up to the subject matter level requirements in the first phase from the design phase. This is required because the output of the phases are defined to different level of details.

NOTE – Not all requirements will be traceable in the design. For example requirements such as availability, redundancy, etc. may not be reflected in particular interface design even though they are to be supported in an implementation. There is no formal mechanism defined in this ITU-T Recommendation for traceability of requirements between the three phases. One approach is to reference clauses and subclauses in the output of the requirements and analysis phases during the design phase.

2.6 Documentation structure

Even though there are three phases, the documentation of the interface may combine their outputs into one or more documents. It is recommended that the requirements and analysis be combined and separate design documents are developed for each specific network management protocol paradigm.

ANNEX A

Guidelines for the Definition of Management Interface (GDMI)

A.1 Introduction

The following are guidelines for the Definition of Management Interface (GDMI). The GDMI template provides a structure for specifying the outputs of the requirements, analysis and design phases.

A.2 GDMI Template

A.2.1 Scope

Define major goals and objectives and the applicable TMN interfaces (and Reference Points) for this specification. Use ITU-T Recommendation M.3200 [2] categorization as a source for identifying the management service(s) supported by this interface.

This subclause should give a clear description of the TMN users benefit, i.e. the reason for performing this management service. Background and context should be added as necessary, but the explanatory and descriptive part should be separated. Supporting background information, where required, should be placed in an appendix.

A.2.2 Requirements

A.2.2.1 Business level requirements

List major requirements in text, and identify use cases with actor/role and resources. The use case should bring out high-level requirements and is distinguished from the specification requirements by not refining to lower levels. Policies related information (e.g. security, persistence) are candidates for inclusion at this level. Numbering the requirements is recommended for traceability.

A.2.2.1.1 Actor roles

A textual description of the actor is included here.

A.2.2.1.2 Telecommunications resources

Textual description of the relevant resources required to support the use cases are presented here.

A.2.2.1.3 High-level use case

A high-level use case diagram is presented. In order to understand the use case by subject matter experts, they should be augmented with textual description for each use case. The description should serve two purposes: to capture the domain experts' knowledge and to validate the models in analysis and design phases with respect to the requirements. An example of a high-level use case is given in Appendix I.

The high-level use cases may identify the various functions sets defined in ITU-T Recommendation M.3400 [3]. These use cases may be further refined as described in the specification requirement subclause below by using stereotypes such as "include" and "extend".

If appropriate, sequence and state chart diagrams may be used. However, at the high-level requirements these diagrams are not expected to be used. When the use cases at this level are further decomposed in the next level of requirements, these diagrams may be more suitable.

The traceability of the next level of requirements from this level may be identified by how each function set further refined with new use cases.

A.2.2.2 Specification level requirements

The high-level use cases are further refined using management functions from ITU-T Recommendation M.3400. Since M.3400 is not exhaustive enough to address all management services for all managed areas, it is expected that new functions will be required. The new functions should be included in the requirements as described below.

A.2.2.2.1 Actor roles

A list of all actors and textual description of actors not already defined in high-level requirements is included here.

A.2.2.2.2 Telecommunications resources

A list of all passive resources and textual description of resources not already defined in high-level requirements are presented here.

A.2.2.2.3 TMN management functions

Management functions identify the interactions between the different actor roles. The requirements may include one or more of the following – use cases, sequence and state charts diagrams for various functions in the problem domain and textual descriptions.

A.2.2.2.4 Use cases

An example of the refinement of the high-level use case diagrams above is presented in Appendix I. The refinement is achieved by using "extend" and "include" stereotypes.

If appropriate sequence and state chart diagrams may be used.

A.2.2.3 Analysis

The analysis clause includes functional decomposition, information flows, class diagrams (including relationships between classes), sequence diagrams and state charts/tables. The class diagrams may be augmented with details of attributes and allowed operations. As with any UML diagram, textual description is required to augment the figures. Including all the attributes along with their properties within the class diagram may hinder the readability. As a presentation issue, it is permissible not to show attributes in all class diagrams.

Detailed descriptions of the Management Functions and interactions between the functions shall be provided.

The information flow associated with each function should generally be captured using simple tables defining the flow. The tables should specify whether the information is mandatory, optional or present subject to a condition. The condition should be defined. The analysis may include state models as a result of information flow. The state transitions may be described using tables or diagrams identifying events and the resulting state. Another area to consider when documenting analysis of the requirements is the exception/error cases.

The scenarios describing the information flow amongst the entities may be described using the sequence diagrams as illustrated in Appendix I.

The sequence diagram is a message flow diagram with time going vertical and messages running horizontal. It models the manager interacting with the managed system via the management information model. Inside the diagram are the object instances, which collaborate via message passing to perform the given functional task.

Pre- and post-conditions may be used to describe the information flows in the interaction diagrams.

Reference numbers shall be assigned to the analysis sections and can be used for traceability from the model or referencing from the model where appropriate.

Traceability is to be provided for the various diagrams and text to the requirements phase by appropriate references to the use cases.

A.2.2.4 Design

Protocol paradigm specific information models are presented in this section (e.g. GDMO/ASN.1, IDL).

ANNEX B

TMN object identifier assignment rules

The following scheme is used for assigning object identifiers when CMIP/GDMO is used in designing a TMN interface.

B.1 TMN object identifier structure

Annex C/X.680[1] [6] defines the first few arcs of the object identifier structure to be used for information items in ITU-T Recommendations. All object identifiers are structured as in Figure B.1 which is a graphical depiction of the following information:

(0) itu-t[2]
- (0) recommendation
- (1) a
- (2) b
- (3) c
- .....

[1] Annex C/X.208 provides the equivalent definitions.

[2] The name "ccitt" was used in Recommendation X.208 (ASN.1) to construct the object identifier hierarchy. New Recommendations should use "itu-t" which is synonymous with "ccitt".

(7) g
(774) g774
.....
(13) m
(3100) m3100
.....
(14) n
.....

For example, the object identifier of ITU-T Recommendation M.3100 is:

{ itu-t(0) recommendation(0) m(13) m3100(3100) }

The leaves of the above structure represent ITU-T Recommendations. The following TMN substructure is to be used beneath each such leaf representing a Recommendation. This substructure is derived from the rules defined in ITU-T Recommendation X.722.

(0) informationModel
(0) standardSpecificExtension
(2) asn1Module
(3) managedObjectClass
(4) package
(5) parameter
(6) nameBinding
(7) attribute
(8) attributeGroup
(9) action
(10) notification
(11) -- *the next two nodes are reserved for use with GRM*
(12)
(1) protocolSupport
(0) applicationContext
(2) managementApplicationsSupport
(0) standardSpecificExtension
(1) functionalUnitPackage
(2) asn1Module
(127) dot -- *for parts of a Recommendation (see B.2)*

It is recommended that value references be defined within an ASN.1 module for the leaves of the above TMN substructure as follows, e.g. for managedObjectClass:

<recommendation>ObjectClass OBJECT IDENTIFIER
:: = { itu-t(0) recommendation(0) <recommendation series letter>(number)
<recommendation>(number) informationModel(0)
managedObjectClass(3) }

Example:

m3100ObjectClass OBJECT IDENTIFIER
:: = { itu-t(0) recommendation(0) m(13) m3100(3100) informationModel(0)
managedObjectClass(3) }

For management information to be communicated or to be reusable in other templates, the template defining that information must be registered. Each such management information template to be registered is identified with an object identifier.

As an example, an object class in M.3100 called exampleObjectClass will have an object identifier assigned as follows:

```
exampleObjectClass MANAGED OBJECT CLASS
                    .
                    .
                    .
  REGISTERED AS { m3100ObjectClass 5 };
```

The same approach should be followed for the other leaves of the TMN substructure.

Included in the Abbreviation clause of the Recommendation the value references and the sequences of values of the object identifier of that value reference, for example:

```
m3100ObjectClass  { itu-t(0) recommendation(0) m(13) m3100(3100)
                        informationModel(0) objectClass(3) }
```

B.2 TMN object identifier structure extended for Recommendation "parts"

The structure in B.1 should also be used for Recommendations that use part numbers as shown below:

(0) itu-t
- (0) recommendation
- (1) a
- (2) b
- (3) c
- .....
- (7) g
- (774) g774
- (127) dot
- (1) part1
- .....
- (13) m
- (3100) m3100
- .....
- (14) n
- .....

For example, the object identifier for ITU-T Recommendation G.774.1 is:

```
{ itu-t(0) recommendation(0) g(7) g774(774) dot(127) part1(1) }
```

The substructure below this level is as defined in ITU-T Recommendation X.722 and outlined in B.1 above.

An example:

```
g774dot1ObjectClass OBJECT IDENTIFIER ::=

{ itu-t(0) recommendation(0) g(7) g774(774) dot(127) part1(1) informationModel(0)
      managedObjectClass(3) }
```

As an example, an object class in ITU-T Recommendation G.774.1 called exampleObjectClass will have an object identifier assigned as follows:

```
exampleObjectClass MANAGED OBJECT CLASS
                    .
                    .
                    .
  REGISTERED AS { g774dot1ObjectClass 5 };
```

References from other should be in the following format:

"Recommendation G.774.1: 1994"

B.3 TMN assignment procedures

The following assignment procedures are recommended:

1) An item of management information is only assigned one object identifier and is defined in only one document. If some item of management information is required in a Recommendation, and the item is already defined elsewhere, a reference to the existing template shall be used. The reference to a template shall identify the Recommendation and date published with the template-label, e.g. "Recommendation M.3100: 1992": example Object Class. Each Recommendation should also include the following statement:

   "When referencing the definitions for the templates in this Recommendation by other documents, the prefix, e.g. "Recommendation M.3100: 1992", should be used to identify the source for the definitions."

2) Each Study Group is responsible (registration authority) for the registration of object identifiers for its Recommendations within the arcs of the TMN substructure defined above.

B.4 Object identifier allocation for a TMN application context

The following object identifier for a TMN application context is defined and registered in ITU-T Recommendation M.3100 and should be used by all TMN applications:

The object identifier value:

{ itu-t(0) recommendation(0) m(13) m3100(3100) protocolSupport(1)
applicationContext(0) tmnApplicationContextOne(1) }

is assigned to the application context that has the same capabilities as the systems application context in ITU-T Recommendation X.701, but also supports the integer encoding for ProbableCause as defined in ITU-T Recommendation M.3100.

Figure B.1 depicts the "upper" part of the TMN object identifier structure.

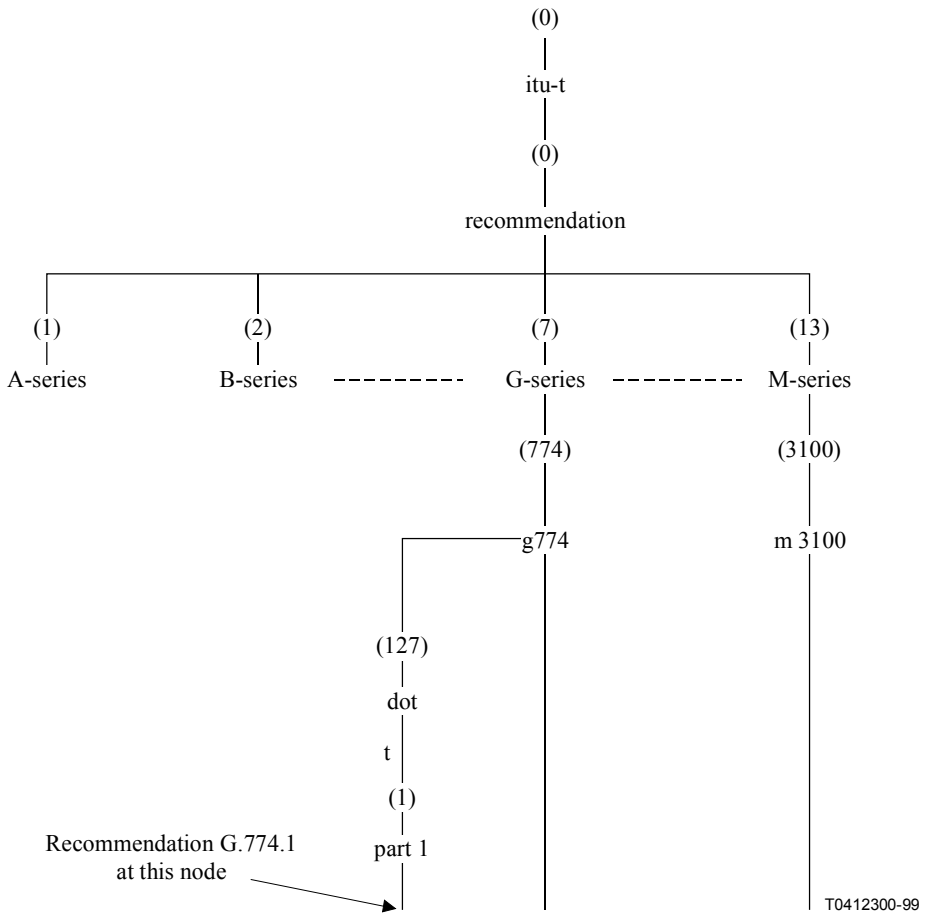


Figure B.1/M.3020 – Graphical representation of the object identifier tree

ANNEX C

Generic definitions

C.1 Generic operations

C.1.1 getAttributes

The generic operation getAttributes (in attributeNamelist: AttributeNameListType, out attributeNameValuePairList: AttributeNameValuePairListType) where the parameter attributeNameList is used to denote that any combination of the attributes (subject to any additional constraints) which are to be returned in the parameter attributeNameValuePairList.

C.1.2 getAllAttributes

The generic operation getAllAttributes (out attributeNameValuePairList: AttributeNameValuePairListType) is used to retrieve values of all the attributes of a class. The response is returned in attributeNameValuePairList.

C.1.3 notifications

To specify notifications, the stereotype <<NotifyDispatch>> has been defined. This is represented diagrammatically using the symbol used for UML classes. The signature of one or more notifications is specified as operations in a <<NotifyDispatch>> stereotype. The semantics are that the operations in a <<NotifyDispatch>> compartment are used by UML classes to initiate the dispatch of notifications through some event distribution mechanism, which is more fully specified in the design phase.

Placing more than one notification operation in a <<NotifyDispatch>> compartment implies the ability to send all of the notification types shown.

Figure C.1 shows an example of the specification of two notification dispatch stereotypes. The alarm example has one notification type, commAlarm, which has a parameter signature which is a simplification of the X.733 communicationAlarm syntax. The configEvents example has four notification types, which have parameter signatures which are simplifications of the syntax defined in ITU-T Recommendations X.730 and X.731.

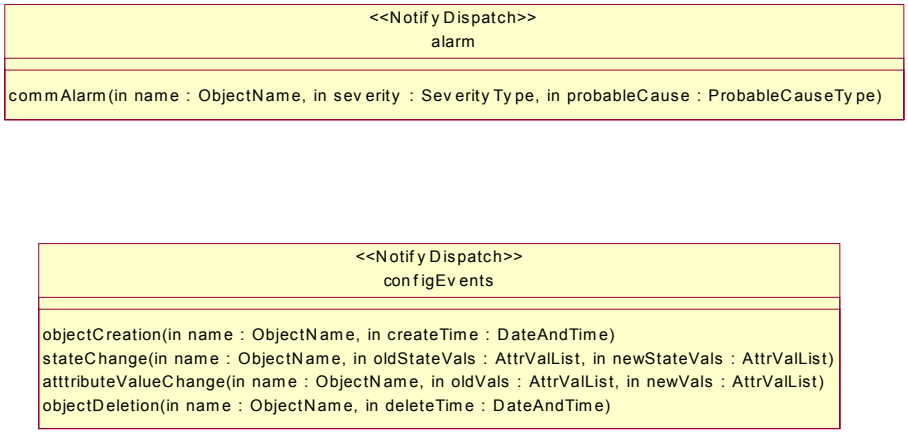


Figure C.1/M.3020 – Specification of Generic Notification sets

Figure C.2 shows an example of how to use the stereotype <<send>> of the UML Association standard element to specify that an instance of a Class must be able to send notifications. When a Class has a <<send>> association with a <<NotifyDispatch>> stereotype, this implies that the Class must be able to send all the notification types shown in the operation compartment of the associated <<NotifyDispatch>> stereotype. For readability, abbreviations for the notification operation signatures (i.e. empty parameter list) may be used, as shown in Figure C.2.

The role of the association end on the side of the Class can have constraints which specify conditions under which specific notification types must be emitted by instances of that Class.

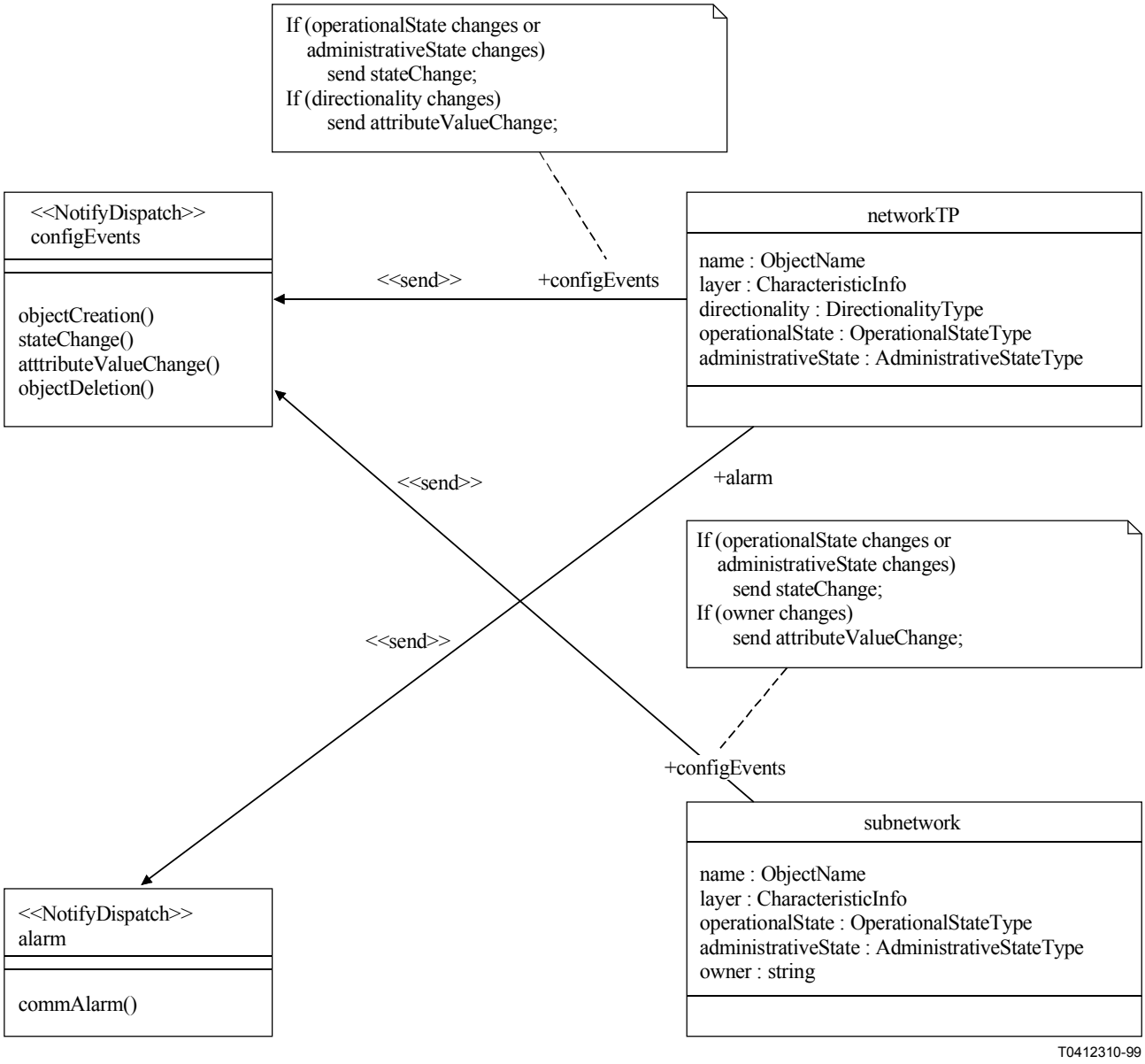


Figure C.2/M.3020 – Specification of Notifications emitted by Classes

APPENDIX I

Example use of GDMI (LCS Provision)

I.1 Introduction

This appendix contains an example use of GDMI template. The example shows how the requirements, analysis and design phases may be documented using a combination of text, UML diagrams and tabular representation. This example is based on ITU-T Recommendation M.3208.1 [5] for provisioning a dedicated leased circuit service. The use of UML in all three phases and how the design phase can reference behaviour definitions in the analysis phase is illustrated here. This covers only a small subset of the requirements contained in ITU-T Recommendation M.3208.1. Where appropriate, existing text from ITU-T Recommendation M.3208.1 is used.

I.2 GDMI template

I.2.1 Scope

This ITU-T Recommendation describes a subset of TMN management services for Dedicated Leased Circuits network identified in ITU-T Recommendation M.3200 as a TMN managed area. Its main focus is on the management services of Customer Administration and Maintenance management for the point-to-point Leased Circuit Services (LCS) that may be offered by one or more service providers and may be controlled by the SC with different levels of visibility. The LCS is defined between a single SC and a single SP. These management services are also applicable for interactions between management systems of different service providers or within a service provider.

I.2.2 Requirements

I.2.2.1 Business level requirements

TMN management services in this ITU-T Recommendation specify interface requirements for Leased Circuit Services between an operations system (OS) and an operations system (OS) to provision and manage Leased Circuit Services. The interfaces addressed by the TMN management services in this ITU-T Recommendation are applicable to both X interfaces across jurisdictional boundaries and Q interfaces within a TMN. Support for the services described in this ITU-T Recommendation are at the discretion of the Service Provider.

In general, the definition of a service should be independent of the particular network used to transport the service. This allows multiple technologies to support the service. Therefore, network level information should not be presented to the service layer. However, specific service features may be defined which allow network or network element information to be presented to a service customer. In this case, an abstraction of the information appropriate to the service feature is transferred.

I.2.2.1.1 Actor Roles

Service customer

Service Customer: See definition of "Customer" in ITU-T Recommendation M.3320. This use of service customer specializes the definition to the context of the TMN Management role for the Service Level.

Service provider

A general reference to an entity that provides telecommunications services to Customers and other users either on a tariff or contract basis. An SP may or may not operate a network. An SP may or may not be a Customer of another SP. In this appendix, the phrase "SP's (sub) network" is used to reference the network(s) used by the SP to provide the LCS.

I.2.2.1.2 Telecommunications resources

Dedicated leased circuit service

The dedicated leased circuit service is a point-to-point connection between two service access points which cannot be changed after creation of the service. The Dedicated LCS uses the Service Name and Service Class to define the value for service specific parameters and to designate which parameters may be changed by the SC following provisioning of the service.

I.2.2.1.3 High-level Use Case

At the top level, M.3200 identifies as the managed area the "Customer Administration of Leased Circuit Service". An actor in the role of the service customer interacts with an actor in the service provider role to perform the various CNM activities. These activities are grouped together using

function set groups defined either in ITU-T Recommendation M.3400 or new definitions to meet the additional capabilities. A service provider may assume the service customer role if the end-to-end service is to be provisioned and maintained by multiple service providers.

Figure I.1 is the highest-level use case where a service customer actor interacts with the service provider actor. The use case customer admin LCS uses the function sets indicated by the three use cases. This ITU-T Recommendation addresses the requirements and analysis corresponding to the service provisioning aspects. Other Recommendations in this series expand on other function set groups.

The use cases representing function set groups are refined further into function sets and finally into management functions. As will be seen later, the functions in one function set use case may extend the functions in another set to accomplish the over all requirements of the management service.

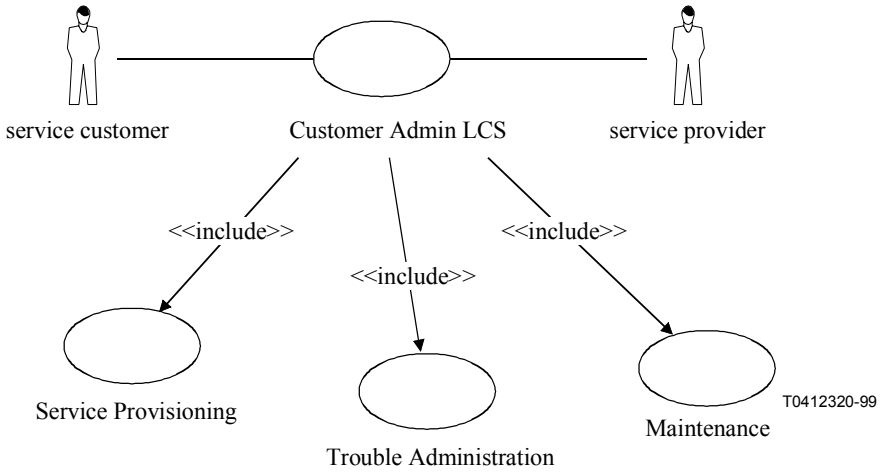


Figure I.1/M.3020 – Customer Administration of LCS Use Case

The service customer actor interacts with the service provider actor to accomplish the requirements for use case called "service provisioning". Service provisioning use case uses two use cases: LCS configuration function set and Link configuration function set. The use case service provisioning describes the requirements for the service customer actor to request the creation of either a leased circuit service or preprovisioned link connections. The latter can be used by the service customer (as an example) to create leased circuit services in real time by selecting specific link connections to be connected (possibly to meet a scheduled major game event) for a period of time during certain times of specific days. The stereotype <<include>> is used to indicate that the use case service provisioning employs the reusable fragments for creating leased circuits and link connections.

The activities of creation of LCS configuration function set use case is extended by the administrative function set use case. The <<extends>> stereotype is used to indicate this as follows. The configuration function set contains activities for the service customer to provide the minimum capabilities for creating service order (request service). When the creation of the service is not possible in real time, the administrative functions are used by the service provider. The SP may inform for example the progression of the request (as an example the availability date for the service has been extended or the requested bandwidth is not what can be provided etc.). The <<extends>> denotes the fact that the use case adds to the functions in the LCS configuration function set by providing (for example) reporting capabilities on the status of the request to create a new service or modify a previously issued service request or an existing service.

Figure I.2 depicts the use cases that are required to meet the service customer's needs for service provisioning activities.

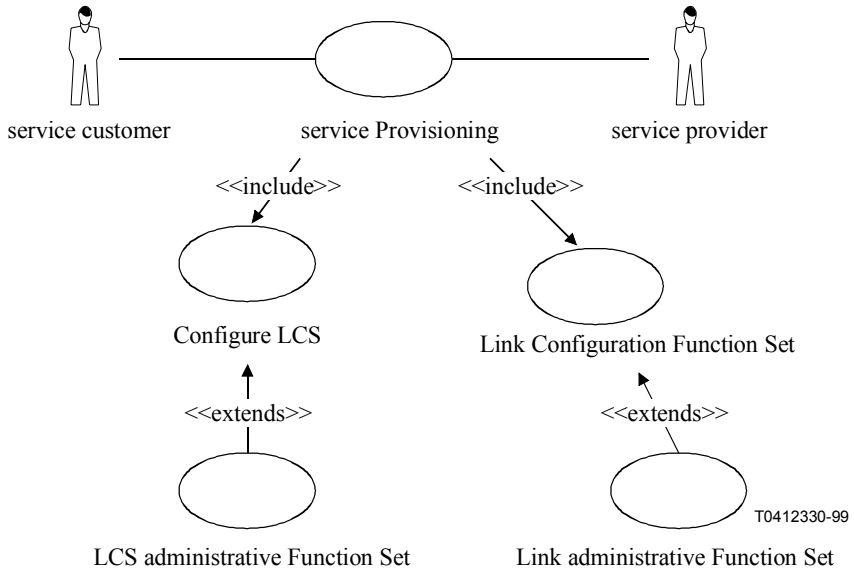


Figure I.2/M.3020 – Service Provisioning Use Case for CNM of LCS and Link Connections

I.2.2.2 Requirements Level Specification

The LCS configuration function set, as noted above, describes the scenarios for a service customer actor to interact with the service provider actor to request provisioning an LCS (non-real time or real time) or a link connection (non-real time). Specializing the generic provisioning function set in ITU-T Recommendation M.3400 for LCS and Link connections refines the use case further.

I.2.2.2.1 Actor Roles

No new actor roles beyond those identified in the discussion of business level requirements are needed for specification level requirements.

I.2.2.2.2 Telecommunications resources

No additional resources beyond those identified in the business level requirements are required.

I.2.2.2.3 TMN management functions

The LCS configuration function set use case has three functions: request creation of a service, delete an existing service or modify either the parameters of a previously issued service creation request or an existing service. The request for creating a service is also referred to service order request or a service customer creating a service order to the service provider. Once the service order is created, a service customer actor may cancel the requested order.

NOTE – Even though the example shows that cancellation of the request is associated with only create service function, it can also be used to extend the other two functions – cancel the deletion request or the modification request. The service level agreement (referred to as SLA or contract) between the service provider and service customer defines the policies associated with such a cancellation. Examples policy decisions include if work has already started based on the request, what is the accounting policy for the completed part, and the security requirements associated with cancelling a request.

I.2.2.2.4 Use Cases

Figure I.3 illustrates the LCS configuration function set in terms of the following use cases: Create LCS, Delete LCS and Modify LCS. The service provider interacts with the configuration function set use case, which uses the three use cases. The create LCS use case defines the requirements for a service customer to issue a service order requesting the name, class as well as the values of the relevant parameters for the requested service.

The Cancel LCS request use case is extended by defining the scenario where the customer can cancel a previously issued service request. If the request contains invalid parameters (for example the requested service class and or name is not valid or not offered by the service provider), then error condition occurs and appropriate error scenarios are generated by the extended use case "Invalid request parameters encountered".

NOTE – New stereotypes may be needed in defining the requirements in some management services and managed areas. This example uses only the UML defined stereotypes.

At the specification level requirements, the CNM functions are identified. The summary associated with these functions can be included here. As an example, the following modified description taken from ITU-T Recommendation M.3208.1 can be used:

Create LCS Use Case: This use case allows the SC to request the creation of one or more Dedicated Leased Circuit Services. The SC shall identify the service to be provisioned, and service features (as specified in the Information Flow), the service availability date requested, the customer contact within the organization, and relevant information about the originating and terminating locations of the service (see Information Flow). The SC may also specify the route of the requested service and a user identifier for the requested leased circuit. The SP may reject the request (described in the extended use case) if the service provider does not offer the requested service.

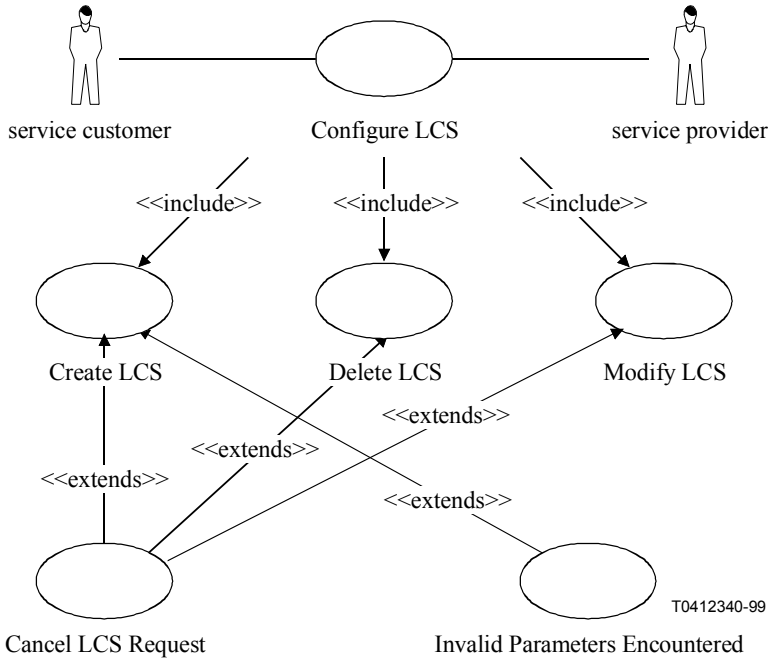


Figure I.3/M.3020 – Decomposition of LCS Configuration Function Set Use Case

The requirements for the use cases are to support the overall requirements discussed in the functions in I.2.2.2.3.

I.2.2.3 Analysis

I.2.2.3.1 Object Classes and State Charts

The classes to support the create LCS and Link Connection use cases are shown in Figure I.4. A service customer object class may issue one or more service requests as shown by the association with cardinality marked as 1 and *. Because there are common properties associated with requesting LCS and Link Connections, a generic class called "service request" is identified. To improve readability, not all attributes and any of the operations permitted on the object classes are shown

here. The attribute "provider request number" is included here without showing the visibility or type (as these are attributes that are exposed on an interface for management, they are to be considered as public, even though they may have different visibility from software perspective). The type is not included because it may be different based on the design paradigm. A generalization relationship exists between the service request to create LCS or link with the generic service request as shown in the figure.

The LCS service request class supports the requirements for the use case Create LCS. The detailed parameters are shown in the information flow table. The interactions between the service customer and service provider for this use case are shown in sequence, collaboration and activity diagrams.

Customer name is a mandatory attribute of the object class service customer. A customer may have several services from a service provider and aggregation is used to describe this. The service object LCS and Link Connection are the result of performing successfully the use cases Create LCS service and Create Link Connection respectively. A service must belong to a customer and the cardinality between the customer and services indicate this requirement.

When a service is created (LCS or Link connection) it is identified using the attribute "service ID". The constraint {frozen} is used to indicate that once the service is created, the value of this attribute may not change during the lifetime of the object. This also implies that the attribute is read-only and cannot be changed either by the service customer or service provider. (This can be designed using set by create capability available with CMIP-GDMO.)

Associated with the service request object is a state transition diagram shown in Figure I.5. The definition of the states can be found in ITU-T Recommendation M.3208.1 with some slight differences to adapt to UML notation. To avoid repetition the description of the states and the transition events are not provided in this appendix and the readers are referred to ITU-T Recommendation M.3208.1. In order to show that cancel can be issued when the request is any one of the three states, a super state called "in progress" has been introduced. Without this, the diagram would have cancel event from each of these states.

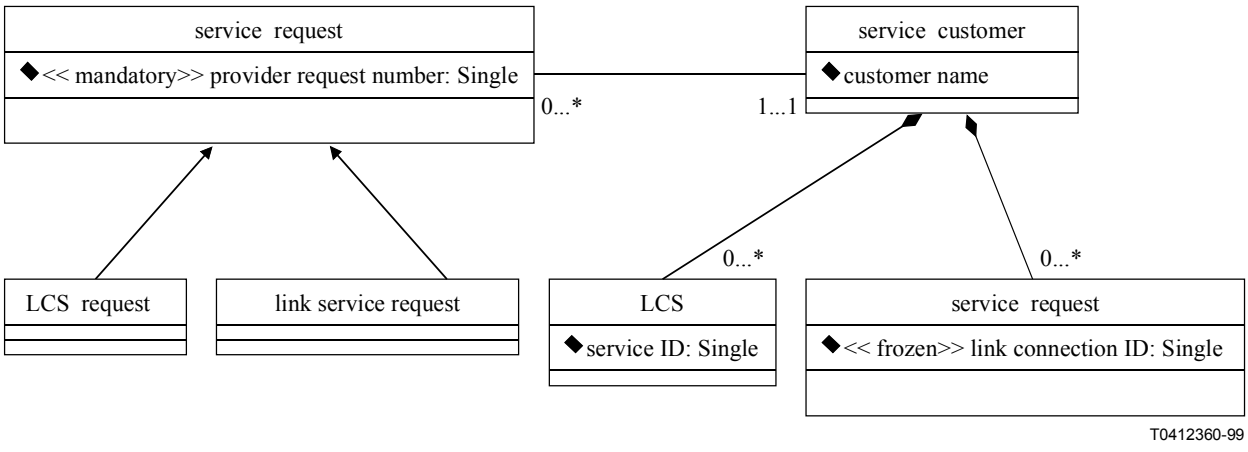


Figure I.4/M.3020 – Class structure for Create LCS and Link Connection

Figure I.5 shows the states and transition between the states for the service request.

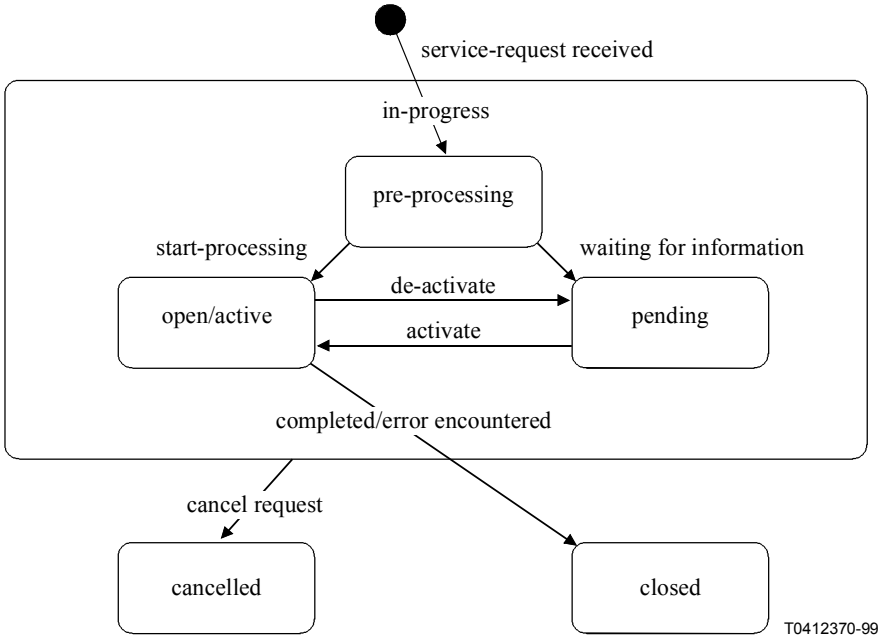


Figure I.5/M.3020 – State Diagram for Service Request Class

The state diagram supports the requirements identified in the text following Figure I.1. The transitions show that the availability of the requested service, if not available on real time, can result in moving to different states. The extension provided by cancel LCS request use case is supported by the event and state called "cancelled".

I.2.2.3.2 Sequence diagram

When a service customer sends a request for creating a service, the sequences of message flows are shown in the sequence chart in Figure I.6.

NOTE – An exhaustive identification of all sequences possible when requesting a service is not included here. Some illustrative cases are shown. The sequence diagram includes messages that are part of different use cases. Some messages specify condition evaluation and depending on the result of evaluating the condition determines if the message is exchanged.

A service customer issues a request for a new service. If the service provider determines the input parameters are valid, then the request is accepted and a service request subclass is created. Instead of defining a constructor or an object factory, the notes feature is used to explain the creation of service request object if the request is accepted by the service provider. Accepting the service request does not imply all requested values associated with creating the new service would be available with the new service. This is clarified when the parameters included in create request are identified later in detail (see Table I.1). The response after creating the request object includes the values of the parameters. As illustrated in Figure I.6, the response includes values for the parameters that may be modified relative to the original request. It is assumed that the service customer, by default, accepted the modifications offered by the service provider. If this is not true, the customer can issue a cancel to the service request. The policies associated with the cancellation should be available in a SLA. In addition, based on the service level agreement, the provider may retain a history of requested changes. If the service is created in real time, there is no need to create a service request object. This case is treated as follows: a service request object is created and deleted immediately and the service customer is notified of the created new service. The service customer will not be able to perform the sequence of messages (e.g. modify service request) indicated below.

Once the service request object is created, the service customer can monitor the progress of the request, request changes to the parameters and be informed autonomously of the progress. Even though the sequence diagram is used to represent the time domain, the report of the status is asynchronous and can be issued by the service provider asynchronously as long as the service request object is created.

After the service request object state changes to closed (assuming successful creation), a notification issued by the service provider indicating the creation of the service with relevant values for the associated parameters. The reporting messages correspond to functions that are supported by the LCS administrative function set use case (which is further refined into other use cases for reporting state changes and attribute value changes along with monitoring function).

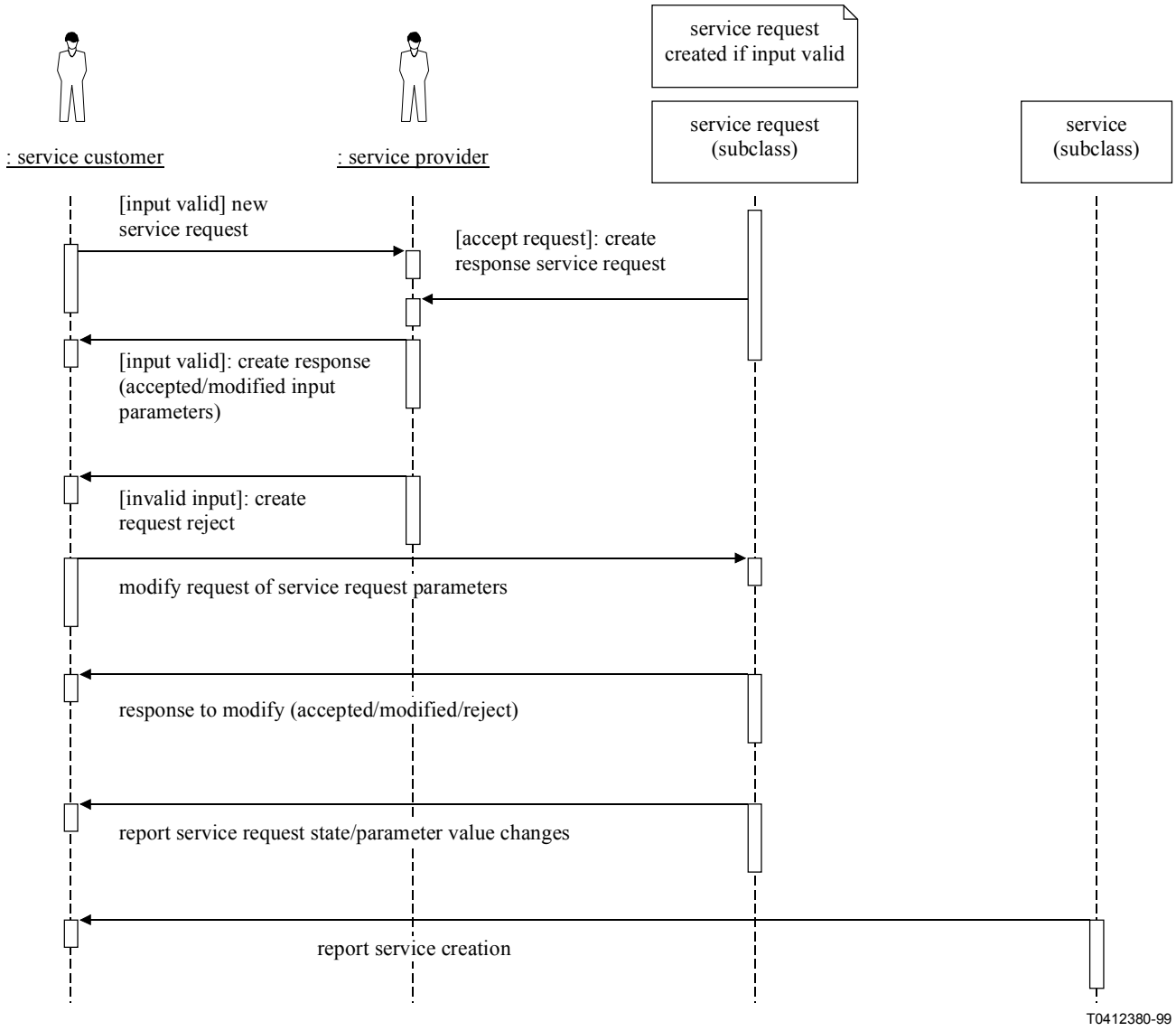


Figure I.6/M.3020 – Sequence diagram: Successful creation of service

I.2.2.3.3 Collaboration diagrams

Corresponding to the sequence diagram depicting the flow of message with respect to time, the collaboration diagram for successful creation of a service is shown in Figure I.7. The interaction between the objects and the messages exchanged by them are shown using sequential order of the messages in the sequence diagram. The actual events corresponding to the numbers in the figure are defined inside the notes.

NOTE – In general it is enough to show sequence diagram in order to illustrate message flows. In this example, the collaboration diagram is included to show that in some cases these diagrams may include additional information and therefore may be necessary. For this simple case, the collaboration diagram does not add any new information beyond the sequence diagram.

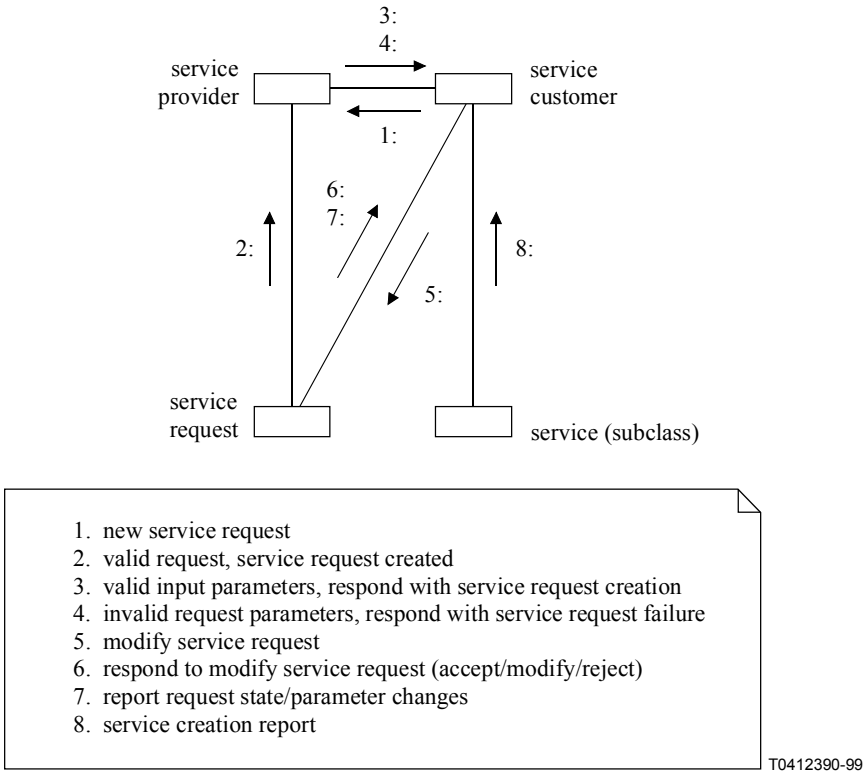


Figure I.7/M.3020 – Collaboration diagram: Successful creation of service

I.2.2.3.4 Information flow

The diagram provides information on the message flows at a high level. If the parameters passed with the messages are to be shown in the diagram, it will be difficult to read. In addition to listing the parameters, often it is necessary to identify whether a parameter is always required in an exchange (mandatory) or may be included if a condition is met or at the discretion of the user. In order to explain the parameters and conditions for their presence or absence, a tabular approach is used. Table I.1 is a simplified extract from ITU-T Recommendation M.3208.1. Depending on the complexity of the application, such a table can provide details not possible to show in a diagram. The convention for "m" etc. is defined in ITU-T Recommendation M.3208.1.

A second advantage to creating such a table is reuse of the definition in another management service for a managed area. For example, the service called "Connection management" can be used by a customer to create LCS in real time. In this case, most of the parameters for creating a leased circuit are common with the non-real time case. Some restriction and augmentation of the parameters may

be necessary. By using the same table and explaining the constraints facilitates reuse of the parameter definitions.

Table I.1/M.3020 – Information Flow for create service request

Service Customer Request and SP Response	Service Customer	Service Provider	Notes
Service Name	m	o	The type of leased circuit service offered by the SP. Service names are not subject to standardization and are defined by the contract between the SC and the SP.
Service Class	o	c	The name of a profile of service characteristics (associated with the service name) defined and supported by the SP. Examples of the service characteristics that may be included in the profile are directionality, channelization, signalling options, protection, quality of service objectives, application, etc. Service Class names not subject to standardization are defined by the Contract. c – If the requested service class is not equal to the class of service provided by the SP, then the SP must supply the value, else it is optional.
Bandwidth	o	c	Requested bandwidth, actual bandwidth returned. c – If the requested bandwidth cannot be provided by the SP, the SP shall return the value together with a reason code indicating that the bandwidth is not available. If the response is not indicating a completion, the SP may report an error condition with a reason code indicating that the available service differs from the customers initial service request.
Quantity	o	c	The number of Leased Circuit Services to be generated by the SP. Following the processing of the LCS function, the SP shall return unique circuit numbers for each LCS generated by the processing of this command by the SP.

I.2.2.3.5 Activity diagrams

To explain the workflow when creating a new service, the activity diagram is used as shown in Figure I.8. The activity diagram shows where synchronization can take place between multiple activities. In this example the workflow being the representation of activities in a service provider, use of notation such as swim lanes (defines concurrent activities corresponding to multiple objects) are not required.

The service provider receives the request, validates it and then creates the service request if it is accepted. The customer is informed of the successful creation and the synchronization bar indicates that processing the request can start concurrently with informing the customer. Only after the customer receives the report of the service request object creation, any modification request to the values of the parameters can be issued. Similarly, the cancel can also be sent because as will be noted later, to cancel the service request, the reference to the created service request object is necessary. Even though not shown in the figure, once that synchronization bar is encountered, changes to the progress of the request may also be reported. The activities resulting from successful or failed request are shown along with the end of the workflow in all cases.

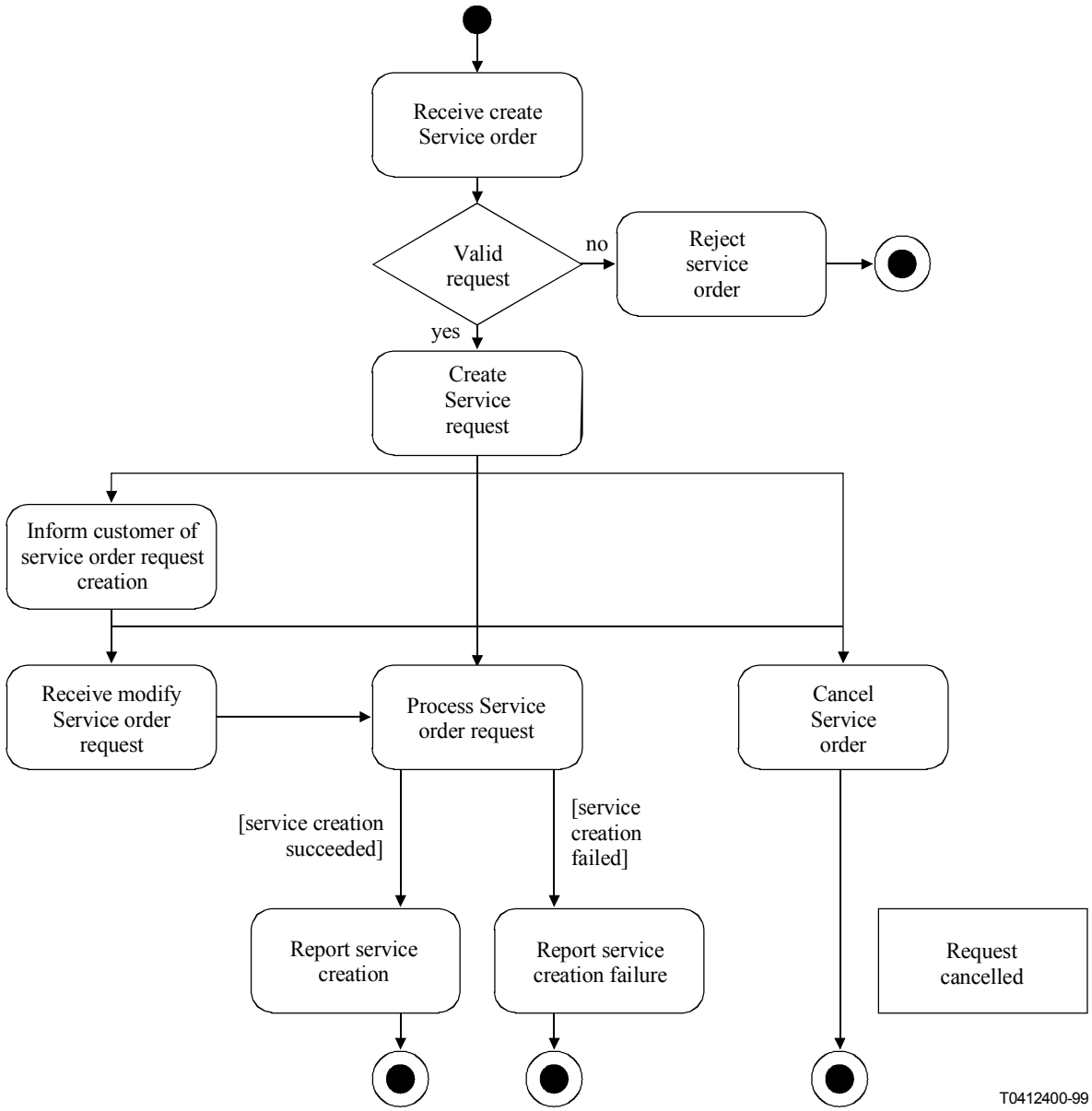


Figure I.8/M.3020 – Activity diagram: Workflow for creating service

NOTE – This example illustrates use of some of the visual-modelling notation from UML to describe the behaviour and activities for the management service. The object classes shown here may have a m:n relation with those in the design phase (for example managed object class with CMIP paradigm and interface in CORBA/IDL). These figures may be referenced in the design to explain the behaviour of the protocol specific entities.

I.2.3 Design

The GDMO definitions for this example are provided in ITU-T Recommendation M.3108.1. The behaviour of the managed object classes can reference the state chart diagram shown in the analysis.

In the context GDMO, the managed object class hierarchy and naming diagrams expressed in UML are shown below. The traceability of the various elements of the GDMO models are shown in ITU-T Recommendation M.3108.1.

I.2.3.1 UML class diagrams for inheritance of M.3208.1 object classes

In these diagrams, classes are shown as boxes with three sections, including: object class name in the top section; the attribute names in the second section (not filled in these figures for readability); and access operations in the bottom section.

The operation "get()" is used in the class diagrams to denote that the class attributes are readable after an instance of that class is created.

The operation "set()" is used to denote that some (at least one) of the class attributes may be modified after an instance of that class is created.

UML class diagrams use large open headed arrows to indicate inheritance relationships. When a class is related to another by inheritance, the operations from the superclass (the one which has the large arrowhead touching it) are also supported for the inherited class, but are not repeated in the operation section of the class box. See Figures I.9-1 to I.9-3.

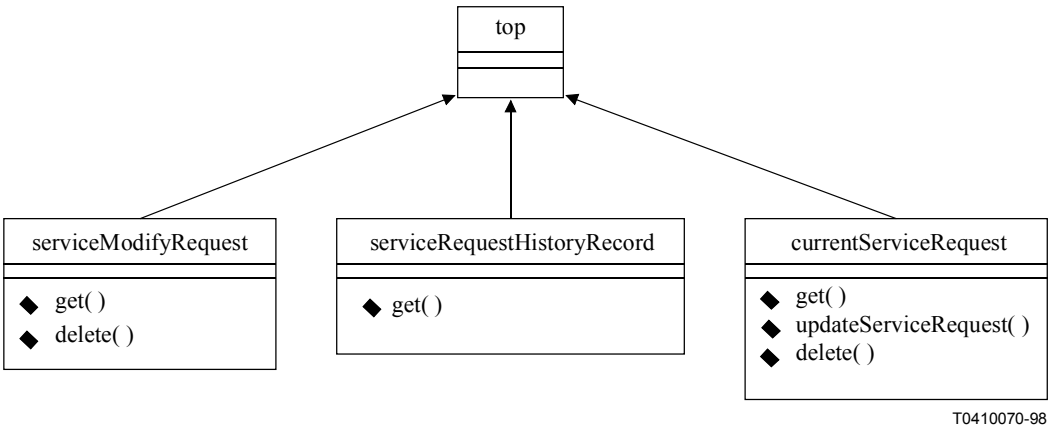


Figure I.9-1/M.3020 – Inheritance relationships for generic service request fragment

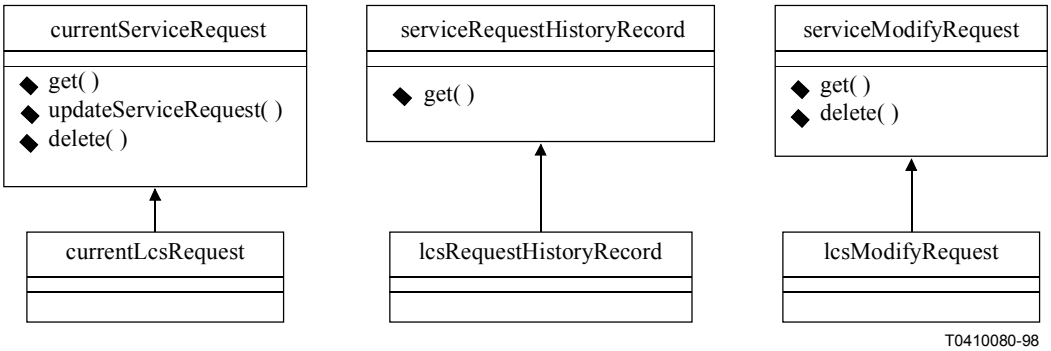


Figure I.9-2/M.3020 – Inheritance relationships for LCS request fragment

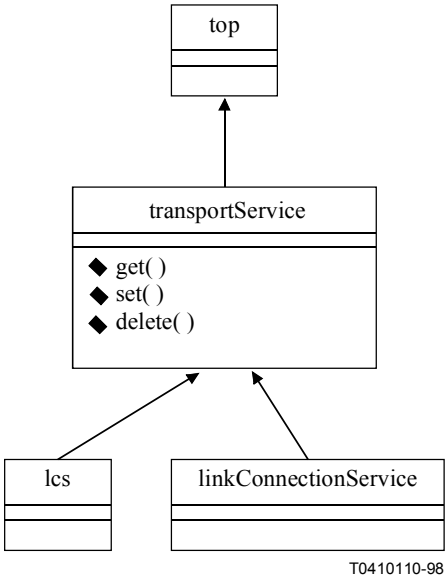


Figure I.9-3/M.3020 – Inheritance relationships for service fragment

I.2.3.2 UML class diagrams for modelling relationships

The possible relationships between instances are shown in UML class diagrams with associations. Containment relationships are denoted by a diamond headed line touching the parent (UML aggregation). Simple associations are shown with lines with roles indicated on the line ends. Relationship cardinalities are indicated by "0 .. *" or "1 .. *" tags on the end of the line representing a relationship. The account object represents the service customer actor shown in the analysis section. See Figure I.9-4.

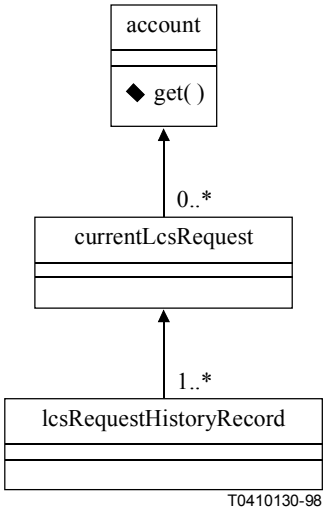
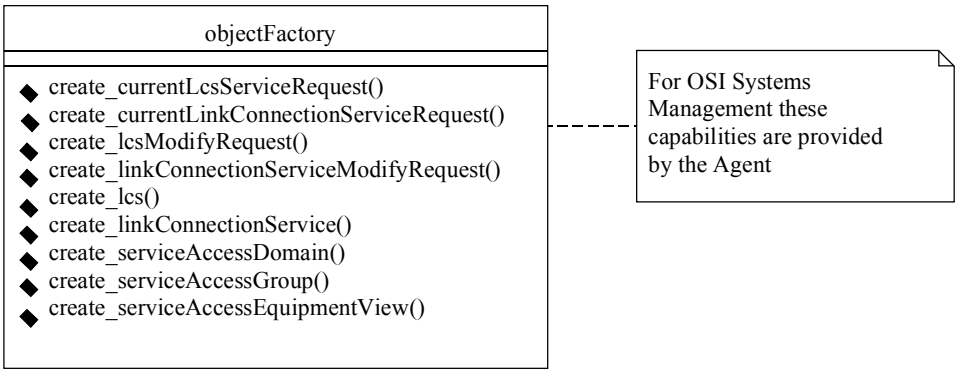


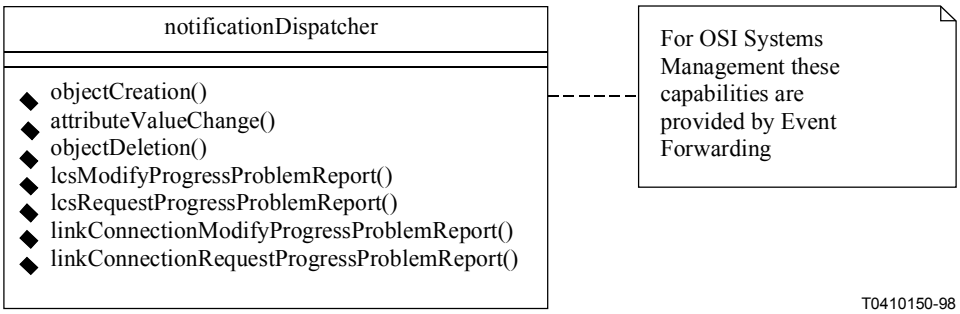
Figure I.9-4/M.3020 – Containment relationships for LCS request with history records

I.2.3.3 UML class diagrams for modelling agent functionality

Some UML classes are introduced (factories and notification dispatcher) to model the actions for creating objects and distributing notifications from objects. Instances of these agent functionality classes appear in the sequence diagrams. When a notification operation invocation is made onto a notification dispatch object, all destinations that have registered interest will receive a copy of that notification. These final delivery flows are not shown in the sequence diagrams in Figure I.9-5, since many objects may be interested in receiving them.



Factory with operations for customer to create service and request objects



Notification Dispatcher to receive and distribute notifications

Figure I.9-5/M.3020 – UML model for agent functions (creating objects and disseminating notifications)

I.2.3.4 UML sequence diagrams to illustrate scenarios of object usage

Figures I.9-1, I.9-2 and I.9-3 illustrate the use of UML sequence diagrams to describe scenarios.

The message flows from the notification dispatcher to the ultimate registered destinations are not shown in these diagrams. It would be normal for the customer (as well as other objects) to be a registered recipient for the notifications shown in these sequence diagrams).

Figure I.9-6 illustrates the message exchanges for creating automatic termination of the LCS. The flow in this diagram uses the object classes representing the agent functionality. Steps 1-3 can be expanded into the steps shown in Figure I.9-6. The intermediate steps that are permitted (modifying the parameter values of the requested service) can be noted by referencing the figure in analysis section. The analysis section explains the criteria for successfully creating the requested service and informing the customer. Step 3 in I.9-6 shows the name of the notification applicable for this design. Other message name may be used with another design.

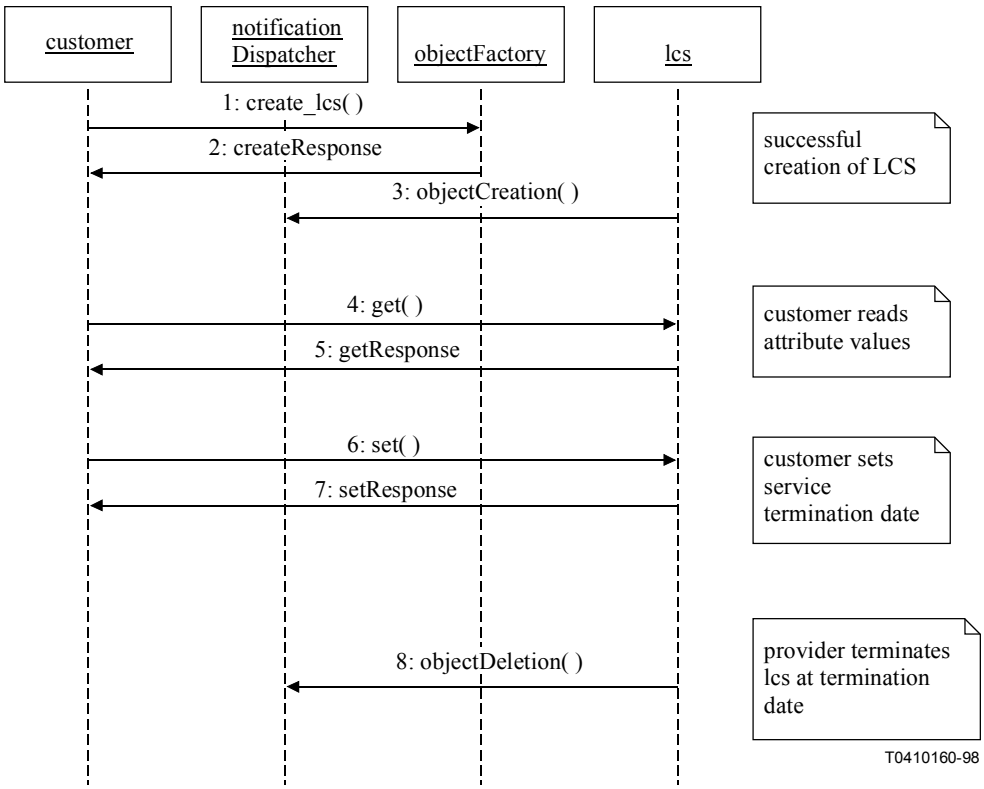


Figure I.9-6/M.3020 – Sequence diagram for explicit LCS create and automatic termination

Figure I.9-7 expands Figure I.9-6 in the analysis with specific managed object classes defined in the GDMO model. The service request object in Figure I.9-6 is realized in the design using the current and history lcs service request objects. The sequence diagram in this design case shows how the history object is created as a result of updates to the original request. The model in the design phase refines further the service request object in the analysis phase so that history of the requested changes can be retained. This meets the optional requirement identified in I.2.2.3.2 as being relevant to SLA. Figure I.9-8 illustrates the case where history of the changes is not retained. In addition, it shows the interaction between the managed objects in the design phase when the service creation is not successful (note the corresponding figure for unsuccessful creation is not shown in the analysis section).

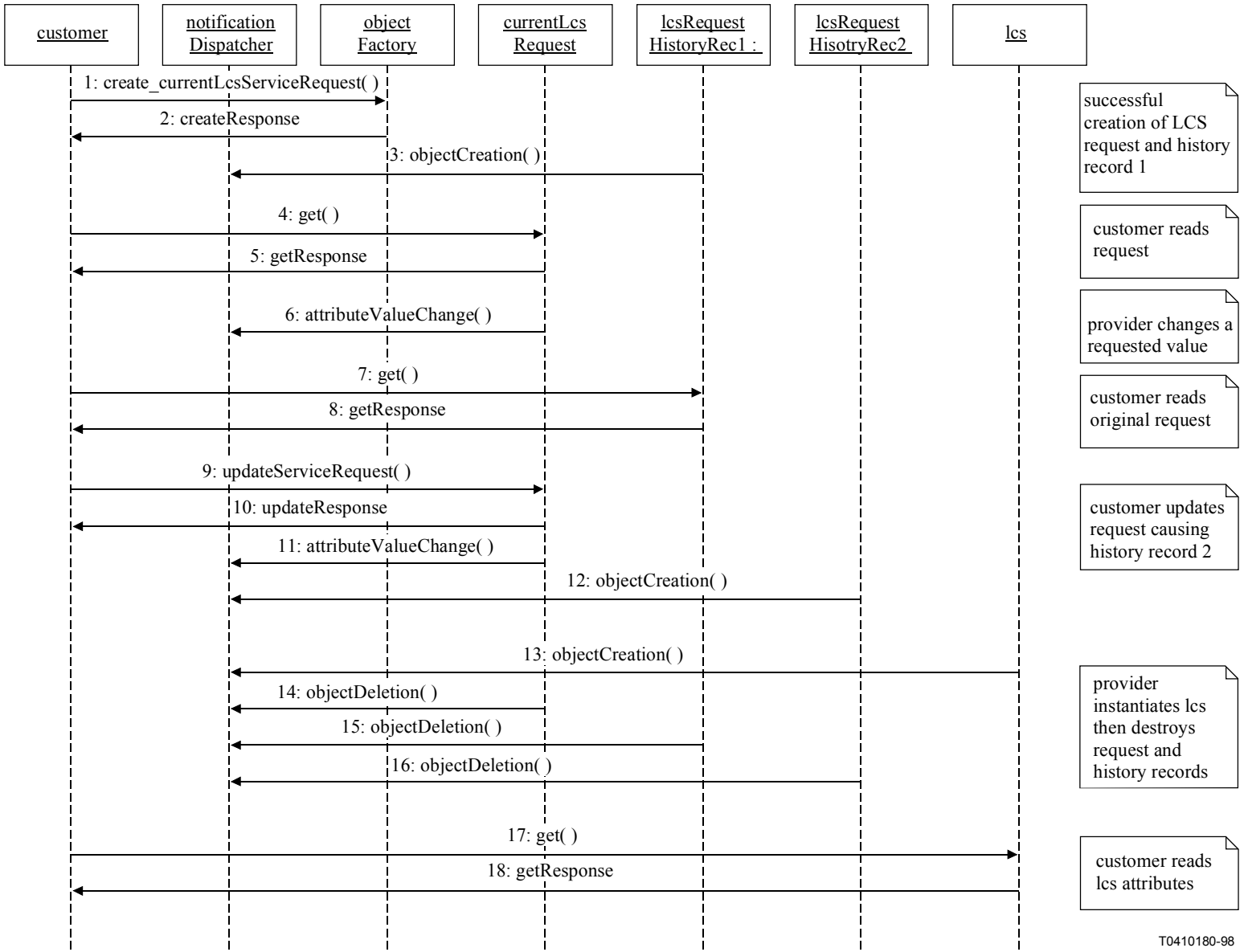


Figure 1.9-7/M.3020 – Sequence diagram for instantiation of LCS using request and history records

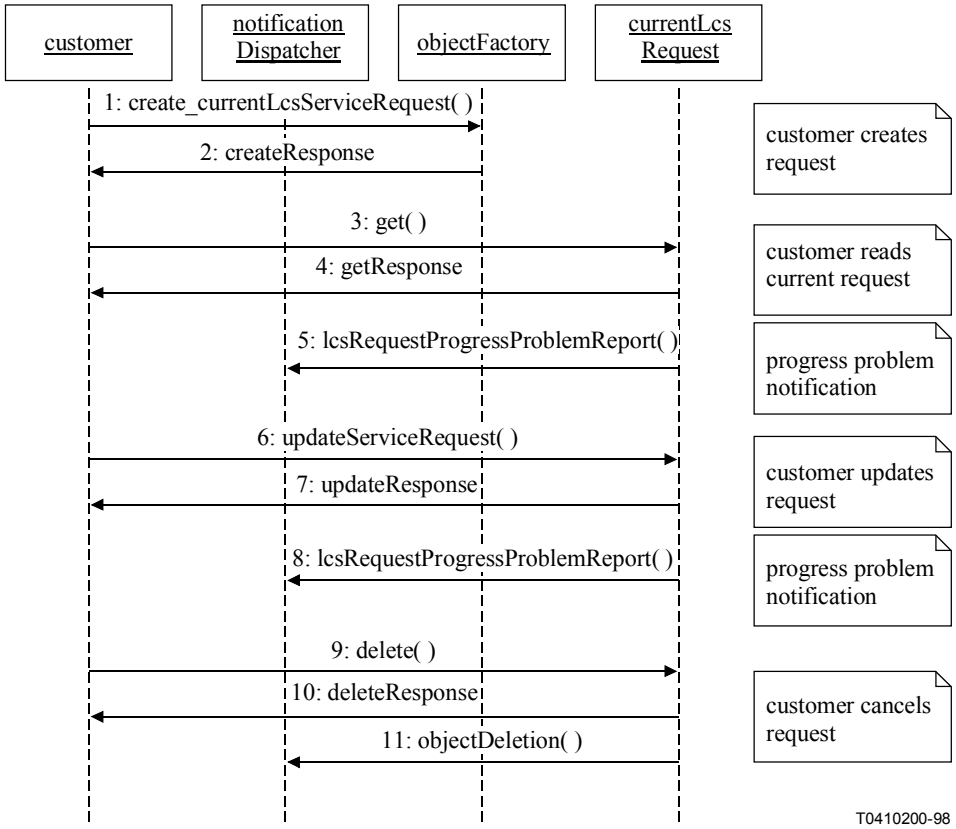


Figure I.9-8/M.3020 – Sequence diagram for unsuccessful LCS instantiation using request without history

ITU-T RECOMMENDATIONS SERIES

Series A	Organization of the work of the ITU-T
Series B	Means of expression: definitions, symbols, classification
Series C	General telecommunication statistics
Series D	General tariff principles
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Construction, installation and protection of cables and other elements of outside plant
Series M	**TMN and network maintenance: international transmission systems, telephone circuits, telegraphy, facsimile and leased circuits**
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Telephone transmission quality, telephone installations, local line networks
Series Q	Switching and signalling
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks and open system communications
Series Y	Global information infrastructure and Internet protocol aspects
Series Z	Languages and general software aspects for telecommunication systems

Geneva, 2001