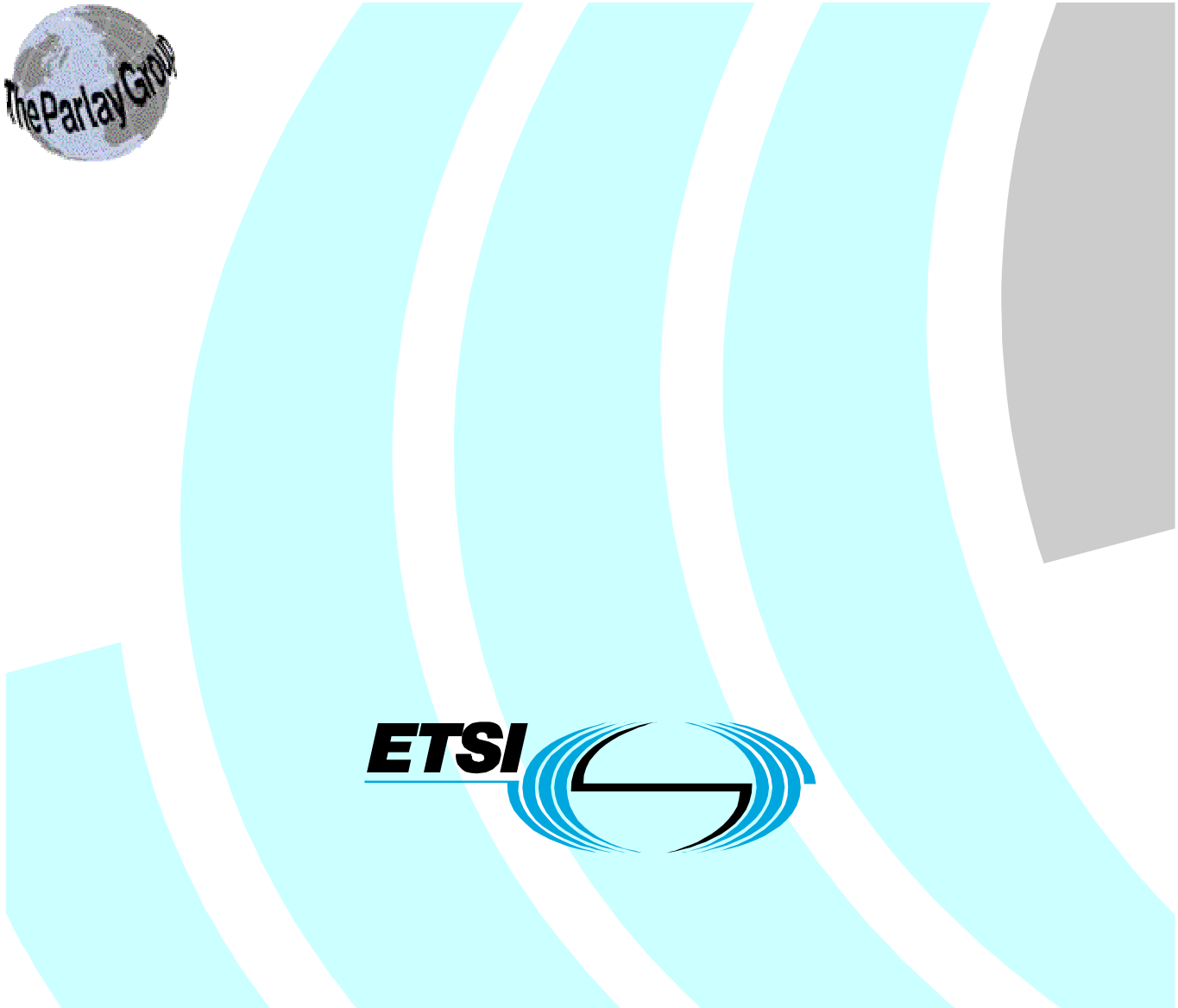
Final draft ETSI ES 202 915-2 V1.4.1 (2006-10)

ETSI Standard

Open Service Access (OSA); Application Programming Interface (API); Part 2: Common Data Definitions (Parlay 4)



Reference
RES/TISPAN-01028-02-OSA

Keywords
API, IDL, OSA, UML

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, please send your comment to one of the following services:
http://portal.etsi.org/chaircor/ETSI_support.asp

Copyright Notification

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

© European Telecommunications Standards Institute 2006.
© The Parlay Group 2006.
All rights reserved.

DECT™, **PLUGTESTS**™ and **UMTS**™ are Trade Marks of ETSI registered for the benefit of its Members.
TIPHON™ and the **TIPHON logo** are Trade Marks currently being registered by ETSI for the benefit of its Members.
3GPP™ is a Trade Mark of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners.

Contents

Intellectual Property Rights 5
Foreword 5
1 Scope 6
2 References 6
3 Definitions and abbreviations 6
3.1 Definitions 6
3.2 Abbreviations 6
4 Common Data Definitions 6
5 Common System Data Definitions 7
5.1 Standard Data Types 7
5.1.1 TpBoolean 7
5.1.2 TpInt32 7
5.1.3 TpFloat 7
5.1.4 TpLongString 7
5.1.5 TpOctet 7
5.1.6 TpOctetSet 7
5.1.7 TpString 7
5.1.8 TpAssignmentID 7
5.1.9 TpSessionID 7
5.1.10 TpSessionIDSet 8
5.1.11 TpAny 8
5.1.12 TpAttribute 8
5.1.13 TpAttributeValue 8
5.1.14 TpAttributeList 8
5.1.15 TpAttributeSet 8
5.1.16 TpInt64 8
5.1.17 TpVersion 9
5.1.18 TpStringSet 9
5.1.19 TpStringList 9
5.1.20 TpAttributeTagInfo 9
5.1.21 TpSimpleAttributeValue 10
5.1.22 TpSimpleAttributeTypeInfo 10
5.1.23 TpStructuredAttributeType 10
5.1.24 TpStructuredAttributeValue 11
5.1.25 TpChar 11
5.1.26 TpWChar 11
5.1.27 TpWString 11
5.1.28 TpInt16 11
5.1.29 TpUnsignedInt16 11
5.1.30 TpUnsignedInt32 11
5.1.31 TpUnsignedInt64 11
5.1.32 TpDouble 11
5.1.33 TpFixed 11
5.1.34 TpXMLString 11
5.1.35 TpUnorderedOctetSet 12
5.2 Other Data Sorts 12
5.2.1 Sequence of Data Elements 12
5.2.2 Tagged Choice of Data Elements 12
5.2.3 Numbered Set of Data Elements 12
5.2.4 Reference 13
5.2.5 Numbered List of Data Elements 13
5.3 Interface Related Data Definitions 13
5.3.1 IpInterface 13

5.3.2 IpInterfaceRef 13
5.4 Exception Classes 13
5.4.1 Underlying Technology Exceptions 13
5.4.2 TpCommonExceptions 14
5.4.3 Constants associated with TpCommonExceptions 14
5.4.4 Exceptions available to all methods on all interfaces 14
5.5 Date and Time Related Data Definitions 15
5.5.1 TpDate 15
5.5.2 TpTime 15
5.5.3 TpDateAndTime 16
5.5.4 TpDuration 16
5.5.5 TpTimeInterval 16
5.6 Address Related Data Definitions 17
5.6.1 TpAddress 17
5.6.2 TpAddressSet 18
5.6.3 TpAddressPresentation 18
5.6.4 TpAddressScreening 18
5.6.5 TpAddressPlan 18
5.6.6 TpAddressError 19
5.6.7 TpAddressRange 19
5.6.8 TpURL 20
5.6.9 TpURN 20
5.7 Price-related Data Definitions 20
5.7.1 TpPrice 20
5.7.2 TpAoCInfo 20
5.7.3 TpAoCOrder 21
5.7.4 TpCallAoCOrderCategory 21
5.7.5 TpChargeAdviceInfo 21
5.7.6 TpCAIElements 21
5.7.7 TpChargePerTime 22
5.7.8 TpLanguage 22
5.8 Data Types Common Across Call Control and Data Session Control 22
5.8.1 TpDataSessionQosClass 22

Annex A (normative): OMG IDL Description of the Common Data definitions 23

Annex B (informative): W3C WSDL Description of the Common Data definitions 24

Annex C (informative): Java™ API Description of the Common Data definitions 25

Annex D (normative): Exception Hierarchy 26

Annex E (informative): Record of changes 34

E.1 Data Definitions 34
E.1.1 New 34
E.1.2 Modified 35
E.1.3 Removed 35

E.2 Exceptions 35
E.2.1 New 35
E.2.2 Modified 36
E.2.3 Removed 36

E.3 Others 36

History 37

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 ETSI Standard (ES) has been produced by ETSI Technical Committee Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN), and is now submitted for the ETSI standards Membership Approval Procedure.

The present document is part 2 of a multi-part deliverable covering Open Service Access (OSA); Application Programming Interface (API), as identified below. The API specification (ES 202 915) is structured in the following parts:

Part 1: "Overview";

Part 2: "Common Data Definitions";

Part 3: "Framework";

Part 4: "Call Control";

Part 5: "User Interaction SCF";

Part 6: "Mobility SCF";

Part 7: "Terminal Capabilities SCF";

Part 8: "Data Session Control SCF";

Part 9: "Generic Messaging SCF";

Part 10: "Connectivity Manager SCF";

Part 11: "Account Management SCF";

Part 12: "Charging SCF";

Part 13: "Policy Management SCF";

Part 14: "Presence and Availability Management SCF".

The present document has been defined jointly between ETSI, The Parlay Group (http://www.parlay.org) and the 3GPP, in co-operation with a number of JAIN™ Community (http://www.java.sun.com/products/jain) member companies.

The present document forms part of the Parlay 4.3 set of specifications.

The present document is equivalent to 3GPP TS 29.198-2 V5.11.0 (Release 5).

1 Scope

The present document is part 2 of the Stage 3 specification for an Application Programming Interface (API) for Open Service Access (OSA).

The OSA specifications define an architecture that enables application developers to make use of network functionality through an open standardised interface, i.e. the OSA APIs.

The present document specifies the Common Data Definitions of the OSA. The Common Data Definitions contain data-types that are common across the rest of the OSA API. All aspects of the Common Data are defined here, these being:

- Data Definitions.
- IDL Description of the data types.
- WSDL Description of the data types.
- Reference to the Java™ API description of the data types.

2 References

The references listed in clause 2 of ES 202 915-1 contain provisions which, through reference in this text, constitute provisions of the present document.

ETSI ES 202 915-1: "Open Service Access (OSA); Application Programming Interface (API); Part 1: Overview (Parlay 4)".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in ES 202 915-1 apply.

3.2 Abbreviations

For the purposes of the present document, the abbreviations defined in ES 202 915-1 apply.

4 Common Data Definitions

The following clauses describe each aspect of the Common data definitions.

The order is as follows:

- The Data Definitions clause shows a detailed expansion of each of the data types associated with the methods within the classes.
- IDL description of the data types (normative annex).
- WSDL description of the data types (informative annex).
- Reference to the Java™ API description of the data types (informative annex).

5 Common System Data Definitions

These data definitions are assumed to be provided by the client operating system.

5.1 Standard Data Types

The APIs assume that the following data types can be supported.

5.1.1 TpBoolean

Defines a Boolean data type.

5.1.2 TpInt32

Defines a signed 32-bit integer.

5.1.3 TpFloat

Defines a single precision real number.

5.1.4 TpLongString

Defines a Byte string, comprising length and data. The length must be at least a 32-bit integer.

5.1.5 TpOctet

Defines an 8-bit quantity that is not translated during transmission.

5.1.6 TpOctetSet

Defines a Numbered List of Data elements of TpOctet. Note that this is an ordered list.

5.1.7 TpString

Defines a Byte string, comprising length and data. The length must be at least a 16-bit integer.

5.1.8 TpAssignmentID

Defines an assignment ID with a value that is unique to an instance of an implementation of a given interface (i.e. an object), irrespective of the method invoked on it. This ID may be used, for example, to identify single or multiple event notifications enabled by an object; or by a requesting object to modify or stop functionality (e.g. event notifications, call load control) associated with a previously supplied assignment ID.

The assignment ID is identical to a TpInt32 type.

5.1.9 TpSessionID

Defines a session ID with a value that is at least unique within the context of a specific instance of an SCF. An instance of an SCF is a single service manager instance plus the associated subordinate instances. For example, a single MultiPartyCallControlManager instance plus all associated MultiPartyCall and MultiPartyCallLeg instances. The session ID is used to identify different sessions (e.g. different call or call leg sessions) of an interface capable of handling multiple sessions.

EXAMPLE 1: myCallObject may implement the IpCall interface. If so, myCallObject may handle multiple call sessions, and each call session will be identified by a call session ID value (e.g. 1, 2, 3) that is unique within the context of the SCF instance.

EXAMPLE 2: myCallAndCallLegObject may implement the IpCall and IpCallLeg interfaces. If so, myCallAndCallLegObject may handle multiple call sessions and multiple call leg sessions. Each call session will be identified by a call session ID value (e.g. 1, 2, 3) that is unique within the context of the SCF instance. Similarly, each call leg session will be identified by a call leg session ID value (e.g. 1, 2, 3, 4, 5, 6) that is also unique within the context of the SCF instance. Because call session IDs and call leg session IDs are different data types, overlapping values are permitted and their uniqueness still remains.

The session ID is identical to a TpInt32 type.

5.1.10 TpSessionIDSet

Defines a Numbered Set of Data Elements of TpSessionID.

5.1.11 TpAny

Defines a type that can hold any type. This is not restricted to only the primitive types.

5.1.12 TpAttribute

This is a `Sequence of Data Elements` containing the attribute name and value.

Sequence Element Name	Sequence Element Type	Notes
AttributeName	TpString	The name of the attribute.
AttributeValue	TpAttributeValue	The typed value(s) for the attribute.

5.1.13 TpAttributeValue

This is a tagged choice of data elements to hold attribute values of different complexity.

	Tag Element Type	
	TpAttributeTagInfo	

Tag Element Value	Choice Element Type	Choice Element Name
P_SIMPLE_TYPE	TpSimpleAttributeValue	SimpleValue
P_STRUCTURED_TYPE	TpStructuredAttributeValue	StructuredValue
P_XML_TYPE	TpXMLString	XMLValue

5.1.14 TpAttributeList

This is a `Numbered List of Data Elements` of type TpAttribute.

5.1.15 TpAttributeSet

This is a `Numbered Set of Data Elements` of type TpAttribute.

5.1.16 TpInt64

Defines a signed 64-bit integer.

5.1.17 TpVersion

This data type is identical to `TpString`. It is used to uniquely identify the implemented version of the framework or an SCF. The syntax for this datatype is defined as:

`P_<publishing body>_<version number>`

Where:

`<publishing body>` is one of the strings listed in the table below.

Character String Value	Description
PARLAY	Specification released by The Parlay Group.
ETSI	Specification released by ETSI.
3GPP	Specification released by 3GPP.

`<version number>` consists of numbers separated by underscores (e.g. 3_1). It is recommended that not more than the two most significant numbers (major and minor version) of the version are used.

Examples of version strings are:

Character String Value	Description
P_PARLAY_3_1	Parlay v3.1.
P_ETSI_2_0	ETSI v2.0.
P_3GPP_4_3	3GPP Release 4.3.

Note that different version strings can be aliases of each other all pointing to the same SCF/Framework version.

5.1.18 TpStringSet

Defines a `Numbered Set of Data Elements` of type `TpString`.

5.1.19 TpStringList

Defines a `Numbered List of Data Elements` of type `TpString`.

5.1.20 TpAttributeTagInfo

TpAttributeTagInfo is an enumerated type used as a discriminator for the TpAttributeValue structure, and can contain the following values:

Name	Value	Description
P_SIMPLE_TYPE	0	Simple type
P_STRUCTURED_TYPE	1	Structured type
P_XML_TYPE	2	XML type

5.1.21 TpSimpleAttributeValue

This is a tagged choice of data elements to hold attribute values of different complexity.

	Tag Element Type	
	TpSimpleAttributeTypeInfo	

Tag Element Value	Choice Element Type	Choice Element Name
P_BOOLEAN	TpBoolean	BooleanValue
P_OCTET	TpOctet	OctetValue
P_CHAR	TpChar	CharValue
P_WCHAR	TpWChar	WCharValue
P_STRING	TpString	StringValue
P_WSTRING	TpWString	WStringValue
P_INT16	TpInt16	Int16Value
P_UNSIGNED_INT16	TpUnsignedInt16	UnsignedInt16Value
P_INT32	TpInt32	Int32Value
P_UNSIGNED_INT32	TpUnsignedInt32	UnsignedInt32Value
P_INT64	TpInt64	Int64Value
P_UNSIGNED_INT64	TpUnsignedInt64	UnsignedInt64Value
P_FLOAT	TpFloat	FloatValue
P_DOUBLE	TpDouble	DoubleValue
P_FIXED	TpFixed	FixedValue

5.1.22 TpSimpleAttributeTypeInfo

TpSimpleAttributeTypeInfo is an enumerated type used as a discriminator for the TpSimpleAttributeValue structure, and can contain the following values:

Name	Value	Description
P_BOOLEAN	0	Attribute type is type TpBoolean.
P_OCTET	1	Attribute type is type TpOctet.
P_CHAR	2	Attribute type is type TpChar.
P_WCHAR	3	Attribute type is type TpWChar.
P_STRING	4	Attribute type is type TpString.
P_WSTRING	5	Attribute type is type TpWString.
P_INT16	6	Attribute type is type TpInt16.
P_UNSIGNED_INT16	7	Attribute type is type TpUnsignedInt16.
P_INT32	8	Attribute type is type TpInt32.
P_UNSIGNED_INT32	9	Attribute type is type TpUnsignedInt32.
P_INT64	10	Attribute type is type TpInt64.
P_UNSIGNED_INT64	11	Attribute type is type TpUnsignedInt64.
P_FLOAT	12	Attribute type is type TpFloat.
P_DOUBLE	13	Attribute type is type TpDouble.
P_FIXED	14	Attribute type is type TpFixed.

5.1.23 TpStructuredAttributeType

This data type is identical to a TpString, and is defined as a string of characters that uniquely identifies the type of a structured data type. Network operator specific capabilities may also be used, but should be preceded by the string "SP_". The pattern of values is defined, where module names and class names map to a fully specified class name.

Character String Value	Description
P_module1/module2/module3/className	An object of the specified, fully qualified class.

5.1.24 TpStructuredAttributeValue

This is a `Sequence of Data Elements` containing the structured attribute type tag and the value to be interpreted using that type.

Sequence Element Name	Sequence Element Type	Notes
Type	TpStructuredAttributeType	The type for the value.
Value	TpAny	The structured values for the attribute.

5.1.25 TpChar

This type is an 8-bit quantity that may undergo conversion when transmitted by the communication system.

5.1.26 TpWChar

This type is a quantity that may undergo conversion when transmitted by the communication system. The size of this type is implementation-dependent.

5.1.27 TpWString

Defines a TpWChar string, comprising length and data. The length shall be at least a 16-bit integer.

5.1.28 TpInt16

Defines a signed 16-bit integer.

5.1.29 TpUnsignedInt16

Defines an unsigned 16-bit integer.

5.1.30 TpUnsignedInt32

Defines an unsigned 32-bit integer.

5.1.31 TpUnsignedInt64

Defines an unsigned 64-bit integer.

5.1.32 TpDouble

Defines a double precision real number.

5.1.33 TpFixed

This data type defines a fixed-point decimal number of up to 31 significant digits. The scale factor is a non-negative integer less than or equal to the total number of digits.

5.1.34 TpXMLString

This data type is TpString containing well-formed XML and may contain a reference to/include a DTD or Schema.

5.1.35 TpUnorderedOctetSet

Defines a Numbered Set of Data elements of TpOctet. Note that this is an un-ordered set.

Note that this type should not be removed from the present document, even if unused by any part of the OSA specifications. It is included to ensure that TpOctetSet is correctly used as a Numbered List of Data Elements, and not a Numbered Set.

5.2 Other Data Sorts

The APIs assumes that the following data syntaxes can be supported.

5.2.1 Sequence of Data Elements

This describes a sequence of data types. This may be defined as a structure (for example, in C++) or simply a sequence of data elements within a structure.

EXAMPLE: The TpAddress data type may be defined in C++ as:

```
typedef struct {
  TpAddressPlan              Plan;
  TpString                AddrString;
  TpString                Name;
  TpAddressPresentation      Presentation;
  TpAddressScreening      Screening;
  TpString            SubAddressString;
} TpAddress;
```

5.2.2 Tagged Choice of Data Elements

This describes a data type which actually evaluates to one of a choice of a number of data elements. This data element contains two parts: a tag data type (the *tag* part) which is used to identify the chosen data type, and the chosen data type itself (the *union* part). This form of data type is also referred to as a tagged union.

This data type can be implemented (for example, in C++) as a structure containing an integer for the *tag* part, and a union for the *union* part.

This data type is implementation specific. Please refer to the appropriate IDL documents (and the resulting language mappings) to see how this data type is implemented.

EXAMPLE: The `TpCallError` data type may be defined in C++ as:

```
typedef struct {
  TpCallErrorType Tag;
  union {
    TpCallErrorInfoUndefined     Undefined;
    TpCallErrorInfoRoutingAborted  RoutingAborted;
    TpCallErrorInfoCallAbandoned   CallAbandoned;
    TpCallErrorInfoInvalidAddress  InvalidAddress;
    TpCallErrorInfoInvalidState   InvalidState;
    TpCallErrorInfoInvalidCriteria  InvalidCriteria;
  } callErrorInfo;
} TpCallError;
```

5.2.3 Numbered Set of Data Elements

This describes a data type which comprises an integer which indicates the total number of data elements in the set (the *number* part), and an **unordered** set of data elements (the *data* part). *Set* data types do not contain duplicate data elements.

Note that TpOctetSet is a Numbered List of Data Elements, not a Numbered Set.

EXAMPLE: The TpAddressSet data type may be defined in C++ as:

```
typedef struct {
  TpInt32 Number;
  TpAddress Set[Number];
} TpAddressSet;
```

5.2.4 Reference

This describes a reference (or pointer) to a data type. When one interface inherits directly from another interface, a reference to the inheriting interface can be used when the base interface reference is required. For example, the references to Multi-Media Call Control interfaces can be used in Multi-Party Call Control methods that have been inherited by Multi-Media Call Control from Multi-Party Call Control interfaces.

5.2.5 Numbered List of Data Elements

This describes a data type which comprises an integer which indicates the total number of data elements in the set (the *number* part), and an **ordered** set of data elements (the *data* part). *List* data types can contain duplicate data elements.

EXAMPLE: The TpStringList data type may be defined in C++ as:

```
typedef struct {
  TpInt32 Number;
  TpString List[Number];
} TpStringList;
```

5.3 Interface Related Data Definitions

5.3.1 IpInterface

Defines the address of a generic interface instance.

5.3.2 IpInterfaceRef

Defines a Reference to type IpInterface.

5.4 Exception Classes

5.4.1 Underlying Technology Exceptions

All methods contain a signature showing, amongst other things, the explicit exceptions that they may throw. In addition to these exceptions, all methods can throw a number of implicit exceptions. These exceptions do not need to be included within the method signatures and are given below.

These exceptions would be thrown by the underlying technology (e.g. CORBA, Java™) as a result of problems encountered, for example, with the way the API method is invoked. They are a minimum set of exceptions that must be throwable by the underlying technology. Depending upon the underlying technology, additional method exceptions may also be thrown.

Description
Invalid Parameter: A method has been passed an invalid parameter argument
Invalid Parameter Value: A method parameter has been passed a value that is out of range
Parameter Missing: A method has not been passed a mandatory parameter argument

5.4.2 TpCommonExceptions

Defines the structure of the exception class which is applicable to all methods.

Structure Element Name	Structure Element Type	Structure Element Description
ExceptionType	TpInt32	Carries a constant from the list in the table below
ExtraInformation	TpString	Carries extra information to help identify the source of the exception, e.g. a parameter name

5.4.3 Constants associated with TpCommonExceptions

Name	Value	Description
P_RESOURCES_UNAVAILABLE	000Dh	The required resources in the network are not available.
P_TASK_REFUSED	000Eh	The requested method has been refused.
P_TASK_CANCELLED	000Fh	The requested method has been cancelled.
P_NO_CALLBACK_ADDRESS_SET	0011h	The requested method is refused because no callback address has been set (this may be the result of a timing issue between setting the callback address and invoking the method).
P_METHOD_NOT_SUPPORTED	0016h	The method is not allowed or supported within the context of the current service agreement.
P_INVALID_STATE	02E8h	Unexpected sequence of methods, i.e. the sequence does not match the specified state diagrams.

5.4.4 Exceptions available to all methods on all interfaces

The following are the list of exception classes which are available to all interfaces of the API.

Name	Description
P_APPLICATION_NOT_ACTIVATED	An application is unauthorised to access information and request services with regards to users that have deactivated that particular application. In case the request was for information related to multiple user identities the reference to user identities that are causing this exception will be returned in the extra information of the exception.
P_INFORMATION_NOT_AVAILABLE	The requested information is not available. A reason might be that the information is unavailable in the core network or that the application is unauthorised to access the information. In case the request was for information related to multiple user identities, the reference to user identities that are causing this exception will be returned in the extra information of the exception.
P_INVALID_ADDRESS	Invalid address specified.
P_INVALID_AMOUNT	Invalid amount specified.
P_INVALID_ASSIGNMENT_ID	The assignment ID is invalid
P_INVALID_CRITERIA	Invalid criteria specified.
P_INVALID_CURRENCY	Invalid currency specified.
P_INVALID_EVENT_TYPE	Invalid event type.
P_INVALID_INTERFACE_NAME	Invalid interface name.
P_INVALID_INTERFACE_TYPE	The interface reference supplied by the client is the wrong type.
P_INVALID_NETWORK_STATE	Although the sequence of method calls is allowed by the gateway, the underlying protocol cannot support it. E.g. in some protocols some methods are only allowed by the protocol, when the call processing is suspended, e.g. after reporting an event that was monitored in interrupt mode.
P_INVALID_SESSION_ID	Invalid session ID.
P_INVALID_TIME_AND_DATE_FORMAT	Invalid date and time format provided.
P_UNAUTHORISED_PARAMETER_VALUE	A method parameter value violates the Service Level Agreement.

Name	Description
P_UNKNOWN_SUBSCRIBER	The subscriber is not known in the network or the application is unauthorised to access information. In case the request was for information related to multiple user identities, the reference to user identities that are causing this exception will be returned in the extra information of the exception.
P_UNSUPPORTED_ADDRESS_PLAN	An address contains an address plan which is not supported.
P_INVALID_VERSION	An invalid version is specified.

5.5 Date and Time Related Data Definitions

5.5.1 TpDate

This data type is identical to a TpString. It specifies the data in accordance with International Standard ISO 8601. This is defined as the string of characters in the following format:

YYYY-MM-DD

where the date is specified as:

YYYY four digits year

MM two digits month

DD two digits day

The date elements are separated by a hyphen character (-).

EXAMPLE: The 4 December 1998, is encoded as the string:

```
1998-12-04
```

5.5.2 TpTime

This data type is identical to a TpString. It specifies the time in accordance with International Standard ISO 8601. This is defined as the string of characters in the following format:

HH:MM:SS.mmm

or

HH:MM:SS.mmmZ

where the time is specified as:

HH two digits hours (24h notation)

MM two digits minutes

SS two digits seconds

mmm three digits fractions of a second (i.e. milliseconds)

The time elements are separated by a colon character (:). The date and time are separated by a space. Optionally, a capital letter Z may be appended to the time field to indicate Universal Time (UTC). Otherwise, local time is assumed.

EXAMPLE: 10:30 and 15 seconds is encoded as the string:

`10:30:15.000`

for local time, or in UTC it would be: `10:30:15.000Z`

5.5.3 TpDateAndTime

This data type is identical to a TpString. It specifies the data and time in accordance with International Standard ISO 8601. This is defined as the string of characters in the following format:

`YYYY-MM-DD HH:MM:SS.mmm`

or

`YYYY-MM-DD HH:MM:SS.mmmZ`

where the date is specified as:

YYYY	four digits year
MM	two digits month
DD	two digits day

The date elements are separated by a hyphen character (-).

The time is specified as:

HH	two digits hours (24h notation)
MM	two digits minutes
SS	two digits seconds
mmm	three digits fractions of a second (i.e. milliseconds)

The time elements are separated by a colon character (`:`). The date and time are separated by a space. Optionally, a capital letter `Z` may be appended to the time field to indicate Universal Time (UTC). Otherwise, local time is assumed.

EXAMPLE: The 4 December 1998, at 10:30 and 15 seconds is encoded as the string:

`1998-12-04 10:30:15.000`

for local time, or in UTC it would be:

`1998-12-04 10:30:15.000Z`

5.5.4 TpDuration

This data type is a TpInt32 representing a time interval in milliseconds. A value of "-1" defines infinite duration and a value of "-2" represents a default duration.

5.5.5 TpTimeInterval

Defines the Sequence of Data Elements that specify a time interval.

Sequence Element Name	Sequence Element Type
StartTime	TpDateAndTime
StopTime	TpDateAndTime

5.6 Address Related Data Definitions

5.6.1 TpAddress

Defines the Sequence of Data Elements that specify an address.

Sequence Element Name	Sequence Element Type
Plan	TpAddressPlan
AddrString	TpString
Name	TpString
Presentation	TpAddressPresentation
Screening	TpAddressScreening
SubAddressString	TpString

The AddrString defines the actual address information and the structure of the string depends on the Plan. The following table gives an overview of the format of the AddrString for the different address plans.

Address Plan	AddrString Format Description	Example
P_ADDRESS_PLAN_NOT_PRESENT	Not applicable	
P_ADDRESS_PLAN_UNDEFINED	Not applicable	
P_ADDRESS_PLAN_IP	For Ipv4 the dotted quad notation is used. Also for IPv6 the dotted notation is used. The address can optionally be followed by a port number separated by a colon.	"127.0.0.1:42"
P_ADDRESS_PLAN_MULTICAST	An Ipv4 class D address or Ipv6 equivalent in dotted notation.	"224.0.0.0"
P_ADDRESS_PLAN_UNICAST	A non-multicast or broadcast IP address in dotted notation.	"127.0.0.1"
P_ADDRESS_PLAN_E164	An international number without the international access code, including the country code and excluding the leading zero of the area code.	"31161249111"
P_ADDRESS_PLAN_AESA	The ATM End System Address in binary format (40 bytes)	01234567890ABCDEF01234567 890ABCDEF01234567
P_ADDRESS_PLAN_URL	A uniform resource locator as defined in RFC 1738	"http://www.parlay.org"
P_ADDRESS_PLAN_NSAP	The binary representation of the Network Service Access Point	490001AA000400010420
P_ADDRESS_PLAN_SMTP	An e-mail address as specified in RFC 822	"webmaster@parlay.org"
P_ADDRESS_PLAN_X400	The X400 address structured as a set of attribute value pairs separated by semicolons.	"C=nl;ADMD= ;PRMD=uninet;O=parlay;S=Doe; I=S;G=John'
P_ADDRESS_PLAN_SIP (note 1)	Any valid address string allowed in RFC 3261 "SIP: Session Initiation Protocol"	"sip:user@parlay.org" "tel:+358-555-1234567;postd=pp22" "<sip:enquiries@1.2.3.4:5060> Enquiries"
P_ADDRESS_PLAN_ANY (note 2)	Not applicable	
P_ADDRESS_PLAN_NATIONAL	Reserved for National Specific use	Refer to relevant National Numbering Plan Specification

NOTE 1: It should be noted that two SIP addresses will be regarded as equivalent by a gateway if they correspond to the same user at the same network address. The textual form of the two addresses need not be the same. For example, sip:enquiries@parlay.org will be deemed to match <sip:Enquiries@1.2.3.4:5060>Enquiries (if parlay.org resolves to 1.2.3.4).
NOTE 2: This is only to be used with TpAddressRange.

5.6.2 TpAddressSet

Defines a Numbered Set of Data Elements of TpAddress.

5.6.3 TpAddressPresentation

Defines whether an address can be presented to an end user.

Name	Value	Description
P_ADDRESS_PRESENTATION_UNDEFINED	0	Undefined
P_ADDRESS_PRESENTATION_ALLOWED	1	Presentation Allowed
P_ADDRESS_PRESENTATION_RESTRICTED	2	Presentation Restricted
P_ADDRESS_PRESENTATION_ADDRESS_NOT_AVAILABLE	3	Address not available for presentation

5.6.4 TpAddressScreening

Defines whether an address can be presented to an end user.

Name	Value	Description
P_ADDRESS_SCREENING_UNDEFINED	0	Undefined
P_ADDRESS_SCREENING_USER_VERIFIED_PASSED	1	user provided address verified and passed
P_ADDRESS_SCREENING_USER_NOT_VERIFIED	2	user provided address not verified
P_ADDRESS_SCREENING_USER_VERIFIED_FAILED	3	user provided address verified and failed
P_ADDRESS_SCREENING_NETWORK	4	Network provided address (Note that even though the application may provide the address to the gateway, from the end-user point of view it is still regarded as a network provided address)

5.6.5 TpAddressPlan

Defines the address plan (or numbering plan) used. It is also used to indicate whether an address is actually defined in a TpAddress data element.

Name	Value	Description
P_ADDRESS_PLAN_NOT_PRESENT	0	No Address Present
P_ADDRESS_PLAN_UNDEFINED	1	Undefined
P_ADDRESS_PLAN_IP	2	IP
P_ADDRESS_PLAN_MULTICAST	3	Multicast
P_ADDRESS_PLAN_UNICAST	4	Unicast
P_ADDRESS_PLAN_E164	5	ITU-T Recommendation E.164
P_ADDRESS_PLAN_AESA	6	AESA
P_ADDRESS_PLAN_URL	7	URL
P_ADDRESS_PLAN_NSAP	8	NSAP
P_ADDRESS_PLAN_SMTP	9	SMTP
<<deprecated>> P_ADDRESS_PLAN_MSMAIL (see note)	10	Microsoft Mail
P_ADDRESS_PLAN_X400	11	ITU-T Recommendation X.400
P_ADDRESS_PLAN_SIP	12	Any URL scheme which is allowed in RFC 3261 "SIP: Session Initiation Protocol"
P_ADDRESS_PLAN_ANY	13	Any address plan is deemed to match (This is only used for TpAddressRange)
P_ADDRESS_PLAN_NATIONAL	14	Reserved for National Specific use
NOTE: This value is not to be used.		

For the case where the P_ADDRESS_PLAN_NOT_PRESENT and P_ADDRESS_PLAN_ANY are indicated, the rest of the information in the TpAddress is not valid.

5.6.6 TpAddressError

Defines the reasons why an address is invalid.

Name	Value	Description
P_ADDRESS_INVALID_UNDEFINED	0	Undefined error
P_ADDRESS_INVALID_MISSING	1	Mandatory address not present
P_ADDRESS_INVALID_MISSING_ELEMENT	2	Mandatory address element not present
P_ADDRESS_INVALID_OUT_OF_RANGE	3	Address is outside of the valid range
P_ADDRESS_INVALID_INCOMPLETE	4	Address is incomplete
P_ADDRESS_INVALID_CANNOT_DECODE	5	Address cannot be decoded

5.6.7 TpAddressRange

Defines the Sequence of Data Elements that specify a range of addresses.

Sequence Element Name	Sequence Element Type
Plan	TpAddressPlan
AddrString	TpString
Name	TpString
SubAddressString	TpString

The AddrString defines the actual address information and the structure of the string depends on the Plan.

An overview of the AddrString formats can be found at the description of the TpAddress data-type.

The difference with TpAddress is that there is no Presentation and Screening elements, the AddrString can contain wildcards and Plan may contain P_ADDRESS_PLAN_ANY.

If P_ADDRESS_PLAN_ANY is set then the TpAddressRange will be deemed by the gateway to match any TpAddress. If a specific Plan is set (including P_ADDRESS_PLAN_NOT_PRESENT) then the address plan of the range must be identical to the plan contained in an address for the two to match.

Two wildcards are allowed: * which matches zero or more characters and ? which matches exactly one character. For E.164 addresses, * which matches zero or more characters and ? are allowed at the beginning or end.

Some examples for E.164 addresses:

- "123" matches specific number;
- "123*" matches all numbers starting with 123 (including 123 itself);
- "123??*" matches all numbers starting with 123 and at least 5 digits long;
- "123???" matches all numbers starting with 123 and exactly 6 digits long;
- "*" matches any address.

The following address ranges are illegal:

- "1?3"
- "1*3"
- "?123*"
- ""

Legal occurrences of the '*' and '?' characters in AddrString should be escaped by a '\' character. To specify a '\' character '\\' must be used.

For e-mail style addresses, the wildcards are allowed at the beginning of the AddrString:

- "*@parlay.org" matches all email addresses in the parlay.org domain.

For SIP addresses, a range of wildcard options may be supported:

- "sip:*@parlay.org" matches all SIP addresses at parlay.org:5060;
- 'sip:*.domain1.com' matches all SIP addresses in the domain domain1.com;
- 'sip:user*@*' matches all users starting with the string 'user';
- 'sip:user*@*:5060' matches all users starting with the string 'user' using port 5060;
- 'sip:*' matches all SIP addresses;
- 'sip:*@*' matches all SIP addresses;
- 'sip:user*@*:5060' matches all users starting with the string 'user' using port 5060.

In addition for SIP addresses, the absence of an explicit port number at the end of a SIP address, shall be considered to indicate an implicit wildcard for any port number.

- 'sip:user1@parlay.org' may be considered equivalent to 'sip:user1@parlay.org:*'

5.6.8 TpURL

This data type is identical to a TpString and contains a URL address. The usage of this type is distinct from TpAddress, which can also hold a URL. The latter contains a user address which can be specified in many ways: IP, e-mail, URL etc. On the other hand, the TpURL type does not hold the address of a user and always represents a URL. This type is used in user interaction and defines the URL of the text or stream to be sent to an end-user. It is therefore inappropriate to use a general address here.

5.6.9 TpURN

This data type is identical to a TpString and contains a URN value. This type is used in SCFs such as Presence Service and defines the URN of a 'presentity' (i.e. a representation of an end-user). The syntax of what constitutes a valid URN is in RFC 1737 e.g. ('urn: foo.com: bar').

5.7 Price-related Data Definitions

5.7.1 TpPrice

This data type is identical to a TpString. It specifies price information. This is defined as a string of characters (digits) in the following format:

DDDDDD.DD

5.7.2 TpAoCInfo

Defines the Sequence of Data Elements that specify the Advice Of Charge information to be sent to the terminal.

Sequence Element Name	Sequence Element Type	Description
ChargeOrder	TpAoCOrder	Charge order
Currency	TpString	Currency unit according to ISO 4217

5.7.3 TpAoCOrder

Defines the Tagged Choice of Data Elements that specify the charge plan for the call.

	Tag Element Type	
	TpCallAoCOrderCategory	

Tag Element Value	Choice Element Type	Choice Element Name
P_CHARGE_ADVICE_INFO	TpChargeAdviceInfo	ChargeAdviceInfo
P_CHARGE_PER_TIME	TpChargePerTime	ChargePerTime
P_CHARGE_NETWORK	TpString	NetworkCharge

5.7.4 TpCallAoCOrderCategory

Name	Value	Description
P_CHARGE_ADVICE_INFO	0	Set of GSM Charge Advice Information elements according to TS 122 024
P_CHARGE_PER_TIME	1	Charge per time
P_CHARGE_NETWORK	2	Operator specific charge plan specification, e.g. charging table name / charging table entry

5.7.5 TpChargeAdviceInfo

Defines the Sequence of Data Elements that specify the two sets of Advice of Charge parameters. The first set defines the current tariff. The second set may be used in case of a tariff switch in the network.

Sequence Element Name	Sequence Element Type	Description
CurrentCAI	TpCAIElements	Current tariff
NextCAI	TpCAIElements	Next tariff after tariff switch

5.7.6 TpCAIElements

Defines the Sequence of Data Elements that specify the Charging Advice Information elements according to TS 122 024.

Sequence Element Name	Sequence Element Type	Description
UnitsPerInterval	TpInt32	Units per interval
SecondsPerTimeInterval	TpInt32	Seconds per time interval
ScalingFactor	TpInt32	Scaling factor
UnitIncrement	TpInt32	Unit increment
UnitsPerDataInterval	TpInt32	Units per data interval
SegmentsPerDataInterval	TpInt32	Segments per data interval
InitialSecsPerTimeInterval	TpInt32	Initial secs per time interval

5.7.7 TpChargePerTime

Defines the Sequence of Data Elements that specify the time based charging information.

Sequence Element Name	Sequence Element Type	Description
InitialCharge	TpInt32	Initial charge amount (in currency units × 0,0001)
CurrentChargePerMinute	TpInt32	Current tariff (in currency units × 0,0001)
NextChargePerMinute	TpInt32	Next tariff (in currency units × 0,0001) after tariff switch Only used in setAdviceOfCharge()

5.7.8 TpLanguage

This data type is identical to a TpString, and defines the language. In case an indication for the language is not needed an empty string must be used. In other cases valid language strings are defined in ISO 639.

5.8 Data Types Common Across Call Control and Data Session Control

5.8.1 TpDataSessionQosClass

Defines the Quality of Service (QoS) classes. This could be for a data session or multi media call session, for example.

Name	Value	Description
P_DATA_SESSION_QOS_CLASS_CONVERSATIONAL	0	Specifies the Conversational QoS class, as specified in TS 123 107.
P_DATA_SESSION_QOS_CLASS_STREAMING	1	Specifies the Streaming QoS class, as specified in TS 123 107.
P_DATA_SESSION_QOS_CLASS_INTERACTIVE	2	Specifies the Interactive QoS class, as specified in TS 123 107.
P_DATA_SESSION_QOS_CLASS_BACKGROUND	3	Specifies the Background QoS class, as specified in TS 123 107.

NOTE: Because of commonality across multiple interface specifications, this data type is promoted from Data Session Control Data Types to Common Data Types. For backward compatibility reasons, the naming of the data type remains unmodified, and hence continues to refer to Data Session Control.

Annex A (normative): OMG IDL Description of the Common Data definitions

The OMG IDL representation of the present document is contained in a text file (osa.idl) contained in archive es_20291502IDL.zip.

This archive can be found in es_20291502v010401m0.zip which accompanies the present document.

Annex B (informative): W3C WSDL Description of the Common Data definitions

Significant changes have occurred in Web Services technologies and understanding of how to best apply Web Services as a realisation of OSA. These changes are not reflected and therefore this realisation is removed. A future activity may provide a replacement for the content of this annex, reflective of current technology and usage expected.

Annex C (informative): Java™ API Description of the Common Data definitions

The Java™ API realisation of this interface specification is produced in accordance with the Java™ Realisation rules defined in ES 202 915-1. These rules aim to deliver for Java™, a developer API, provided as a realisation, supporting a Java™ API that represents the UML specifications. The rules support the production of both J2SE™ and J2EE™ versions of the API from the common UML specifications.

The J2SE™ representation of this interface specification is provided as Java™ Code, contained in archive 20291502J2SE.zip.

The J2EE™ representation of this interface specification is provided as Java™ Code, contained in archive 20291502J2EE.zip.

Both these archives can be found in es_20291502v010401m0.zip which accompanies the present document.

Annex D (normative): Exception Hierarchy

This clause arranges the OSA exceptions as a set of hierarchies that, depending upon the technology realisation, may or may not be utilised to simplify software developers' code.

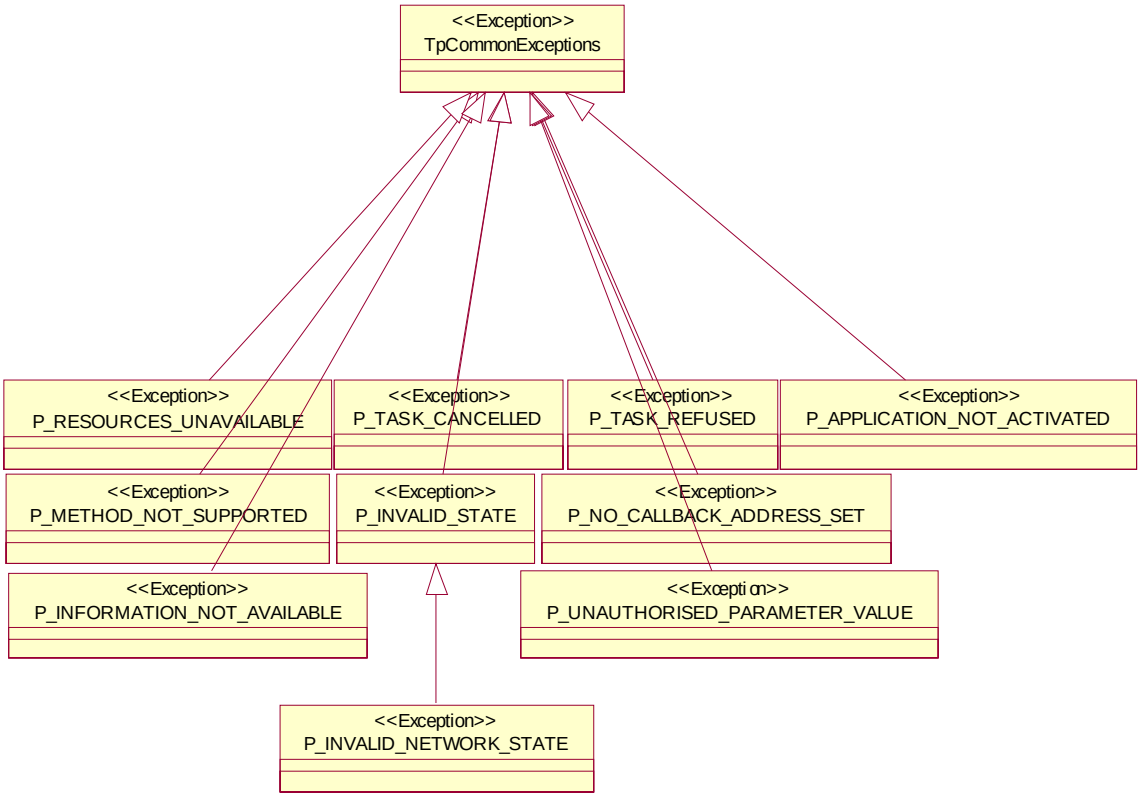
If the exception hierarchy is used in a particular realisation, the following lists all the OSA abstract exceptions:

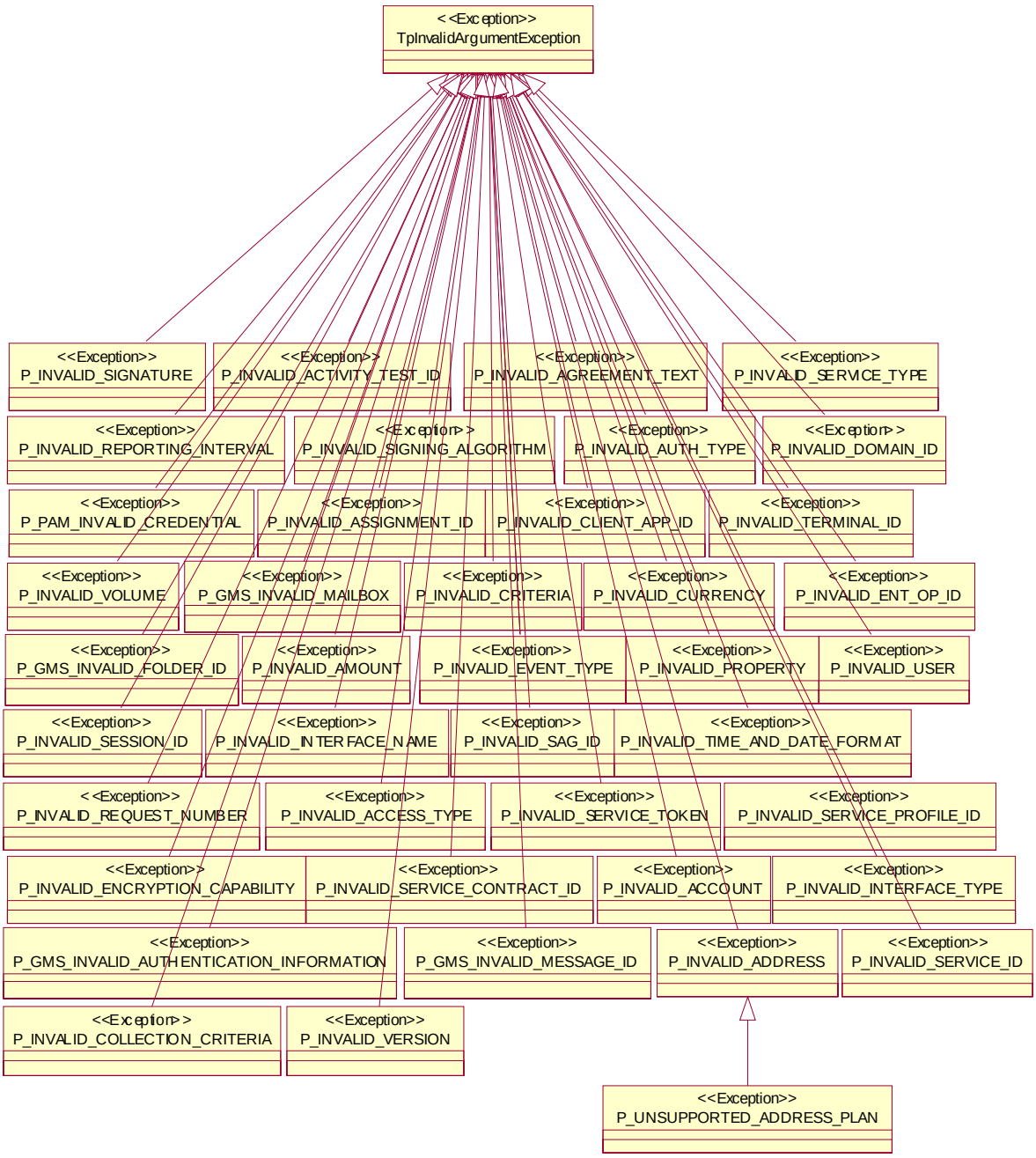
- TpCommonExceptions.
- TpInvalidArgumentException.
- TpDataSessionException.
- TpAccountException.
- TpConnectivityException.
- TpFrameworkException.
- TpMobilityException.
- TpMessagingException.
- TpPamException.
- TpPolicyException.

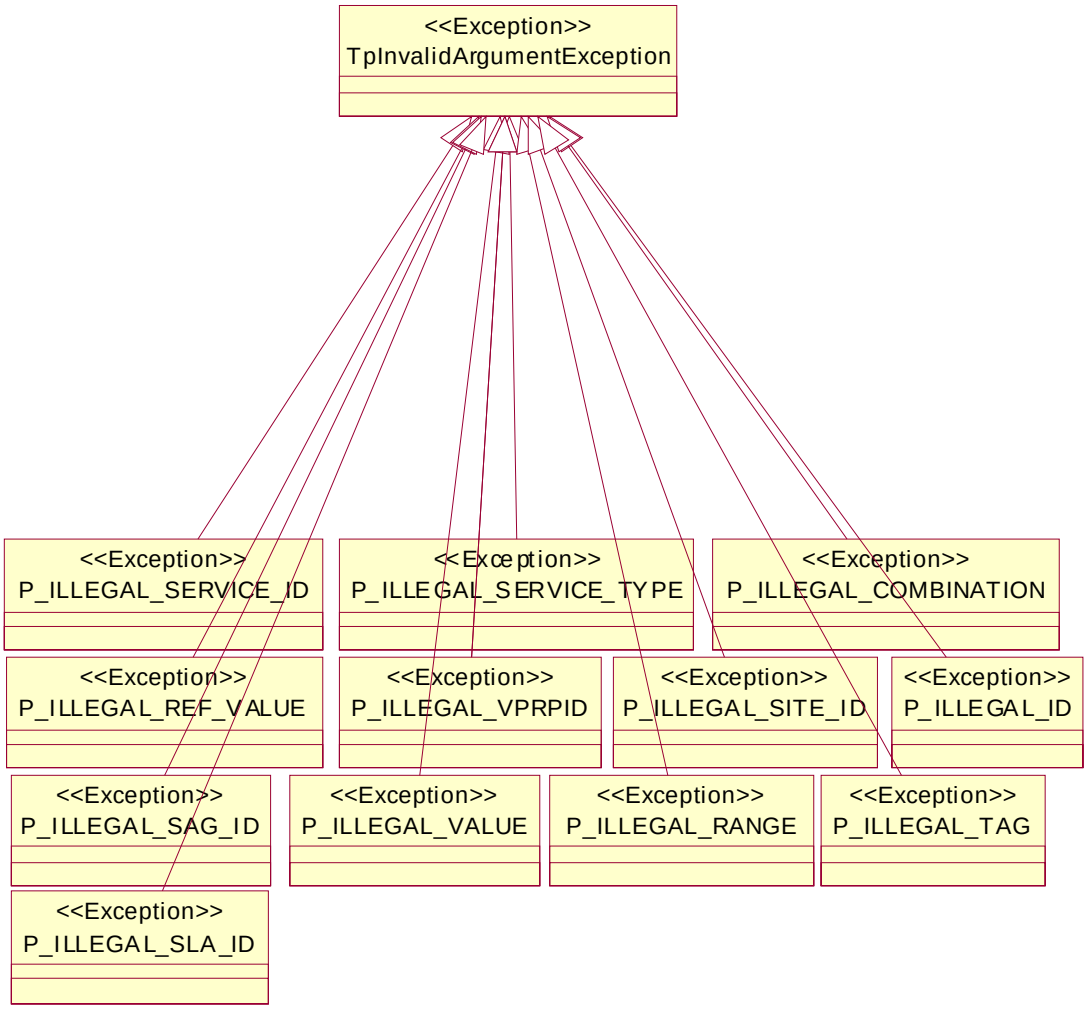
If the exception hierarchy is being used in a particular realisation, a software developer will have the option to catch these abstract exceptions and/or the detailed exceptions which extend them.

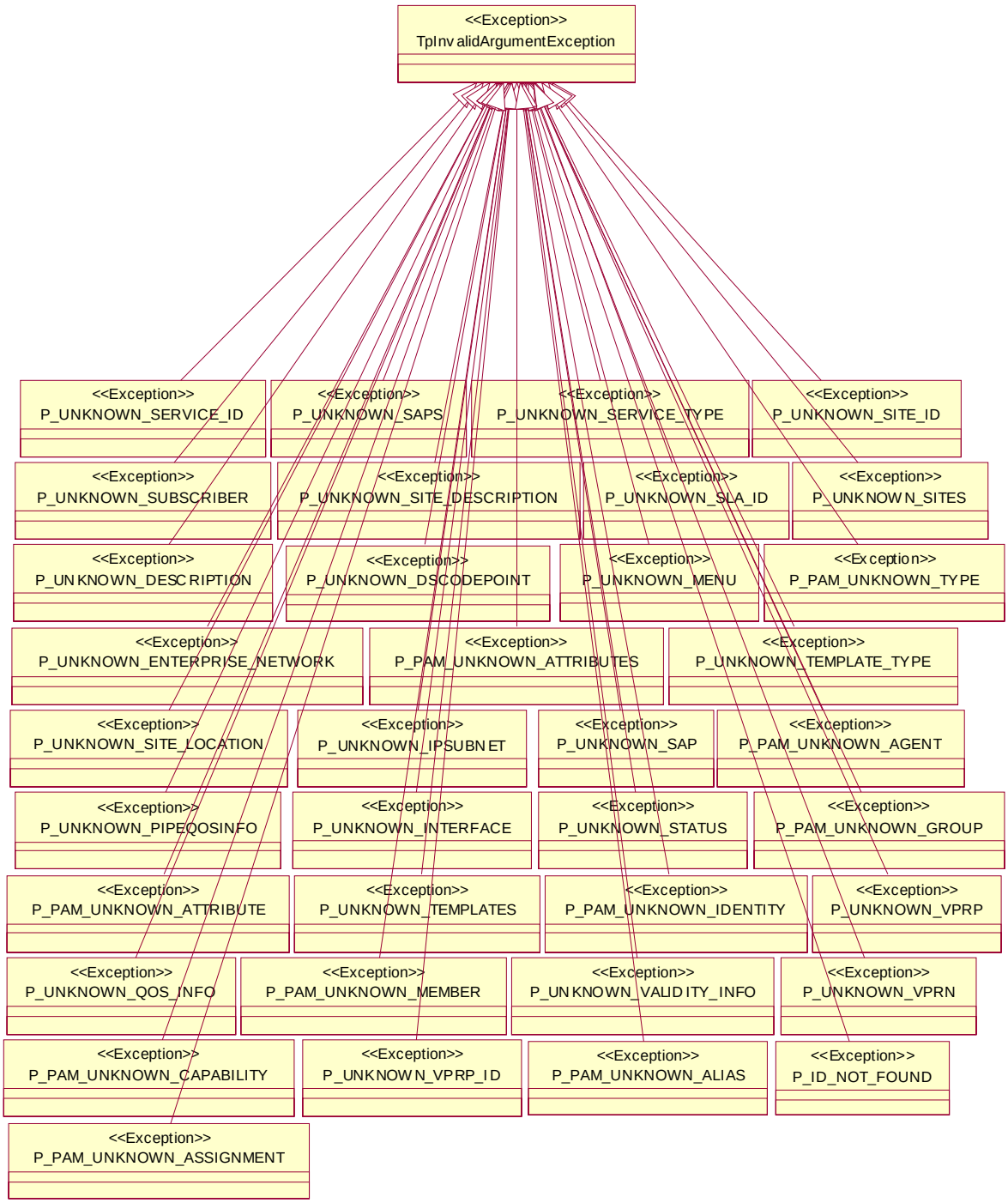
The following diagrams show all the OSA detailed exceptions, and how they relate to the abstract exceptions shown previously.

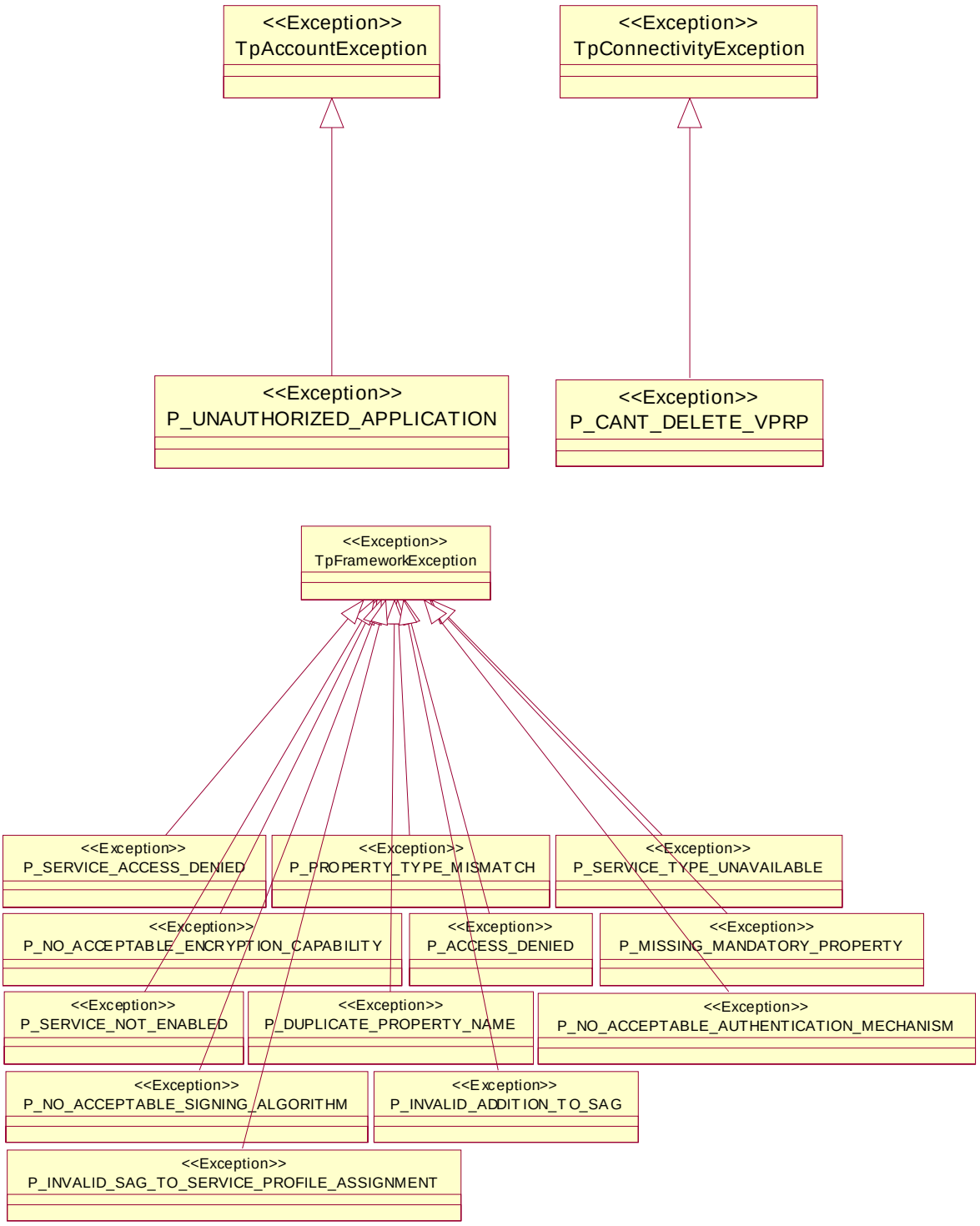
It should be noted that for those OSA methods that raise TpCommonExceptions, the P_RESOURCES_UNAVAILABLE, P_TASK_CANCELLED, P_TASK_REFUSED, P_METHOD_NOT_SUPPORTED, P_INVALID_STATE and P_NO_CALLBACK_ADDRESS_SET detailed exceptions should be raised individually in the method signature. The software developer will thus have the option of catching them individually or catching the TpCommonExceptions abstract exception.

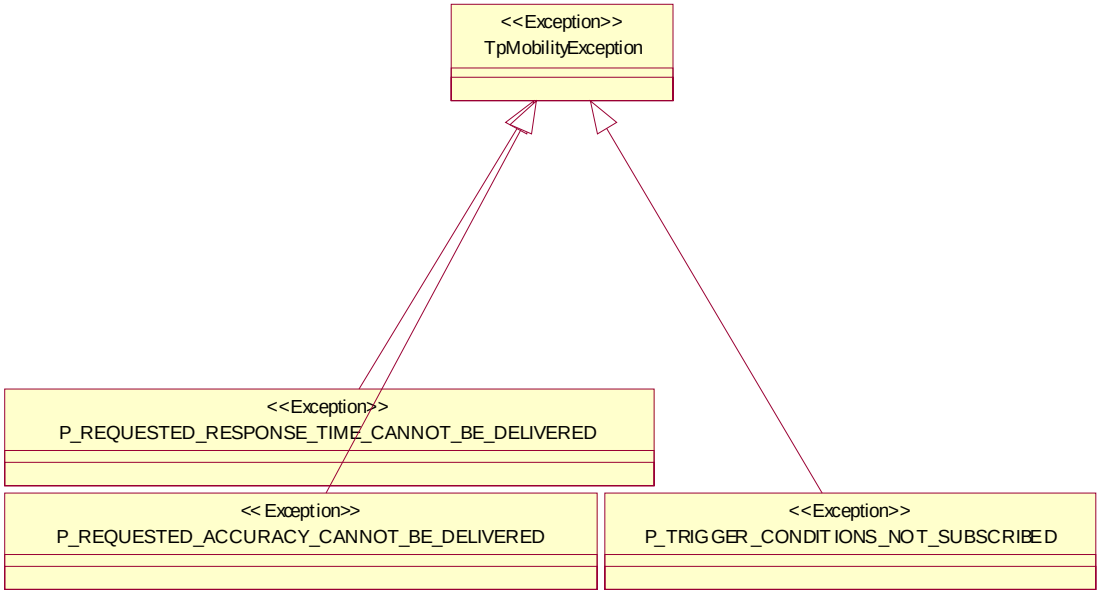

<<Exception>>
TpCommonExceptions
<<Exception>>
P_RESOURCES_UNAVAILABLE
<<Exception>>
P_TASK_CANCELLED
<<Exception>>
P_TASK_REFUSED
<<Exception>>
P_APPLICATION_NOT_ACTIVATED
<<Exception>>
P_METHOD_NOT_SUPPORTED
<<Exception>>
P_INVALID_STATE
<<Exception>>
P_NO_CALLBACK_ADDRESS_SET
<<Exception>>
P_INFORMATION_NOT_AVAILABLE
<<Exception>>
P_UNAUTHORISED_PARAMETER_VALUE
<<Exception>>
P_INVALID_NETWORK_STATE

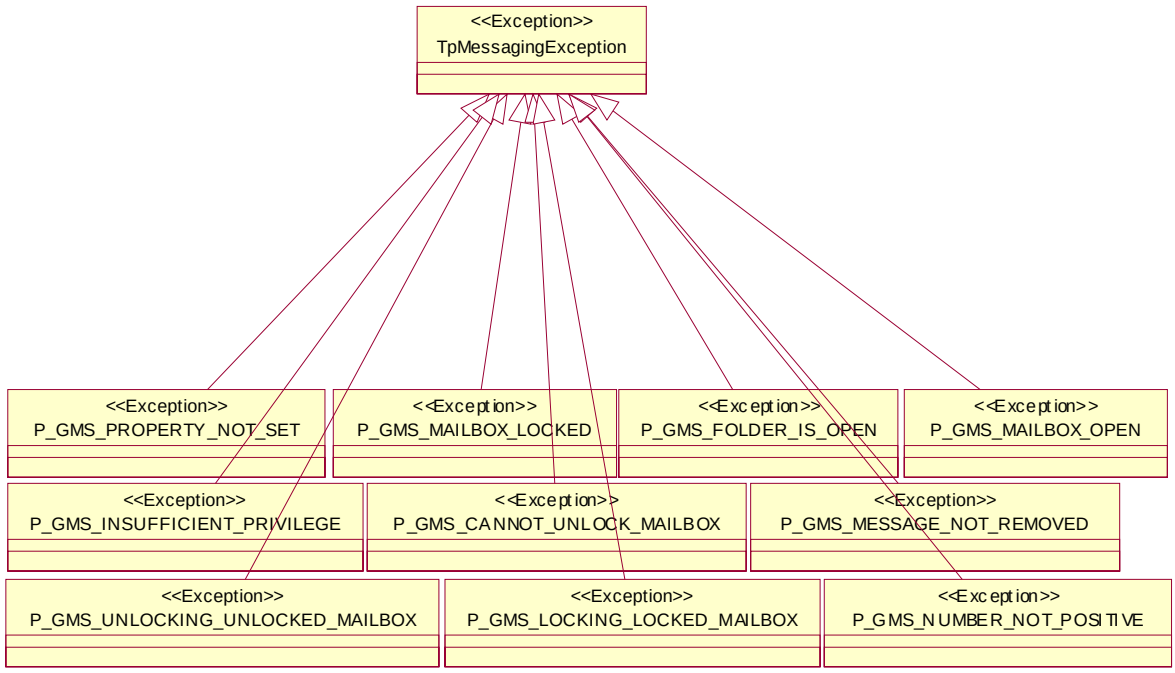

<<Exception>>
TpInvalidArgumentException
<<Exception>>
P_INVALID_SIGNATURE
<<Exception>>
P_INVALID_ACTIVITY_TEST_ID
<<Exception>>
P_INVALID_AGREEMENT_TEXT
<<Exception>>
P_INVALID_SERVICE_TYPE
<<Exception>>
P_INVALID_REPORTING_INTERVAL
<<Exception>>
P_INVALID_SIGNING_ALGORITHM
<<Exception>>
P_INVALID_AUTH_TYPE
<<Exception>>
P_INVALID_DOMAIN_ID
<<Exception>>
P_PAM_INVALID_CREDENTIAL
<<Exception>>
P_INVALID_ASSIGNMENT_ID
<<Exception>>
P_INVALID_CLIENT_APP_ID
<<Exception>>
P_INVALID_TERMINAL_ID
<<Exception>>
P_INVALID_VOLUME
<<Exception>>
P_GMS_INVALID_MAILBOX
<<Exception>>
P_INVALID_CRITERIA
<<Exception>>
P_INVALID_CURRENCY
<<Exception>>
P_INVALID_ENT_OP_ID
<<Exception>>
P_GMS_INVALID_FOLDER_ID
<<Exception>>
P_INVALID_AMOUNT
<<Exception>>
P_INVALID_EVENT_TYPE
<<Exception>>
P_INVALID_PROPERTY
<<Exception>>
P_INVALID_USER
<<Exception>>
P_INVALID_SESSION_ID
<<Exception>>
P_INVALID_INTERFACE_NAME
<<Exception>>
P_INVALID_SAG_ID
<<Exception>>
P_INVALID_TIME_AND_DATE_FORMAT
<<Exception>>
P_INVALID_REQUEST_NUMBER
<<Exception>>
P_INVALID_ACCESS_TYPE
<<Exception>>
P_INVALID_SERVICE_TOKEN
<<Exception>>
P_INVALID_SERVICE_PROFILE_ID
<<Exception>>
P_INVALID_ENCRYPTION_CAPABILITY
<<Exception>>
P_INVALID_SERVICE_CONTRACT_ID
<<Exception>>
P_INVALID_ACCOUNT
<<Exception>>
P_INVALID_INTERFACE_TYPE
<<Exception>>
P_GMS_INVALID_AUTHENTICATION_INFORMATION
<<Exception>>
P_GMS_INVALID_MESSAGE_ID
<<Exception>>
P_INVALID_ADDRESS
<<Exception>>
P_INVALID_SERVICE_ID
<<Exception>>
P_INVALID_COLLECTION_CRITERIA
<<Exception>>
P_INVALID_VERSION
<<Exception>>
P_UNSUPPORTED_ADDRESS_PLAN

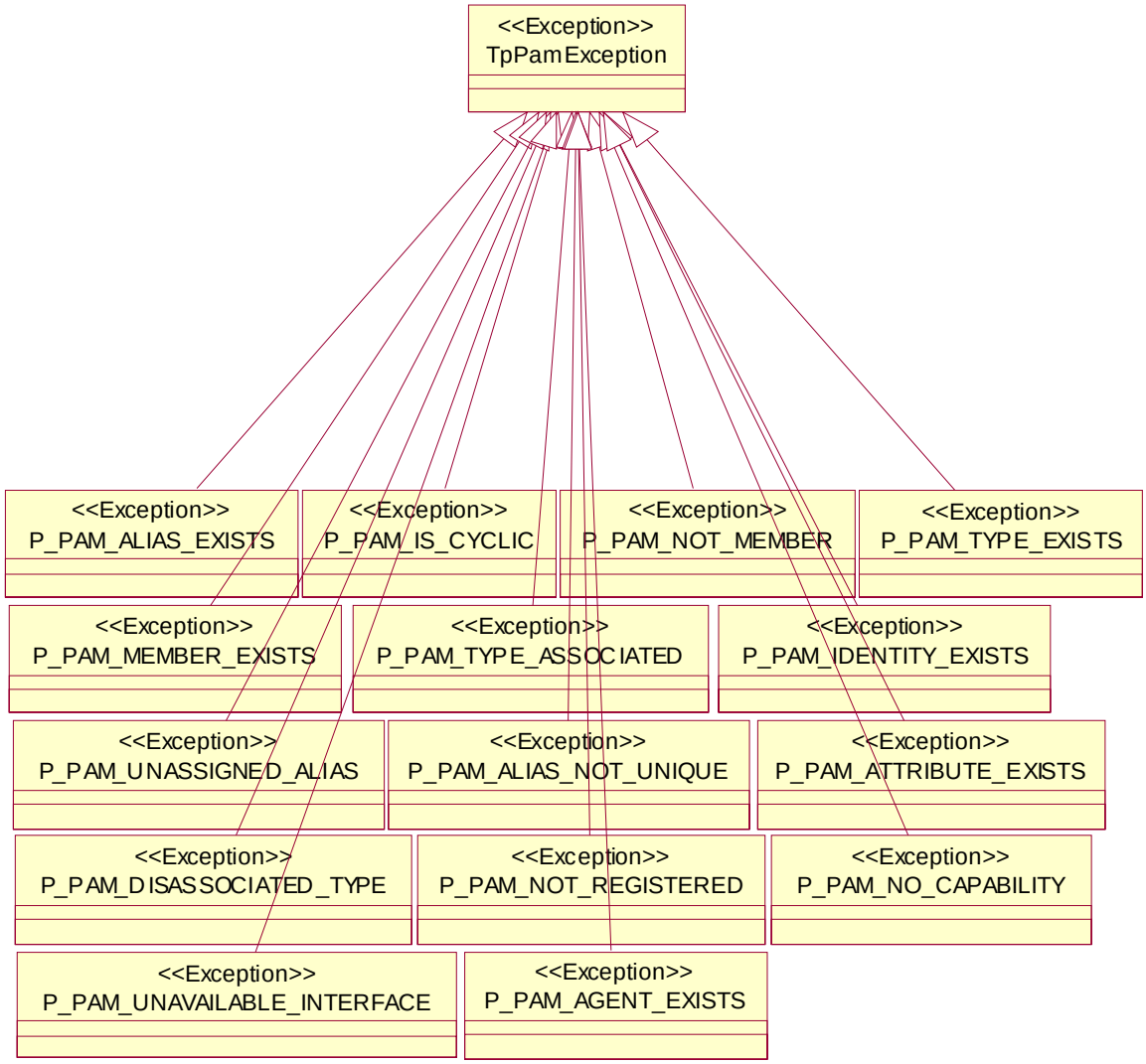

<<Exception>>
TpInvalidArgumentException
<<Exception>>
P_ILLEGAL_SERVICE_ID
<<Exception>>
P_ILLEGAL_SERVICE_TYPE
<<Exception>>
P_ILLEGAL_COMBINATION
<<Exception>>
P_ILLEGAL_REF_VALUE
<<Exception>>
P_ILLEGAL_VPRPID
<<Exception>>
P_ILLEGAL_SITE_ID
<<Exception>>
P_ILLEGAL_ID
<<Exception>>
P_ILLEGAL_SAG_ID
<<Exception>>
P_ILLEGAL_VALUE
<<Exception>>
P_ILLEGAL_RANGE
<<Exception>>
P_ILLEGAL_TAG
<<Exception>>
P_ILLEGAL_SLA_ID

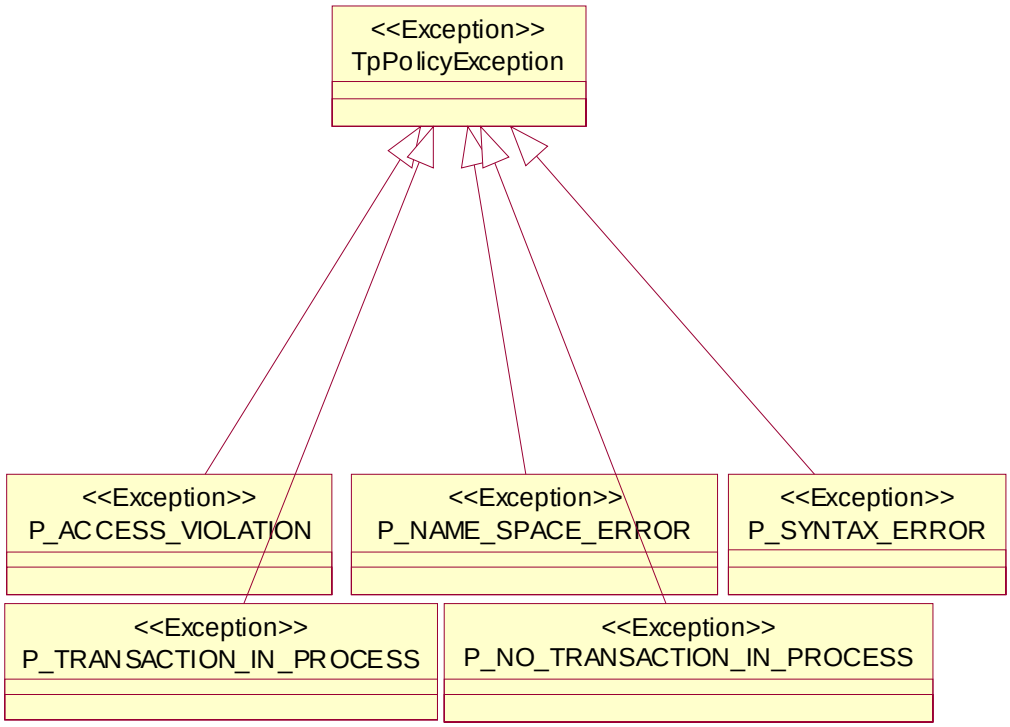

<<Exception>>
TpInvalidArgumentException
<<Exception>>
P_UNKNOWN_SERVICE_ID
<<Exception>>
P_UNKNOWN_SAPS
<<Exception>>
P_UNKNOWN_SERVICE_TYPE
<<Exception>>
P_UNKNOWN_SITE_ID
<<Exception>>
P_UNKNOWN_SUBSCRIBER
<<Exception>>
P_UNKNOWN_SITE_DESCRIPTION
<<Exception>>
P_UNKNOWN_SLA_ID
<<Exception>>
P_UNKNOWN_SITES
<<Exception>>
P_UNKNOWN_DESCRIPTION
<<Exception>>
P_UNKNOWN_DSCODEPOINT
<<Exception>>
P_UNKNOWN_MENU
<<Exception>>
P_PAM_UNKNOWN_TYPE
<<Exception>>
P_UNKNOWN_ENTERPRISE_NETWORK
<<Exception>>
P_PAM_UNKNOWN_ATTRIBUTES
<<Exception>>
P_UNKNOWN_TEMPLATE_TYPE
<<Exception>>
P_UNKNOWN_SITE_LOCATION
<<Exception>>
P_UNKNOWN_IPSUBNET
<<Exception>>
P_UNKNOWN_SAP
<<Exception>>
P_PAM_UNKNOWN_AGENT
<<Exception>>
P_UNKNOWN_PIPEQOSINFO
<<Exception>>
P_UNKNOWN_INTERFACE
<<Exception>>
P_UNKNOWN_STATUS
<<Exception>>
P_PAM_UNKNOWN_GROUP
<<Exception>>
P_PAM_UNKNOWN_ATTRIBUTE
<<Exception>>
P_UNKNOWN_TEMPLATES
<<Exception>>
P_PAM_UNKNOWN_IDENTITY
<<Exception>>
P_UNKNOWN_VPRP
<<Exception>>
P_UNKNOWN_QOS_INFO
<<Exception>>
P_PAM_UNKNOWN_MEMBER
<<Exception>>
P_UNKNOWN_VALIDITY_INFO
<<Exception>>
P_UNKNOWN_VPRN
<<Exception>>
P_PAM_UNKNOWN_CAPABILITY
<<Exception>>
P_UNKNOWN_VPRP_ID
<<Exception>>
P_PAM_UNKNOWN_ALIAS
<<Exception>>
P_ID_NOT_FOUND
<<Exception>>
P_PAM_UNKNOWN_ASSIGNMENT


<<Exception>>
TpAccountException
<<Exception>>
TpConnectivityException
<<Exception>>
P_UNAUTHORIZED_APPLICATION
<<Exception>>
P_CANT_DELETE_VPRP
<<Exception>>
TpFrameworkException
<<Exception>>
P_SERVICE_ACCESS_DENIED
<<Exception>>
P_PROPERTY_TYPE_MISMATCH
<<Exception>>
P_SERVICE_TYPE_UNAVAILABLE
<<Exception>>
P_NO_ACCEPTABLE_ENCRYPTION_CAPABILITY
<<Exception>>
P_ACCESS_DENIED
<<Exception>>
P_MISSING_MANDATORY_PROPERTY
<<Exception>>
P_SERVICE_NOT_ENABLED
<<Exception>>
P_DUPLICATE_PROPERTY_NAME
<<Exception>>
P_NO_ACCEPTABLE_AUTHENTICATION_MECHANISM
<<Exception>>
P_NO_ACCEPTABLE_SIGNING_ALGORITHM
<<Exception>>
P_INVALID_ADDITION_TO_SAG
<<Exception>>
P_INVALID_SAG_TO_SERVICE_PROFILE_ASSIGNMENT


<<Exception>>
TpMobilityException
<<Exception>>
P_REQUESTED_RESPONSE_TIME_CANNOT_BE_DELIVERED
<<Exception>>
P_REQUESTED_ACCURACY_CANNOT_BE_DELIVERED
<<Exception>>
P_TRIGGER_CONDITIONS_NOT_SUBSCRIBED


<<Exception>>
TpMessagingException
<<Exception>>
P_GMS_PROPERTY_NOT_SET
<<Exception>>
P_GMS_MAILBOX_LOCKED
<<Exception>>
P_GMS_FOLDER_IS_OPEN
<<Exception>>
P_GMS_MAILBOX_OPEN
<<Exception>>
P_GMS_INSUFFICIENT_PRIVILEGE
<<Exception>>
P_GMS_CANNOT_UNLOCK_MAILBOX
<<Exception>>
P_GMS_MESSAGE_NOT_REMOVED
<<Exception>>
P_GMS_UNLOCKING_UNLOCKED_MAILBOX
<<Exception>>
P_GMS_LOCKING_LOCKED_MAILBOX
<<Exception>>
P_GMS_NUMBER_NOT_POSITIVE


<<Exception>>
TpPamException
<<Exception>>
P_PAM_ALIAS_EXISTS
<<Exception>>
P_PAM_IS_CYCLIC
<<Exception>>
P_PAM_NOT_MEMBER
<<Exception>>
P_PAM_TYPE_EXISTS
<<Exception>>
P_PAM_MEMBER_EXISTS
<<Exception>>
P_PAM_TYPE_ASSOCIATED
<<Exception>>
P_PAM_IDENTITY_EXISTS
<<Exception>>
P_PAM_UNASSIGNED_ALIAS
<<Exception>>
P_PAM_ALIAS_NOT_UNIQUE
<<Exception>>
P_PAM_ATTRIBUTE_EXISTS
<<Exception>>
P_PAM_DISASSOCIATED_TYPE
<<Exception>>
P_PAM_NOT_REGISTERED
<<Exception>>
P_PAM_NO_CAPABILITY
<<Exception>>
P_PAM_UNAVAILABLE_INTERFACE
<<Exception>>
P_PAM_AGENT_EXISTS


<<Exception>>
TpPolicyException
<<Exception>>
P_ACCESS_VIOLATION
<<Exception>>
P_NAME_SPACE_ERROR
<<Exception>>
P_SYNTAX_ERROR
<<Exception>>
P_TRANSACTION_IN_PROCESS
<<Exception>>
P_NO_TRANSACTION_IN_PROCESS

Annex E (informative): Record of changes

The following is a list of the changes made to the present document for each release. The list contains the names of all changed, deprecated, added or removed items in the specifications and not the actual changes. Any type of change information that is important to the reader is put in the final clause of this annex.

Changes are specified as changes to the prior major release, but every minor release will have its own part of the table allowing the reader to know when the actual change was made.

E.1 Data Definitions

E.1.1 New

Identifier	Comments
Data Definitions added in ES 202 915-2 version 1.1.1 (Parlay 4.0)	
TpInt64	
TpVersionID	
TpStringSet	
TpStringList	
Data Definitions added in ES 202 915-2 version 1.2.1 (Parlay 4.1)	
TpDataSessionQosClass	Moved from Part 8 (Data Session Control)
Data Definitions added in ES 202 915-2 version 1.3.1 (Parlay 4.2)	
TpURN	
TpUnorderedOctetSet	Defined as Numbered Set of TpOctet
TpAttributeValue	
TpAttributeTagInfo	
TpSimpleAttributeValue	
TpSimpleAttributeTypeInfo	
TpStructuredAttributeType	
TpStructuredAttributeValue	
TpChar	
TpWChar	
TpWString	
TpInt16	
TpUnsignedInt16	
TpUnsignedInt32	
TpUnsignedInt64	
TpDouble	
TpFixed	
TpXMLString	
Data Definitions added in ES 202 915-2 version 1.4.1 (Parlay 4.3)	

E.1.2 Modified

Identifier	Comments
Data Definitions modified in ES 202 915-2 version 1.1.1 (Parlay 4.0)	
TpAddressPlan	P_ADDRESS_PLAN_MSMAIL is depreciated
TpAssignmentID	Scope of uniqueness modified
TpSessionID	Scope of uniqueness modified
Data Definitions modified in ES 202 915-2 version 1.2.1 (Parlay 4.1)	
TpAssignmentID	Description updated
TpAddressPlan	P_ADDRESS_PLAN_NATIONAL added
Data Definitions modified in ES 202 915-2 version 1.3.1 (Parlay 4.2)	
TpAddressRange	SIP wildcard options extended
TpOctetSet	Defined as Numbered List of TpOctet
TpAttribute	AttributeType field removed, AttributeValue field type changed to TpAttributeValue
Data Definitions modified in ES 202 915-2 version 1.4.1 (Parlay 4.3)	
TpAddressPlan	Enumeration values corrected in J2SE code in AddressPlan.java

E.1.3 Removed

Identifier	Comments
Data Definitions removed in ES 202 915-2 version 1.1.1 (Parlay 4.0)	
Data Definitions removed in ES 202 915-2 version 1.2.1 (Parlay 4.1)	
Data Definitions removed in ES 202 915-2 version 1.3.1 (Parlay 4.2)	
TpAttributeType	
Data Definitions removed in ES 202 915-2 version 1.4.1 (Parlay 4.3)	

E.2 Exceptions

E.2.1 New

Identifier	Comments
Exceptions added in ES 202 915-2 version 1.1.1 (Parlay 4.0)	
P_INVALID_VERSION	
Exceptions added in ES 202 915-2 version 1.2.1 (Parlay 4.1)	
Exceptions added in ES 202 915-2 version 1.3.1 (Parlay 4.2)	
Exceptions added in ES 202 915-2 version 1.4.1 (Parlay 4.3)	

E.2.2 Modified

Identifier	Comments
Exceptions modified in ES 202 915-2 version 1.1.1 (Parlay 4.0)	
Exceptions modified in ES 202 915-2 version 1.2.1 (Parlay 4.1)	
P_INVALID_STATE	Value modified to match IDL
Exceptions modified in ES 202 915-2 version 1.3.1 (Parlay 4.2)	
Exceptions modified in ES 202 915-2 version 1.4.1 (Parlay 4.3)	

E.2.3 Removed

Identifier	Comments
Exceptions removed in ES 202 915-2 version 1.1.1 (Parlay 4.0)	
P_SET_LENGTH_EXCEEDED	
Exceptions removed in ES 202 915-2 version 1.2.1 (Parlay 4.1)	
Exceptions removed in ES 202 915-2 version 1.3.1 (Parlay 4.2)	
Exceptions removed in ES 202 915-2 version 1.4.1 (Parlay 4.3)	

E.3 Others

ES 202 915-2 V1.4.1:

WSDL files in annex B deleted.

History

Document history		
V1.1.1	January 2003	Publication
V1.2.1	August 2003	Publication
V1.3.1	March 2005	Publication
V1.3.2	April 2005	Publication
V1.4.1	October 2006	Membership Approval Procedure MV 20061215: 2006-10-17 to 2006-12-15