TSGS#26(04)0793

Source: SA5 (Telecom Management)

Title: 4 Rel-6 CR 32.300/2/3 Configuration Management (CM); Name convention for Managed Objects / IS / CORBA SS

Document for: Approval

Agenda Item: 7.5.3

Doc1stevel	Specific a	CR	R	Phase	Subject	Ca	VersCu	Doc2ndLev	WorkitemsI D
SP-040793	32.300	003	--	Rel-6	Correct and convert formal specification from BNF syntax to EBNF with corrections	F	5.0.1	S5-047132	OAM-NIM
SP-040793	32.302	006	--	Rel-5	Add missing rules on how to construct the string NotificationCategory	F	5.1.0	S5-046982	OAM-NIM
SP-040793	32.302	007	--	Rel-6	Add missing rules on how to construct the string NotificationCategory	A	6.0.0	S5-046983	OAM-NIM
SP-040793	32.303	013	--	Rel-6	Remove filter requirement in IDL comments in the Notification IRP CORBA SS	F	6.1.0	S5-047039	OAM-NIM

CR-Form-v7

CHANGE REQUEST

⌘ 32.300 CR 003 ⌘rev - ⌘ Current version: 5.0.1 ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: UICC apps⌘ ☐ ME ☐ Radio Access Network X Core Network X

Title:	⌘	Correct and convert formal specification from BNF syntax to EBNF with corrections
Source:	⌘	**SA5** (Nortel Networks – Suzèle Lariven – lariven@nortelnetworks.com)
Work item code:	⌘	OAM-NIM
Date:	⌘	19/11/2004
Category:	⌘	**F**
Release:	⌘	Rel-6

Use one of the following categories:
F *(correction)*
A *(corresponds to a correction in an earlier release)*
B *(addition of feature),*
C *(functional modification of feature)*
D *(editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 (GSM Phase 2)
R96 (Release 1996)
R97 (Release 1997)
R98 (Release 1998)
R99 (Release 1999)
Rel-4 (Release 4)
Rel-5 (Release 5)
Rel-6 (Release 6)

Reason for change:	⌘	The formal definition is not correct and requires additional definition. The existing specification makes no distinction between the different name types such as RDNs for Domain Components, RDNs for object instances with an "Id" naming attribute, and escaped characters. This tightens the formal definition without adding new features.

Summary of change:	⌘	• Convert the existing BNF syntax to EBNF
		• Amend the formal definition to distinguish between the different name types:
		  • Null distinguished name
		  • Regular and Domain Component specific RDNs
		  • RDNs for object instances of classes with & without "Id" naming attribute
		  • Characters which are escaped
		• Explicit which characters are globally allowed for DN string representation
		• Add an informative annex to describe how to interpret EBNF
		• Editorial corrections

Consequences if not approved:	⌘	Ambiguities in DN name format which could lead to interworking problems

Clauses affected:	⌘	2, 3.1.7, 3.2, 7.A, 7.B, 7.1.1, 7.1.3, 7.2, 7.3, 8, 9.1, annex A, annex B, annex C, New annex D

	Y	N		
Other specs affected: ⌘		X	Other core specifications	⌘
		X	Test specifications	
		X	O&M Specifications	

Other comments:	⌘	Further work, discussion, and decision are required with regard to further tightening of EBNF definitions "ClassName" and "NamingAttributeName". Once finalized, this point will be addressed through complementary CR.

Change in Clause 2

2 References

[…]

[10] Void.~~IETF RFC 733: "Standard for the format of ARPA network text messages".~~

[11] 3GPP TS 32.101: "3G Telecom Management principles and high level requirements".

[12] 3GPP TS 32.102: "3G Telecom Management Architecture".

[13] ISO/IEC 14977: "Information technology – Syntactic metalanguage – Extended BNF".

[14] ISO/IEC 646: "Information technology – ISO 7-bit coded character set for information interchange".

[15] ISO/IEC 10646: "Information technology – Universal multiple-octet Coded Character Set (UCS)".

End of Change in Clause 2

Change in Clause 3.1.7

3.1.7 Distinguished Name and Relative Distinguished Name

[…]

```
~~DistinguishedName ::= RDNSequence~~
~~RDNSequence ::= SEQUENCE OF RelativeDistinguishedName~~
~~      RelativeDistinguishedName ::= SET SIZE (1..MAX) OFAttributeTypeAndValue~~
~~AttributeTypeAndValue ::= SEQUENCE {~~
~~          type AttributeType, value AttributeValue}~~

DistinguishedName ::= RDNSequence
RDNSequence ::= SEQUENCE OF RelativeDistinguishedName
RelativeDistinguishedName ::= SET SIZE (1..MAX) OF AttributeTypeAndValue
AttributeTypeAndValue ::= SEQUENCE {type AttributeType, value AttributeValue}
```

[…]

End of Change in Clause 3.1.7

Change in Clause 3.2

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

[…]

EBNF Extended Backus-Naur Form

[…]

~~EM Element Manager~~

[…]

~~MIB Management Information Base~~
~~MIM Management Information Model~~

[…]

~~NM Network Manager~~

[…]

~~OMG Object Management Group~~

[…]

~~UML Unified Modelling Language (OMG)~~

End of Change in Clause 3.2

Change in Clause 7.A

7 String Representation of DN

7.A Overview

This clause specifies the string representation of DN. This work is based on IETF RFC 2253 [7]. A DN string representation, using the string-encoding scheme specified in the present document, is also a valid DN string according to IETF RFC 2253 [7].

The string-encoding scheme specified in the present document imposes further restrictions as compared to IETF RFC 2253 [7]. The most important restrictions are:

- Multi-valued RDN is not supported in the subject name convention.
- Character ~~star~~ asterisk ~~('*', ASCII 42)~~ is used to denote wildcard in the subject name convention.

End of Change in Clause 7.A

Change in New Clause 7.B

7.B Allowed character sets

Subject to further restrictions described in the following subclauses, the allowed characters for the string representation of DN ~~defined in this specification and exchanged over the Itf-N are to comply with~~ are:

- Characters of ~~the IRV G0 set from~~ ISO/IEC 646 [14] International Reference Version (IRV) coded character set, and
- Characters of standard coded character sets supporting and extending ISO/IEC 646 [14] IRV coded character set, e.g. ISO/IEC 10646 [15] coded character set.

NOTE 1: ISO/IEC 646 [14] IRV coded character set is the international equivalent to the ANSI X3.4 ASCII coded character set.

NOTE 2: The character set of ISO/IEC 646 [14] IRV corresponds to the subset of characters that range from U+0000 to U+007F in the character set of ISO/IEC 10646 [15].

NOTE 3: The ISO/IEC 646 [14] IRV characters specifically referenced in this specification are further identified using ISO/IEC 10646 [15] character short identifier notation form "U+XXXX".

7.1 Converting DN from ASN.1 to a String

End of Change in New Clause 7.B

Change in Clause 7.1.1

7.1.1 Converting RDNSequence

[…]

The encoding of adjacent RDNs are separated by a comma character (~~",", ASCII 44~~', ', U+002C), to be consistent with IETF RFC 2253 [7].

White spaces adjacent to the ~~slash~~ comma character shall be ignored.

End of Change in Clause 7.1.1

Change in Clause 7.1.3

7.1.3 Converting AttributeTypeAndValue

The AttributeTypeAndValue is encoded as the string representation of the AttributeType, followed by an equals sign character (~~'=', ASCII 61~~'=', U+003D), followed by the string representation of the AttributeValue.

[…]

String representation of AttributeValue allows character escape mechanism such as the use of a ~~backslash~~ reverse solidus character ('\', U+005C) followed by two hexadecimal digits to replace a character in a string. String representation of AttributeType does not allow character escape mechanism.

EXAMPLE: "CN=Before\0DAfter,O=Test,C=GB". In this example, the ~~backslash~~ reverse solidus character and the two hexadecimal digits form a single byte in the code of the escaped character. The ~~backslash~~ reverse solidus character followed by "~~0D~~0D" indicates a carriage return character. See annex B for a rule for MO designers to avoid ambiguity concerning the AttributeType of a DN string.

End of Change in Clause 7.1.3

Change in Clause 7.2

7.2 Character syntax

[…]

1. Any character except ~~<special> where <special> is~~ :
   - ~~","~~comma character (', ', U+002C),
   - ~~"="~~equals sign character ('=', U+003D),
   - ~~<CR>~~carriage return character (U+000D),
   - ~~<LF>~~line feed character (U+000A),
   - ~~"+"~~plus sign character ('+', U+002B),
   - ~~"<"~~less-than sign character ('<', U+003C),
   - ~~">"~~greater-than sign character ('>', U+003E),
   - ~~"#"~~number sign character ('#', U+0023),
   - ~~";"~~semicolon character (';', U+003B),

- "\"reverse solidus character ('\', U+005C), ~~or~~

- """quotation mark character ('"', U+0022).

2. The ~~dot~~ full stop character (~~'.', ASCII 46~~'.', U+002E). This character shall be used in the `AttributeValue` whose `AttributeType` is "DC". An example is "DC=~~lme.companyZ.se~~marketing.CompanyXYZ.com". This ~~dot~~ full stop character shall not be used in `AttributeType`.

3. The ~~star~~ asterisk character (~~'*', ASCII 42~~'*', U+002A) is reserved to denote wild~~-~~card. Wild~~-~~card character(s) can appear in `AttributeType` and `AttributeValue`.

End of Change in Clause 7.2

Change in Clause 7.3

7.3 EBNF of DN String Representation

The formal definitions provided within this subclause consolidate several rules and concepts (null distinguished name, DN prefix, local DN, domain component type, class names starting with upper case characters, attribute names starting with lower case characters, ~~class names~~ classes with or without an "Id" naming attribute, attribute type and attribute value allowed characters). The definition is more detailed to clarify these naming rules, and will not introduce compliancy issues for implementations compliant with ~~the previous~~ Rel-5 version of this specification.

The following is the EBNF for DN in string representation (Extended Backus-Naur Form ~~is popular in IETF specifications to define format syntax. See~~; see ~~IETF RFC 733 [10]~~ ISO/IEC 14977 [13] for more information):

```
~~DistinguishedName := RDNSequence~~
DistinguishedName               =   NullDN       (* Distinguished Names shall not exceed   *)
                                  | RegularDN ;  (* 400 octets as specified in section 7.4 *)

NullDN                          = ;  (* empty string; null DN is specified in subclause 7.1.1 *)

RegularDN                       =   DNPrefixPlusRDNSeparator  (* DN prefix and local DN *)
                                  , LocalDN ;                 (* are defined in annex C *)

DNPrefixPlusRDNSeparator        =   ( NullDN , NullRDNSeparator )
                                  | ( DNPrefixWithDomainComponent , RDNSeparator )
                                  | ( DNPrefixWithoutDomainComponent , RDNSeparator ) ;

NullRDNSeparator                = ;  (* empty string *)

~~RDNSequence := RDNSequence <spaced-separator> RDNSequence | RelativeDistinguishedName~~
DNPrefixWithDomainComponent     =   DomainComponentRDN
                                  , { RDNSeparator , DomainComponentRDN }
                                  , { RDNSeparator , RegularRDN } ;

DNPrefixWithoutDomainComponent =   RegularRDN
                                  , { RDNSeparator , RegularRDN } ;

LocalDN                         =   LocalRDN
                                  , { RDNSeparator , LocalRDN } ;

~~<spaced-separator> ::= <optional-space> <separator> <optional-space>~~
~~<separator> ::= ","~~
RDNSeparator                    =   [ RDNSeparatorWhiteSpace ]    (* use of optional white space  *)
                                  , CommaChar                     (* is recommended to be avoided *)
                                  , [ RDNSeparatorWhiteSpace ] ;

~~<optional-space> ::= ( <CR> ) *( " " )~~
RDNSeparatorWhiteSpace          =   [ CarriageReturnChar ]
                                  , { SpaceChar } ;

~~RelativeDistinguishedName := AttributeTypeAndValue~~
DomainComponentRDN              = DCAttributeTypeAndValue ;

RegularRDN                      = RegularAttributeTypeAndValue ;

LocalRDN                        = LocalDNAttributeTypeAndValue ;
```

```
AttributeTypeAndValue := AttributeType "=" AttributeValue
DCAttributeTypeAndValue          =   DCAttributeType
                                   , AttributeTypeAndValueSeparator
                                   , DCAttributeValue ;

RegularAttributeTypeAndValue     =   RegularAttributeType
                                   , AttributeTypeAndValueSeparator
                                   , RegularAttributeValue ;

LocalDNAttributeTypeAndValue     =   LocalDNAttributeType
                                   , AttributeTypeAndValueSeparator
                                   , RegularAttributeValue ;

AttributeTypeAndValueSeparator = EqualsSignChar ;

AttributeType := <one or more StringChar>
AttributeValue := <one or more StringChar>
DCAttributeType                  = "DC" ;  (* ISO/IEC 646 IRV U+0044/0043 Latin capital letters D&C *)

DCAttributeValue                 =   DCLabel                          (* this is specified *)
                                   , { FullStopChar , DCLabel } ;  (* in IETF RFC 1035  *)

DCLabel                          =   LetterChar                              (* this is specified *)
                                   , [   { LetterDigitHypenMinusChar }       (* in IETF RFC 1035  *)
                                       , LetterDigitChar                 ] ;

RegularAttributeType             =   LetterChar                         (* this is specified *)
                                   , { LetterDigitHypenMinusChar } ;  (* in IETF RFC 2253  *)

LocalDNAttributeType             =   NameOfClassWithIdAttribute        (* definition selected shall *)
                                   | NamesOfClassAndNamingAttribute ;  (* be in accordance with the *)
                                                                       (* rules defined in annex B  *)

NameOfClassWithIdAttribute       = ClassName ;  (* see rules defined in annex B *)

NamesOfClassAndNamingAttribute =   ClassName              (* see rules defined in annex B *)
                                   , FullStopChar
                                   , NamingAttributeName ;

ClassName                        =   CapitalLetterChar                (* see recommendation on       *)
                                   , { LocalDNAttributeTypeChar } ;  (* characters for class names  *)
                                                                     (* in 3GPP TS 32.622 annex A   *)

NamingAttributeName              =   SmallLetterChar
                                   , { LocalDNAttributeTypeChar } ;

RegularAttributeValue            =   ( AttributeValueChar – SpaceChar )          (* this is       *)
                                   , [   { AttributeValueChar }                  (* specified in  *)
                                       , ( AttributeValueChar – SpaceChar ) ] ;  (* IETF RFC 2253 *)

StringChar := any character except <special>
LocalDNAttributeTypeChar         = DNChar ;

AttributeValueChar               = DNChar | EscapedCharSequence ;

DNChar                           = DNCharUnrestricted - ReservedChar ;

DNCharUnrestricted               =     ? Character of ISO/IEC 646 IRV ?
                                   | ? Character of standard coded character set
                                       supporting and extending ISO/IEC 646 IRV  ? ;

EscapedCharSequence              =   ReverseSolidusChar          (* this is specified  *)
                                   , 2 * HexadecimalDigitChar ;  (* in subclause 7.1.3 *)

<special> ::= "," |"=" |<CR> |<LF> |"+" |"<" |">" |"#" |";" |"\" | """
ReservedChar                     =   Rfc2253ReservedChar | CarriageReturnChar | LineFeedChar ;

Rfc2253ReservedChar              =   CommaChar | EqualsSignChar | PlusSignChar | LessThanSignChar
                                   | GreaterThanSignChar | NumberSignChar | SemiColonChar
                                   | ReverseSolidusChar | QuotationMarkChar ;

LetterChar                       = CapitalLetterChar | SmallLetterChar ;

LetterDigitChar                  = LetterChar | DigitChar ;

LetterDigitHypenMinusChar        = LetterDigitChar | HypenMinusChar ;
```

```
HexadecimalDigitChar          = DigitChar | CapitalLetterAtoFChar | SmallLetterAtoFChar ;

LineFeedChar                  = ? ISO/IEC 646 IRV U+000A character line feed ? ;

CarriageReturnChar            = ? ISO/IEC 646 IRV U+000D character carriage return ? ;

SpaceChar                     = ' ' ;  (* ISO/IEC 646 IRV U+0020 character space *)

QuotationMarkChar             = '"' ;  (* ISO/IEC 646 IRV U+0022 character quotation mark *)

NumberSignChar                = '#' ;  (* ISO/IEC 646 IRV U+0023 character number sign *)

PlusSignChar                  = '+' ;  (* ISO/IEC 646 IRV U+002B character plus sign *)

CommaChar                     = ',' ;  (* ISO/IEC 646 IRV U+002C character comma *)

HypenMinusChar                = '-' ;  (* ISO/IEC 646 IRV U+002D character hyphen-minus *)

FullStopChar                  = '.' ;  (* ISO/IEC 646 IRV U+002E character full stop *)

DigitChar                     =   '0' | '1' | '2' | '3' | '4'    (* ISO/IEC 646 IRV U+0030-0039 *)
                                | '5' | '6' | '7' | '8' | '9' ;  (* digits 0 to 9               *)

SemiColonChar                 = ';' ;  (* ISO/IEC 646 IRV U+003B character semicolon *)

LessThanSignChar              = '<' ;  (* ISO/IEC 646 IRV U+003C character less-than sign *)

EqualsSignChar                = '=' ;  (* ISO/IEC 646 IRV U+003D character equals sign *)

GreaterThanSignChar           = '>' ;  (* ISO/IEC 646 IRV U+003E character greater-than sign *)

CapitalLetterAtoFChar         =   'A' | 'B' | 'C'    (* ISO/IEC 646 IRV U+0041-0046  *)
                                | 'D' | 'E' | 'F' ;  (* Latin capital letters A to F *)

CapitalLetterChar             =   CapitalLetterAtoFChar | 'G' | 'H'    (* ISO/IEC 646 IRV *)
                                | 'I' | 'J' | 'K' | 'L' | 'M' | 'N'    (* U+0041-005A     *)
                                | 'O' | 'P' | 'Q' | 'R' | 'S' | 'T'    (* Latin capital   *)
                                | 'U' | 'V' | 'W' | 'X' | 'Y' | 'Z' ;  (* letters A to Z  *)

ReverseSolidusChar            = '\' ;  (* ISO/IEC 646 IRV U+005C character reverse solidus *)

SmallLetterAtoFChar           =   'a' | 'b' | 'c'    (* ISO/IEC 646 IRV U+0061-0066 *)
                                | 'd' | 'e' | 'f' ;  (* Latin small letters a to f  *)

SmallLetterChar               =   SmallLetterAtoFChar | 'g' | 'h'      (* ISO/IEC 646 IRV *)
                                | 'i' | 'j' | 'k' | 'l' | 'm' | 'n'    (* U+0061-007A     *)
                                | 'o' | 'p' | 'q' | 'r' | 's' | 't'    (* Latin small     *)
                                | 'u' | 'v' | 'w' | 'x' | 'y' | 'z' ;  (* letters a to z  *)
```

End of Change in Clause 7.3

Change in Clause 8

8 Examples of DN in string representation

[…]

EXAMPLE 1: "`DC=com,DC=CompanyXYZ,DC=~~M~~marketing,IRPAgent=ATMPVCBilling,Log=19990101131000,AccountingRecord=100098`". In this example, the name space aligns with DNS. The `AttributeType` of the top three RDN are "`DC`". Concatenation of the corresponding `AttributeValues` produces the DNS registered name, i.e. "`marketing.~~c~~CompanyXYZ.com`". The top RDN is the Global Root because DNS defines "`DC=com`" as the root of its name space. That top RDN is the Local Root as well.

EXAMPLE 2: "`DC=marketing.CompanyXYZ.com,IRPAgent=ATMPVCBilling,Log=19990101131000,AccountingRecord=100098`". In this example, the name space aligns with DNS as well. Instead of using three RDNs to represent the DNS registered name, this example chooses to use one RDN. The top RDN is the Global Root (and Local Root as well).

EXAMPLE 3: "IRPNetwork=ABCNetwork,Subnet=TN2,BSS=B5C0100". In this example, the name space designer chooses not to name its objects under the DNS nor X.500 scheme. The name space designer chooses to use "IRPNetwork=ABCNetwork" as the Local Root of its name space (by looking at the DN string, it is not possible to say if the Local Root is the Global Root). DNs in this name space will start with that string as their Local Root. One string ("IRPNetwork") for AttributeType (of the AttributeTypeAndValue of the RDN) starts with "IRP". This indicates that this string is mapped from the MO class names specified in NRM of [9] or other domain specific NRMs (see the Introduction clause). Other strings do not start with "IRP", indicating that those strings are not mapped from MO class names specified in NRM of [9] or other domain specific NRMs. They are probably mapped from MO classes that are specific for a particular product and thus specified in a product-specific NRM ~~(MIM)~~.

EXAMPLE 4: The following example illustrates the use of ~~","~~ the comma character as separator for RDNs. It also illustrates the use of space and ~~period~~ full stop characters as part of the legal character syntax for RDNs~~.~~:

"CN=John T. Mills, O= Cyber System Consulting~~, L= Göteborg, C=SE~~"

End of Change in Clause 8

Change in Clause 9.1

9.1 DN prefix usage

[…]

5. The NE is sold to a customer. The customer administrator knows his Enterprise name space, the topology of his network and where the NE will be deployed. Based on the information, he configures the DN prefix of the NE.

EXAMPLE 2: The customer administrator can configure it to:

~~"DC=CompanyXYZ.com,Net=DS3BackBone,Station=TMR".~~

"DC=marketing.CompanyXYZ.com,Net=DS3BackBone,Station=TMR"

~~The Global Root in this case is "DC=CompanyXYZ.com".~~

The Global Root in this case is "DC=marketing.CompanyXYZ.com".

6. At run time, whenever NE is reporting an alarm on Port=3 via the IRP, the following string will be in the MOI field of the alarm record~~.~~:

~~"DC=CompanyXYZ.com,Net=DS3BackBone,Station=TMR,Node=1,Port=3".~~

"DC=marketing.CompanyXYZ.com,Net=DS3BackBone,Station=TMR,Node=1,Port=3"

End of Change in Clause 9.1

Change in Clause Annex A

Annex A (normative): Mapping of RDN AttributeType to Strings

[…]

There is one AttributeType that is not defined in NRM of 3GPP TS 32.622 [9] or other domain specific NRMs as listed in the Introduction clause. This special AttributeType is used to denote the domain component of the DNS.

The following partial DN string representations are examples to illustrate the valid use of "~~DC~~DC" strings for the three DNS domain components of "~~lme.companyZ.se~~marketing.CompanyXYZ.com"~~.~~:

- ~~DC=se.companyZ.lme,..~~
- "DC=com.CompanyXYZ.marketing,..."
- ~~DC=se,DC=companyZ,DC=lme,..~~
- "DC=com,DC=CompanyXYZ,DC=marketing,..."
- ~~DC=se,DC=companyZ.lme,..~~
- "DC=com,DC=CompanyXYZ.marketing,..."
- ~~DC=se.companyZ,DC=lme,..~~
- "DC=com.CompanyXYZ,DC=marketing,..."

[…]

End of Change in Clause Annex A

Change in Clause Annex B

Annex B (normative): Rule for MO Designers regarding `AttributeType` interpretation

[…]

First interpretation

[…]

If this (first) interpretation is used for constructing the DN string, then the DN will be "~~..~~"..., `id=123`". MO class name cannot be derived from the DN string. The value of the `AttributeValue` contains the value of the naming attribute.

Second interpretation

[…]

If this interpretation is used for constructing the DN string, then the DN will be "~~..~~"..., `Bsc=123`". The name of the naming attribute cannot be derived from the DN string. The value of the `AttributeValue` contains the value of the naming attribute.

Rule

Given the two interpretations, a DN reader cannot know how to interpret the `AttributeType`, i.e. if the `AttributeType` identifies class or naming attribute. To avoid ambiguity, the following rules shall apply:

- If `AttributeType` of a naming attribute is not a concatenation of MO class name and "Id" (ignoring case for both), then the DN shall use "~~...,Yyy.zzz=123,..~~..., `Yyy.zzz=123,...`" where "~~Yyy~~Yyy" is the MO class name and "~~zzz~~`zzz`" is the naming attribute (preserving case for both).

EXAMPLE 1: If "`Bsc`" is the MO class name and if its naming attribute is "`serialNumber`", then the DN shall be "~~,,Bsc.serialNumber=123,..~~..., `Bsc.serialNumber=123,...`".

- If `AttributeType` of a naming attribute is a concatenation of MO class name and "Id" (ignoring case for both), then the DN shall use "~~..,Xxx=123,..~~..., `Xxx=123,...`" where "~~Xxx~~Xxx" is the MO class name (preserving case).

EXAMPLE 2: If "Bsc" is the MO class name and if its naming attribute is "bscId", then the DN shall be "~~,,Bsc=123,..~~...,Bsc=123,...".

End of Change in Clause Annex B

Change in Clause Annex C

Annex C (informative): DN Prefix and Local Distinguished Name (LDN)

[…]

Suppose the Enterprise name space is organized hierarchically and is partitioned into 4 sub-sets as shown in figure C.1.

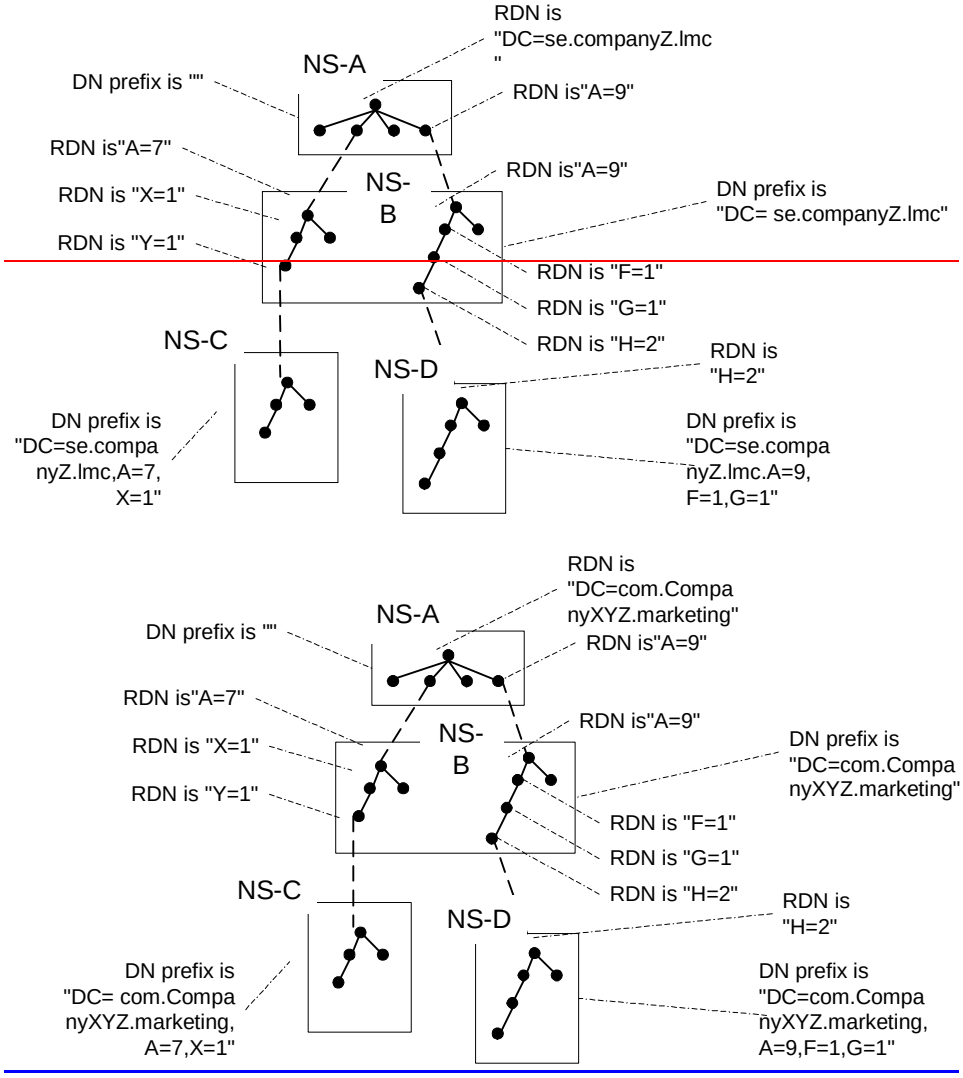

Figure C.1: Name space partitions

NS (name space)-A contains 5 objects. DN prefix is NULL. The Global Root and Local Root of NS-A is "DC=~~se.companyZ.lmc~~com.CompanyXYZ.marketing" (see the Note below). DN of top object is "DC=~~se.companyZ.lmc~~com.CompanyXYZ.marketing". RDNs of the other four objects are, from bottom left to bottom right, "A=1", "A=7", "~~-~~A=3" and "A=9". DNs of the same four objects are "DC=~~se.companyZ.lmc~~com.CompanyXYZ.marketing,A=1", "DC=~~se.companyZ.lmc~~com.CompanyXYZ.marketing,A=7",

"DC=~~se.companyZ.lmc~~com.CompanyXYZ.marketing,A=3" and "DC=~~se.companyZ.lmc~~com.CompanyXYZ.marketing,A=9". The second and fourth objects are reference objects to MOs in NS-B.

NS-B contains two branches. They have the same DN prefix that is "DC=~~se.companyZ.lmc~~com.CompanyXYZ.marketing". The Global Root is "DC=~~se.companyZ.lmc~~com.CompanyXYZ.marketing".

The Local Root and RDN of top object of the right branch is "A=9". Its DN is "DC=~~se.companyZ.lmc~~com.CompanyXYZ.marketing,A=9". RDNs of other objects are shown in figure C.1. DN of the bottom object is "DC=~~se.companyZ.lmc~~com.CompanyXYZ.marketing,A=9,F=1,G=1,H=2". This object refers to object of another name space called NS-D.

The Local Root and RDN of the top object of the left branch is "A=7". Its DN is "DC=~~se.companyZ.lmc~~com.CompanyXYZ.marketing,A=7". RDNs of other objects are shown in figure C.1. DN of the bottom object is "DC=~~se.companyZ.lmc~~com.CompanyXYZ.marketing,A=7,X=1,Y=1". This object refers to object of another name space called NS-C.

NS-C contains a branch of 4 objects. Its DN prefix is "DC=~~se.companyZ.lmc~~com.CompanyXYZ.marketing,A=7,X=1". The Local Root an RDN of the top object is "Y=1".

NS-D contains a branch of 5 objects. Its DN prefix is "DC=~~se.companyZ.lmc~~com.CompanyXYZ.marketing,A=9,F=1,G=1". The Local Root and RDN of the top object is "H=2".

In figure C.1, the bottom object of NS-B right branch has the following names:

- ~~DN is "DC=se.companyZ.lmc,A=9,F=1,G=1,H=2".~~

- DN is "DC=com.CompanyXYZ.marketing,A=9,F=1,G=1,H=2".

- ~~LDN is "A=9,F=1,G=1,H=2".~~

- LDN is "A=9,F=1,G=1,H=2".

- ~~RDN is "H=2".~~

- RDN is "H=2".

[…]

NOTE: Use of "~~DC~~DC" in "~~DC=se.companyZ.lmc~~DC=com.CompanyXYZ.marketing" is an attempt to align the RDN with DNS name associated with the named organisation. The "~~DC~~DC" stands for Domain Component and is an attribute name defined by IETF for use in directory work. Annex A specifies other valid ways to align RDN with DNS as well. Equally valid, the example can choose to align the RDN with the X.500 convention. In such case, the subject string can be "~~C=se,O=CompanyZ,L=lmc~~O=com,O=CompanyXYZ,OU=marketing" where ~~C,~~ O and ~~L~~ OU are X.500 standard attributes denoting ~~country,~~ organisation and ~~location~~ organization unit respectively. The alignment choice belongs to the name space designer of each operator. The choice will be reflected in the value of the DN prefix, probably a product configuration parameter. See clause 7 for more information.

End of Change in Clause Annex C

Change in Clause New Annex D

Annex D (informative): Interpreting EBNF [13]

This annex provides a very simplified summary of EBNF, and does not modify in any way the reference text in ISO/IEC 14977: "Information technology – Syntactic metalanguage – Extended BNF" [13].

ISO/IEC 14977 [13] specification also provides far greater coverage supported by numerous examples which are not included within this annex.

The EBNF metalanguage is useful for defining rigorous syntax notations and is a notation for defining syntax rules.

The language uses sequences of formal definitions.

Definitions may have several layers of definition. The definitions which are refined are termed as "non terminal symbols".

A term which cannot be defined at a lower level of detail is known as a "terminal symbol". I.e. the "terminal symbols" cannot be further decomposed.

The language permits sentences to be constructed.

The sentences consist of a non terminal, or a terminal symbol, followed by an equality symbol, followed by a formal definition of the symbol.

Each sentence terminates with the semicolon `';'` terminal symbol.

Ideally the definitions are read from the top across to the right hand side of the page and downwards.

A definition commences with an identifier (of the thing being defined) followed by an equality sign.

The thing is defined by the symbols and identifiers to the right hand side of the equality symbol, up to the next semicolon `';'`.

There is a natural breaking down of definitions, by other definitions until a point is reached that a terminal symbol is reached – which cannot be further defined (e.g. the leaves of definition hierarchy).

There are terminal symbols which permit optional choice, sequence, exclusion, comments to be included in the sentence.

The set of terminal symbols as defined in table 1 of ISO/IEC 14977 [13] are below.

The normal character representing each operator of Extended BNF and its implied precedence is (highest precedence at the top):

- `'*'` repetition-symbol
- `'-'` except-symbol
- `','` concatenate-symbol
- `'|'` definition-separator-symbol
- `'='` defining-symbol
- `';'` terminator-symbol

The normal precedence is over-ridden by the following pairs of terminal symbols:

`"'"`	first-quote-symbol	first-quote-symbol	`"'"`
`'"'`	second-quote-symbol	second-quote-symbol	`'"'`

"(*" start-comment-symbol end-comment-symbol "*)"

'(' start-group-symbol end-group-symbol ')'

'[' start-option-symbol end-option-symbol ']'

'{' start-repeat-symbol end-repeat-symbol '}'

'?' special-sequence-symbol special-sequence-symbol '?'

Examples:

```
letter    =   "A" | "B" | "C" | "D" | "E" | "F" | "G" | "H" | "I" | "J" | "K" | "L"
            | "M" | "N" | "O" | "P" | "Q" | "R" | "S" | "U" | "V" | "W" | "X" | "Y" | "Z" ;

vowel     = "A" | "E" | "I" | "O" | "U" ;  (* a subset of letters *)

consonant = letter – vowel ;  (* the set of letters except vowels *)
```

End of Change in Clause New Annex D
End of Document

Annex ~~D~~ E (informative): Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
Jun 2001	S_12	SP-010283	--	--	Approved at TSG SA #12 and placed under Change Control	2.0.0	4.0.0
Dec 2001	S_14	SP-010641	001	--	Alignment of Figure C.1 with text in annex C	4.0.0	4.1.0
Sep 2001	S_17	SP-020481	002	--	Upgrade to Rel-5 (Remove information in the Introduction that is only relevant to Rel-4)	4.1.0	5.0.0
Dec 2002	--	--	--	--	Cosmetics	5.0.0	5.0.1

CR-Form-v7

CHANGE REQUEST

⌘ 32.302 CR 006 ⌘rev - ⌘ Current version: 5.1.0 ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: UICC apps⌘ ☐ ME ☐ Radio Access Network [X] Core Network [X]

Title:	⌘	Add missing rules on how to construct the string NotificationCategory
Source:	⌘	**SA5** (olaf.pollakowski@siemens.com)
Work item code:	⌘	OAM-NIM
Date:	⌘	01/10/2004
Category:	⌘	**F**
Release:	⌘	Rel-5

Use one of the following categories:
F *(correction)*
A *(corresponds to a correction in an earlier release)*
B *(addition of feature),*
C *(functional modification of feature)*
D *(editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:

2	*(GSM Phase 2)*
R96	*(Release 1996)*
R97	*(Release 1997)*
R98	*(Release 1998)*
R99	*(Release 1999)*
Rel-4	*(Release 4)*
Rel-5	*(Release 5)*
Rel-6	*(Release 6)*

Reason for change:	⌘	It is not defined how to derive the notification category string.
Summary of change:	⌘	It is specified that the notification category string shall be derived in the same way as the IRP ducument version number string.
Consequences if not approved:	⌘	It would be not clear how the derive the notification category string resulting in different implementations which in turn hampers interoperability.

Clauses affected:	⌘	2, 3.1

Other specs affected:	Y	N			
⌘		X	Other core specifications	⌘	
		X	Test specifications		
	X		O&M Specifications		Rel-6 32.302

Other comments:	⌘	Rel-6 Mirror CR in S5-046983.

Change in Clause 2

2 References

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

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] 3GPP TS 32.111-2: "Telecommunication management; Fault Management; Part 2: Alarm Integration Reference Point (IRP): Information service".

[2] ITU-T Recommendation X.734 (1992): "Information technology - Open Systems Interconnection - Systems management: Event report management function".

[3] 3GPP TS 32.300: "Telecommunication management; Configuration Management (CM); Name convention for Managed Objects".

[4] OMG: "OMG Notification Service". http://www.omg.org/technology/documents/

[5] 3GPP TS 32.101: "Telecommunication management; Principles and high level requirements".

[6] 3GPP TS 32.102: "Telecommunication management; Architecture".

[7] 3GPP TS 32.301: "Telecommunication management; Configuration Management (CM); Notification Integration Reference Point (IRP): Requirements".

[8] 3GPP TS 32.602: "Telecommunication management; Configuration Management (CM); Basic CM Integration Reference Point (IRP) information service".

[9] 3GPP TS 32.622: "Telecommunication management; Configuration Management (CM); Generic network resources Integration Reference Point (IRP): Network Resource Model (NRM)".

[10] 3GPP TS 32.312: "Telecommunication management; Generic Integration Reference Point (IRP) management;÷ Information service".

[11] 3GPP TS 32.311: "Telecommunication management; Generic Integration Reference Point (IRP) management: Requirements".

End of Change in Clause 2

Change in Clause 3.1

3.1 Definitions

For the purposes of the present document, the terms and definitions given in 3GPP TS 32.101 [5], 3GPP TS 32.102 [6] and 3GPP TS 32.301 [7] and the following apply:

IRPAgent: See 3GPP TS 32.102 [6].

IRPManager: See 3GPP TS 32.102 [6].

event: it is an occurrence that is of significance to network operators, the NEs under surveillance and network management applications

Events can indicate many types of network management information, such as network alarms, network configuration change information and network performance data.

notification: it refers to the transport of information regarding events from event producer to consumer (receiver)
In this IRP, notification is used to carry information about network events from IRPAgent to IRPManager. Producer sends notifications to consumers as soon as new events occur. Consumer does not need to check ("pull") for events.

IRP: See 3GPP TS 32.102 [6].

notification category: The semantics and syntax of the notification category string are identical to that defined for "IRP document version number string", see 3GPP TS 32.311 [11] subclause 3.1.~~it refers to the set of notifications of one 3GPP IRP Information Service specification~~
~~A Notification Category is identified by the name of the IRP specification and the IRP specification version number.~~

qualifiers: the meaning of qualifiers for operations, parameters and information attributes (whether they are Mandatory(M)/ Conditional(C)/ Optional(O)) defined in the present (Information Service) document is provided in 3GPP TS 32.102 [6]
Moreover, qualifiers of information attributes, when those information attributes are re-used in other IRP ISs, obey to the following rule: Mandatory and Conditional qualifiers of information attributes shall always be the same in other IRPs ISs, Optional qualifiers of information attributes may be set to either Optional or Mandatory in the other IRP ISs.

End of Change in Clause 3.1

CR-Form-v7

CHANGE REQUEST

⌘ 32.302 CR 007 ⌘rev - ⌘ Current version: 6.0.0 ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: UICC apps⌘ ☐ ME ☐ Radio Access Network [X] Core Network [X]

Title:	⌘	Add missing rules on how to construct the string NotificationCategory
Source:	⌘	**SA5** (olaf.pollakowski@siemens.com)
Work item code:	⌘	OAM-NIM
Date:	⌘	01/10/2004
Category:	⌘	**A**
Release:	⌘	Rel-6

Use one of the following categories:
F *(correction)*
A *(corresponds to a correction in an earlier release)*
B *(addition of feature),*
C *(functional modification of feature)*
D *(editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:

2	*(GSM Phase 2)*
R96	*(Release 1996)*
R97	*(Release 1997)*
R98	*(Release 1998)*
R99	*(Release 1999)*
Rel-4	*(Release 4)*
Rel-5	*(Release 5)*
Rel-6	*(Release 6)*

Reason for change:	⌘	It is not defined how to derive the notification category string.
Summary of change:	⌘	It is specified that the notification category string shall be derived in the same way as the IRP ducument version number string.
Consequences if not approved:	⌘	It would be not clear how the derive the notification category string resulting in different implementations which in turn hampers interoperability.

Clauses affected:	⌘	2, 3.1

Other specs affected: ⌘

Y	N	
	X	Other core specifications
	X	Test specifications
	X	O&M Specifications

⌘

Other comments: ⌘ Rel-6 Mirror CR to S5-046982.

Change in Clause 2

2 References

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

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] 3GPP TS 32.111-2: "Telecommunication management; Fault Management; Part 2: Alarm Integration Reference Point (IRP): Information service".

[2] ITU-T Recommendation X.734 (1992): "Information technology - Open Systems Interconnection - Systems management: Event report management function".

[3] 3GPP TS 32.300: "Telecommunication management; Configuration Management (CM); Name convention for Managed Objects".

[4] OMG: "OMG Notification Service". http://www.omg.org/technology/documents/

[5] 3GPP TS 32.101: "Telecommunication management; Principles and high level requirements".

[6] 3GPP TS 32.102: "Telecommunication management; Architecture".

[7] 3GPP TS 32.301: "Telecommunication management; Configuration Management (CM); Notification Integration Reference Point (IRP): Requirements".

[8] 3GPP TS 32.602: "Telecommunication management; Configuration Management (CM); Basic CM Integration Reference Point (IRP) information service".

[9] 3GPP TS 32.622: "Telecommunication management; Configuration Management (CM); Generic network resources Integration Reference Point (IRP): Network Resource Model (NRM)".

[10] 3GPP TS 32.312: "Telecommunication management; Generic Integration Reference Point (IRP) management: Information service".

[11] 3GPP TS 32.311: "Telecommunication management; Generic Integration Reference Point (IRP) management: Requirements".

End of Change in Clause 2

Change in Clause 3.1

3.1 Definitions

For the purposes of the present document, the terms and definitions given in 3GPP TS 32.101 [5], 3GPP TS 32.102 [6] and 3GPP TS 32.301 [7] and the following apply:

IRPAgent: See 3GPP TS 32.102 [6].

IRPManager: See 3GPP TS 32.102 [6].

event: it is an occurrence that is of significance to network operators, the NEs under surveillance and network management applications
Events can indicate many types of network management information, such as network alarms, network configuration change information and network performance data.

notification: it refers to the transport of information regarding events from event producer to consumer (receiver)
In this IRP, notification is used to carry information about network events from IRPAgent to IRPManager. Producer sends notifications to consumers as soon as new events occur. Consumer does not need to check ("pull") for events.

IRP: See 3GPP TS 32.102 [6].

notification category: ~~it refers to the set of notifications of one 3GPP IRP Information Service specification~~
~~A Notification Category is identified by the name of the IRP specification and the IRP specification version number~~The semantics and syntax of the notification category string are identical to that defined for "IRP document version number string", see 3GPP TS 32.311 [11] subclause 3.1.

qualifiers: the meaning of qualifiers for operations, parameters and information attributes (whether they are Mandatory(M)/ Conditional(C)/ Optional(O)) defined in the present (Information Service) document is provided in 3GPP TS 32.102 [6]
Moreover, qualifiers of information attributes, when those information attributes are re-used in other IRP ISs, obey to the following rule: Mandatory and Conditional qualifiers of information attributes shall always be the same in other IRPs ISs, Optional qualifiers of information attributes may be set to either Optional or Mandatory in the other IRP ISs.

Change in Clause 3.1

3GPP TSG-SA5 (Telecom Management) **S5-047039**
Meeting #40, Sanya, CHINA, 15 - 19 November 2004

CR-Form-v7

CHANGE REQUEST

⌘ **32.303** **CR** **013** ⌘ **rev** **-** ⌘ Current version: **6.1.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: UICC apps⌘ ☐ ME ☐ Radio Access Network **X** Core Network **X**

Title:	⌘	Remove filter requirement in IDL comments in the Notification IRP CORBA SS
Source:	⌘	**SA5** (edwin.tse@ericsson.com)
Work item code:	⌘	OAM-NIM
Date:	⌘	19/11/2004
Category:	⌘	**F**
Release:	⌘	Rel-6

Use one of the following categories:
F *(correction)*
A *(corresponds to a correction in an earlier release)*
B *(addition of feature),*
C *(functional modification of feature)*
D *(editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:

2	*(GSM Phase 2)*
R96	*(Release 1996)*
R97	*(Release 1997)*
R98	*(Release 1998)*
R99	*(Release 1999)*
Rel-4	*(Release 4)*
Rel-5	*(Release 5)*
Rel-6	*(Release 6)*

Reason for change:	⌘	Incorrect filter requirement wordings in IDL comments
Summary of change:	⌘	Remove filter requirement wordings in IDL comments.
Consequences if not approved:	⌘	The filtering requirement text in the IDL comments is not identical to that specified in Yyy IRP CORBA SS such as EP IRP.
Clauses affected:	⌘	A.5

	Y	N		
Other specs affected: ⌘		X	Other core specifications	⌘
		X	Test specifications	
		X	O&M Specifications	

Other comments: ⌘

Change in A.5

A.5 IDL specification (file name "NotificationIRPNotifications.idl")

```
//File: NotificationIRPNotifications.idl

#ifndef _NOTIFICATIONIRPNOTIFICATIONS_IDL_
#define _NOTIFICATIONIRPNOTIFICATIONS_IDL_

#include <NotificationIRPConstDefs.idl>

// This statement must appear after all include statements
#pragma prefix "3gppsa5.org"

module NotificationIRPNotifications
{

    interface Notify
    {
         /**
          Notification IRP IS defines 6 attributes for the notification header.
          They are: objectClass, objectInstance, notificationId, eventTime,
          systemDN and notificationType.

          The first 2 attributes are mapped into 1 name-value pair.
          of the filterable body of the CORBA structured event.  The name of
          the mapped IDL construct is MANAGED_OBJECT_INSTANCE.  The const
          string of this mapped IDL construct is defined here.

          The notificationId, eventTime and systemDN are respectively mapped
          into 3 name-value pairs. of the filterable body.  The const string(s) of
          these 3 mapped IDL
          constructs are defined here.

          The notificationType is not mapped into any name-value pair of the
          filterable body
          but is mapped into the type_name position-dependent
          field of the CORBA structured-event.  There is no need for a const string
          definition for it.
          */


        ...


        ...
```

Change in A.5