

IPWorks ENUM LDAP CUDB Interface

Interwork Description

Copyright

© Ericsson AB 2015, 2016. All rights reserved. No part of this document may be reproduced in any form without the written permission of the copyright owner.

Disclaimer

The contents of this document are subject to revision without notice due to continued progress in methodology, design and manufacturing. Ericsson shall have no liability for any error or damage of any kind resulting from the use of this document.

Trademark List

All trademarks mentioned herein are the property of their respective owners. These are shown in the document Trademark Information.



Contents

1	Introduction	1
1.1	Prerequisites	1
1.2	Related Information	1
2	Interface Overview	3
2.1	Interface Role	3
2.2	Services	3
2.3	Encapsulation and Addressing	3
3	Procedures	5
3.1	Creating and Deleting Entries	5
3.2	Modifying ENUM Data Modification	10
3.3	Searching ENUM Subscriber Data	12
4	Information Model	15
4.1	Attributes	15
4.2	Object Class	21
4.3	ENUM FE DIT Entries	23
5	Data Model	29
5.1	Data Model Overview	29
5.2	ENUM FE Directory Information Tree	29
6	Error Handling	31
	Reference List	33





1 Introduction

This document describes the interface between the IPWorks ENUM FE and the Centralized User Database(CUDB).

Scope

This document covers the following topics:

- Describes the LDAP user Database for accessing the CUDB.
- Provides the Ericsson reference DIT for accessing ENUM data.
- Specifies the LDAP operations required to be supported by the CUDB.

Target Groups

This document is intended for personnel who need to understand the logical entity, including interfaces and protocols, of IPWorks.

1.1 Prerequisites

Not Applicable.

1.2 Related Information

Definition and explanation of acronyms and terminology, trademark information, and typographic conventions can be found in the following documents:

- *Glossary of Terms and Acronyms*
- *Trademark Information*
- *Typographic Conventions*

For the standards related to this interface, see Standards.





2 Interface Overview

This section describes the interface LDAP exposed on IPWorks ENUM FE.

The CUDB is a network entity in a layered architecture domain that serves as the central storage point for ENUM data and other applications (AAA, HSS, and so on). The CUDB is built as an LDAP directory server, containing the necessary entries and attributes according to the defined schema for the different applications. The LDAP server accesses the ENUM data, which is then restructured according to LDAP specifications. That is, the data is displayed in a tree structure format from an external LDAP client. The tree structure is called Directory Information Tree (DIT), where each entry is identified by a Relative Distinguished Name (RDN), referring to the path to the tree root, that is, the Distinguished Name (DN).

2.1 Interface Role

This section describes the role of the IPWorks ENUM FE in the LDAP interface between ENUM FE and CUDB.

The LDAP interface uses the LDAP Protocol to access the CUDB and provides the reference Ericsson DIT for accessing the ENUM data. The interface also specifies the LDAP operations required to be supported by the CUDB.

2.2 Services

This section describes the services offered by the LDAP interface as shown in Table 1.

Table 1 Services

Service	Description
Search/add/delete/modify	Specifies the LDAP operations required to be supported by the CUDB.

2.3 Encapsulation and Addressing

This section describes the lower-level protocols on the LDAP interface:

- TCP
- LDAP





3 Procedures

This section describes the procedures used within the LDAP interface.

The operations are used by the ENUM FE and Provisioning Gateway (PG) to communicate with the CUDB to handle ENUM data.

The following LDAP operations are used:

- Add
- Delete
- Modify
- Search

These LDAP operations are described in Reference [7] and Reference [10].

The PG is the client responsible for provisioning the CUDB Server node. It adds, modifies, deletes entries in the CUDB Server, and performs any search operation the PG needs. Subscriber common data operations from the PG are performed using the branch under `servCommonData`.

The PG can perform following operations:

- Add new entries
- Delete entries
- Modify entries
- Search for an entry

The ENUM FE is the client responsible for traffic operations. It performs any search operation the ENUM FE needs for traffic. All operations from the ENUM FE are performed using the branch under `enum`.

3.1 Creating and Deleting Entries

All creation and deletion operations are triggered by PG to CUDB.

3.1.1 Creating Service Common Data Enum Entry

The Service Common Data `enum` entry is an entry of the `EnumDnSched` and `EnumDnRange` structural Object Class.

Create the entry by adding an LDAP according to Table 2.

*Table 2 Creating Service Common Data enum Entry*

AddRequest	
Entry	dn: serv=enum, ou=servCommonData, dc=<CUDB root entry>
Attributes	objectclass: top
	objectclass: CUDBService
	serv = "enum"

3.1.2 Creating Service Common Data EnumDnSched Entry

Create the entry by adding an LDAP according to Table 3:

Table 3 Creating Service Common Data EnumDnSched Entry

AddRequest	
Entry	dn: ou= EnumDnSched, serv=enum, ou= servCommonData, dc=<CUDB root entry>
Attributes	objectclass: top
	objectclass: organizationalUnit
	ou = " EnumDnSched"

3.1.3 Creating Service Common Data EnumDnRange Entry

Create the entry by adding an LDAP according to Table 4:

Table 4 Creating Service Common Data EnumDnRange Entry

AddRequest	
Entry	dn: ou= EnumDnRange, serv=enum, ou= servCommonData, dc=<CUDB root entry>
Attributes	objectclass: top
	objectclass: organizationalUnit
	ou = " EnumDnRange"

3.1.4 Creating FQDN Entry

The FQDN entry is an entry of the FQDN structural Object Class.

Create the entry by adding an LDAP according to Table 5 and Table 6:



Table 5 Creating EnumDnSched FQDN Entry

AddRequest	
Entry	dn: fqdn = <fqdn>, ou=EnumDnSched, serv=enum, ou = servCommonData, dc=<CUDB root entry>
Attributes	objectclass: top
	Objectclass: EnumFQDN
	fqdn = <fqdn>

Note: FQDN is up to 48M in EnumDnSched.

Table 6 Creating EnumDnRange FQDN Entry

AddRequest	
Entry	dn: fqdn = <fqdn>, ou= EnumDnRange, serv=enum, ou = servCommonData, dc=<CUDB root entry>
Attributes	objectclass: top
	Objectclass: EnumFQDN
	fqdn = <fqdn>

Note: FQDN is up to 18M in EnumDnRange.

3.1.5

Creating EnumView Entry

Create the entry by adding an LDAP according to Table 7 and Table 8:

Note: Viewid is from 0 to 19 under one FQDN.

Table 7 Creating EnumView Entry under EnumDnSched

AddRequest	
Entry	dn: viewid=<viewid>, fqdn = <fqdn>, ou= EnumDnSched, serv=enum, ou=servCommonData, dc=<CUDB root entry>
Attributes	objectclass: top
	objectclass: EnumView
	viewid=<viewid>

Table 8 Creating EnumView Entry under EnumDnRange

AddRequest	
Entry	dn: viewid=<viewid>, fqdn = <fqdn>, ou= EnumDnRange, serv=enum, ou=servCommonData, dc=<CUDB root entry>



Attributes	objectclass: top
	objectclass: EnumView
	viewid=<viewid>

3.1.6 Creating NAPTRRecord Entry

The `NAPTRRecord` entry is an entry of the `NAPTRRecord` structural Object Class.

Record is up to 5 in one view. The value of `<recordid>` is 1 to 5. PG must check every domain in `NAPTRRecord` in a record. Make every record unique under the specific FQDN and view.

Create the entry by adding an LDAP according to Table 9 and Table 10:

Table 9 Creating NAPTRRecord Entry

AddRequest	
Entry	dn: recordid =<recordid>, viewid=<viewid>, fqdn = <fqdn>, ou=EnumDnSched, serv=enum, ou = servCommonData, dc=<CUDB root entry>
Attributes	objectclass: top
	Objectclass: NAPTRRecord
	recordid = < recordid >
	NaptrOrder=< NaptrOrder>
	NaptrPreference=< NaptrPreference>
	NaptrService=< NaptrService>
	NaptrFlags=< NaptrFlags>
	NaptrTxt=< NaptrTxt>

Table 10 Creating NAPTRRecord Entry

AddRequest	
Entry	dn: recordid =<recordid>, viewid=<viewid>, fqdn = <fqdn>, ou=EnumDnRange, serv=enum, ou = servCommonData, dc=<CUDB root entry>



Attributes	objectclass: top
	Objectclass: NAPTRRecord
	recordid = < recordid >
	NaptrOrder=< NaptrOrder>
	NaptrPreference=< NaptrPreference>
	NaptrService=< NaptrService>
	NaptrFlags=< NaptrFlags>
	NaptrTxt=< NaptrTxt>

3.1.7 Deleting NAPTRRecord Entry

Delete NAPTRRecord in EnumDnSched entry according to Table 11:

Table 11 Deleting NAPTRRecord in EnumDnSched Entry

DelRequest	dn: recordid =<recordid>, viewid = <viewid>, fqdn = <fqdn>, ou=EnumDnRange, serv=enum, ou = servCommonData, dc=<CUDB root entry>
DelRequest	dn: recordid =<recordid>, viewid = <viewid>, fqdn = <fqdn>, ou=EnumDnSched, serv=enum, ou = servCommonData, dc=<CUDB root entry>

3.1.8 Deleting EnumView Entry

Delete EnumView in EnumDnSched entry according to Table 12:

Table 12 Deleting EnumView in EnumDnSched Entry

DelRequest	dn: viewid = <viewid>, fqdn = <fqdn>, ou=EnumDnRange, serv=enum, ou = servCommonData, dc=<CUDB root entry>
DelRequest	dn: viewid = <viewid>, fqdn = <fqdn>, ou=EnumDnSched, serv=enum, ou = servCommonData, dc=<CUDB root entry>

3.1.9 Deleting FQDN Entry

Delete FQDN in EnumDnRange entry according to Table 13:

*Table 13 Deleting FQDN in EnumDnRange Entry*

DelRequest	dn: fqdn = <fqdn>, ou= EnumDnSched, serv=enum, ou = servCommonData, dc=<CUDB root entry>
DelRequest	dn: fqdn = <fqdn>, ou= EnumDnRange, serv=enum, ou = servCommonData, dc=<CUDB root entry>

3.1.10 Deleting Service Common Data EnumDnSched

Delete the service common data `EnumDnSched` according to Table 14:

Table 14 Deleting EnumDnSched Entry

DelRequest	dn: ou= EnumDnSched, serv=enum, ou = servCommonData, dc=<CUDB root entry>
------------	---

3.1.11 Deleting Service Common Data EnumDnRange Entry

Delete the service common data `EnumDnRange` according to Table 15:

Table 15 Deleting EnumDnRange Entry

DelRequest	dn: ou= EnumDnRange, serv=enum, ou = servCommonData, dc=<CUDB root entry>
------------	---

3.1.12 Deleting Service Common Data ENUM Entry

Delete the service common data `EnumView` according to Table 16:

Table 16 Deleting EnumView Entry

DelRequest	dn: serv=enum, ou = servCommonData, dc=<CUDB root entry>
------------	--

3.2 Modifying ENUM Data Modification

This section includes examples of ENUM data modification.

3.2.1 EnumDnSched Modification (Replace)

The `EnumDnSched` subscriber data is an entry of the `EnumDnSched` Object Class.



The following table presents an example of possible `EnumDnSched` subscriber data modification, where the authentication method is changed for a subscriber. In this example, an LDAP replace operation is performed.

Table 17 EnumDnSched Modification (Replace)

ModifyRequest	dn: recordid =<recordid>, viewid = <viewid>, fqdn = <fqdn>, ou=EnumDnSched, serv=enum, ou = servCommonData, dc=<CUDB root entry>
Operation	replace
	attributetype: NaptrOrder
	NaptrOrder=< NaptrOrder>
	attributetype: NaptrPreference
	NaptrPreference=< NaptrPreference>
	attributetype: NaptrService
	NaptrService=< NaptrService>
	attributetype: NaptrFlags
	NaptrFlags=< NaptrFlags>
	attributetype: NaptrTxt
	NaptrTxt=< NaptrTxt>

3.2.2 EnumDnRange Modification (Replace)

The `EnumDnRange` subscriber data is an entry of the `EnumDnRange` Object Class.

The following table presents an example of possible `EnumDnRange` subscriber data modification, where the authentication method is changed for a subscriber. In this example, an LDAP replace operation is performed.

Table 18 EnumDnRange Modification (Replace)

ModifyRequest	dn: recordid =<recordid>, viewid = <viewid>, fqdn = <fqdn>, ou=EnumDnRange, serv=enum, ou = servCommonData, dc=<CUDB root entry>
Operation	replace



	attributetype: NaptrOrder
	NaptrOrder=< NaptrOrder>
	attributetype: NaptrPreference
	NaptrPreference=< NaptrPreference>
	attributetype: NaptrService
	NaptrService=< NaptrService>
	attributetype: NaptrFlags
	NaptrFlags=< NaptrFlags>
	attributetype: NaptrTxt
	NaptrTxt=< NaptrTxt>

3.3 Searching ENUM Subscriber Data

3.3.1 Searching Subscriber Common Data

The subscriber data is an entry of the ENUM Object Class. To search the subscriber data, an LDAP search operation must be performed.

It is possible to request the following:

- All data of a subscriber
- Individual subscriber data of a subscriber

3.3.1.1 Searching for All Data of a Subscriber for EnumDnRange

The `EnumDnRange` subscriber data is an entry of the `EnumDnRange` structural Object Classes. To search for `EnumDnRange` subscriber data, an LDAP search operation must be performed.

`EnumDnRange` could contain up to 10K FQDNs. One FQDN could contain up to 20 views. One view could contain up to 5 NAPTRRecord records.

Searching for all data of a subscriber can be performed from the `EnumDnRange` branch with the scope set to `wholeSubtree`. The following scenarios trigger this operation:

- ENUM FE gets all `EnumDnRange` data when it initializes.
- ENUM FE refreshes the cache data in 7 days.

Search for all data of a subscriber from the `EnumDnRange` entry according to Table 19:



Table 19 Searching for All Data of a Subscriber from EnumDnRange Entry

SearchRequest	
BaseObject	dn: ou= EnumDnRange, serv = enum, ou = servCommonData, dc = <CUDB root entry>
Scope	wholeSubtree
DerefAliases	neverDerefAliases
Filter	(objectclass= *)
Attributes	NULL

3.3.1.2

Searching for Individual Subscriber Data of EnumDnSched

The EnumDnSched individual subscriber data is an entry of the EnumDnSched structural Object Class. To search for EnumDnSched individual subscriber data, an LDAP search operation must be performed.

Searching for the individual subscriber data of a subscriber can be performed from the enum branch with the scope set to singleLevel. The following scenario triggers this operation:

- ENUM FE fetches one EnumDnSched data when the ENUM FE server doesnot find it in the local cache.

Search for the individual subscriber data of a subscriber from the enum entry according to Table 20.

Table 20 Searching for Individual Subscriber Data of a Subscriber from "enum" Entry

SearchRequest	
BaseObject	dn: viewid = <viewid>, fqdn=< fqdn >, ou= EnumDnSched, serv = enum, ou = servCommonData, dc = <CUDB root entry>
Scope	singleLevel
DerefAliases	neverDerefAliases
Filter	(objectclass= *)
Attributes	NULL

3.3.1.3

Searching for Individual Subscriber Data of EnumDnRange

The EnumDnSched individual subscriber data is an entry of the EnumDnSched structural Object Class. To search for EnumDnSched individual subscriber data, an LDAP search operation must be performed.



Searching for the individual subscriber data of a subscriber can be performed from the `enum` branch with the scope set `singleLevel`. The following scenario triggers this operation:

- ENUM FE fetches one `EnumDnRange` data when the ENUM FE server doesnot find it in the local cache.

Search for the individual subscriber data of a subscriber from the `enum` entry according to Table 21:

Table 21 Searching for Individual Subscriber Data of a Subscriber from "enum" Entry

SearchRequest	
BaseObject	dn: viewid = <viewid>, fqdn=<fqdn>, ou= EnumDnRange, serv = enum, ou = servCommonData, dc = <CUDB root entry>
Scope	singleLevel
DerefAliases	neverDerefAliases
Filter	(objectclass= *)
Attributes	NULL

4 Information Model

This section describes the information model, including the description and the format of the messages.

4.1 Attributes

This section provides an introduction to the most common attributes.

4.1.1 Definition

An entry consists of a set of attributes that holds information about the object that the entry represents.

An attribute is an attribute description (a type and zero or more options) with one or more associated values.

The tables in the following sections specify all the attributes for `Identities` entries in UDC DM. In addition, the attributes for `AdministrativeData` entries in the `Association` entries are also specified.

Table 22 *Attributes Description*

Attributes		
Attribute	Format, Remark	Example
A brief description of the attribute and the Object Identifier (OID) ⁽¹⁾ of the attribute	A description of the LDAP syntax characteristics of the attribute	An example with one of the possible values of the attribute

(1) The *OID* is used as a unique identifier for attributes. The *OIDs* can have different number prefixes depending on the different objects and functionalities, for example 1.3.6.1.4.1.193.169.2.(2n).

4.1.2 Convention

Several types of data exist in the CUDB Managed Object LDAP hierarchy. This section only describes the ones used in `Identities` entries.

The following data types are available in CUDB:

- Directory String



A value of the Directory String syntax is a string of one or more arbitrary characters from the Universal Character Set (UCS). The LDAP-specific encoding of a value of this syntax is the UTF-8 encoding of the character string, [UTF-8, a transformation format of ISO 10646, RFC 3629](#).

- Bit String

A value of the Bit String syntax is a sequence of binary digits.

- Numeric String

A value of the Numeric String syntax is a sequence of one or more numerals (digits 0–9) and spaces.

- IA5 String

A value of the IA5 String syntax is a string of zero, one, or more characters from International Alphabet 5.

- DN

A value of the DN syntax is the (purported) distinguished name of an entry, [Lightweight Directory Access Protocol \(LDAP\): Directory Information Models, RFC 4512](#). The DN syntax corresponds to the N ASN.1 type from [X.501].

4.1.3 ENUM FE Attributes

The following sections contain description of `Identities` attributes.

4.1.3.1 fqdn Attribute

Table 23 describes the `fqdn` attribute:

Table 23 *fqdn Attribute*

fqdn		
Attribute	Format, Remark	Example
<p>This attribute identifies the fqdn identity. The e164 Directory Number of the object. This should be full URI, including the associated EnumZone component.</p> <p>1.3.6.1.4.1.193.204.12.1</p>	<p>Type: Directory String</p> <p>Single-value attribute</p> <p>Value range: 1-255 characters</p>	<p>"1.2.3.4.5 .6.7.8.9.0. 3.3.1.e16 4.apar.net "</p>

4.1.3.2

viewid Attribute

Table 24 describes the `viewid` attribute:

Table 24 *viewid Attribute*

viewid		
Attribute	Format, Remark	Example
<p>This attribute is the ID of the view with which the EnumDnRange is associated. This view must be related to the EnumZone to which the EnumDnRange belongs. This field is used to implement the split namespace for EnumDnRange.</p> <p>1.3.6.1.4.1.193.204.12.2</p>	<p>Type: Numeric String</p> <p>Single-value attribute</p> <p>Value range: 0-19 digits</p> <p>Each digit is 0 - 9</p>	<p>'2'</p>

4.1.3.3

NaptrFlags Attribute

Table 25 describes the `NaptrFlags` attribute:

*Table 25 NaptrFlags Attribute*

NaptrFlags		
Attribute	Format, Remark	Example
This attribute identifies the IMPU identity. 1.3.6.1.4.1.193.204.12.3	Type: Directory String Single-value attribute Value range: 1 - 2 characters	"n"

4.1.3.4 NaptrOrder Attribute

Table 26 describes the `NaptrOrder` attribute:

Table 26 NaptrOrder Attribute

NaptrOrder		
Attribute	Format, Remark	Example
This attribute identifies A 16-bit unsigned integer that indicates the order that NAPTR records should be processed, low numbers being processed before high numbers. The administrator should normally provision the same value of <code>NaptrOrder</code> for all NAPTR resource records for a given DN. 1.3.6.1.4.1.193.204.12.4	Type: Numeric String Single-value attribute Value range: 1-5 digits Each digit is 0-9	'100'

4.1.3.5 NaptrPreference Attribute

Table 27 describes the `NaptrPreference` attribute:



Table 27 NaptrPreference Attribute

NaptrPreference		
Attribute	Format, Remark	Example
<p>This attribute identifies A value used to derive the value of NaptrPreference for slot 1 depending on the type of entry in this slot.</p> <p>1.3.6.1.4.1.193.204.12.5</p>	<p>Type: Integer syntax</p> <p>Single-value attribute</p> <p>Value range: 1-5 digits</p> <p>Each digit is 0-9</p>	'100'

4.1.3.6

NaptrService Attribute

Table 28 describes the NaptrService attribute:

Table 28 NaptrService Attribute

NaptrService		
Attribute	Format, Remark	Example
<p>This attribute is the value to be returned in the NaptrService field for slot 1. This field is NULL for an unused slot, a Call Server entry or a DestNode entry.</p> <p>1.3.6.1.4.1.193.204.12.6</p>	<p>Type: Directory String</p> <p>Single-value attribute</p> <p>Value range: 1 - 32 characters</p>	"E2U+SIP"

4.1.3.7

NaptrTxt Attribute

Table 29 describes the NaptrTxt attribute:

*Table 29 NaptrTxt Attribute*

NaptrTxt		
Attribute	Format, Remark	Example
This attribute is the text for slot 1, depending on the type of entry defined by the <code>naptrFlags</code> parameter 1.3.6.1.4.1.193.204.12.7	Type: Directory String Single-value attribute Value range: 1 - 255 characters	"/^.*\$/sip:861955620033@cs87.iptelco.com/"

4.1.3.8 recordid Attribute

Table 30 describes the `recordid` attribute:

Table 30 recordid Attribute

recordid		
Attribute	Format, Remark	Example
This attribute is a mask used to count services using the identity. 1.3.6.1.4.1.193.204.12.8	Type: Numeric String Single-value attribute Value range: 1 digits characters Each digit is 1-5	'1'

4.1.3.9 NaptrTtl Attribute

Table 31 describes the `NaptrTtl` attribute:



Table 31 NaptrTtl Attribute

NaptrTtl		
Attribute	Format, Remark	Example
<p>This attribute is the resource record 'time to live', or TTL, as specified in RFC 1035: a 32 bit signed integer that specifies the time interval that the resource record may be cached before the source of the information is again consulted. Zero values are interpreted to mean that the resource record can only be used for the transaction in progress, and should not be cached.</p> <p>1.3.6.1.4.1.193.204.12.9</p>	<p>Type: Numeric String</p> <p>Single-value attribute</p> <p>Value range: 1-11 digits characters</p> <p>Each digit is 0-9</p>	'24'

4.2 Object Class

The following sections introduce Object Classes, information on the Object Class identity, and the `mscIdentities` Object Class.

4.2.1 Definition

Object Classes define what type of entries are possible in an LDAP directory.

An Object Class identifies the set of attributes in an entry in the DIT, both mandatory and optional attributes. The Object Classes can be structural or auxiliary.

According to the [Lightweight Directory Access Protocol \(LDAP\): Directory Information Models, RFC 4512](#), an entry can be initiated with attributes corresponding to several Object Classes.



Table 32 shows describes what is Object Class, and the tables in the following sections specify all the Object Classes and their attributes for Identity entries in UDC data. In addition, the attributes for AdministrativeData entries in Association entries in UDC data are also specified.

Table 32 Object Class Description

Object Class Name			
Object Identifier	Object Class Type	Attribute	Required
The OID of the object ⁽¹⁾	Structural or Auxiliary	The list of attributes included in the Object Class	Mandatory or Optional

(1) The OID is used as a unique identifier for Object Classes. The OIDs can have different number prefixes depending on the object and its functionality; 1.3.6.1.4.1.193.169.1.(1n).

For more information about the syntaxes and matching rules, refer to [Lightweight Directory Access Protocol \(LDAP\): Directory Information Models, RFC 4512](#).

4.2.2

EnumFQDN Object Class

The EnumFQDN Object Class contains the attributes to characterize fqdn in the EnumDnSched and EnumDnRange entry.

Table 33 describes the attributes to handle the EnumFQDN Object Class:

Table 33 EnumFQDN Object Class

EnumFQDN			
Object Identifier	Object Class Type	Attribute	Required
1.3.6.1.4.1.193.204.11.1	Auxiliary	fqdn	Mandatory
		Zoneld	Optional
		DSUnitGroup	Optional

4.2.3 EnumView Object Class

The EnumEnumView Object Class contains the attributes to distinguish different views in the FQDN entry.

Table 34 describes how to handle EnumView Object Class:

Table 34 EnumView Object Class

EnumView			
Object Identifier	Object Class Type	Attribute	Required
1.3.6.1.4.1.193.204.11.2	Auxiliary	viewid	Mandatory

4.2.4 EnumNAPTRRecord Object Class

The EnumNAPTRRecord Object Class contains the attributes to characterize the Naptr record in the EnumView entry in ENUM DIT.

Table 35 describes how to handle the EnumNAPTRRecord Object Class:

Table 35 EnumNAPTRRecord Object Class

EnumNAPTRRecord			
Object Identifier	Object Class Type	Attribute	Required
1.3.6.1.4.1.193.204.11.3	Auxiliary	recordid	Mandatory
		NaptrFlags	Mandatory
		NaptrOrder	Mandatory
		NaptrPreference	Mandatory
		NaptrService	Mandatory
		NaptrTxt	Mandatory
		NaptrTtl	Optional

4.3 ENUM FE DIT Entries

This section provides information on the ENUM DIT entries.



4.3.1 Definition

Table 36 describes the UDC DIT and the tables in the following sections specify all the Identities entries, alias entries, or Identities service entries included in Figure 1.

Table 36 UDC DIT Description

Entry DN			
Object Class	Attribute	Value Type	Example
The list of the Object Classes which make the entry	The list of the attributes included in the Object Classes	Fixed value or Variable value ⁽¹⁾	An example with one of the possible values of the attribute

(1) If the value type is fixed, the value is given.

4.3.2 Enum Entry

This entry contains is a child of the `servCommonData` entry.

The `Identities` entry is the container, holding the different Identities for the `MultiServiceConsumer` or `Association` grouped by Identity domains as shown in Table 37.

Table 37 Enum Entry

serv=enum, ou = servCommonData, dc=<CUDB root entry>			
Object Class	Attribute	Value Type	Example
top	serv	Fixed value = enum	-
CUDBSer vice			

4.3.3 EnumDnSched Organizational Unit Entry

This entry contains is a child of the `servCommonData` entry. `enum` entry only has one `servCommonData EnumDnSched` entry.



Table 38 EnumDnSched Entry

ou=EnumDnSched, serv=enum, ou = servCommonData, dc=<CUDB root entry>			
Object Class	Attribute	Value Type	Example
top	ou	Fixed value = EnumDn Sched	-
organizationalUnit			

4.3.4 EnumDnRange Organizational Unit Entry

This entry contains is a child of the `enum` entry. `enum` entry only has one `servCommonData EnumDnRange` entry.

Table 39 EnumDnRange Entry

ou=EnumDnRange, serv=enum, ou = servCommonData, dc=<CUDB root entry>			
Object Class	Attribute	Value Type	Example
top	ou	Fixed value = EnumDn Range	-
organizationalUnit			

4.3.5 FQDN about EnumDnSched Entry

This entry contains the `EnumDnSched` data and it is a child of the `servCommonData EnumDnSched` entry. Each `servCommonData EnumDnSched` entry can include up to 48M `EnumDnSched` entrys. PG should check this rule when provisioning data.

Table 40 FQDN Entry

fqdn = <fqdn>, ou=EnumDnSched, serv=enum, ou = servCommonData, dc=<CUDB root entry>			
Object Class	Attribute	Value Type	Example



top EnumFQ DN	Fqdn	Variable	"1.2.3.4.5 .6.7.8.9.0. 3.3.1.e16 4.apar.net "
	Zoneld	Variable	CUDB LDAP Interwork Descripti on, 1/15 519-HDA 104 03/2 VERSION
	DSUnitGr oup	Variable	CUDB LDAP Interwork Descripti on, 1/15 519-HDA 104 03/2 VERSION

4.3.6

FQDN about EnumDnRange Entry

This entry contains the EnumDnRange data and it is a child of the servCommonData EnumDnRange entry. Each servCommonData EnumDnRange entry can include up to 10K EnumDnRange entrys. PG should check this rule when provisioning data.

Table 41 FQDN Entry

fqdn = <fqdn>, ou=EnumDnRange, serv=enum, ou = servCommonData, dc=<CUDB root entry>			
Object Class	Attribute	Value Type	Example



top EnumFQDN	Fqdn	Variable	"*.5.6.7.8.9.0.3.3.1.e164.apar.net" Or
			"11-33.5.6.7.8.9.0.3.3.1.e164.apar.net"
	Zoneld	Variable	CUDB LDAP Interwork Description, 1/15 519-HDA 104 03/2 VERSION
	DSUnitGroup	Variable	CUDB LDAP Interwork Description, 1/15 519-HDA 104 03/2 VERSION

4.3.7

EnumView Entry

This entry contains the view id and it is a child of the EnumDnRange and EnumDnSched entry. Each EnumDnRange and EnumDnSched can include 20 view ids, which view id is from 0 to 19. The default value is 0. PG should check this rule when provisioning data.

Table 42 IMPU Entry

viewid=<viewid>, fqdn = <fqdn>, ou=EnumDnSched, serv=enum, ou = servCommonData, dc=<CUDB root entry>			
Object Class	Attribute	Value Type	Example
top	Viewid	Variable	"0"
EnumView			



4.3.8 NAPTRRecord Entry

This entry contains the `NAPTRRecord` data and it is a child of the `EnumView` entry. Each View id can include 5 `NAPTRRecords`. PG should check this rule when provisioning data.

Table 43 NAPTRRecord Identities Service Entry

recordid =<recordid>, viewid=<viewid>, fqdn = <fqdn>, ou=EnumDnSched, serv=enum, ou = servCommonData, dc=<CUDB root entry>			
Object Class	Attribute	Value Type	Example
top NAPTRRecord	recordid	Variable	"1"
	NaptrFlags	Variable	"n"
	NaptrOrder	Variable	"100"
	NaptrPreference	Variable	"100"
	NaptrService	Variable	"E2U+SIP"
	NaptrTxt	Variable	"/^.*\$/sip: 86195562 0033@cs 87.iptelco. com/"
	NaptrTtl	Variable	"24"



5 Data Model

This section describes the UDC Data Model and the UDC information tree.

5.1 Data Model Overview

The CUDB is built as an LDAP directory server, containing the required entries and attributes according to the defined schema for the different services.

LDAP is a client/server based directory access protocol that provides both read and update access.

The LDAP data information models provide the structures and data types necessary for building an LDAP directory tree. The LDAP hierarchy has a tree structure. This tree is called the Directory Information Tree (DIT). It is composed of entries that have one or more key attributes and a set of attributes grouped in Object Classes, characterizing that entry. The entries have one key attribute. The key attribute and its value form the Relative Distinguish Name (RDN) of an object. The concatenation of the RDNs of the sequence of entries from a particular object to the root entry of the tree forms the Distinguished Name (DN). This DN uniquely identifies an object inside the tree.

5.2 ENUM FE Directory Information Tree

From the LDAP modeling perspective in ENUM FE, the DIT is built into the following separated structures under the root entry:

- Service associated to ENUM:
(serv=enum, ou=servCommonData, <root_DN>)
- Branch EnumDnSched Data:
(ou=EnumDnSched, serv=enum, ou=servCommonData, <root_DN>)
- Branch EnumDnRange Data:
(ou=EnumDnRange, serv=enum, ou=servCommonData, <root_DN>)

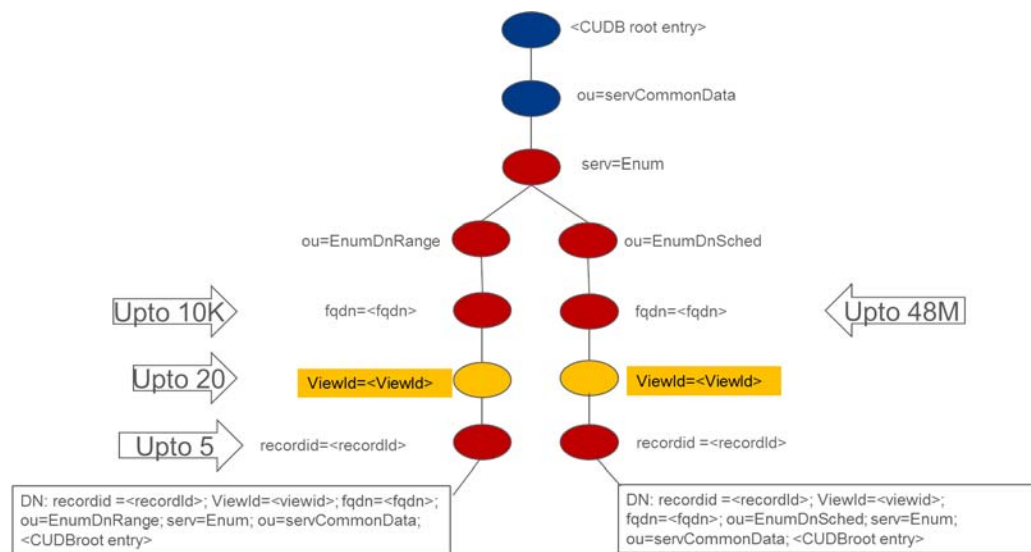


Figure 1 ENUM Directory Information Tree

5.2.1 Entries in ou=EnumDnSched

- FQDN container:
(fqdn=<fqdn>, ou=EnumDnSched, serv=enum, ou=servCommonData, <root entry>)
- EnumView container:
(viewed=<viewid>, fqdn=<fqdn>, ou=EnumDnSched, serv=enum, ou=servCommonData, <root entry>)
- NAPTRRecord container:
(recordid=<recordid>, viewed=<viewid>, fqdn=<fqdn>, ou=EnumDnSched, serv=enum, ou=servCommonData, <root entry>)

5.2.2 Entries in ou=EnumDnRange

- FQDN container:
(fqdn=<fqdn>, ou=EnumDnRange, serv=enum, ou=servCommonData, <root entry>)
- EnumView container:
(viewid=<viewid>, fqdn=<fqdn>, ou=EnumDnRange, serv=enum, ou=servCommonData, <root entry>)
- NAPTRRecord container:
(recordid=<recordid>, viewid=<viewid>, fqdn=<fqdn>, ou=EnumDnRange, serv=enum, ou=servCommonData, <root entry>)



6 Error Handling

Not Applicable.





Reference List

IPWorks Library Documents

- [1] *Glossary of Terms and Acronyms*
- [2] *Trademark Information*
- [3] *Typographic Conventions*

Other Ericsson Documents

- [4] *CUDB LDAP Interwork Description, 1/15519-HDA 104 03/2*
- [5] *ENUM Front End LDAP Data Schema*

Standards

- [6] [Lightweight Directory Access Protocol \(LDAP\): Syntaxes and Matching Rules RFC 4517](#)
- [7] [Lightweight Directory Access Protocol \(LDAP\): The Protocol RFC 4511](#)
- [8] [Lightweight Directory Access Protocol \(LDAP\): Directory Information Models RFC 4512](#)
- [9] [Lightweight Directory Access Protocol \(LDAP\): Schema for User Applications RFC 4519](#)
- [10] [Lightweight Directory Access Protocol \(LDAP\): Modify-Increment Extension RFC 4525](#)