

CUDB Technical Product Description

CUDB 1

TECHN PRODUCT DESCR

Copyright

© Ericsson AB 2016-2018. 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	Purpose and Scope	1
1.2	Revision Information	1
1.3	Target Groups	3
1.4	Prerequisites	3
1.5	Typographic Conventions	3
2	Overview	5
2.1	Supported Hardware	6
3	Functions	9
3.1	General	9
3.2	CUDB Functionality	9
3.3	Security	24
4	Interfaces	25
4.1	Reference Model	25
4.2	Protocols	25
5	Architecture	29
5.1	Logical Components	29
6	OAM	31
6.1	Fault Management	31
6.2	Performance Management	31
6.3	Log Management	31
6.4	Configuration and Software Management	32
7	Configuration	33
7.1	CUDB Node Configuration	33
7.2	CUDB System Configuration	34
8	Characteristics	37
8.1	Availability and Reliability	37
8.2	Scalability	37
	Glossary	39



Reference List

41



1 Introduction

This document provides an overview of the Ericsson Centralized User Database (CUDB).

CUDB is an extensible, high-performance, subscriber-centric database system, able to consolidate subscriber data for several application Front Ends (FEs). This telecom-grade, real-time, geographically distributed database system can store both static and dynamic data, as well as subscriber-related service data.

CUDB provides controlled and enhanced access to subscriptions and allows seamless traffic business growth by offering a single point of provisioning, as well as effortless introduction of new services in a network through simplified subscriber and service management with significant cost savings.

Note: This document applies to the node main release and all its consecutive software drops. The delivery dates of specific functions may diverge. For specific details on General Availability dates, check UDM Roadmap.

1.1 Purpose and Scope

This document describes the architecture, functions, interfaces, element management, subscriber data management, and characteristics of CUDB.

1.2 Revision Information

Rev. A

Initial release.

Rev. B

Editorial changes only.

Rev. C

Other than editorial changes, this document has been revised as follows:

- Updated virtualization terminology throughout the document.

Rev. D

Other than editorial changes, this document has been revised as follows:

- Added Local Reads function.



- Updated virtualization terminology throughout the document.

Rev. E

Other than editorial changes, this document has been revised as follows:

- Removed **Layered N+1 Redundancy in HLR Classic Networks** as a CUDB functionality.

Rev. F

Other than editorial changes, this document has been revised as follows:

- Removed EMA from the Glossary.

Rev. G

Other than editorial changes, this document has been revised as follows:

- Section 3.2.1 on page 9: Added support for IoT profiles.

Rev. H

Other than editorial changes, this document has been revised as follows:

- Section 2.1 on page 6: Updated supported hardware combinations.
- Section 3.2.22 on page 21: Added certified or certification-level delivery model, migration from native to virtualized CUDB, and supported workflows.
- Section 4.1 on page 25: Updated Figure 6.

Rev. J

Editorial changes only.

Rev. K

Other than editorial changes, this document has been revised as follows:

- Section 3.2 on page 9: Editorial updates.
- Section 3.2.1 on page 9: Updated the list of profiles.
- Section 3.2.17 on page 17: Updated Automatic Mastership Change (AMC) function.
- Section 3.2.22 on page 21: Updated the list of delivery models. Added the list of deliveries.



- Section 7.2 on page 34: Updated description.

1.3 Target Groups

This document is intended as an introduction to CUDB for network operators, network and service planners, as well as system engineers and administrators.

1.4 Prerequisites

Users of this document must have a basic knowledge of data communication and telecommunication.

1.5 Typographic Conventions

Typographic conventions can be found in the following document:

- *Typographic Conventions*





2 Overview

Ericsson evolved its telecommunications subscriber database used by its mobile circuit switch and packet switch networks, its IP Multimedia System (IMS), and Evolved Packet Core (EPC) solutions. The result of the development is the Ericsson User Data Consolidation (UDC), a system based on a layered architecture and the physical or logical decoupling of subscriber data, application service, and business logic into different network elements.

The network elements are as follows:

- Back End (BE), acting as the centralized database, such as CUDB.
- Front Ends (FEs), running the application and business logic, such as Home Location Register (HLR) FE, Home Subscriber Server (HSS) FE, Authentication, Authorization and Accounting (AAA) FE, and so on.

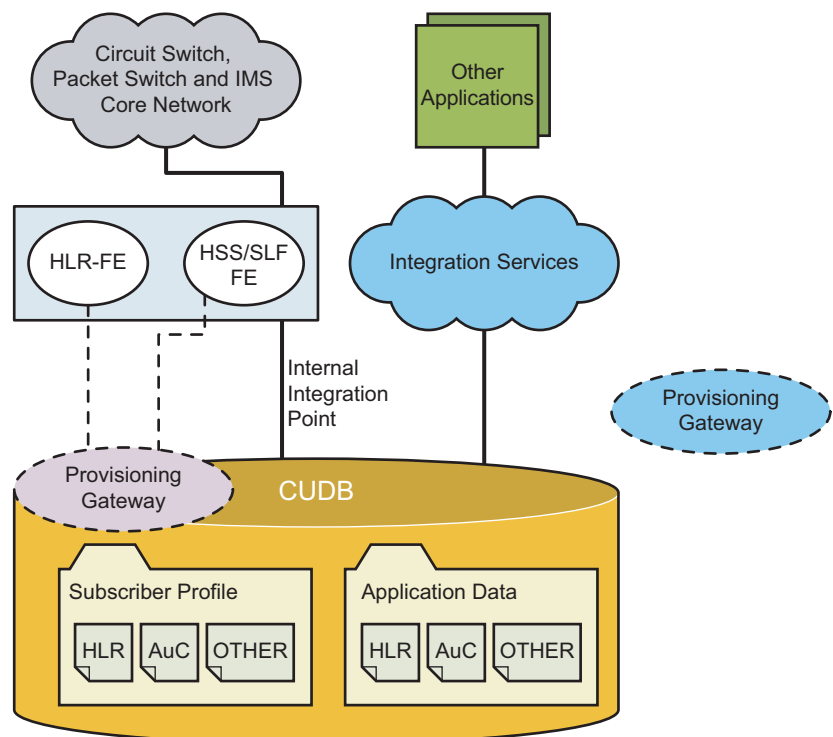


Figure 1 UDC

All subscriber-related data is stored and managed in the CUDB data layer, while service logic that was previously triggered and executed in the classic HLR monolithic architecture is now performed on the HLR-FE node. Similarly, the service logic provided by the classic Home Subscriber Server (HSS) /

Subscription Locator Function (SLF) Monolithic Architecture is executed on the HSS/SLF FE node, while the subscriber data is stored in CUDB. The same strategy is applied to other application FEs.

CUDB is a distributed system of connected CUDB nodes that cooperate to provide seamless database service. A CUDB node is the main architectural unit of this network, comprising both hardware and software components. It is a physical cabinet or a Virtualized Network Function (VNF) that provides access to CUDB system services. CUDB nodes can be placed in different network locations for distribution and redundancy purposes.

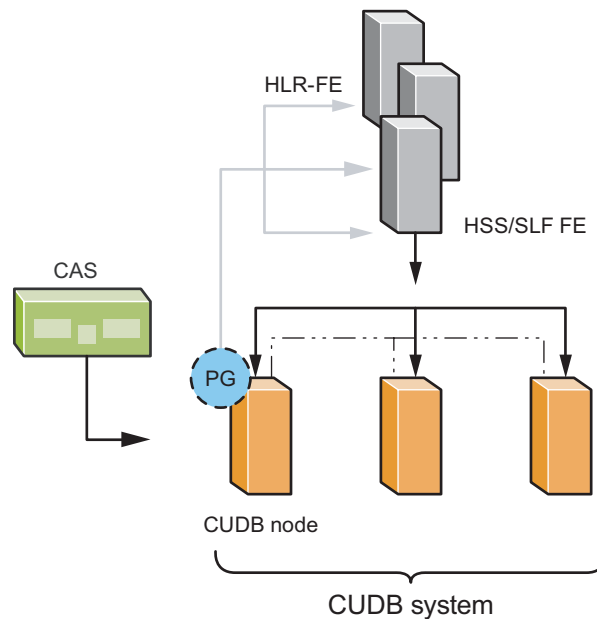


Figure 2 Ericsson UDC Overview

CUDB stores subscriber profiles and provisioned (static) or non-provisioned (dynamic) service data associated to subscribers. As a subscriber-centric database, CUDB also holds the service profiles for the supported applications.

CUDB supports application FEs, such as HLR and HSS/SLF. Other application FEs can be supported through integration services.

2.1 Supported Hardware

CUDB supports the Ericsson Blade Server Platform (BSP) 8100 hardware platform with Generic Ericsson Processor version 5 (GEP5) blades or Generic Ericsson Processor version 3 (GEP3) blades.

CUDB supports virtualized environments. CUDB is verified to run on the Ericsson Cloud Execution Environment (CEE), on BSP 8100 hardware with GEP5 blades. Support for other virtualized environments or hardware is possible, but must be secured through an integration project.



It is possible to combine nodes with different hardware platforms in the same CUDB system. Mixing the following platform combinations in hybrid systems is allowed:

- GEP3 and GEP5
- GEP3 and virtualized CUDB on GEP5 BSP 8100
- GEP5 and virtualized CUDB on GEP5 BSP 8100

Note: Combining GEP3 and GEP5 blades in the same node is not supported.





3 Functions

This section provides details on CUDB functions.

3.1 General

CUDB is a highly available, geographically redundant, real-time, distributed database exposed as a Lightweight Directory Access Protocol (LDAP) directory to HLR, Authentication Center (AuC), HSS/SLF, or any other applications.

3.2 CUDB Functionality

CUDB is a geographically distributed system that provides local single logical points of access for traffic and provisioning. Each CUDB node provides access to the entire subscription database, regardless of data distribution across the interconnected CUDB nodes. This is called a single logical point of access, physically implemented as an IP address and port pair, for each CUDB node. The internal CUDB network and node architecture are transparent to applications.

When subscriber data is being provisioned, CUDB automatically balances the amount of stored data among the available data partitions, although custom data distribution policies may also be defined. In both cases, CUDB makes it possible to store data on particular subscribers in CUDB nodes located in specific geographical areas.

3.2.1 Structured, Flexible and Extensible Common Data Model

CUDB stores subscription information using an LDAP Directory Information Tree (DIT) as the application data model. The LDAP tree is divided into different parts, managed and handled in different database clusters or cluster groups, becoming a distributed LDAP directory. For further details on LDAP, refer to *CUDB LDAP Interwork Description*, Reference [1].

Subscriber data for multiple applications or network elements is supported simultaneously.

CUDB supports the Ericsson 2G/3G (HLR-FE), 4G, 5G and IMS (HSS-FE), AuC, Service-Aware Policy Controller (SAPC), Machine-to-Machine (M2M), AAA, Mobile Number Portability (MNP), E.164 Number Mapping (ENUM), Equipment Identity Register (EIR), and Internet of Things (IoT) profiles.

The structured data model is flexible and extensible. Extensibility is achieved through configuration. Data model extension is carried out online without system degradation. New types of subscriber application profiles, application



common data profiles, and subscriber common data profiles can be created, and as such, new services can be introduced with a shorter time to market and without impacting traffic.

CUDB supports multiple identifiers per subscriber. New types of identifiers can also be added dynamically without system degradation.

3.2.2 Data Distribution

The subscriber data set is divided into disjoint subsets. Each of these subsets is stored on a data partition, or Data Store Unit Group (DSG). A DSG is a subscriber data partition, the basic distribution unit of subscriber data. CUDB stores every information related to each subscriber in one DSG only, but data related to different subscribers can be allocated to different DSGs.

Each CUDB node hosts a number of database clusters. These database clusters are replicated and geographically distributed across different network sites, making up a distributed Data Store (DS) Layer. These database clusters are called DS units.

A DS unit can be configured to host data for any DSG. For example, in Figure 3, DS2 hosts data for DSG2 in CUDB node 1, but the same DSG2 is hosted by DS1 in CUDB node 5. Not all DSGs must be hosted in all CUDB nodes. However, the data stored on any DSG is accessible from any CUDB node in the system.

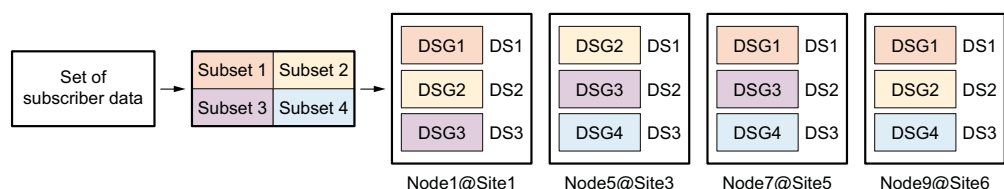


Figure 3 Subscriber Data Distribution

One of the DS units of the DSG is called the DS Master replica. All high-priority read and all write operations related to the subscriber data in that partition are executed in the master replica. This mechanism maintains data consistency in the CUDB system.

Modifications are replicated asynchronously to the rest of the DS units in the partition. The other DS units are in charge of low priority tasks such as massive queries, or export operations. If the DS Master replica in a DSG fails for any reason, another DS unit in that partition is appointed as the new master replica.

The Processing Layer Database (PLDB) contains the rest of the data, as aliases, profiles, and so on. PLDB also stores information on which DSG hosts data for each subscriber.



By default, subscribers are automatically distributed among the DSGs assigned to them, based on DSG occupancy. However, specifying the DSG or geographical zone where the data must be stored is also possible.

CUDB supports custom DSG distribution algorithms during provisioning. This is performed by configuring and loading the desired algorithm rules in a dynamic library.

For further information on data distribution, refer to *CUDB LDAP Data Access*, Reference [2].

3.2.3 Data Backup and Restore

A backup of the whole system can be ordered and executed without stopping traffic. The backup is carried out over each subscriber subset, including the contents of the entire database. During backup, traffic and provisioning are allowed. As a rule, the resources used for backup are not used for traffic or provisioning.

Recovery can be executed for the whole system or only for a database partition. For more details on the backup and restore procedures, refer to *CUDB Backup and Restore Procedures*, Reference [3].

3.2.4 Import and Export

CUDB supports large-scale import and export operations for the entire database, or selected partitions, performed with the provided import and export tool commands.

Import operations can be optimized, for example, by using separate import files for different sets of data, divided by subscribers and applications. As such, a large amount of data can be imported simultaneously through multiple connections, minimizing time.

Export can be optimized by parallel execution of different partial exports, based on hierarchy or branch criteria, as well as filtering.

For more details on data import and export, refer to *CUDB Import and Export Procedures*, Reference [4].

3.2.5 Access Control

Access Control policies for LDAP, provisioning, and traffic can be dynamically configured through the definition of users, groups, and their associated set of data access privileges.

Access Control is performed when an application FE attempts to access subscriber data stored in CUDB, using data consumer authentication and data access authorization.

These privileges are set by using flexible Access Control Lists (ACLs) supporting regular expressions.

Access Control is performed based on the following factors:

- LDAP client type.
- Level of the service data attempted to be accessed, such as branch level, object class level, or attribute level.
- Access type, such as read or write.

3.2.6 Notifications

CUDB supports outbound notifications. When a piece of data is modified in a subscriber profile, CUDB can send Simple Object Access Protocol (SOAP) based notifications towards a configurable endpoint destination.

The support for notifications allows extensive configuration, including the following options:

- Monitored attribute.
- Condition sets of the monitored attribute, other attributes, or both.
- Notification message contents.
- Destination endpoint list, that is, application FE list.

Several receiver endpoints, that is, notification destinations, can be configured for every defined notification type. Based on the number of destinations, two types of notifications are available:

- Notifications sent to all members of the endpoint.
- Notifications sent to one member of the endpoint.

CUDB also enables individual SOAP-based notification connections to be protected with Transport Layer Security (TLS) / Secure Socket Layer (SSL).

For more details on notifications, refer to *CUDB Notifications*, Reference [5].

3.2.7 EPC Mobility Support

CUDB supports HLR and HSS/SLF interoperability based on notification services to notify application FEs on database updates in EPC mobility scenarios.

3.2.8

Geographical Zone Administration

Geographical zones offer customized data distribution, allowing the reduction of bandwidth across geographically distant CUDB sites using ad-hoc subscription distribution. Ad-hoc distribution is based on the zone concept shown in Figure 4.

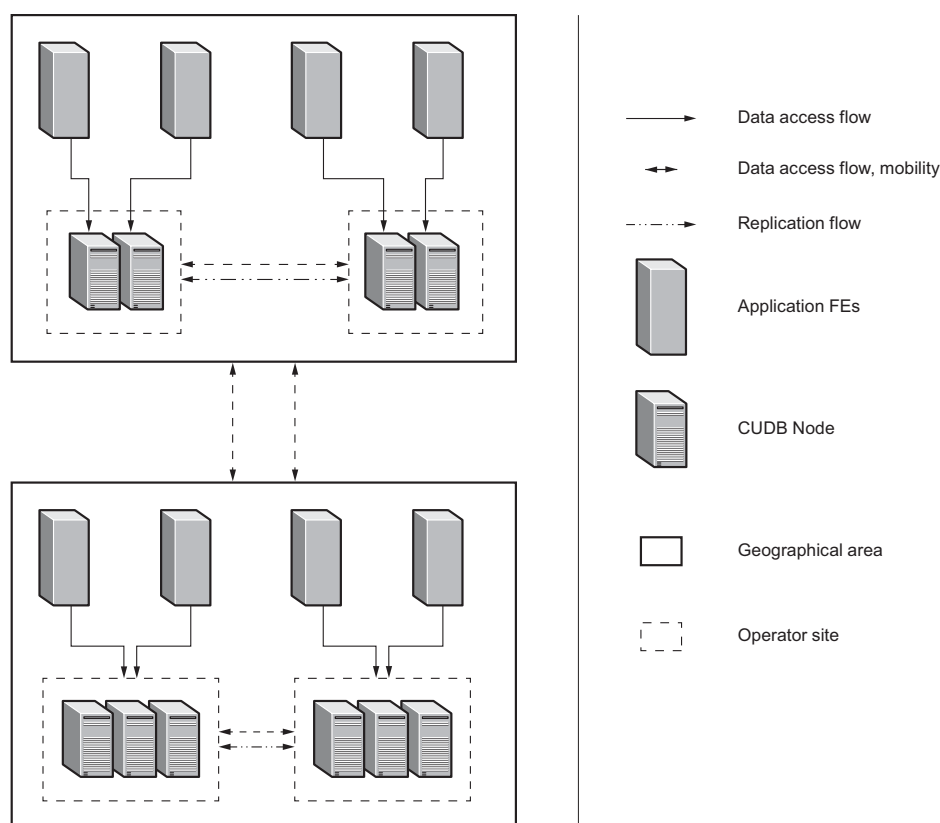


Figure 4 Geographical Zone Administration Concepts

Individual subscribers are allocated to a zone based on customer choice, reducing data flow among areas. Each zone has its own independent storage and processing resources.

When a subscriber profile is created, the subscriber is assigned to a zone. The profile is, in general, accessed from the zone the subscriber was allocated to, keeping the network traffic local. For mobility scenarios, such as when the subscriber visits a different part of the country, the subscription data is transparently accessible. Smart zone use improves data access latency and network costs for large areas or between areas where data transport fees are expensive, as replication traffic is kept local to nodes within the zone.

For more details, refer to *CUDB Multiple Geographical Areas*, Reference [6].



3.2.9 BLOB Support

CUDB supports the definition of Binary Large Objects (BLOBs) in the data model. Each BLOB can be stored in the memory or on disk, based on individual configuration. For more information, refer to *CUDB Binary Large Object Attributes Management*, Reference [7].

3.2.10 External Data Access Support

Applications connected to CUDB can access any data stored by them and by other application FEs, with the proper access rights (see Section 3.2.5 on page 11).

3.2.11 Subscription Reallocation

CUDB supports reallocating data from one DSG to another DSGs to ensure that all DSGs have optimal data occupancy.

Node-, site-, or zone-level data reallocation is also possible, through the proper handling of the origin and the target DSGs. The destination DSG does not have to be empty or belong to a different CUDB node.

For more details on reallocation, refer to *CUDB Subscription Reallocation*, Reference [8].

3.2.12 Geographically Redundant Deployments

Geographical redundancy allows storing copies of the same data in different CUDB nodes in different sites, providing a high redundancy and resiliency level. This function is available in the following two configurations:

- Double Geographical Redundancy, also known as 1+1 redundancy, deploys two replicas of each DSG, hosted in a different node located in a different site.
- Triple Geographical Redundancy, also known as 1+1+1 redundancy, deploys three replicas of each DSG, hosted in a different node located in a different site. Triple Geographical Redundancy is available in the Advanced Network Protection Value Package.

Note: Standalone configuration without geographical redundancy is only supported for Customer Trial, Customer Test, and Ericsson Internal systems.

For more information on geographical redundancy, refer to *CUDB High Availability*, Reference [9].



3.2.13

Enhanced Overload Protection and Cooperative Load Regulation FE-BE

CUDB nodes incorporate self-protection mechanisms and internal load regulation between components, so that excessive incoming traffic (overload) does not lead to service interruption. If the overload is up to 150% of the engineered capacity, CUDB guarantees that at least 90% of the throughput at the engineered capacity is served.

CUDB is designed to withstand traffic peaks up to 300% of its engineered capacity.

CUDB nodes also support UDC Cooperative Load Regulation FE-BE, which provides automated traffic throttling in the overloaded UDC system. This is implemented in both the application FE and the BE, that is, CUDB,, and it is triggered by the application FE when the BE is facing difficulties handling the incoming traffic.

In overload situations, CUDB reports the overload status to the application FEs to trigger FE-BE Cooperative Load Regulation. After receiving overload notifications, the FE can initiate graceful throttling that limits traffic throughput in the UDC system, allowing CUDB to work at its engineered capacity and to handle the UDC system overload as fast as possible while maximizing service availability.

The FE-BE Load Regulation mechanism may occur in the following two ways:

- Overload of a DS unit in a CUDB node

When a DS is overloaded, the CUDB node initiates its internal load regulation between the local LDAP FEs and the affected DS unit. Excessive incoming traffic to the affected DS is rejected with a specific LDAP error code.

After receiving this specific error code, the application FE notifies the network with the proper signal, such as, sending MAP TC-ABORT.

As only excessive traffic is rejected, only subscribers in the overloaded DS are affected. Subscribers in other DS units, and the UDC system remain unaffected.

- Overload of PLDB or LDAP servers in a CUDB node

If PLDB or LDAP servers in a CUDB node are overloaded, excessive incoming traffic requests to the node are answered with an overload-specific error code, indicating the node congestion situation.

After receiving the overload-specific error code, the application FE measures the level of congestion in the CUDB node.

Depending on the measured congestion level, the application FE may start Cooperative Load Regulation, that is, the early rejection of some traffic



coming from the core network before sending it to the overloaded CUDB node.

The amount of traffic that the application FEs reject early is dynamically adapted to the measured congestion errors received from the CUDB node.

When the overload ends, the application FE normalizes traffic handling and smoothly reduces traffic rejection as the number congestion errors from the BE decrease.

Statistics about the rejected overload traffic are logged in the affected nodes.

3.2.14 Traffic Prioritization per Application under Overload

The “Traffic Prioritization per Application under Overload” function makes it possible to configure how CUDB handles incoming LDAP traffic from different application FEs when the DS layer is overloaded.

CUDB defines five priority levels under overload: one to five. LDAP users, representing application FEs, may be configured with a specific priority level through the `overloadRejectionWeight` attribute. Unless configured with a priority level, LDAP users keep the default priority level, which is one.

If database clusters are overloaded and traffic must be rejected, CUDB checks the priority level of the LDAP user issuing the LDAP request. Low-priority LDAP users will see their traffic more aggressively rejected than high-priority LDAP users. Traffic from high-priority LDAP users is processed at the expense of traffic from low-priority LDAP users. Thus, high-priority LDAP users have a higher successful-to-rejected traffic ratio than low-priority LDAP users.

The priority assigned to an LDAP user can be changed at any time, even in an overload situation. Priority-level changes take effect immediately.

3.2.15 Data Schema Management Tool

CUDB includes a Graphical User Interface (GUI) called the Data Schema Management Tool to manage the LDAP schema files. The tool allows the following operations:

- Creating new LDAP schema files.
- Browsing existing LDAP schema files.
- Editing and saving existing LDAP schema files.

The CUDB Data Schema Management GUI is included in the product by default. The tool handles LDAP schema files, but does not directly load schema files from the LDAP FEs running in the CUDB nodes.



The tool allows the following operations on schema files:

- Including schema files from different application FEs in one project, if different schemas share the same object classes and attributes.
- Creating, opening and saving projects.
- Creating a new schema file within a project.
- Adding existing schema files to a project.
- Removing a schema file from a project.
- Updating and saving schema files.

The following actions related to LDAP schema objects can also be performed:

- Creating attribute types.
- Editing attribute types.
- Creating objects.
- Editing objects.

The tool performs integration constraints validation in the schema files loaded in a project. The checked criteria are as follows:

- No duplicate or invalid object identifiers (OIDs) are used.
- No duplicate attribute or object name.
- Schema is consistent and valid..

The Data Schema Management Tool is executed on any Linux machine with Java or in the Operation and Maintenance (OAM) server blades installed in each CUDB node.

This GUI does not directly apply any schema change to a running node. Before proceeding with the schema updates, contact Ericsson personnel.

3.2.16 Consistency Check

CUDB Consistency Check is a basic function that looks for divergences in data stored on a slave PLDB or DSG replica, and the corresponding master replica, by scanning the database on both slave and master sides.

Consistency Checks can be ordered through the Command Line Interface (CLI), and their result is placed in output log files. Alarms are raised if data divergence is detected. Alarm severity level depends on the amount of inconsistency found.

For more details on CUDB Consistency Check, refer to *CUDB Consistency Check*, Reference [10].



3.2.17 Automatic Mastership Change

The Automatic Mastership Change (AMC) function serves the following purposes in CUDB:

- To secure that PLDB and DSG master replicas work at full capacity. So, if the current master replica gets crippled for any reason, AMC executes a mastership change, and assigns the master role to an available and healthy replica of the same partition.
- To eventually return the master role of PLDB or DSGs to the replica configured with the highest priority if it is ready for service, and the master replica is elsewhere. If AMC is enabled, it periodically checks if the PLDB and DSG masters are located on the preferred replica of each group. If AMC detects that a PLDB master or a DSG master is not located on the preferred replica of the group, it executes a mastership change if the preferred replica is healthy and ready for service.

For more information on the AMC function, refer to the *Automatic Mastership Change* section of *CUDB High Availability*, Reference [9].

3.2.18 Automatic Handling of Network Isolation

This functionality maximizes service availability in network isolation scenarios, reducing the chance of partial loss of service in an isolated site or sites during a network incident.

After a network isolation incident, CUDB preserves both data consistency and data durability using a sequence of automatically-triggered internal procedures, while ensuring service availability for end users.

During a network incident, CUDB can be split in two or more groups of CUDB sites with no communication between them.

A group with less than half of the sites in the CUDB system is called minority. A group with more than half of the sites is called majority.

In a symmetrical split, the system is split in two halves with the same number of sites in each group. In this case, the system is divided into two identical halves without minority or majority.

In an asymmetrical split, there are one or more minorities with or without a majority. Typically, but not necessarily, one majority with half or more than half of the sites exists.

CUDB handles such scenarios in different ways, providing different service availability per site based on the type of group each site belongs to: majority, half, or minority. System behavior and service maximization are described below:



- Symmetrical split:

The two system halves provide continuous service. During a network incident, CUDB works in multi-master mode when both halves make changes locally, but cannot replicate to the other half, temporarily behaving as two divergent databases.

- Asymmetrical split:

By default, the sites in a minority stop providing service and only the group with the majority or half of the sites provides service. All incoming network traffic reaches this majority group through automatic FE and network failovers. Therefore, the affected CUDB system does not enter multi-master mode and no recovery steps are needed when the split ends. No temporary database divergence occurs.

However, if it is necessary due to network design, a specific isolation event, or any other reason, minorities can be forced to provide service by command or through configuration, allowing service maximization in the isolated network regions, meaning that the system potentially enters multi mode, when the network incident or network design does not allow redirecting traffic to the majority group through automatic network failovers or application FE failovers.

When the network connectivity is recovered, CUDB exits multi-master mode if it was used. All data modifications from multi-master mode are identified and merged into the master to achieve the highest possible data durability.

After data repair is finished, CUDB automatically orders data backup in the current master replica and data restore in the current slave replica or replicas to synchronize data. Optionally, self-ordered backup and restore can be disabled by configuration to make it a manual step.

Once recovery is complete, geographical replication is immediately restarted.

Summarized Automatic Handling of Network Isolation provides the following:

- Provisioning during symmetrical split situations.
- Entering in multi-master mode in a symmetrical split situation.
- Enter multi-master mode using a command or through configuration in minority situations, thus providing service in the isolated minority as needed.
- Automatically identifying data changes in any master replica other than the current master replica, when connectivity is restored after multi-master mode.
- Automatically updating data in the current master replica to smartly inject data that changed in the other master replicas, during multi-master mode.



- Automatically ordering and executing the required data backup in masters and data recovery in slaves, as part of the recovery actions after network split.

Automatic repair is based on time stamps.

The Automatic Handling of Network Isolation function secures that provisioned data is never impacted or overwritten.

If the data repair functionality cannot resolve a conflict, such as the same LDAP entry being modified in two sites during the network incident, a report is generated identifying any LDAP entry that requires further investigation.

3.2.19 Provisioning Assurance after CUDB Mastership Change

This functionality preserves provisioning data durability following an unexpected or unplanned mastership change, to ensure service availability for end users.

Following a mastership change, CUDB automatically addresses the potential risk of having lost, or having only partially executed some provisioning operations. Such operations may have not been replicated in time to the new elected master before or during the mastership failover.

In such cases, CUDB informs the Provisioning Gateway to re-execute the affected operations, securing that any potentially lost or partially executed provisioning operation is properly recovered.

3.2.20 Multi-Application Support

CUDB on BSP 8100 (GEP5) platform includes multi-application support, which allows the following:

- Co-habitation of applications within a subrack.
- Expansion within the same subrack, cabinet, or with additional cabinets.
- One application can span over subracks or cabinets.
- Common infrastructure shared by all applications.
- Relocation of hardware between applications on command.

3.2.21 LDAP Data Views

The LDAP Data Views function makes it possible for application FEs to access data through a customized tree structure, called a custom DIT, with a custom schema. LDAP users can be configured to access CUDB either using the core DIT or to use a defined LDAP data view.

Multiple LDAP data views can be defined to accommodate application FEs.



Accessing CUDB through a data view applies some restrictions on the `write` operations, that is, `add`, `delete`, or `modify`, to guarantee the coherence of the core DIT data (for more information, refer to *CUDB LDAP Data Views*, Reference [11]). Ericsson strongly recommends that provisioning-type application FEs access CUDB through the core DIT.

LDAP data views are not available for export and import procedures or for notifications. Notifications must be configured according to core DIT Distinguished Names (DNs) and attributes. Data in SOAP messages are, likewise, to be identified by their core DIT DN and attributes.

The LDAP Data Views function is included in the Application Facilitator Value Package.

3.2.22

VNF Support

The VNF Support function allows decoupling software and hardware through virtualization, enabling the harmonization of hardware across multiple products and vendors, and the optimization of hardware utilization.

Virtualization has benefits that change the way infrastructures and applications are deployed, used, maintained, and supported. It enables operators to move or deploy applications into the cloud.

The support of virtualization in CUDB is intended for early adopters of cloud technology.

VNF Support covers installation, configuration, and adaptation of CUDB OAM functions.

Virtualized CUDB provides the same functionality as a native CUDB system by:

- Keeping the logical architecture. A physical blade in the native system is mapped to a single Virtual Machine (VM), while a CUDB node is mapped to a VNF.
- Supporting the same external interfaces. Interoperability with non-virtualized application FEs is guaranteed.
- Mapping and adapting existing OAM functions. Procedures present in native CUDB systems can also be executed in virtual deployments, with the exception of those related to hardware, networking, or routing, handled by the applicable Virtual Infrastructure Manager.

CUDB is available through three different delivery models:

- System verification delivery model: virtualized CUDB is system-verified over full Ericsson cloud infrastructure (ECEE running on either BSP 8100 or HDS 8000), which ensures CUDB functionality and performance, and maximizes predictability.

- Certified or certification-level testing delivery model: virtualized CUDB is delivered on a cloud infrastructure where certification tests have been run. Functionality and performance of the solution is secured through a streamlined system integration project. Lifecycle management is provided for CUDB only, although end-to-end support may be requested, if needed. CUDB is certified to run on VMware.
- Software-only delivery model: when third party cloud infrastructure is required, functionality and performance are secured through a system integration project, and life cycle management is provided only for CUDB. CUDB provides information on the minimum requirements on the cloud infrastructure.

CUDB deliveries include the following:

- a VMware specific package
- a HOT package

It is possible to migrate a native CUDB system to a virtualized one using two different procedures:

- Migration through backup and restore.
- Migration through subscription reallocation.

CUDB supports the following workflows on the full Ericsson cloud infrastructure (ECEE & BSP 8100):

- Instantiation.
- Termination.

3.2.23 Flexible PL Deployment

The Flexible Processing Layer (PL) Deployment function allows the optimization of the number of PLDB servers in the system. Operators can decide the level of redundancy for PLDB per site, as long as at least one instance of PLDB is kept per site.

Until CUDB 1, a CUDB system consisted of a set of CUDB nodes of a single type. All nodes contained both a PLDB replica and one or more Data Store units.

With Flexible PL Deployment, it is possible to define nodes with only Data Store units and no PLDB replica. These nodes must not be configured to receive direct traffic from application FEs.

This function allows the optimization of equipment in existing customers by:

- Reducing the number of servers required in a system.



- Converting redundant PLDB servers into Data Storage servers.

The allowed CUDB deployments are:

- All nodes with PLDB servers, already supported in all previous releases
- Some nodes with PLDB and some nodes without PLDB. In such a deployment, nodes without PLDB do not handle direct traffic, and at least one node with PLDB per site is mandatory.

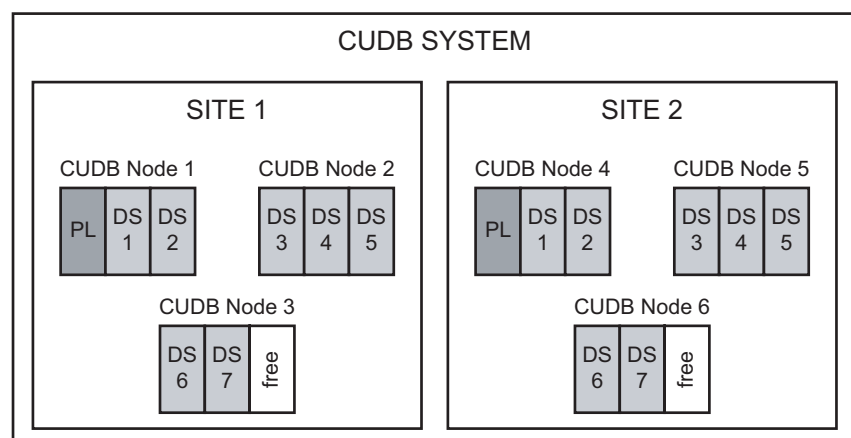


Figure 5 Example of a CUDB System Setup with Flexible PL Deployment (one PLDB Instance per Site)

3.2.24

Local Reads

The Local Reads function makes it possible for application FEs to read data from the closest replica even if it is not a master replica. The Local Reads function provides shorter response times, reduced traffic through the IP backbone, and a more efficient use of the available processing capacity since proxy traffic between CUDB nodes is minimized.

CUDB only allows local reads on slave replicas if the data there is recent enough, that is, the replication distance between the slave replica and its master is not greater than a configurable threshold. The threshold is defined per application FE or LDAP user.

The Local Reads function is configured per LDAP user, so it is possible to have application FEs with and without Local Reads in the same network. The Local Reads configuration for a given LDAP user can be changed at runtime without traffic disturbance.

Application FEs configured to use Local Reads can choose to override the Local Reads configuration, and perform a read on the master replica by an LDAP extension control, on a per LDAP request basis.



The Local Reads function is included in the Deployment Flexibility Value Package.

3.3 Security

CUDB supports the following interfaces over a secure protocol:

- Data can be reached through LDAP over TLS/SSL.
- Access to the CLI for configuration purposes is achieved through Secure Shell (SSH).
- For sending notifications, SOAP is used over Hypertext Transfer Protocol Secure (HTTPS), with either simple or mutual authentication.
- Statistics and logs are accessible through Secure File Transfer Protocol (SFTP).
- CUDB can be configured to support TLS/SSL-protected replication between nodes.

Authentication is supported by user credentials, that is, username and password, through the LDAP and OAM interfaces.

CUDB supports Simple Authentication, Security Layer (SASL) and TLS authentication for the LDAP interface.

Access to data can be limited using ACLs. For further information on ACLs, see Section 3.2.5 on page 11.

It is possible to delegate authentication of OAM users to a properly configured external LDAP/LDAPS authentication server.

For more details on CUDB security, refer to *CUDB Security and Privacy Management*, Reference [12].

4 Interfaces

This section provides information on CUDB interfaces.

4.1 Reference Model

The interfaces and the protocols used within them are shown in Figure 6.

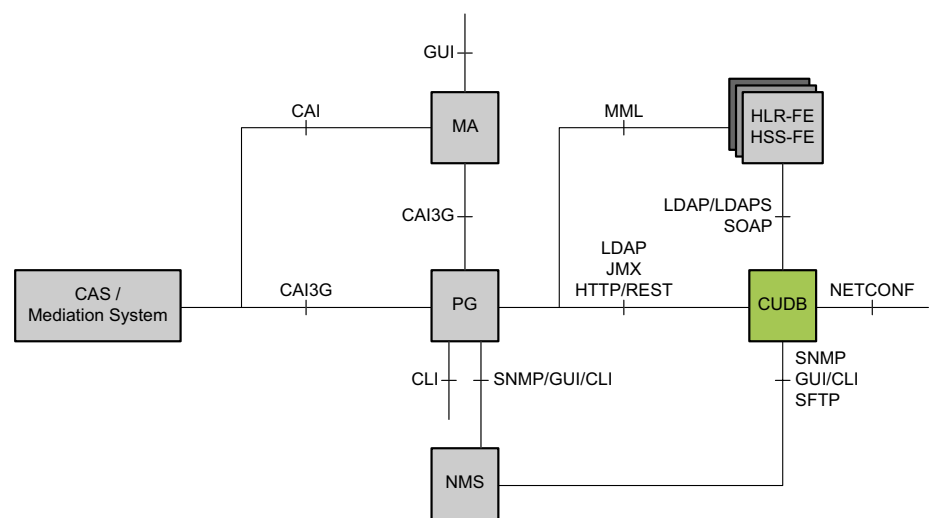


Figure 6 Reference Network Model

4.2 Protocols

A general outline of the protocols and the necessary stacks used by CUDB are depicted in Figure 7.

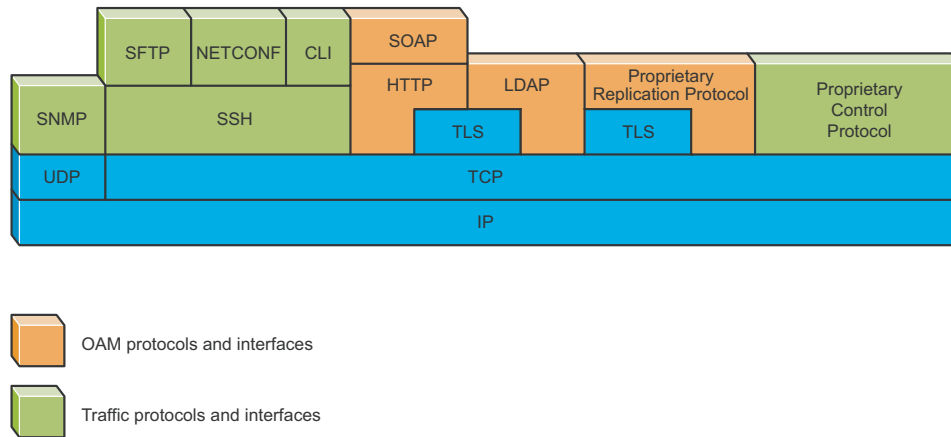


Figure 7 Protocols and Interfaces

The protocols used by CUDB and depicted in Figure 7 are as follows:

- **LDAP/LDAPS V3**

LDAP is an application protocol for querying and modifying directory services running over TCP/IP. CUDB supports the Secure LDAP protocol suite as defined by the Internet Engineering Task Force (IETF). For more information on LDAP, refer to *RFC 4510: Lightweight Directory Access Protocol (LDAP)*, Reference [21]. For more information on the enhancements on standard LDAP that CUDB introduces, refer to *CUDB LDAP Interwork Description* Reference [19].

CUDB provides LDAP FE with the SASL Digest authentication procedure to authenticate a remote LDAP server.

CUDB supports client-side authentication of the TLS/SSL LDAP interface and optional server-side authentication.

- **SFTP**

SFTP allows the transfer of files to and from CUDB, and is also used to retrieve performance information and log files. Data is encrypted during transfer to prevent packet sniffers from extracting useful information from the data packets.

The protocol itself does not provide authentication or security; it relies on SSH to provide such features. Currently, there is no standard for this protocol; only IETF drafts exist.

- **SSH**

SSH is a network protocol used for data exchange through a secure channel between two computers. Encryption provides



confidentiality and data integrity over an insecure network, such as the Internet. For more information on SSH, refer to *RFC 4252: The Secure Shell (SSH) Authentication Protocol*, Reference [22].

- **CLI**

CUDB provides a CLI for its configuration and software management. That CLI is used to alter CUDB parameters and order OAM requests, such as data backups or software upgrade operations. Users or applications accessing the CLI must provide login credentials.

- **NETCONF**

CUDB also provides configuration management through the IETF Network Configuration Protocol (NETCONF). NETCONF uses XML-based data encoding and protocol. For more information on NETCONF, refer to *RFC 4741: NETCONF Configuration Protocol*, Reference [23].

- **SNMP**

The Simple Network Management Protocol (SNMP) allows logically remote users to inspect or alter the management information of a network element. CUDB supports SNMP for Fault Management (FM).

- **SOAP**

SOAP is a simple and extensible XML-based protocol for exchanging structured information between applications over the Internet. CUDB uses this protocol to send notifications over HTTP/HTTPS. For further details on CUDB using SOAP, refer to *CUDB SOAP Interwork Description*, Reference [13].

- **JMX**

The Java Management Extension (JMX) protocol is used between CUDB nodes and the Provisioning Gateway (PG) to optionally withhold provisioning during CUDB system data backups.

- **HTTP/REST**

The HTTP/REST protocol is used between CUDB nodes and PG to trigger Provisioning Assurance after CUDB Mastership Change.

CUDB also makes use of the following internal protocols:

- **Proprietary Replication Protocol**

The proprietary replication protocol is used to keep the replicas up-to-date. When the protocol is active, changes in the database are replicated, keeping the system continuously updated.

- **Proprietary Control Protocol**



The proprietary control protocol is used to maintain accurate information on the status of the CUDB system in each node.

- **LDAP/LDAPS v3**

The LDAP protocol is also used internally between CUDB nodes for internal proxy functions.



5 Architecture

This section describes the CUDB architecture from two different points of view: logical and hardware.

5.1 Logical Components

CUDB consists of the three logical components described in the following subsections.

5.1.1 PL

This component provides the following functions:

- Load balancing

CUDB offers connection load balancing by removing single points of failure and virtualizing the network.

- PLDB

The PL on a CUDB node typically holds a replicated copy of the PLDB. PLDB is a clustered database, holding indexing information for the data in the DSGs, as well as storing a complete set of subscriber identities. PLDB can also store common subscriber data, application FE data, or any other type of data as needed.

It is possible to deploy CUDB nodes without a PLDB replica. See Section 3.2.23 on page 22 for more information.

- LDAP Gateway and Proxy Servers

The LDAP gateway and proxy servers perform the following tasks:

- Locating subscriber information across the distributed CUDB system by looking up the subscriber identity defined in PLDB.
- Providing protocol conversion from LDAP to the internal DS technology.
- Massive search request execution.

5.1.2 DS Layer

This component provides the following functions:

- DS unit: A database cluster storing data of a data partition, that is, a DSG.



- DSG: A group of DS units storing exactly the same subset of subscribers, that is, the same partition, configured in geographical redundancy replication.
- Data Replication: Replicates all modifications on the master replica of a DSG to the rest of DSs in the same DSG.

5.1.3 **Monitoring and OAM Components**

This component provides the following functions:

- Internal detection of the CUDB system status, failures, reacting to report alarms, service maximization, data consistency protection, and internal failover to the redundant components when needed.
- FM, Performance Management (PM), Configuration Management (CM), backup and restore, and so on.



6 OAM

CUDB provides different interfaces for different administrative purposes, such as configuration and supervision. Focusing on customer needs, the goal is to provide operators with a flexible, cost-effective, and secure management system. CUDB offers OAM on a per node level.

Configuration management is supported through a CLI or NETCONF.

SNMP is used to provide FM alarms. For PM, statistics and logs are accessible through the SSHv2 suite of tools, like SFTP.

To visualize system status, the UDC Cockpit in User Profile Gateway (UPG) product provides a single, easy-to-use web interface.

6.1 Fault Management

Alarms are delivered through an SNMP flow to the Network Management System (NMS), allowing system administrators to detect faults and malfunctions in the node. Based on the alarm information, the system administrator can locate the event source and the documentation relevant to resolving or preventing failures. For more information on CUDB FM, refer to *CUDB Node Fault Management Configuration Guide*, Reference [14].

6.2 Performance Management

CUDB node elements generate statistics and counters, enabling the operator to monitor the performance of each CUDB node. Statistics are collected by internal performance components in each CUDB node and are made available to the NMS as standard 3GPP XML files. For more information on the related 3GPP standard, refer to *3GPP TS 32.435: Telecommunication Management; Performance Measurement: eXtensible*, Reference [24]. For further details on CUDB PM, refer to *CUDB Performance Guide*, Reference [15].

6.3 Log Management

CUDB node elements generate log files, enabling the operator to monitor events on each CUDB node. Logs can be easily collected by external management applications using SFTP.

Authentication and command logs can be sent to an external logging server.

For more details on CUDB logs, refer to *CUDB Node Logging Events*, Reference [16].



6.4 Configuration and Software Management

Both NETCONF and a CLI interface are available for handling CUDB node configuration. Configuration management is available through a Managed Objects tree with configurable parameters, or parameters that offer information on the status of certain components of CUDB, such as the status of a specific DS, or whether a specific DS is the master on its partition. CUDB includes tools to help define the initial Managed Objects tree and to verify if it is correct.

Certain CUDB operations are performed by executing ad-hoc commands.

Software inventory in a CUDB node shows all software package versions installed on each server of the CUDB node.

CUDB supports creating backups of the system configuration per CUDB node.

The LDAP Schema Management GUI ensures user-friendliness by providing a graphical interface to perform schema management operations, such as adding support for new applications. For further information, refer to *CUDB LDAP Schema Management Graphical User Interface*, Reference [17].



7 Configuration

Configuration of the system must take both the node level and the system level into account. Both levels are described in the following sections.

7.1 CUDB Node Configuration

A CUDB node consists of a maximum of 36 blades per cabinet or 36 VMs per VNF.

There are two types of CUDB nodes:

- Nodes with a PLDB.
- Nodes without a PLDB, only allowed if the Flexible PL Deployment function is used.

The components of nodes **with** PLDB are as follows:

- OAM servers: 2 blades or VMs.
- PLDB
 - Minimum size:
 - 4 blades for BSP 8100 (GEP3).
 - 2 blades for BSP 8100 (GEP5).
 - 2 VMs for virtual deployments.
 - Maximum size: 16 blades or VMs.
- DS units:
 - Minimum number of DS units: 1 DS unit, meaning 2 DS blades or VMs.
 - Maximum number of DS units: 16 DS units, meaning 32 DS blades or VMs.

The components of nodes **without** PLDB are as follows:

- OAM servers: 2 blades or VMs.
- DS units:
 - Minimum number of DS units: 2 DS units, meaning 4 DS blades or VMs.
 - Maximum number of DS units: 17 DS units, meaning 34 DS blades or VMs.

PLDB can be configured with any *even* number of servers between the minimum and maximum values. A larger PLDB size means that a lower number of DS units fits in nodes with a PLDB.

Note: For CUDB systems deployed on native BSP 8100 (GEP5), when the “BSP Capturing Unit Option” is used, the subrack capacity is decreased by two blades. For more information, refer to the BSP 8100 CPI.

7.2 CUDB System Configuration

CUDB supports several system configurations. Geographical redundancy can be set to either 1+1 or 1+1+1 modes. The maximum number of sites in a deployment is six. Figure 8 and Figure 9 show examples for all deployment types.

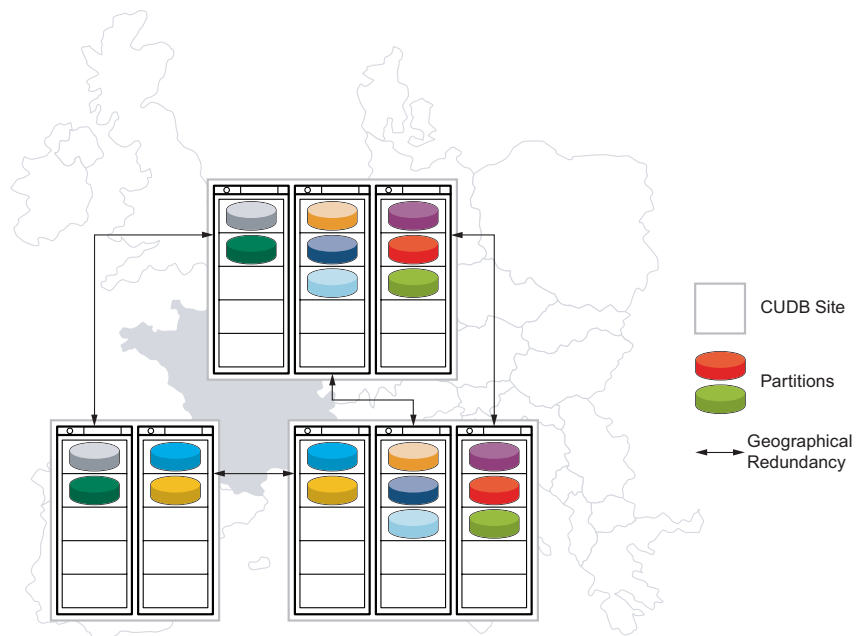


Figure 8 CUDB Deployment with Double Geographical (1+1) Redundancy

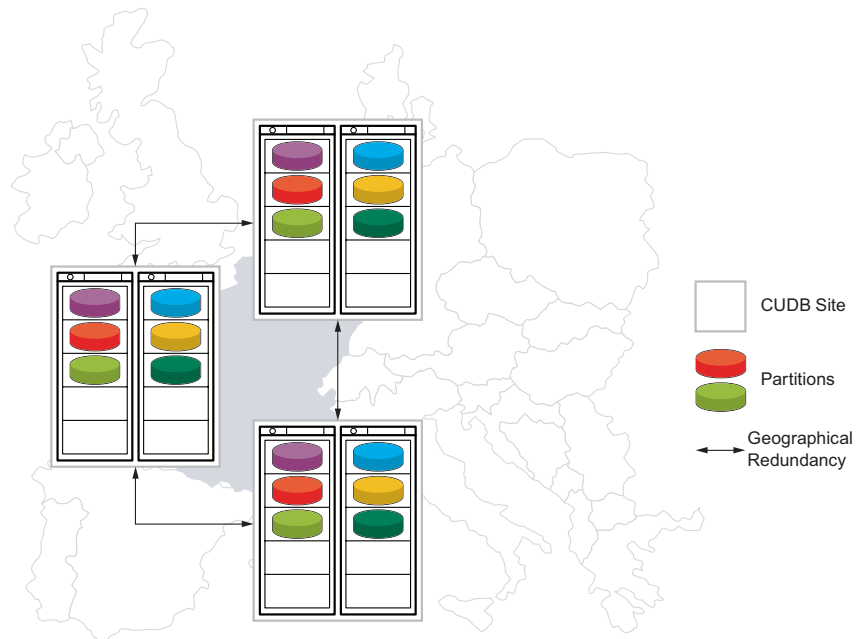


Figure 9 CUDB Deployment with Triple Geographical (1+1+1) Redundancy

During installation, CUDB can be deployed on either all-IPv4 or all-IPv6 network configurations.

Actual CUDB configurations must be set up according to the operator requirements after network planning. For further details, refer to *CUDB Deployment Guide*, Reference [18].





8 Characteristics

The following section highlights the most relevant characteristics of CUDB.

8.1 Availability and Reliability

The availability of CUDB is 99,999% at system level. The system is deployed with geographical redundancy; therefore, the service is not interrupted, even if a site is down. This is because the other CUDB sites keep processing traffic, thereby providing non-stop, uninterrupted data access.

CUDB architecture is also transparent from the data access point of view. Each CUDB node with a replica of PLDB provides access to the entire subscriber database, regardless of the data distribution across CUDB nodes. Data transparency is also supported by geographical redundancy. CUDB supports double or triple geographical redundancy (for more information, see Section 3.2.12 on page 14).

Every element is replicated at least twice inside a CUDB node. In this way, single points of failure are avoided, and maintenance operations, such as replacing a failing switch or network card, can be performed without outage. Therefore, CUDB is continuously available during software and hardware upgrades.

Data is replicated between the DS units of a partition and inside a DS cluster. Each DS cluster holds two copies of the data stored on the partition.

Periodical backups can be ordered for the entire system.

8.2 Scalability

CUDB has a high-performance data layer that provides linear scalability both in capacity and performance, with bounded access latency. A CUDB system can increase the amount of stored data by adding further DSGs. This operation does not require downtime, and does not affect traffic. DSGs can be added by adding more server blades to the existing CUDB nodes, or by setting up additional CUDB nodes within the system.





Glossary

AAA

Authentication, Authorization and Accounting

ACL

Access Control List

AMC

Automatic Mastership Change

AuC

Authentication Center

BE

Back End

BLOB

Binary Large Object

BSP

Blade Server Platform

CLI

Command Line Interface

CUDB

Ericsson Centralized User Database

DIT

Directory Information Tree

DN

Distinguished Name

DS

Data Store

DSG

DS Unit Group

EIR

Equipment Identity Register

ENUM

E.164 Number Mapping

EPC

Evolved Packet Core

FE

Front End

FM

Fault Management

GEP3

Generic Ericsson Processor version 3

GEP5

Generic Ericsson Processor version 5

GUI

Graphical User Interface

HLR

Home Location Register

HSS

Home Subscriber Server

HTTP

Hypertext Transfer Protocol

HTTPS

Hypertext Transfer Protocol Secure

IETF

Internet Engineering Task Force

IMS

IP Multimedia Subsystem

IoT

Internet of Things

JMX

Java Management Extension

LDAP

Lightweight Directory Access Protocol

LDAPS

LDAP over SSL

M2M

Machine-to-Machine



MNP

Mobile Number Portability

NETCONF

Network Configuration

NMS

Network Management System

OAM

Operation and Maintenance

PG

Provisioning Gateway

PL

Processing Layer

PLDB

Processing Layer Database

PM

Performance Management

SAPC

Service-Aware Policy Controller

SASL

Simple Authentication and Security Layer

SFTP

Secure File Transfer Protocol

SLF

Subscription Locator Function

SNMP

Simple Network Management Protocol

SOAP

Simple Object Access Protocol

SSH

Secure Shell

SSL

Secure Socket Layer

TLS

Transport Layer Security

UDC

User Data Consolidation

UPG

User Profile Gateway

VNF

Virtualized Network Function



Reference List

CUDB Documents

- [1] *CUDB LDAP Interwork Description*, 1/15519-HDA 104 03/10
- [2] *CUDB LDAP Data Access*, 5/15534-HDA 104 03/10
- [3] *CUDB Backup and Restore Procedures*, 1/1553-CNH 160 6130/10
- [4] *CUDB Import and Export Procedures*, 6/1553-HDA 104 03/10
- [5] *CUDB Notifications*, 9/15534-HDA 104 03/10
- [6] *CUDB Multiple Geographical Areas*, 13/15534-HDA 104 03/10
- [7] *CUDB Binary Large Object Attributes Management*, 2/15534-CRH 109 0494/10
- [8] *CUDB Subscription Reallocation*, 12/15534-HDA 104 03/10
- [9] *CUDB High Availability*, 7/15534-HDA 104 03/10
- [10] *CUDB Consistency Check*, 17/1553-CSH 109 067/10
- [11] *CUDB LDAP Data Views*, 15/15534-HDA 104 03/10
- [12] *CUDB Security and Privacy Management*, 8/1553-HDA 104 03/10
- [13] *CUDB SOAP Interwork Description*, 3/15519-HDA 104 03/9
- [14] *CUDB Node Fault Management Configuration Guide*, 3/1553-CSH 109 067/10
- [15] *CUDB Performance Guide*, 4/1553-HDA 104 03/10
- [16] *CUDB Node Logging Events*, 4/1553-CSH 109 067/10
- [17] *CUDB LDAP Schema Management Graphical User Interface*, 1/1553-CNH 160 6161/10
- [18] *CUDB Deployment Guide*, 2/1553-HDA 104 03/10
- [19] *CUDB LDAP Interwork Description*, 1/15519-HDA 104 03/10
- [20] *CUDB Glossary of Terms and Acronyms*, 0033-HDA 104 03/10

Other Documents and Online References



- [21] *Lightweight Directory Access Protocol (LDAP)*. RFC 4510
<http://www.rfc-editor.org/rfc/rfc4510.txt>
- [22] *The Secure Shell (SSH) Authentication Protocol*. RFC 4252
<http://www.rfc-editor.org/rfc/rfc4252.txt>
- [23] *NETCONF Configuration Protocol*. RFC 4741 <http://www.rfc-editor.org/rfc/rfc4741.txt>
- [24] *Telecommunication Management, Performance Measurement: eXtensible Markup Language (XML) File Format Definition*. 3GPP TS 32.435
<http://www.3gpp.org/ftp/Specs/html-info/32435.htm>