

Function Specification Dynamic Activation Execution Environment

Ericsson Dynamic Activation 1

FUNCTION SPECIFICATION

Copyright

© Ericsson AB 2017. 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	Target Groups	1
1.3	Typographic Conventions	1
1.4	Prerequisites	1
2	Application Functions	2
2.1	Northbound Interfaces	2
2.2	Authentication	6
2.3	License Management	8
2.4	Event Service	10
2.5	Logging Service	11
2.6	Southbound Connectivity	14
2.7	Asynchronous Request Handler and Scheduling	14
2.8	Resilient Activation	18
2.9	Graphical User Interface	23
2.10	Batch Handler	26
2.11	Customization	28
3	System Characteristics	29
3.1	Deployment	29
3.2	Operation and Maintenance	36
3.3	Expansion of Dynamic Activation	43
4	Synchronization between Clusters	43
4.1	Synchronization of License Counters	44
4.2	Synchronization of Configuration	44
5	Limitations	45
6	Appendix A - Licenses Deployed on Dynamic Activation	46
	Reference List	49





1 Introduction

This section contains information about the prerequisites, purpose, scope, and target group for the document. This section also contains explanations of typographic conventions used in this document.

1.1 Purpose and Scope

The purpose of this document is to give detailed description about the common functions in Ericsson™ Dynamic Activation (EDA) execution environment and to provide an entry of related documentation. Descriptions of functional behavior for the features hosted by Dynamic Activation can be found in the respective Function Specification.

For the full list and description of Dynamic Activation documents, see *Library Overview*, Reference [1].

1.2 Target Groups

The target groups for this document are as follows:

- System Administrator
- Application Administrator
- Any other personnel who intend to learn about Dynamic Activation

For more information about different target groups, see *Library Overview*, Reference [1].

1.3 Typographic Conventions

Typographic conventions are described in the document *Library Overview*, Reference [1].

1.4 Prerequisites

It is assumed that the readers have basic product knowledge of Dynamic Activation. The document recommended for such information is *Product Overview*, Reference [2].

2 Application Functions

The execution environment of the Dynamic Activation provides common functions to support the **Resource Activation** and **Resource Configuration** services.

The business logics of both resource activation and resource configuration services are runtime deployable Java™ code that provides specific business features. For details of a feature, refer to the *Function Specification Resource Activation*, Reference [3] and *Function Specification Resource Configuration*, Reference [4].

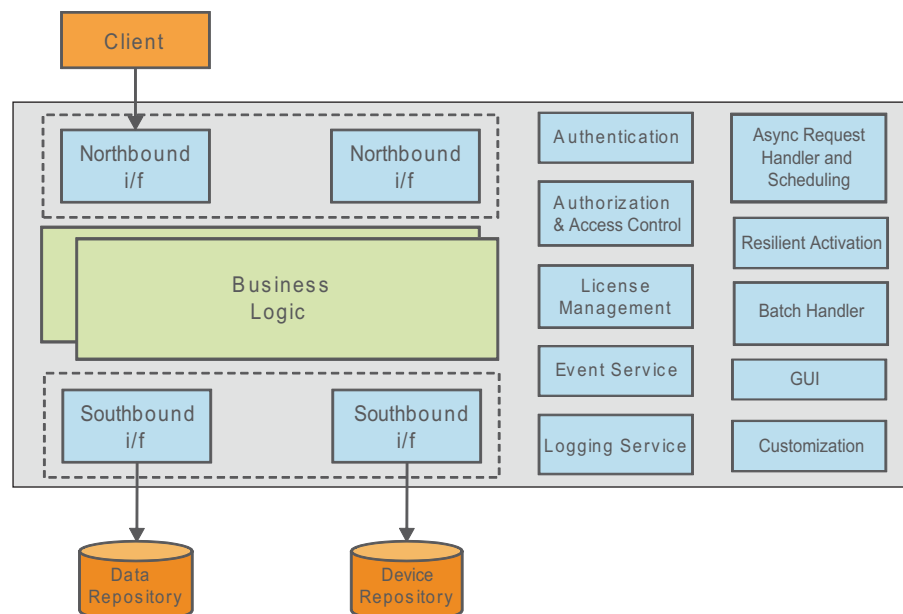


Figure 1 Dynamic Activation Logical Architecture

As shown in Figure 1, Dynamic Activation contains a number of conceptual components. Those components provides the common function of the platform, such as access control and routing of requests to the deployed Business Logic Modules. The common function also handles threading and queuing of requests. This function works in the background.

The major functional components are described in the coming sections.

2.1 Northbound Interfaces

The northbound interfaces handle the connectivity with northbound systems and internally takes the responsibility to convert an inbound interface to the internal data representation supported by the business logic modules.



The default inbound interfaces are described in *Function Specification Resource Activation*, Reference [3] and *Function Specification Resource Configuration*, Reference [4].

In case the northbound system prefers another interface than the default ones, it is possible to tailor a customer solution as a service.

2.1.1

CAI3G

Customer Administration Interface Third Generation (CAI3G) is aimed to provide a simple, up-to-date, and unified provisioning interface for NEs in the telecommunication or other Information Technology (IT) networks. The physical implementation of the networks, protocols, and interactions among NEs is hidden.

Attention!

The Dynamic Activation generated CAI3G XML format, and namespace definition can change in-between releases, even though the document content is exactly the same.

CAI3G 1.2 is based on the Simple Object Access Protocol (SOAP) 1.1, and contains five levels in its protocol stack as shown in Figure 2.

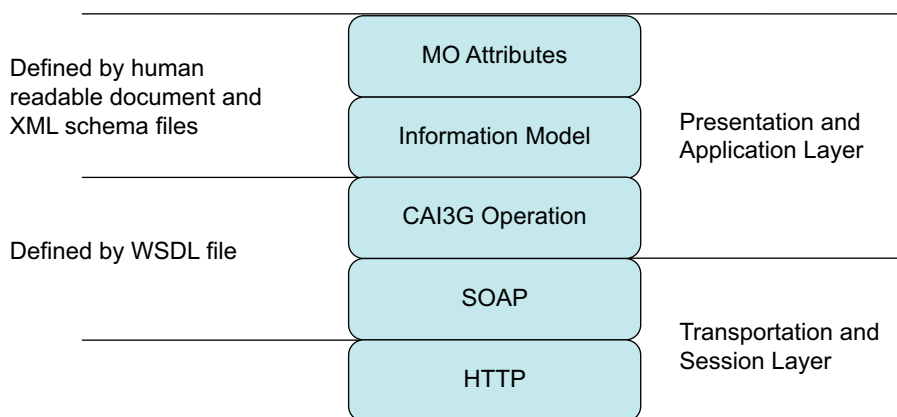


Figure 2 CAI3G Communication Protocol Stack

Any application using CAI3G can use the SOAP requests directly. Though, since SOAP creates some overhead information, it is recommended to use the SOAP Third Party Products (3PP) client for this purpose. By using a SOAP client together with the CAI3G Web Services Description Language (WSDL) file, that comes with a business logic, the application obtains a simple Application Programming Interface (API). The actual SOAP messages are hidden by the SOAP client. CAI3G traffic can be protected by using HTTP over SSL/TLS.

The main features of CAI3G are:

- SOAP message-based presentation and session layer
- Multiple concurrent sessions (maximum number of concurrent sessions are 25000 per system)
- Programming language and platform independency
- Possibility to work in a secure environment with a firewall
- A wide variety of 3PP support
- The supported character set in a CAI3G interface is UTF-8.
- Uses Web Services Security (WS-Security) as authentication mechanism while CAI3G also supports login-logout authentication.

WS-Security is an extension to SOAP to apply security to Web services.

An example of a CAI3G message flow is shown in Figure 3.

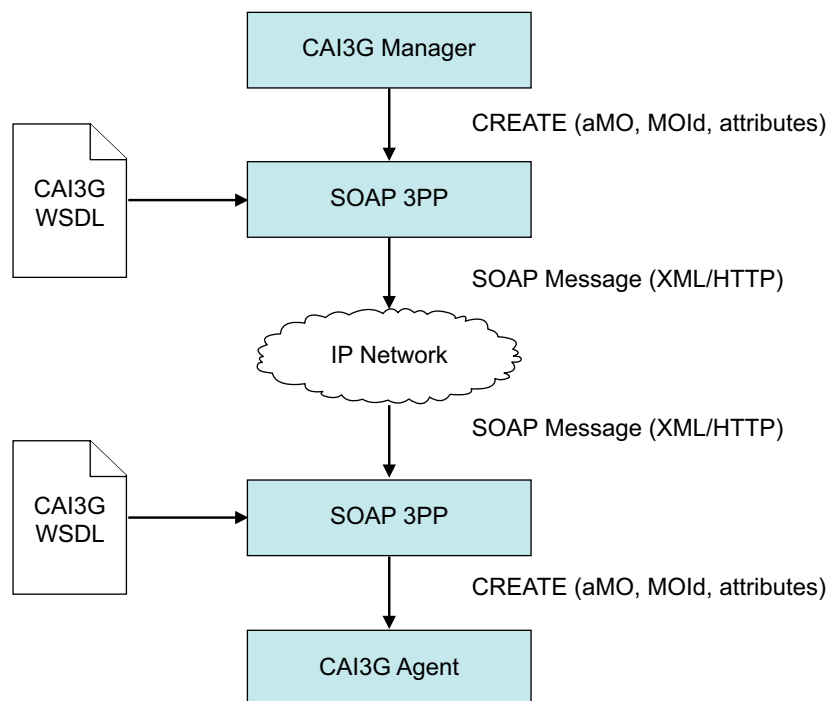


Figure 3 CAI3G 1.2 Message Flow Example

For more information, see *Generic CAI3G Interface 1.2*, Reference [5].

2.1.2 Async CAI3G

The Asynchronous Customer Administration Interface Third Generation (Async CAI3G) defines an asynchronous Web Service interface based on CAI3G.



For information about CAI3G interface, refer to *Generic CAI3G Interface 1.2*, Reference [5]. Figure 4 shows the Async CAI3G Protocol Stack.

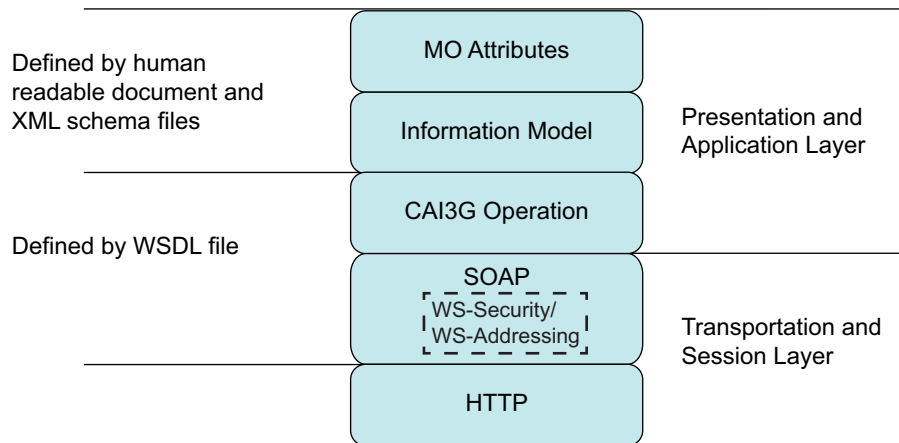


Figure 4 Async CAI3G Protocol Stack

Attention!

The Dynamic Activation generated Async CAI3G XML format, and namespace definition can change in-between releases, even though the document content is exactly the same.

Compared to CAI3G, Async CAI3G:

- Is an asynchronous interface while CAI3G is synchronous interface.
- Uses Web Services Addressing (WS-Addressing) to correlate request and notification (the execution result).

WS-Addressing is a standardized way of including message routing data within SOAP headers.

- Introduces extra optional parameters for asynchronous purpose.
- Uses WS-Security as authentication mechanism while Async CAI3G also supports login-logout authentication.
- Does not support the `Search` operation in the current release.
- The supported character set in an asynchronous CAI3G interface is UTF-8.

For more information, refer to *Asynchronous CAI3G Interface Specification 1.2*, Reference [6].



2.2 Authentication

Access to Dynamic Activation can be performed via machine to machine interfaces as well as the human to machine interface. Machine to machine interface requires a provisioning client and human to machine interface requires a GUI user.

Administration and authentication of provisioning clients and GUI users are separated.

For security reasons, use a strong password, that is, password with an adequate combination of length, complexity, and unpredictability is enforced, and the rules are configurable. For provisioning clients and GUI users, two different sets of password rules are applied independently.

2.2.1 Machine to Machine Interface

Client authentication is made when provisioning clients accesses the inbound interfaces, either provisioning interface, or REST API such as for **Replayer** and **Device Management**.

The provisioning client credentials are encrypted and stored in the system.

Provisioning clients are created and managed in the **Access Control** of the **Operation and Management** GUI. When creating a new password for a provisioning client, a set of criteria needs to be fulfilled. For more information about authentication and password restrictions, refer to *User Guide for Resource Activation*, Reference [7].

2.2.2 Human to Machine Interface

GUI user authentication is made when user accesses the graphical interfaces.

The human to machine interface authentication is developed following the OAuth2 and OpenID connect standards. Dynamic Activation provides an authentication solution of the shelf (internal authentication) but also allows integration with external providers for authentication (external authentication). However, internal authentication and external authentication cannot be used at the same time.

Regardless of internal or external authentication, a GUI user can either have a System User or a System Administrator role. A System User has access to all GUIs except the User Management GUI, and a System Administrator has access to all GUIs.

The sensitive user data is encrypted at both transit and rest.

For internal authentication:



- A GUI user is enforced to change password as first login and thereafter at a system configurable interval.
- Users can change their own password, view and edit their own profile.
- It is possible to configure strong password rules for the system. For more information on how to set password rules for GUI users, refer to:
 - *System Administrators Guide for Native Deployment*, Reference [10]
 - *System Administrators Guide for Virtual and Cloud Deployment*, Reference [22]

For external authentication integration:

- Users are managed outside Dynamic Activation. For example, user management GUI in Dynamic Activation is not applicable.
- Users can view their own profile in the GUI but cannot modify the content.
- Guideline is provided on how to configure external OpenID Connect Provider. For details, refer to:
 - *System Administrators Guide for Native Deployment*, Reference [10]
 - *System Administrators Guide for Virtual and Cloud Deployment*, Reference [22]

2.2.3

Access Control

Dynamic Activation provides authorization framework for defining user access to the application function as well as provisioning clients for the underlying network resources.

A default `System Administrator` GUI user is created during the installation. The default user can create and manage other GUI users.

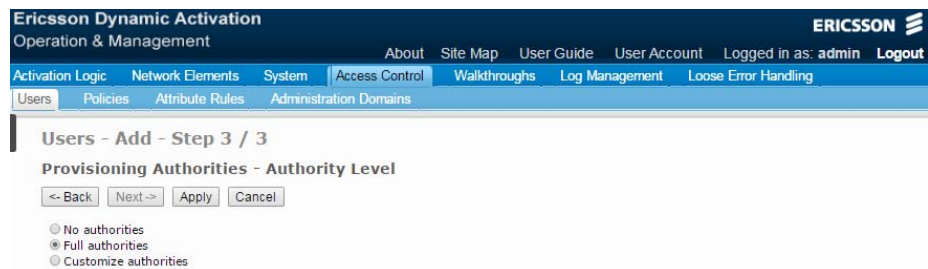
A GUI user has access to the **Operation and Management > Access Control** GUI, at where to create and manage provisioning users, and to grant provisioning users Provisioning Access Control (PAC) authority for provisioning users.

A provisioning client is granted PAC authority on two levels.

- For Resource Activation solution, refer to *Function Specification Resource Activation*, Reference [3] for more information on PAC authorities.
- For Resource Configuration solution, when creating a provisioning client for **Device Management** function:
 - 1 Set the first level PAC to `No authorities`.



- 2 Set the second level PAC to Full authorities.



2.3 License Management

This section contains information on Dynamic Activation licenses architecture.

2.3.1 Licenses

There are two types of license that are used to activate the features in a Dynamic Activation system. Standard product license and product customization license. A standard product license . For information about the available standard product licenses

- Standard product license – Activates either a basic package or a value package, which is a collection of several features bound together.

Standard product licenses are obtained based on contractual content.

For information about the available standard product licenses, see Section 6 on page 46.

- Product customization license – Activates a specific product customization.

Product customization license is provided together with the product customization deliverable.



2.3.2 Architecture

Dynamic Activation provides a generic license management service for the business logic, and a license agent process for application level license enforcement.

This enables the business logic to enforce which features to be used, and on which resource instances the application is entitled to run.

Dynamic Activation uses its own license server to provide the license service.

As shown in Figure 5, the license service provides uniform access method for business logics. Licenses for standard product features are managed by a license server while product customization licenses are stored separately.

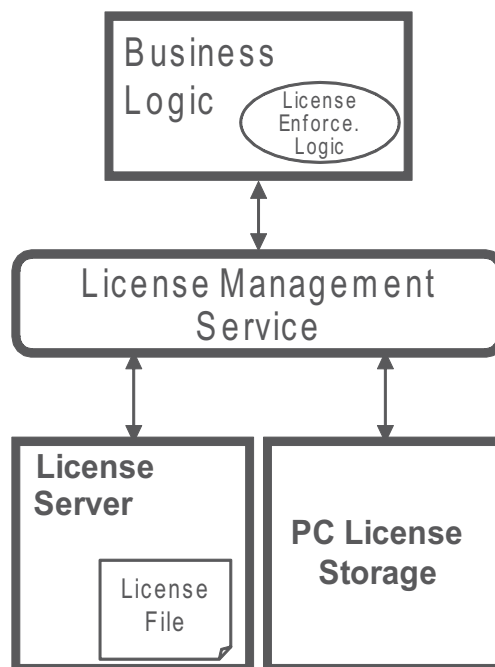


Figure 5 Business Logic License Management Architecture

As shown in Figure 6, the license agent process provides license enforcement on how many resources instance one deployment is comprised of (in terms of number of server instances and CPU resources) and where the application modules are entitled to run.

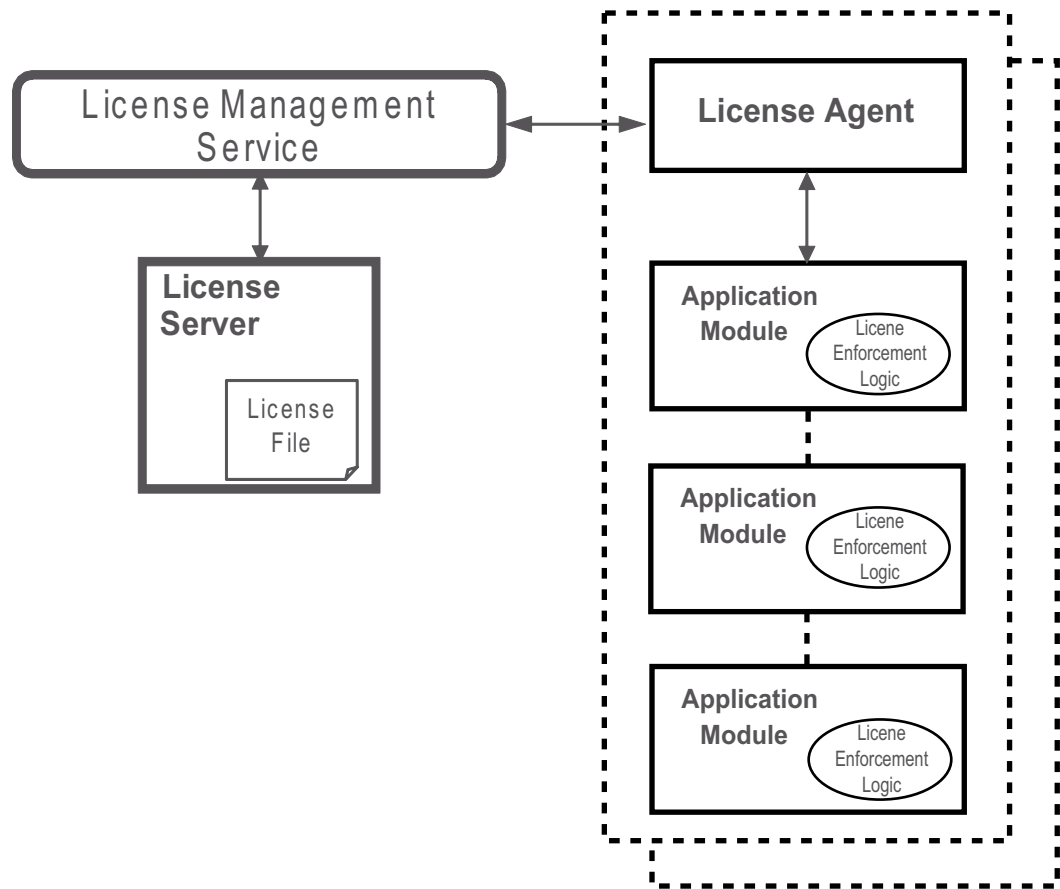


Figure 6 Application Module License Management Architecture

The license information can be viewed in the GUI in a deployed system.

License capacity alerts:

- A yellow indicator is shown when the use is above the warning threshold of the contractual capacity. A notification is sent to pre-defined receivers and an alarm can be triggered.
- A red indicator is shown, together with the number of remaining days when the enforcement logic takes effect, when the use exceeds the contractual capacity. A notification is sent to pre-defined receivers and an alarm can be triggered.

2.4 Event Service

All components in Dynamic Activation, including the business logics it hosts, generate events. The Event Service listens to all the events. It judges the events against the pre-configured alarm criteria. When an event matches one of the alarm criteria, the Event Service converts the message and sends it to the alarm management agent.



For more information about the alarm management agent, see Section 3.2.1 on page 37.

2.5 Logging Service

Dynamic Activation uses a flexible logging mechanism. This allows a multitude of different events to be logged, such as errors and warnings, as well as tracing information and debug printouts. The following logs are produced:

- Application log

The application log contains information about specific events in the Dynamic Activation system, such as deployment of data views, successful logons, and other events that are of interest to the administrative user. This log is also useful as a debug log when developing new or updating processing logic to be deployed.

- Access log

There are two separate files for successful and unsuccessful provisioning requests. They can be used as a base to create statistics or a historical log of incoming requests.

- Partially succeeded log

The partially succeeded log contains information about partially executed provisioning commands towards Centralized User Database (CUDB). It provides input for repair actions.

- Audit log

The audit log contains information about failed authorization attempts on the provisioning interfaces.

- Processing log

The processing log records provisioning transaction going through the system. The information is essential for provisioning statistics as well as trouble shooting of provisioning errors, or other purpose such as for **Replayer**. For detailed information, see Section 2.5.1 on page 11.

2.5.1 Processing Log

Figure 7 illustrates the information that can be logged in the processing log.

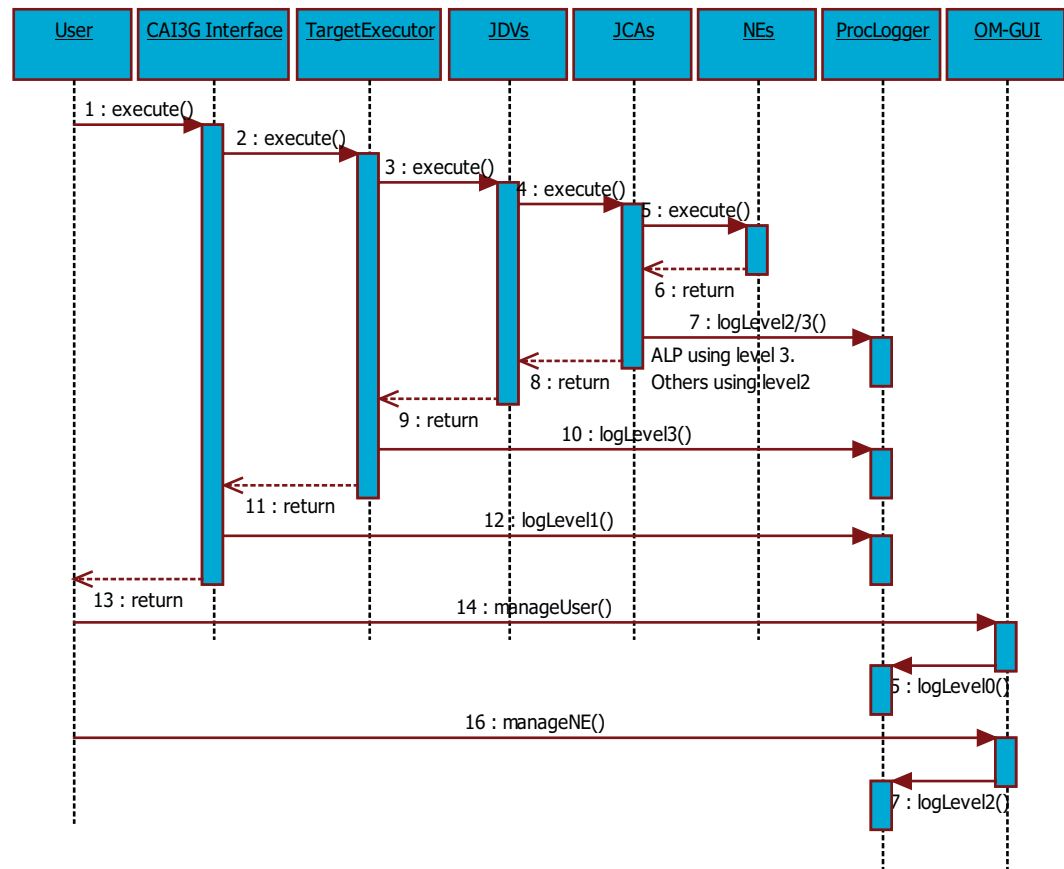


Figure 7 Processing Log Generation Example

By configuring the log level on the target system, the user decides which information is kept in the log files.

Log contents are classified in four levels:

- Level 0: User management activities via Operation and Maintenance (O&M) GUI.
- Level 1: northbound requests and their responses.
- Level 2: Network Element (NE) management activities in O&M GUI as well as southbound requests from Dynamic Activation to external NEs.

This is the default log level.

If **Replayer** function is used, Level 2 is the minimum level.

- Level 3: requests exchanged between internal components.

When log level is set to 2 or 3, filters on southbound link type level can be set.

Log items generated by provisioning components are stored in both virtual database system and flat processing log files, see Figure 8:

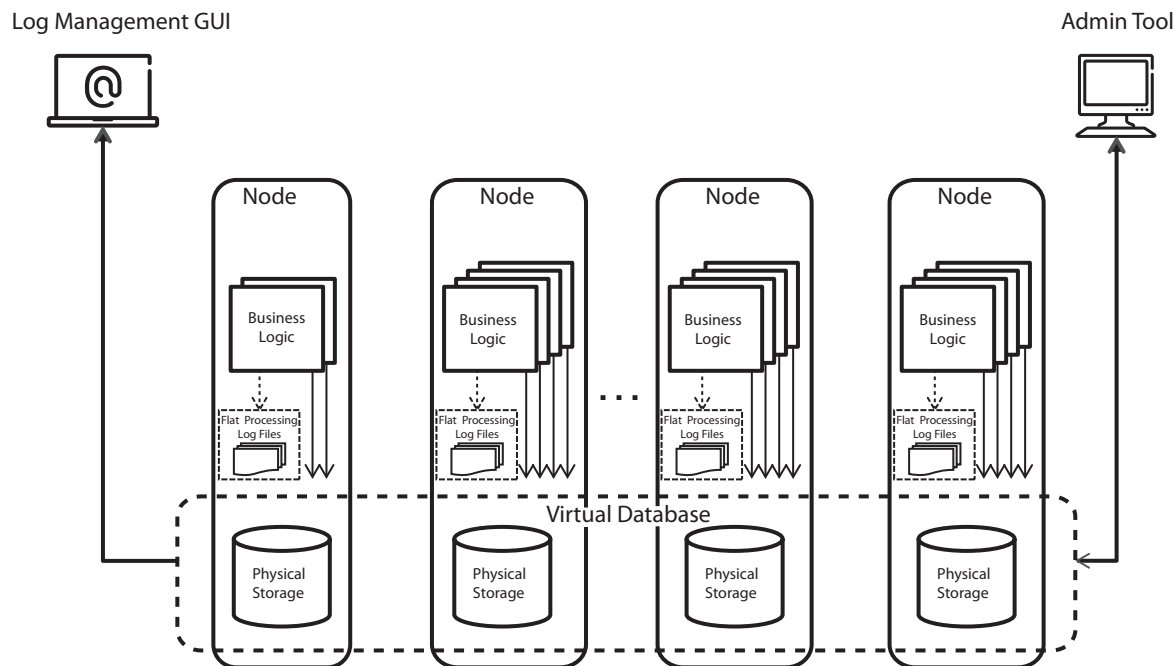


Figure 8 Log Management Overview

- The virtual database system uses the physical storage capability provided by all nodes. It ensures availability of log data in case of single node failure.

A GUI is provided to facilitate easy trouble shooting of provisioning errors. For detailed information, refer to section **Log Management** in *User Guide for Resource Activation*, Reference [7].

An administration tool is provided for export and import of log items from the database. For detailed information, refer to section **Processing Log Admin Tool** in *System Administrators Guide for Native Deployment*, Reference [10] or *System Administrators Guide for Virtual and Cloud Deployment*, Reference [22].

A safety script cleans up old logs when storage space exceeds a specified limit. The limit is configurable with the default value set to 90%. For detailed information of configuring the limit, refer to section **Removal of Old Processing Logs** in *System Administrators Guide for Native Deployment*, Reference [10] or *System Administrators Guide for Virtual and Cloud Deployment*, Reference [22].

- The flat processing log file stores the runtime operations/request/transaction in a flat-formatted file on each nodes. It provides a way to monitor logs in real-time in order to do runtime trace and analysis rather than retrieve the logs from database. For detailed information about Flat Processing Log files, refer to *System Administrators Guide for Native Deployment*, Reference [10] or *System Administrators Guide for Virtual and Cloud Deployment*, Reference [22]

2.6 Southbound Connectivity

Dynamic Activation provides unified southbound connection management solution that can be configured in runtime to serve business logics.

The connectivity of default outbound interfaces is described in *Function Specification Resource Activation*, Reference [3] and *Function Specification Resource Configuration*, Reference [4].

2.7 Asynchronous Request Handler and Scheduling

Asynchronous Request Handler and Scheduling provides Dynamic Activation with a web-service based asynchronous interface that saves the business system from waiting for the requests completion.

The ability to store Customer Service Orders (CSO) for later execution is valuable in service campaign scenarios, which enables the operator to make a service available for end users at a certain time.

As a prerequisite of the Asynchronous Request Handler function, Asynchronous CAI3G interface must be used to receive asynchronous requests from the business system. Therefore in customize adaptation cases, async WSDL and schema files must be created according to *Asynchronous CAI3G Interface Specification 1.2*, Reference [6].

2.7.1 Asynchronous Request Process

After receiving an asynchronous request from the BSS, Dynamic Activation sends successful acknowledgment to BSS immediately, and notifies the business system when the execution is completed.

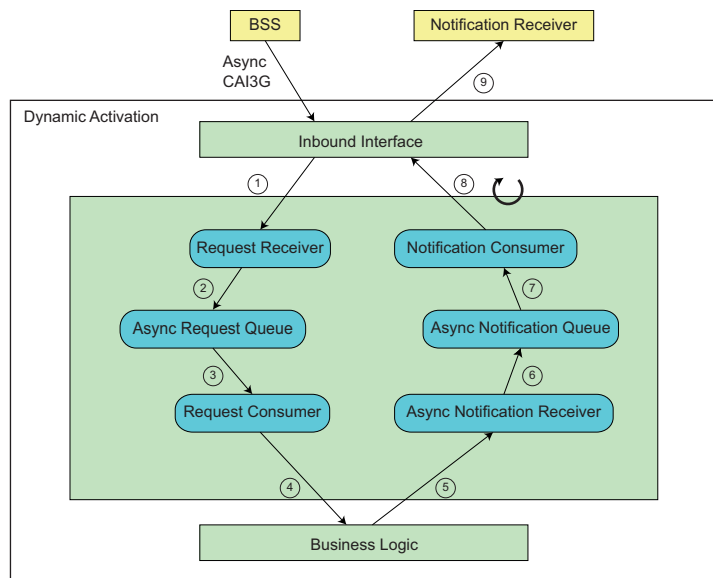


Figure 9 Asynchronous Request Process Flow

The processes of Asynchronous Request Handler are as follows:

1. Inbound interface sends the asynchronous request to Request Receiver when receiving an Asynchronous request from Business Support System (BSS).
2. The Request Receiver stores the command in the Asynchronous Request Queue, and sends a successful acknowledgment to BSS.
3. If the request is ready for execution, the Request Consumer retrieves this request from the queue.
4. The business logic executes the request, and provisions the corresponding Network Elements (NE).
5. The business logic sends the provisioning result to Asynchronous Notification Receiver.
6. Asynchronous Notification Receiver stores the provisioning result into Asynchronous Notification Queue.
7. Notification Consumer retrieves the notification from the queue to execute.
8. Notification Consumer sends the notification to Inbound Interface, or performs notification retry for a final notification that is not acknowledged by the Notification Receiver.
9. The inbound interface sends the provision result to the Notification Receiver. The Notification Receiver sends an acknowledgement to Dynamic Activation if the result is accepted.

For information on notification retry, see Section 2.7.7 on page 17.



If Resilient Activation is used, Notification Receiver receives both intermediate and final notification. For information about Resilient Activation, see Section 2.8 on page 18.

2.7.2 Scheduling

Scheduling enables Dynamic Activation to store CSOs for later execution based on the scheduled fire time

The scheduled fire time is provided in the CSOs and the CSOs are stored in the Asynchronous Request Queue until it is the time for execution.

2.7.3 Asynchronous CAI3G Priority Support

The Asynchronous CAI3G request priority can be defined in the following two ways:

- Define the priority value in SOAP headers of the Asynchronous CAI3G Request.
- Configure the priority rule as MOType and specified Xpath in the **Priority Configuration** GUI.

If the priority has been configured in both ways, the priority value defined in the Asynchronous CAI3G Request will be used.

2.7.4 Asynchronous Request States

An asynchronous request has one of the following states throughout its lifecycle:

- Pending – The request is stored in the Asynchronous Request Queue.
- Executing – The request is retrieved by the Request Consumer and being executed by the Business Logic.
- Failed – The request execution is failed. A final notification of the failure is sent to the Asynchronous Notification Receiver.
- Successful – The request execution is successful. A successful final notification is sent to the Asynchronous Notification Receiver.

In the **Asynchronous Request Management** GUI, users can view the information of requests that have not been executed successfully, and delete the ones that have been pending for a long time.

2.7.5 Request Grouping

Requests can be grouped to preserve the order of execution. The grouping can be done in two ways:



- Grouped done by Business System sending explicit group ID in Asynchronous requests
- Grouped automatically if the asynchronous queue contains several CSOs that are related to the same MO ID

Grouped requests are executed in the same order as they were sent from the BSS. Therefore error situation caused by out of order can be eliminated.

See Section 5 on page 45 for a known limitation of the Request Grouping.

2.7.6 Asynchronous Queue Capacity Control

Asynchronous Queue Capacity Control monitors and manages the capacity of asynchronous requests in Asynchronous Request Handler. The maximum queue size and capacity check interval can be configured by operator. For detailed configuration, refer to *System Administrators Guide for Native Deployment*, Reference [10] or *System Administrators Guide for Virtual and Cloud Deployment*, Reference [22].

When the number of asynchronous requests in Asynchronous Queue exceeds the maximum queue size, Asynchronous Request Receiver stops receiving asynchronous request, and sends an event to Ericsson Simple Network Management Protocol (SNMP) Agent (ESA).

When the number of asynchronous requests in Asynchronous Queue drop below 85% of the maximum queue size, Asynchronous Request Receiver starts receiving asynchronous request.

Asynchronous Request Receiver can be manually disabled by operator in the **Asynchronous Control** GUI. In this case, no asynchronous request is received until operator manually enables the Asynchronous Request Receiver again.

2.7.7 Notification Retry

If a final notification fails to send due to temporary network issues or Notification Receiver errors, Notification Consumer is able to retry the final notification a number of times during a configurable Max Retry Time.

- If there are multiple Notification Receivers, error of one Notification Receiver does not impact notifications send to other Notification Receivers.
- Toward one Notification Receiver, Notification Consumer retries only one final notification at one time.

Any other final notifications toward the same Notification Receive are waiting in the Asynchronous Notification Queue until Notification Retry on the current final notification ends.

- If the Notification Receiver or network issues restored during notification retry, Notification Consumer still waits until current retry interval ends before performing the next retry.
- If the Max Retry Time is reached, but the current final notification still fails to send, Asynchronous Request Handler will send an event to ESA. And the failed asynchronous notification will be recorded in Processing log. The Notification Consumer will move on to send the next notification in the queue.
- For Notification Retry, the following behavior can be defined via updating of system properties:

– ARHNOTIFICATION_RETRY_FACTOR

Retry Factor increases the retry time interval. The first retry happens after the retry time interval is reached. For the second retry and onward, the delay duration is increased by the Retry Factor, expressed as:

Present Retry Interval= Retry Factor ^{retry times}

– ARHNOTIFICATION_MAX_RETRY_TIME

Max Retry Time defines the total retry duration of sending asynchronous notification to Notification Receiver. The retry attempts are determined by the retry factor and Max Retry time, see the example in Table 1.

For how to configure system properties, refer to *System Administrators Guide for Native Deployment*, Reference [10] or *System Administrators Guide for Virtual and Cloud Deployment*, Reference [22].

Table 1 Example of Retry Behavior for Asynchronous Notification

Asynchronous Notification Retry Configuration: Retry Factor: 2	Present Retry Interval	Wait time of every retry attempt
1 st retry delay	2 s	2 s
2 nd retry delay	4 s	2+4=6 s
3 rd retry delay	8s	2+4+8=14 s
...
n th retry delay	2 ⁿ s	t = 2+4+8+...2 ⁿ s (t <=ARHNOTIFICATION_MAX_RETRY_TIME)

2.8 Resilient Activation

As prerequisites of Resilient Activation, in the Service Realization layer, Service Model developed by using the **Designer Studio** must be deployed. Such a Service Model:



- Enables asynchronous orchestration and sequencing of sub-requests based on model configuration.
- Requires a SW Advanced license to run on the Dynamic Activation.

Note: If a Java Data View (JDV) Business Logic is deployed in the Service Realization layer, Resilient Activation is not provided.

2.8.1

Resilient Activation Operation

Figure 10 shows an overview of the Resilient Activation provided in Dynamic Activation.

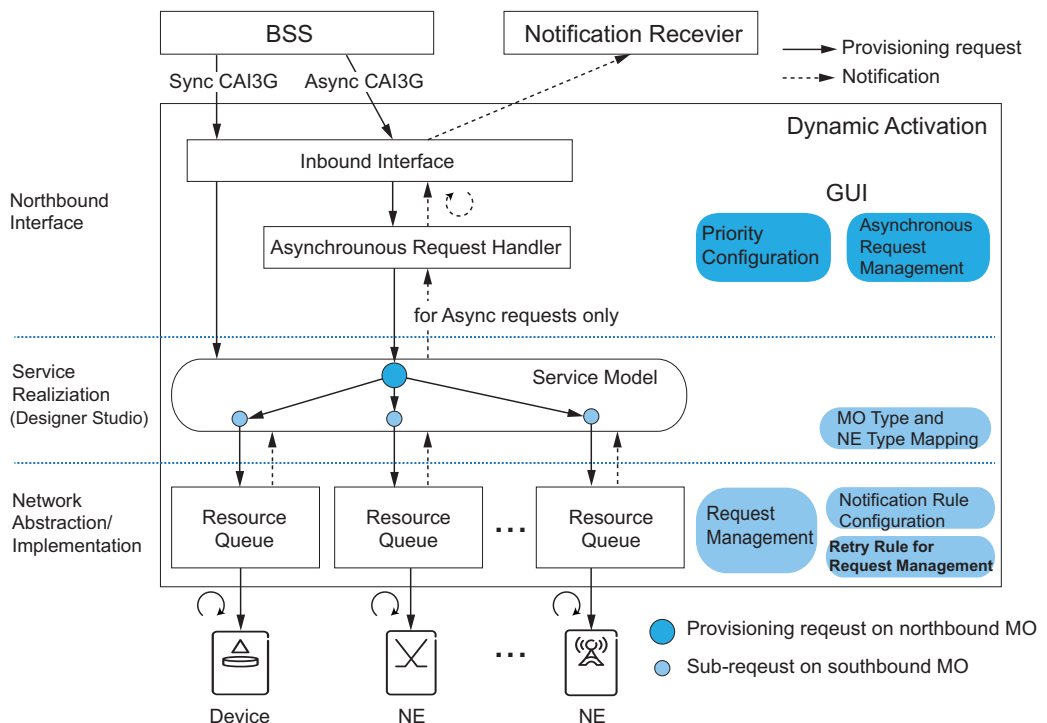


Figure 10 Resilient Activation Overview

Above the Service Realization layer

Above the Service Realization layer (Business Logic), provisioning requests received via the Async and Sync CAI3G interfaces are processed differently:

- Asynchronous provisioning requests are processed by the Asynchronous Request Handler as described in Section 2.7.1 on page 14.

During the asynchronous request execution, the Asynchronous Request Handler sends both final notification and intermediate notification (see Section 2.8.3 on page 22) to Notification Receiver, but only perform Notification Retry (see Section 2.7.7 on page 17) for the final notification.

- Synchronous provisioning requests are sent to the Service Model for execution one by one, bypassing the Asynchronous Request Handler. For each request, Dynamic Activation sends an execution result back to BSS when either the execution is finished, or a timeout is reached for the request.

Users can configure the timeout (up to 300 s) to specify how long BSS can wait for the execution result. For configuration instruction, refer to:

- *System Administrators Guide for Native Deployment*, Reference [10]
- *System Administrators Guide for Virtual and Cloud Deployment*, Reference [22]

Service Realization and lower layers

No matter Async or Sync CAI3G interface is used, as long as a provisioning request reaches the Service Model in Service Realization layer (Business Logic), the request is processed as follows:

1. Dynamic Activation orchestrates the provisioning request to multiple sub-requests based on model configuration.
2. Dynamic Activation consumes the sub-requests for each Resource queue (see Section 2.8.4 on page 22) in the order of priority, and dispatch them to the corresponding Activation Logic component (NE Abstraction/Implementation).
3. Each response from the Activation Logic corresponding to a specific target resource is evaluated against the configured retry-rules and notification-rules, in order to determine what action to be taken for the sub-request:
 - For successful response, the sub-request is removed from the queue and forwarded back to the Service Model.
 - For failed response:
 - If no matching retry rule, the sub-request is removed from the queue, and the failed response is forwarded back to the Service Model.
 - If matching retry rule, the sub-request is rescheduled in the Resource Queue, and to be executed after the configured retry delay.
 - If all retry attempts have been exhausted, the sub-request is marked as expired in the Resource Queue, and needs manual intervention.
 - For asynchronous provisioning request, intermediate notification can be sent on successful-, failed-, and matching retry rule response, depending on the notification rules configuration.



The different states of the sub-requests are described in Section 2.8.5 on page 22.

4. For each successful or failed network response, the response is evaluated towards any conditional dependencies in the model configuration, to determine if the request sequence may progress with the next sub-request in sequence, or if the overall request is completed.
5. Once all sub-requests defined by the model configuration have been processed, Dynamic Activation does one of the following, and any remaining states of the request are removed.
 - For asynchronous provisioning request, Service Model sends a final notification to the Asynchronous Notification Receiver.
 - For synchronous provisioning request, an execution result is sent back to BSS.

2.8.2 Resilient Activation Configuration

As shown in Figure 10, Dynamic Activation provides the following GUIs for Resilient Activation configuration:

- Prioritization of activation tasks based on business needs by configuring priorities for the corresponding request content in the **Priority Configuration** GUI.
- Asynchronous provisioning requests are queued and scheduled in order of priority and timestamp.

Users can monitor the requests queue in the **Asynchronous Request Management** GUI.

- Provisioning requests operating on a northbound interface (MO type) are processed according to the model configuration, which is operating towards one or multiple southbound interfaces (NE types).

For Recourse Activation solution, users can use **MO Type and NE Type Mapping** GUI to route sub-requests to the corresponding Resource Queues.

- Users can use **Notification Rules Configuration** GUI to configure whether to send intermediate notification when a sub-request is performed successfully, failed, or is rescheduled (retried).

Note: The **Notification Rules Configuration** takes no effect on provisioning requests via Sync CAI3G interface.

- Users can configure retry rules for each target resource in the **Retry Rule for Request Management** GUI. Whereas, failed sub-requests can be rescheduled in the Resource Queue.



- Users can view visualized statistics of the Resource Queues, and manage queued/expired sub-requests in the **Request Management** GUI.

For more information about configuration of Resilient Activation, refer to *User Guide for Resource Activation*, Reference [7].

2.8.3 Asynchronous Notification

2.8.3.1 Intermediate notifications

Configurable notifications being sent after matching a notification rule for a specific NE. Based on notification rule configuration, notifications may be sent based on successful-, failed-, and/or matching retry rule response.

Multiple intermediate notifications may be sent throughout the lifecycle of an asynchronous request, depending on notification rule configuration.

For more information on how to configure intermediate notifications, refer to *User Guide for Resource Activation*, Reference [7].

2.8.3.2 Final notifications

Notifications being sent after completing an asynchronous request, either successfully or failed.

2.8.4 Resource Queue

The Resource Queue is a logical representation of target resources towards which the sub-request is operating on via the activation logic.

One Resource Queue is allocated for each active target resource, and sub-requests are only dispatched to the corresponding activation logic component if the target resource is active or available. If a target resource is deactivated for maintenance or unavailable for other reasons, the processing of sub-requests in the corresponding Resource Queue is put on hold.

2.8.5 Sub-request States

Throughout its lifecycle, each provisioning request and sub-request goes through multiple persisted states before sending a final notification (for an asynchronous request) or a execution result (for a synchronous request) back to the client.

A sub-request in an Resource Queue has one of the following states:

Queued	Initial state of sub-request in an Resource Queue, prior to being consumed.
---------------	---



Rescheduled	Pending state of sub-request in an Resource Queue after matching a retry rule after being consumed at least once.
Expired	Pending state of sub-request in an Resource Queue, after exhausting all retry attempts for the matching retry rule.
Successful	Final state of sub-request, which means the sub-request is executed successfully and is removed from the corresponding Resource Queue.
Failed	Final state of sub-request, which means the sub-request failed and is removed from the corresponding Resource Queue. Service Model will receive corresponding error messages of the failure.

2.9 Graphical User Interface

The GUI is the web application that facilitates system and application configuration tasks.

The GUI is accessible through a web browser. Connection to the GUI is protected with HTTPS. Access to the GUI content is guarded by the user's authority.

The GUI supports only characters from the ISO-8859-1 character set, that is:

- The numbers from 0 through 9
- The uppercase and lowercase English alphabet
- The characters used in Western European countries
- Commonly used special characters such as “&”, “-”

Through the GUI an authorized user can perform some or all tasks.

For more information about the GUIs and use instruction, refer to *User Guide for Resource Activation*, Reference [7] and *User Guide for Resource Configuration*, Reference [8].

2.9.1 Common Tasks

Common tasks and tools in both Resource Activation and Resource Configuration are:

Note: A SW Basic license is required if not specified otherwise.

- Monitor of



- **Dashboard:** 36 hours (SW Basic license)/7 days (SW Advanced license) of provisioning performance statistics.
- Configuration of:
 - **Asynchronous Control:** enable/disable asynchronous Request receiver/consumer.
 - **CAI3G DC:** view/manage routing of CAI3G requests towards an external system that supports CAI3G protocol.
 - **Notification Rules:** (SW Advanced license) manage notification rules that trigger intermediate notification towards external system when particular conditions are met.
 - **Operation & Management**
 - **Activation Logic:** view activation logic, edit property
 - **System:**
 - Switch on/off the alarm function
 - Define maximum consecutive unsuccessful logon attempt for a user.
 - Lock/unlock a provisioning client.
 - Configure an e-mail address for notification when license capacity has been exceeded.
 - Synchronize configuration data
 - **Access Control:** manage provisioning clients, assign provisioning authority and restriction.
 - **Priorities:** manage priority rules for northbound requests.
 - **Retry Rule** for processing queue: (SW Advanced license) manage retry rules for processing queue.
 - **Retry Rule** for request management: (SW Advanced license) manage retry rules for request management.
- Management of:
 - **Asynchronous Request:** view/search asynchronous northbound requests, delete pending requests.
 - **Log:** view/search northbound requests/response and its southbound requests/responses.



- **Processing Queue:** (SW Advanced license) view and manage provisioning commands that are executed in parallel for different subscribers.
- **Request:** (SW Advanced license) view request management dashboard, search for and view detail of a request, resend/delete expired sub-requests.
- Other tools:
 - **Loose Error Handling:** manage loose error handling rules for a business logic.
 - **Batch Handler:** manage bulk provisioning job.
 - **User Management:** manage GUI users.

2.9.2 Resource Activation Specific Tasks

Tasks that are specific for Resource Activation are:

- **Network Element:** view/manage network element endpoint, group, routing as well as cluster strategy
- **Cluster Strategy:** (SW Advanced license) view/manage cluster strategy instance
- **MO Type and NE Type Mapping:** (SW Advanced license) manage mapping between MO type and NE type to enable routing sub-requests to the corresponding Resource Queues.

2.9.3 Resource Configuration Specific Tasks

Tasks that are specific for Resource Configuration are:

- **Service Visualization:** view service detail, compare service at different point in time, restore service configuration.
- **Device Management:** add/import devices, manage device, view/edit/restore device configuration, device reconciliation.
- **Feature Model:** import/export/edit feature model.
- **Vendor Template:** create/import/export template, manage template definition.

2.10 Batch Handler

2.10.1 Batch Handler Overview

Batch Handler provides an easy, flexible, and automatic way to process a large number of provisioning requests that are defined by the batch file and the scheme file. Therefore, user can schedule the provisioning of a range of subscribers at a certain time to avoid high peak traffic times in the network. Batch Handler supports CAI3G and synchronous CAI requests.

Batch Handler can simplify manual operation in many user cases, such as:

- Activate or deactivate the subscription data for a range of subscriber numbers.
- Modify the subscription data for a list of subscriber numbers.
- Migrate a large number of subscribers with individual subscription data from one network element to another.

The following figure shows an overview of the Batch Handler function.

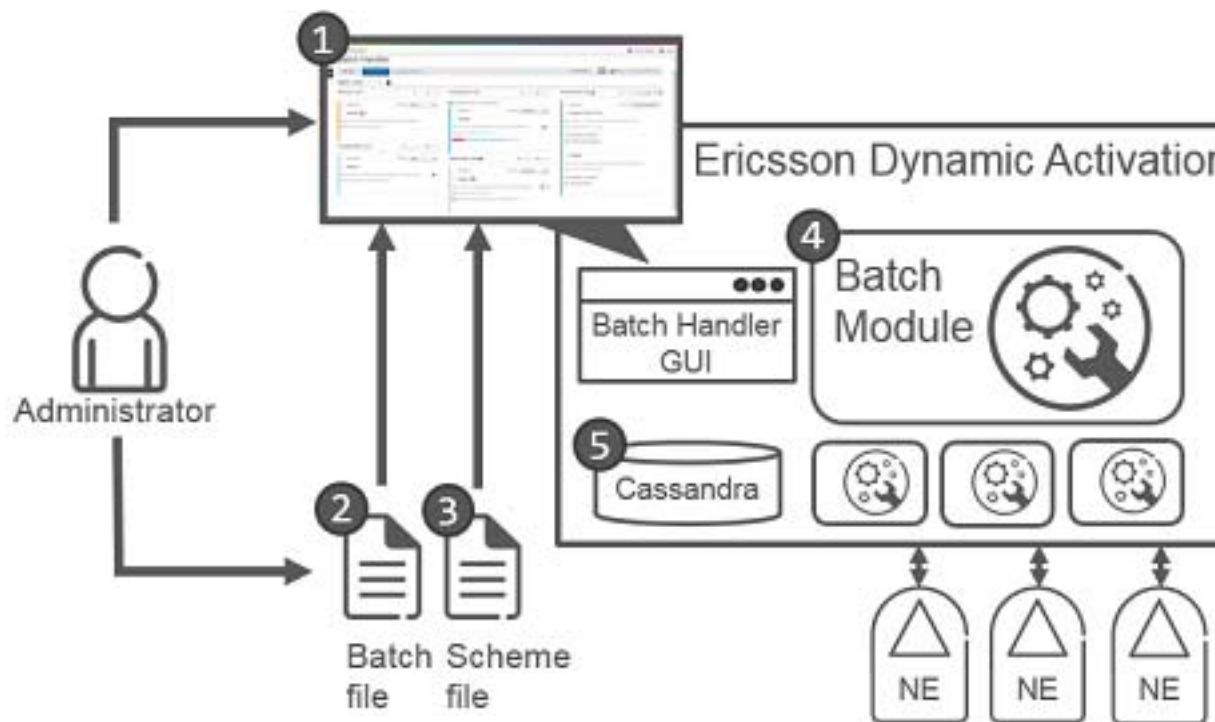


Figure 11 Overview of Batch Handler

- **Batch Handler GUI** - User can manage and monitor the batch jobs and batch files in GUI. For details, refer to *User Guide for Batch Handler*, Reference [34].

- ### 2.10.2 Batch Handler Functionality

Figure 1: State transition diagram of the batch job lifecycle. The diagram illustrates the various states a batch job can be in and the actions that trigger transitions between them. States are categorized as permanent (rectangles) or transitional (ovals). Transitions are labeled with actions in circles.

Legend:

- Permanent state (rectangle)
- Transitional state (oval)
- Regular workflow (solid arrow)
- Abnormal workflow (dashed arrow)
- Start batch job (ST)
- Scheduled batch job (SC)
- Cancel batch job (C)
- Pause batch job (P)
- Resume batch job (R)
- Automatically delete batch job (AD)
- Manually delete batch job (MD)

Workflow:

- Start** (black oval) leads to **New** (rectangle).
- New** (rectangle) can transition to **Scheduled** (rectangle) via **C** (Cancel batch job), to **Preparing** (oval) via **ST** (Start batch job), or to **Deleting** (oval) via **MD** (Manually delete batch job).
- Scheduled** (rectangle) can transition to **Preparing** (oval) via **SC** (Scheduled batch job) or to **Deleting** (oval) via **MD** (Manually delete batch job).
- Preparing** (oval) leads to **Running** (oval), **Paused** (rectangle), or **Cancelling** (oval).
- Running** (oval) can transition to **Paused** (rectangle) via **P** (Pause batch job) or **R** (Resume batch job).
- Paused** (rectangle) can transition to **Running** (oval) via **C** (Cancel batch job) or to **Cancelling** (oval) via **C** (Cancel batch job).
- Cancelling** (oval) leads to **Cancelled** (rectangle) or **Failed** (rectangle).
- Cancelled** (rectangle) leads to **Deleting** (oval) via **AD** (Automatically delete batch job).
- Failed** (rectangle) leads to **Deleting** (oval) via **AD** (Automatically delete batch job).
- Deleting** (oval) leads to **End** (black oval).
- A dashed arrow indicates an abnormal workflow from **Cancelling** (oval) to **Failed** (rectangle).

A batch job can has one of the following states:

- **New:** A batch job has been created.
 - **Scheduled:** The batch job has been scheduled for later execution.
 - **Preparing:** The batch job is in preparing process where the batch handler allocates sessions for the batch job and prepares individual requests to be sent.
 - **Running:** The batch job is in running process where the batch handler executes requests one by one utilizing the specified sessions.
 - **Paused:** The batch job has been paused. A batch job can only be manually set to this status.
 - **Cancelling:** The batch job is in cancelling process, where:
 - If from preparing state, the batch handler releases the sessions allocated for this batch job.
 - If from running state, the batch handler finishes the ongoing requests and then releases the sessions allocated for this batch job.
 - **Finished:** The batch job execution is done without any exception, it indicates all requests in the batch job have been executed. It contains failed and successful requests. Export function can be used to generate a new batch file for failed requests.
 - **Cancelled:** The batch job execution is terminated by `Cancel` operation or by parameter `quit` configured in scheme file. In this state, the requests in the batch job are not completely executed. It contains failed, successful, and un-executed requests. Export function can be used to generate a new batch file for failed and un-executed requests.
 - **Failed:** The batch job execution is aborted due to unexpected system failure. In this state, the requests in the batch job might not be completely executed. It contains failed, successful, and un-executed requests, Export function could be used to generate a new batch file for failed and un-executed requests.
 - **Deleting:** The batch job is in deleting process where the batch handler removes the batch job. However, the batch file will not be deleted.
- Note:** Export function is offered by Batch Handler to deal with the failed or un-executed provisioning requests. It generates a new batch file which contains the failed and un-executed requests from the specified batch job. For how to use Export function, refer to *User Guide for Batch Handler*, Reference [34].

2.11 Customization

Dynamic Activation software platform facilitates adaptation to fulfill operator's specific provisioning demands, and to benefit from strong platform features, such as access control, (loose) error handling, and high-availability support.



For more information, refer to the following documentations:

- *Customization - Architectural Overview*, Reference [30]
- *Customer Adaptation Development Guide for Resource Activation*, Reference [31]
- *Customer Adaptation Guide for Resource Configuration*, Reference [32]

Customer adaptation can be implemented by system integrator. Adaptations outside the customer adaptation scope can be offered as product customization. For more information about product customization, contact your local Ericsson representative.

Data Unit Processing (DUP) migration tool is provided to convert Multi Vendor Network Element (MVNE) DUP customer adaptation from Classic Multi Activation 15.0 (or earlier) to Dynamic Activation 1. Manual code update and test by System Integrators (SI) are still needed to complete the migration. For more information about DUP migration tool, refer to *DUP Customer Adaptation Migration Guide*, Reference [33].

3 System Characteristics

3.1 Deployment

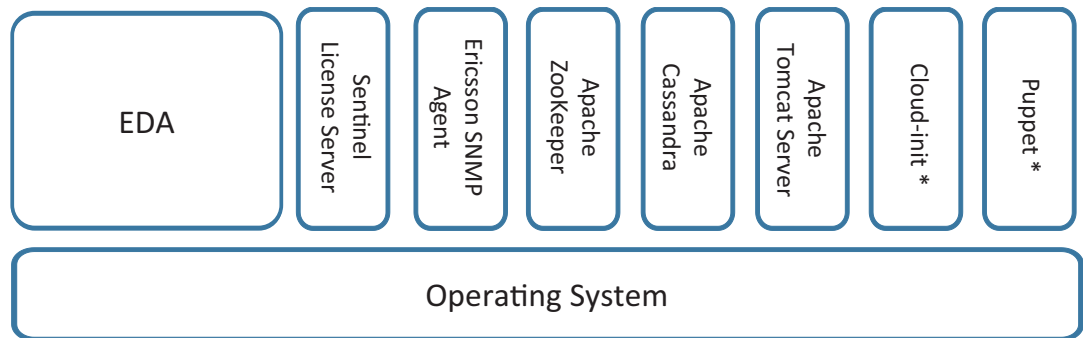
Dynamic Activation is a Java-based software that needs to be deployed on an environment that has the compatible Java Runtime Environment.

To secure telecom grade provisioning characteristics and ease product launch, Dynamic Activation is verified on the following deployments:

- EBS x86 BSP8100 hardware platform for Native deployment.
- Kernel Based Virtual Machine (KVM) or VMware vSphere ESXi for virtual deployments
- Ericsson Cloud Execution Environment (ECEE) or OpenStack for cloud deployment

Other deployment scenario can be handled as a service.

As shown in Figure 13, in addition to Dynamic Activation, the verified software system consists of the following components:



* This component is only applicable for Dynamic Activation on virtualized and cloud deployment.

Figure 13 System Components

- Operating System (OS)
 - Linux™ Distribution Extensions with SUSE (LDEwS)

The adapted variation of the SUSE™ Linux operating system that comes with built-in high-availability capability. This OS is used in a native deployment.
 - Red Hat Enterprise Linux (RHEL) Server

Standard RHEL™ operating system with HA proxy load balancer add-on. This OS is used in a virtual deployment.
- Apache Tomcat Server

The application server that manages the GUI for Dynamic Activation.
- Sentinel™ License Server

The License Server that is a part of the license management architecture.
- Ericsson Simple Network Management Protocol (SNMP) Agent (ESA)

The agent that provides fault management function over the SNMP on application and system level. It also provides server performance management function on the servers.
- Apache ZooKeeper

Centralized service for distributed synchronization within the cluster for Dynamic Activation application.
- Apache Cassandra

Distributed database used for Asynchronous Request Handler, and Processing Log.
- Cloud-init



Multi-distribution package that handles early initializations of cloud instances. Used for Dynamic Activation virtual and cloud deployment.

- Puppet

Configuration management toolkit. Used for Dynamic Activation virtual and cloud deployment.

High-availability deployment details for native deployment can be found in Section 3.1.1 on page 31, for virtualized deployment in Section 3.1.2 on page 33.

Dynamic Activation is stateless in the sense that it does not contain any persistent data apart from configuration data. Geographically redundant deployment can be achieved by deploying multiple Dynamic Activation systems. It is up to the configuration of the client to support multiple Dynamic Activation for desirable effect, for example, redundancy or load-sharing.

3.1.1

Native Deployment

Information described in this section is for the reference deployment scenario for native Dynamic Activation using BSP8100.

Dynamic Activation is deployed in two tiers, as shown in Figure 14, the control tier and the processing tier.

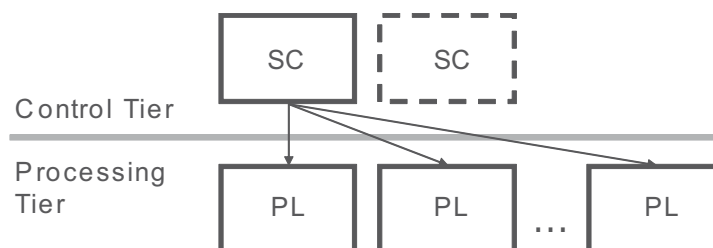


Figure 14 Tiered Deployment

The control tier is the controller of the LDEwS system. It hosts the distributed file system and the license server. The control tier always consists of two dedicated servers in redundant configuration for high-availability reason - one server is active while the other is in standby mode. These servers are called the System Control (SC) node.

The processing tier executes the provisioning traffic. For high-availability reason, at least two servers are mandatory in the processing tier. Such server is called the Payload (PL) node. Processing capacity is scaled by adding more PL nodes to the system.

This makes the minimum system a four-server configuration.

A basic native deployment includes a cabinet and a subrack which contains two routers, two switches and space for up to 12 blades. The Figure 15 shows the

logical view of such system. The routers handle external connectivity that is independent of the number of blades within the shelf. The switches together with the back panel and SW deployed on the blades provide load balancing functionality among the PL nodes.

For security and safety reason, the default IP infrastructure setup handles provisioning traffic and O&M traffic in two different IP networks. An example of typical provisioning traffic is individual subscriber provisioning via CAI/CAI3G/MML and massive operations via CLI. O&M traffic can be accessed through GUI, command line for system O&M as well as SNMP connection for alarm handling.

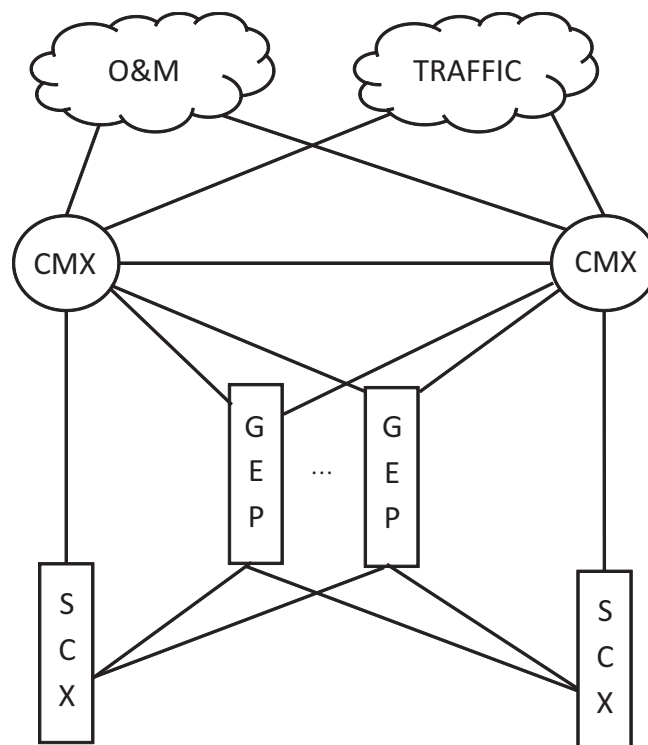


Figure 15 Logical View of a Basic Native Deployment

The maximum configuration of a Dynamic Activation system consists of two SC nodes and 10 PL nodes.

Dynamic Activation provides a high-availability deployment architecture for commercial systems that covers the following aspects:

- HW redundancy: all HW components (server, router, switch) are at least duplicated
- IP backbone connectivity redundancy: either VRRP or BFD can be applied
- Inbound interfaces redundancy: traffic interface and O&M interface are managed by eVIP

For more information about installation, see the following documents:



- *Customer Questionnaire for Native Deployment*, Reference [11]
- *Parameter List for Native Deployment*, Reference [12]
- *Hardware Installation and IP Infrastructure Setup for Native Deployment GEP3*, Reference [13]
- *Software Installation for Native Deployment*, Reference [14]

For system upgrade instruction, see *System Upgrade to Ericsson Dynamic Activation 1*, Reference [15].

3.1.2 Virtual and Cloud Deployment

Information described in this section is for the reference deployment scenario using KVM, VMware ESXi hypervisor, ECEE, or OpenStack.

In a virtual and cloud deployment Dynamic Activation is supplied only as an Open Virtual Appliance (OVA), including guest OS and Dynamic Activation application. Host HW and hypervisor are not provided by Ericsson, see Figure 16.

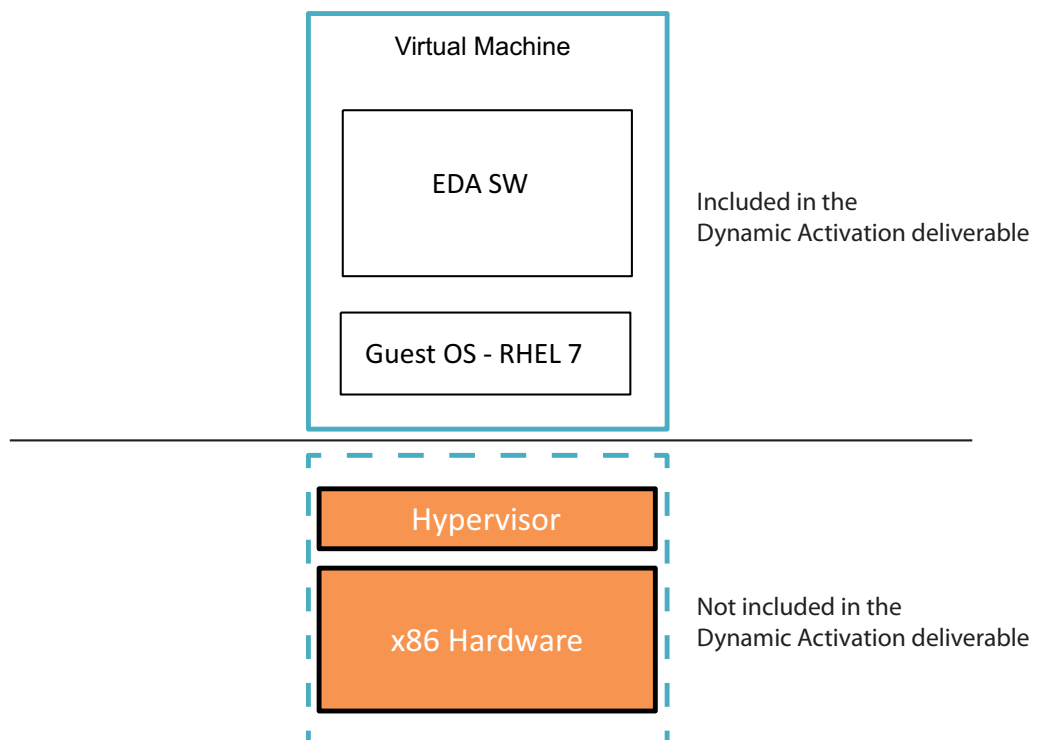


Figure 16 Open Virtual Appliance for Virtual and Cloud Deployment

For virtual and cloud deployments, all system components are running in a Virtual Machine (VM). Multiple VMs can be deployed on one or multiple hosts, see Figure 17.

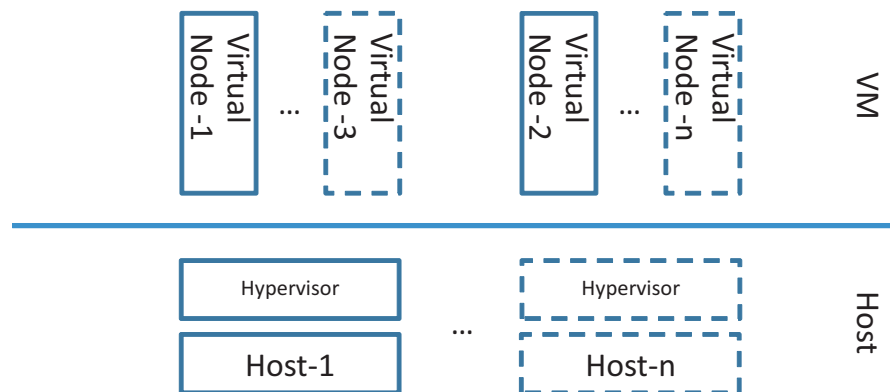


Figure 17 Virtual and Cloud Deployment

Dynamic Activation on Virtual Environment can be deployed in one of the following configurations:

- Standalone VM configuration for customer trial purpose
- Three VM nodes high-availability configuration
- Scalable high-availability configuration

Dynamic Activation executes provisioning traffic on each VM instance. Common services like license server and distributed synchronization service are spread among the VMs.

Stand alone systems only have one license server and one ZooKeeper server instance.

To secure high-availability capabilities, the license server is hosted on two Dynamic Activation instances and the distributed synchronization service (ZooKeeper) is hosted on a minimum of three Dynamic Activation instances. This makes the minimum high-availability system a three-VM configuration.

ZooKeeper uses ensembles with a minimum of three ZooKeeper hosts and the ensemble must be in majority to secure data integrity. The recommendation is to have physical separation of all three ZooKeeper hosts for high-availability reasons, see Figure 18.

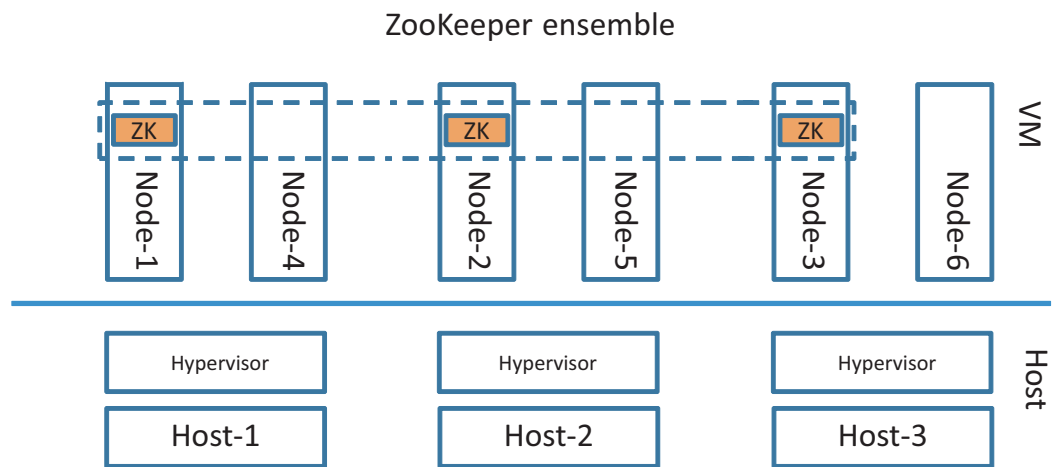


Figure 18 ZooKeeper Host Distribution

Each Dynamic Activation instance requires a minimum of two bridged network adapters; one for internal cluster connectivity and one for external connectivity. For security and safety reasons, the recommended IP infrastructure setup handles provisioning traffic and O&M traffic in two different logical IP networks.

Load balancing is handled by the Keepalived and HAProxy solutions. Keepalived handles the Virtual IP, and HAProxy distributes the load between the VMs. See Figure 19.

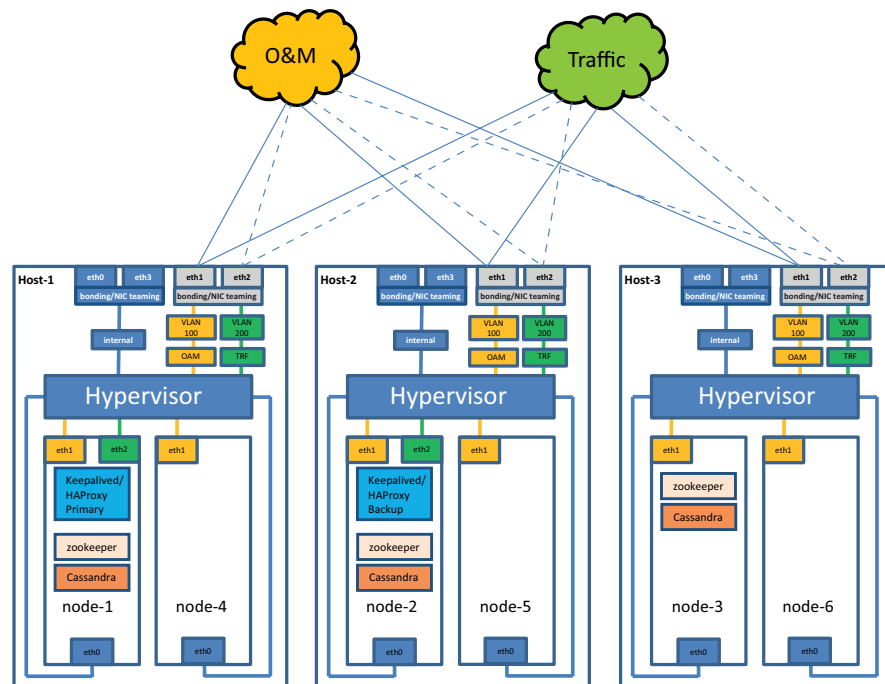


Figure 19 Virtual IP Infrastructure Setup

The scalable high-availability system is configured in the same way as the minimum high-availability system with the addition of extra Dynamic Activation instances.

For more information, see the following documents:

- *Customer Questionnaire for Virtual and Cloud Deployment*, Reference [16]
- *Parameter List for Virtual Deployment*, Reference [18]
- *Parameter List for CEE Deployment*, Reference [19]
- *Parameter List for OpenStack Deployment*, Reference [20]
- *Requirements on Virtualization and Cloud Infrastructure*, Reference [17]
- *Network Description and Configuration for Virtual and Cloud Deployment*, Reference [21]
- *Software Installation for Virtual and Cloud Deployment*, Reference [23]
- *System Administrators Guide for Virtual and Cloud Deployment*, Reference [22]

3.2 Operation and Maintenance

Operation and maintenance solutions, guidelines, and instructions are provided for the reference deployment scenario.



Operation and maintenance supports characters from the ISO-8859-1 character set, that is:

- The numbers from 0 through 9
- The uppercase and lowercase English alphabet
- The characters used in Western European countries
- Commonly used special characters such as “&”, “-”

3.2.1 Fault Management

The fault management solution monitors system behavior, sends notification when abnormal condition occurs and provides method for system recovery.

Dynamic Activation uses ESA to monitor the status of different parts of the software and hardware of the server. ESA receives different events and sends them as alarms over SNMP to the Operations Support System (OSS).

Supported SNMP protocol versions are v2c and v3, default is v2c.

The alarms cover the following errors:

- Too high CPU use over a period of time
- Too high memory use
- Running out of disk space on monitored partitions
- Dynamic Activation application not available
- Mounted disk connection failure
- Communication failure with NE
- License expiration/capacity threshold warning
- SC/PL/VM node not responding

All possible alarms and relevant information, such as probable cause and repair action are described in the user documentation.

Dynamic Activation can be monitored by different systems, such as Ericsson OSS-RC where alarms can be viewed and cleared. It is crucial that the OSS system sends heartbeat to ESA and keeps track of the alarms related to node failure.

It is possible to view and clear alarms in the ESA active alarm list on each blade without connection to an OSS.

In addition to ESA, each Dynamic Activation blade has a local server log file that hosts other types of events such as deployment of business logic.

ESA is configured in cluster mode, to provide a single monitoring point for Dynamic Activation.

Figure 20 shows how ESA in cluster mode integrates towards OSS through one virtual IP (VIP) address. Alarms generated by any of the PL nodes are forwarded to the ESA administrators (masters) running on the SC nodes. The active master then sends the alarm over SNMP to the OSS.

To be able to determine where an alarm originates from, since the sender IP address is always the VIP address, each SNMP trap contains originating source IP information.

For more information, see *System Administrators Guide for Native Deployment*, Reference [10], and *System Administrators Guide for Virtual and Cloud Deployment*, Reference [22].

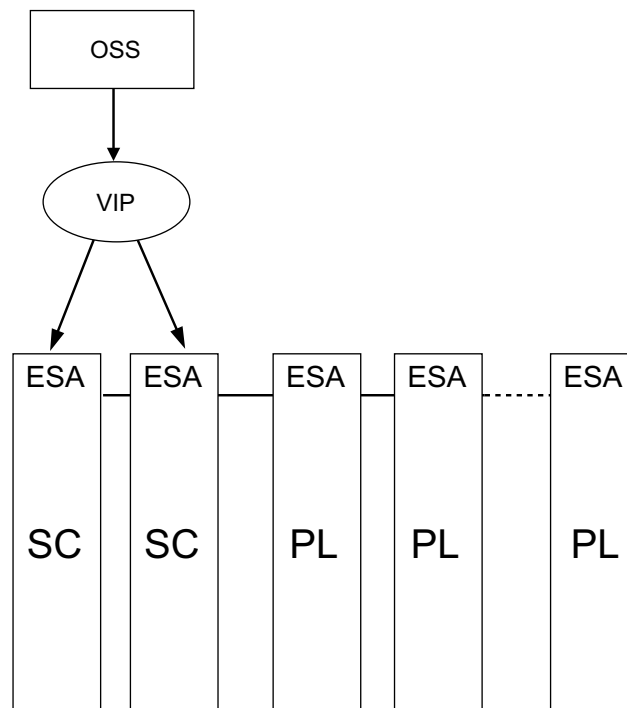


Figure 20 Logical View of ESA Cluster Mode

Note: For virtual and cloud deployments, the SC/PL nodes are replaced with VMs. For more information, see *Software Installation for Virtual and Cloud Deployment*, Reference [23].

3.2.2 Software Lifecycle Management

The software lifecycle management of a Dynamic Activation system is handled in two different ways:

- 3PP services, for example: Zookeeper, Cassandra, are handled by RPMs.
- Dynamic Activation modules are handled by the bootloader component.



Bootloader is an Execution Environment (EE) independent component, which manages the software lifecycle of Dynamic Activation modules. It provides the following services to the system:

- Centralized Software Repository:
 - Holds multiple versions of individual modules and sub-modules.
 - Single point of SW management.
- Server Deployment:
 - Controls the deployment of artifacts on nodes.
 - Provides a CLI interface for management of software artifacts.

For detailed information, refer to *System Administrators Guide for Native Deployment*, Reference [10].

3.2.3 CPU Load Monitoring

ESA is also used for monitoring the CPU load.

The CPU load is measured on per CPU core on each node within the cluster for a pre-defined time interval. The result is stored in a file on the respective nodes file system. Result files are kept in the file system for a configurable time period. Older files are removed automatically. The files can be fetched through Secure Shell File Transfer Protocol (SFTP).

The file contains measures of:

- Actual CPU load in percentage
- Maximum CPU load in percentage
- Minimum CPU load in percentage

For an example of a 3GPP file, see Example 1.



```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<measCollecFile>
  <fileHeader fileFormatVersion="3GPP_PM_32.435_XML_v11.0.0">
    <measCollec beginTime="2017-01-13T09:20:01+01:00"/>
  </fileHeader>
  <measData>
    <measInfo>
      <job jobId="CPULoadCounter"/>
      <granPeriod endTime="2017-01-13T09:25:01+01:00" duration="PT300S"/>
      <repPeriod duration="PT300S"/>
      <measType p="0">CPULoadCounter1.actual</measType>
      <measType p="1">CPULoadCounter1.min</measType>
      <measType p="2">CPULoadCounter1.max</measType>
      <measValue measObjLdn="">
        <r p="0">4</r>
        <r p="1">1</r>
        <r p="2">18</r>
      </measValue>
    </measInfo>
  </measData>
  <fileFooter>
    <measCollec endTime="2017-01-13T09:25:01+01:00"/>
  </fileFooter>
</measCollecFile>
```

Example 1 3GPP XML File

It is possible to modify the following monitoring behavior:

- Output directory.
- The time interval in hours of how long the 3GPP files should be saved.
- The time interval in minutes of how often the system should scan for old 3GPP files.
- The time interval in seconds of how often the system should read the data.
- The time interval in minutes of how often the system should collect data before any 3GPP file is written to file.

For more information, see *System Administrators Guide for Native Deployment*, Reference [10], and *System Administrators Guide for Virtual and Cloud Deployment*, Reference [22].

3.2.4 Internal Heartbeat Monitoring

Dynamic Activation uses ESA to monitor internal hardware resources.

ESA sends the SNMP *Get* operation for internal heartbeat signalling and ESA Performance Management (PM) counters keep track of available SC, PL, and VM nodes. ESA PM Agent has configurable thresholds for each counter and raises alarms upon threshold breach. Separate events are sent to identify the failing node.

The alarms cover the following errors:

- Number of available SC nodes have reached minimum threshold



- Number of available PL nodes have reached minimum threshold
- Number of available VMs have reached minimum threshold

The events cover the following errors:

- When an SC node is not responding
- When a PL node is not responding
- When a VM is not responding

3.2.5 Backup and Restore

A backup and restoration solution is provided for the reference deployment scenario.

3.2.5.1 Backup and Restore for Native Deployment

For Native deployment, it is possible to back up:

- Network File System (NFS) of the cluster
- Configuration data, such as NE configuration, ESA configuration, and user configuration

Backup files created by the system must be transferred to a secondary storage.

Performing a complete system restore is dependent on the condition of nodes. If the SC node can be started, then restoration of the backup is sufficient. If the SC node cannot be started, a maiden installation should be performed.

For more information, see *Backup and Restore Guideline for Native Deployment*, Reference [24].

3.2.5.2 Backup and Restore for Virtual Deployment

For virtual deployment, it is possible to back up:

- Full and Incremental disk backup of running VMs
- Configuration data, such as NE configuration, ESA configuration, and user configuration

Backup files created by the Dynamic Activation system must be transferred to a secondary storage.

How to perform a complete Dynamic Activation system restore is dependent on the condition of the VMs and the host machine. If the host machine is fully functional, then restoration from full or incremental disk backup of each VM is

sufficient. If the host machine is not functional, each VM should be restored on a new host machine.

For more information, see *Backup and Restore Guideline for Virtual and Cloud Deployment*, Reference [25].

3.2.6 Configuration Management

Configuration Management can be performed on the application level and the operating system level.

Apart from the application level configuration functions provided by the GUI, described in Section 2.9 on page 23, additional configuration tasks for the application as well as the system can be performed via command line from a client.

The application level configuration (not including data for optional components) can be synchronized, to other clusters from the GUI, see Section 2.9 on page 23.

The command line provides access to the operating system level. For security reasons, only the Secure Shell (SSH) is allowed. Plain text telnet is disabled.

From the command line the Operating System administrator can perform various administrative tasks, such as check system load, monitor disk usage and others.

For more information about all relevant maintenance tasks, see *System Administrators Guide for Native Deployment*, Reference [10], and *System Administrators Guide for Virtual and Cloud Deployment*, Reference [22].

3.2.7 Security

Security for a deployed Provisioning System is achieved in different areas and aspects. The following is a summary of the security solution:

- Interface security - external interfaces can be protected by secured protocols. For more information, see *System Administrators Guide for Native Deployment*, Reference [10], and *System Administrators Guide for Virtual and Cloud Deployment*, Reference [22].
- Authentication - access to the system is protected on per user bases
- Operating System security - inherits Linux security features
- Node hardening - all ports and system services that are not used are disabled to restrict access. For more information, see *Hardening Guideline for Native Deployment*, Reference [26].



3.2.8 Privacy

Personal data consists of information related to an identifiable person. An identifiable person is one that can be identified directly, for example by its MSISDN, IMSI, or passport number, or indirectly, for example name, date of birth and postal address.

Dynamic Activation processes personal data during the provisioning process. In some cases, the personal data can be saved in certain logs and result files. Dynamic Activation checks the time stamp on those logs and result files regularly. All files older than the systems retention time are deleted. The default retention time is 30 days and it can be adapted to fit the operator's privacy requirement.

For more information about changing the retention time, see *System Administrators Guide for Native Deployment*, Reference [10], and *System Administrators Guide for Virtual and Cloud Deployment*, Reference [22].

3.3 Expansion of Dynamic Activation

For an existing customer, it is possible to expand in the following scenarios:

- Adding new Dynamic Activation system. This is a new installation of Dynamic Activation system. For detailed instruction, see *Software Installation for Native Deployment*, Reference [14], or *Software Installation for Virtual and Cloud Deployment*, Reference [23].
- Adding new PL/VM node to an existing Dynamic Activation system. For detailed instructions, see *System Expansion for Native Deployment*, Reference [28], or *System Expansion for Virtual and Cloud Deployment*, Reference [29].
- Adding new licenses to an existing Dynamic Activation system. For detailed instruction, see *System Administrators Guide for Native Deployment*, Reference [10], or *System Administrators Guide for Virtual and Cloud Deployment*, Reference [22].

4 Synchronization between Clusters

When having more than one Dynamic Activation cluster, these can be used in an active/passive or active/active setup.

The following type of data is possible to synchronize between clusters:

- License usage data - automatically synchronized
- Configuration data (not including Resource Configuration data) - manually triggered a synchronization from the GUI

Since Dynamic Activation does not hold any subscriber or service data, there is no need for synchronizing this type of data.

Up to three different clusters can be synchronized.

4.1 Synchronization of License Counters

License counters are synchronized between the separate clusters once every hour. The synchronization is performed using SFTP to transfer the license usage data between the clusters.

Each cluster has its own license server where a license covering the total needed capacity for all clusters needs to be available.

The used capacity of each cluster is communicated to all other configured clusters. The total used capacity is aggregated in each cluster and this is the capacity that is used for license purposes.

For more information about the synchronization of license counters, refer to:

- *System Administrators Guide for Native Deployment*, Reference [10]
- *System Administrators Guide for Virtual and Cloud Deployment*, Reference [22]

4.2 Synchronization of Configuration

Note: When synchronizing configuration, it is important that all clusters are on the same software level. If using an active/active setup the traffic should be disabled on all clusters, except for the cluster that the synchronization is initiated from, before initiating the configuration synchronization.

If several separate clusters are to have the exact same application level configuration, there is an option to manually synchronize the configuration to all configured clusters. This synchronization is done from the GUI and enables the use of a single administration interface for several clusters.

The synchronization is performed using SFTP to transfer the configuration data between the clusters.

For more information about the synchronization of configuration data, see *User Guide for Resource Activation*, Reference [7].



4.2.1 Synchronized Configuration Data

The following configuration data are synchronized among clusters:

- Network Elements control data, including configuration data under **Operation & Management > Network Elements** GUI, including:
 - **Network Elements**
 - **Network Element Groups**
 - **Routing**
- System configuration data under **Operation & Management > System** GUI, including:
 - **Options**
 - **Notification E-Mail Configuration**
- Configuration data in the **Asynchronous Control** GUI.
- Resilient Activation configuration in the following GUIs:
 - **Priority Configuration**
 - **Retry Rule Configuration for Request Management**
 - **Notification Rules Configuration**
- Configuration data in the **Retry Rule Configuration for Processing Queue** GUI.

Users can also choose to include provisioning client data for synchronization. Such data are configured in all tabs of the **Operation & Management > Access Control** GUI.

Note: When user authentication data is chosen for synchronization, the performance of the receiving cluster will be impacted.

For more information about the synchronization of configuration data, see *User Guide for Resource Activation*, Reference [7].

5 Limitations

The maximum number of HTTP keep-alive connections is 250 per PL/VM node. This number of connections is shared between CAI3G provisioning traffic and



GUI use. However, the number of connections used by GUI is insignificant as one GUI connection can be shared by a number of concurrent users.

If BSS uses the Request Grouping and notification at same time, Asynchronous Request Handler cannot preserve the execution order during the process of disabling or enabling Asynchronous Request Consumer. Therefore, in this case, it is suggested to complete all grouped requests before disabling or enabling the Asynchronous Request Consumer. For information about enabling the Asynchronous Request Consumer, refer to *Asynchronous Control* in *User Guide for Resource Activation*, Reference [7].

6 Appendix A - Licenses Deployed on Dynamic Activation

Table 2 License Key List

Value Package License	
License ID	Name
FAT1023848/1	EDA Maximum Servers
FAT1023848/2	EDA Max VMs
FAT1023848/3	EDA Maximum CPUs
FAT1023849/2	EDA SW Advanced
FAT1023849/5	EDA 2G 3G
FAT1023849/6	EDA Subscriber Services
FAT1023849/7	EDA IMS Core
FAT1023849/8	EDA LTE EPC
FAT1023849/9	EDA Multimedia Telephony
FAT1023849/10	EDA Policy Control
FAT1023849/11	EDA MV Subscriber Services
FAT1023849/12	EDA Charging System
FAT1023849/13	EDA Billing and CBiO
FAT1023849/14	EDA Wi-Fi Calling
FAT1023849/15	EDA eSIM
FAT1023849/16	EDA Shared Networks
FAT1023849/20	EDA Access
FAT1023849/21	EDA Aggregation
FAT1023849/22	EDA Core/Edge



Value Package License	
License ID	Name
FAT1023849/23	EDA Service Configuration
FAT1023849/24	EDA Small Access
FAT1023849/40	EDA Enterprise Core Base
FAT1023849/42	EDA Communication
FAT1023849/101	EDA Base Package





Reference List

Ericsson Documents

- [1] *Library Overview*, 18/1553-CSH 109 628 Uen
- [2] *Product Overview*, 1550-CSH 109 628 Uen
- [3] *Function Specification Resource Activation*, 3/155 17-CSH 109 628 Uen
- [4] *Function Specification Resource Configuration*, 19/155 17-CSH 109 628 Uen
- [5] *Generic CAI3G Interface 1.2*, 2/155 19-FAY 302 0003 Uen
- [6] *Asynchronous CAI3G Interface Specification 1.2*, 34/155 19-CSH 109 628 Uen
- [7] *User Guide for Resource Activation*, 1/1553-CSH 109 628 Uen
- [8] *User Guide for Resource Configuration*, 11/1553-CSH 109 628 Uen
- [9] *Configuration Manual for Resource Activation*, 2/1543-CSH 109 628 Uen
- [10] *System Administrators Guide for Native Deployment*, 1/1543-CSH 109 628 Uen
- [11] *Customer Questionnaire for Native Deployment*, 4/1057-CSH 109 628 Uen
- [12] *Parameter List for Native Deployment*, 5/1057-CSH 109 628 Uen
- [13] *Hardware Installation and IP Infrastructure Setup for Native Deployment GEP3*, 2/1531-CSH 109 628 Uen
- [14] *Software Installation for Native Deployment*, 1/1531-CSH 109 628 Uen
- [15] *System Upgrade to Ericsson Dynamic Activation 1*, 1/154 31-CSH 109 628 Uen
- [16] *Customer Questionnaire for Virtual and Cloud Deployment*, 2/1057-CSH 109 628 Uen
- [17] *Requirements on Virtualization and Cloud Infrastructure*, 2/2135-CSH 109 628 Uen
- [18] *Parameter List for Virtual Deployment*, 3/1057-CSH 109 628 Uen
- [19] *Parameter List for CEE Deployment*, 6/1057-CSH 109 628 Uen



- [20] *Parameter List for OpenStack Deployment*, 7/1057-CSH 109 628 Uen
- [21] *Network Description and Configuration for Virtual and Cloud Deployment*, 1/1551-CSH 109 628 Uen
- [22] *System Administrators Guide for Virtual and Cloud Deployment*, 3/1543-CSH 109 628 Uen
- [23] *Software Installation for Virtual and Cloud Deployment*, 4/1531-CSH 109 628 Uen
- [24] *Backup and Restore Guideline for Native Deployment*, 2/1553-CSH 109 628 Uen
- [25] *Backup and Restore Guideline for Virtual and Cloud Deployment*, 6/1553-CSH 109 628 Uen
- [26] *Hardening Guideline for Native Deployment*, 1/154 43-CSH 109 628 Uen
- [27] *CUDB Subscription Repair and Remove Procedures*, 4/1553-CSH 109 628 Uen
- [28] *System Expansion for Native Deployment*, 2/154 31-CSH 109 628 Uen
- [29] *System Expansion for Virtual and Cloud Deployment*, 3/154 31-CSH 109 628 Uen
- [30] *Customization - Architectural Overview*, 20/1553-CSH 109 628 Uen
- [31] *Customer Adaptation Development Guide for Resource Activation*, 5/1553-CSH 109 628 Uen
- [32] *Customer Adaptation Guide for Resource Configuration*, 14/1553-CSH 109 628 Uen
- [33] *DUP Customer Adaptation Migration Guide*, 19/1553-CSH 109 628 Uen
- [34] *User Guide for Batch Handler*, 12/1553-CSH 109 628 Uen