

Generic CLI Interface Specification

Ericsson Dynamic Activation 1

INTERFACE DESCRIPTION

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 | Communication Protocol | 3 |
| 2.1 | Direct Socket | 3 |
| 2.2 | TELNET | 3 |
| 2.3 | SSL | 3 |
| 2.4 | Ports | 4 |
| 2.5 | Login Procedure | 4 |
| 2.6 | Concurrent Sessions | 4 |
| 2.7 | Concurrent Jobs | 5 |
| 3 | About the Commands | 7 |
| 3.1 | Command Syntax | 7 |
| 3.1.1 | Delimiters | 7 |
| 3.1.2 | Command Descriptions | 8 |
| 3.2 | Request and Response Flow | 8 |
| 3.2.1 | Command with File Response | 9 |
| 3.2.2 | Command with Direct Client Response | 11 |
| 3.2.3 | Errors | 13 |
| 4 | Control Commands | 17 |
| 4.1 | Connect | 17 |
| 4.2 | Cancel | 18 |
| 4.3 | Status | 18 |
| 4.4 | Remove File | 20 |
| 4.5 | Exit | 20 |
| | Reference List | 23 |





1 Introduction

This document covers the Ericsson™ Dynamic Activation (EDA) Command Line Interface (CLI), in this document only referred to as “the CLI”.

1.1 Purpose and Scope

This document is an introduction to the CLI. The CLI is used for operations that can take long time to execute, or have large responses, or both. The purpose of this document is to describe syntaxes, control commands, and some general information for the CLI.

1.2 Target Groups

The target groups for this document are as follows:

- System Administrators

1.3 Typographic Conventions

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

In addition, this document uses the following to indicate operations:

| | |
|----------|--------|
| C | Create |
| S | Set |
| G | Get |
| D | Delete |

1.4 Prerequisites

To use this document fully, users must meet the following prerequisites:

- Basic knowledge about the Dynamic Activation product





2 Communication Protocol

The CLI is an interface to execute a proprietary Ericsson presentation and session layer.

The following are basic features of the CLI:

- Asynchronous operation; several commands can be entered in one session.
- ASCII-character-based presentation
- Multiple concurrent sessions

The following standards can be used as transport and network layers:

- Direct Socket (Raw Socket)
- Direct Socket (Raw Socket) with SSL
- TELNET
- TELNET with SSL

2.1 Direct Socket

The Direct Socket interface is a raw TCP socket over IP. It is a connection-oriented protocol providing a reliable, full-duplex byte stream for a user process. TCP takes care of communication details such as acknowledgements, timers, and retransmissions.

2.2 TELNET

Standard TELNET procedures are used to open a connection. Each end is assumed to act as a network virtual terminal. The client is always the initiating side and option negotiation can take place within the structure of the TELNET Protocol. Refer to RFC854. Any side can choose to clear the connection.

2.3 SSL

SSL can be used together with both the Direct Socket and the TELNET interface, to provide a secure communication between the client and Dynamic Activation. The client application must support SSL or use an external SSL tunnel to secure the communication.

For instruction how to set up SSL, refer to *System Administrators Guide for Native Deployment*, Reference [2] .



2.4 Ports

If an SSL is used, 8992 is the default CLI port. For plain TELNET or Direct Socket connections, the default port is 8023.

2.5 Login Procedure

The login procedure is as follows:

```
login as:  <Application_user_name>
```

```
password:  <password>
```

```
<hostname>
```

The connection is done towards OAM-VIP, and VIP is directing the connection to a Payload (PL) node.

When executing massive commands, the result files will be stored on the PL node that the user is logged in to. Therefore, the hostname indicates where the files are saved. For example, hostname CL21-PL-10 means that the user is currently logged in to PL10.

Also, if VIP is sending the login to another PL node and there is current jobs ongoing that was started on another PL, it is not possible to view the ongoing jobs. That is, ongoing jobs are only known on the current logged in PL node.

2.6 Concurrent Sessions

Several users can be logged in to the CLI at the same time. The maximum number of concurrent sessions can be set in the file `/cluster/home/bootloader/config/module_config_files/UDC-Application/cli.properties`. Uncomment and change the value of `maxNumberOfSessions`. The default value is 16. Save the file and activate the changes.

From an SC node, activate the modifications by running the following command for each affected PL node, one by one:

Warning!

Wait for each PL node to be activated before starting with the next one, otherwise traffic disturbances occur. Do not run `all`.

```
# bootloader.py node activate \
```




```
--host <hostname>
```

<hostname> is the hostname of the PL node that is to be activated.

2.7 Concurrent Jobs

Several ongoing jobs are possible simultaneously in the CLI. The maximum number of concurrent jobs can be edited in the file `/cluster/home/bootloader/config/module_config_files/UDC-Application/cli.properties`. Uncomment and change the value of `maxNumberOfTransactions`. The default value is 16. Save the file and activate the changes.

It is only possible to view ongoing jobs on the PL node that the user is currently logged in to.

Note: The value must not be larger than the number of CUDB connections, which is configured for massive operation. Also, the value must be less than the configured number of CUDB connections if the connections are shared with other commands, for example individual provisioning.

From an SC node, activate the modifications by running the following command for each affected PL node, one by one:

Warning!

Wait for each PL node to be activated before starting with the next one, otherwise traffic disturbances occur. Do not run `all`.

```
# bootloader.py node activate \  
--host <hostname>
```

<hostname> is the hostname of the PL node that is to be activated.





3 About the Commands

This section contains some general information about the CLI commands. The following topics are covered here:

- Command Syntax, Section 3.1 on page 7
- Request and Response Flow, Section 3.2 on page 8

3.1 Command Syntax

This section describes the syntax for the CLI commands.

Each command is represented by a short name (up to seven letters). For example, Print Conditional Subscriber List has the short name `HEMSLIP`. The short name is always the one written in the command prompt. The following is an example of the command Print Conditional Subscriber List:

```
hostname>HEMSLIP:IMSI=755,SUD=CFU-1&TRC-1;
```

Note: All CLI commands must be written in uppercase. However, parameter values can be written in lowercase by using the text string delimiter.

3.1.1 Delimiters

The following table contains delimiters for the CLI commands.

Table 1 Delimiters

| Delimiter | Description |
|-----------|---------------------------------------------------------------------------------------------------------------------|
| : | Separates the command name from its parameters. |
| = | To the left of this symbol is a parameter name. To the right of this symbol is the parameter value. |
| & | Separates two different values for the same parameter. |
| && | Separates the start value and end value in a value range. Start and end values must have the same number of digits. |
| ; | Finishes the command. |
| " | Text String delimiter. |



3.1.2 Command Descriptions

The following is an example of the `Print Conditional Subscriber List` command description.

```
HEMSLIP: [ [MSISDNS=msisdns  
            [IMSIS=imsis] [SUD=sud...]  
            SUD=sud... ] [RID=rid...];
```

Note: The parameters must be entered in the order they appear in the command description.

Table 2 explains how the command description is built.

Table 2 Syntaxes in Command Descriptions

| Syntax | Description |
|----------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| [A] | A is optional. |
| $\begin{bmatrix} A \\ B \end{bmatrix}$ | A and B are optional, but they cannot be used together. |
| $\begin{bmatrix} A \\ B \end{bmatrix}$ | Use either A or B (choice). |
| ... | The parameter supports the use of "&" or "&&" ⁽¹⁾ , for multiple values. It is, however, never required to enter multiple values. |

(1) "&&" can only be used on numeric ranges. This means "&&" is not applicable on, for example, `SUD`.

Note: The Command Description syntaxes can be nested inside each other.

3.2 Request and Response Flow

The message flow described in this section is that between the client and the CLI node.

The CLI has two common request and response flows; one that is used when the response is potentially large, and one that is used when the response is reasonably small. Commands with potentially large responses write the response in a file. The file must be fetched by SFTP. Commands with small responses write the response at the prompt. See the following subsections for detailed information.



The conditional search commands are examples of commands that have their responses written to a file. The control commands are all examples of commands that write their responses directly at the prompt.

3.2.1 Command with File Response

Commands with potentially large responses save their results in a file. The file can be found in the directory `/var/dve/cli/` on the server where the command was run. Log in as `dvecli` to be able to access the files. The request and response flow for commands with file response is shown in Figure 1.

Note: Since the result files are likely to contain personal data, they are automatically deleted when they are 30 days old. For more information about personal data, see *System Administrators Guide for Native Deployment*, Reference [2].

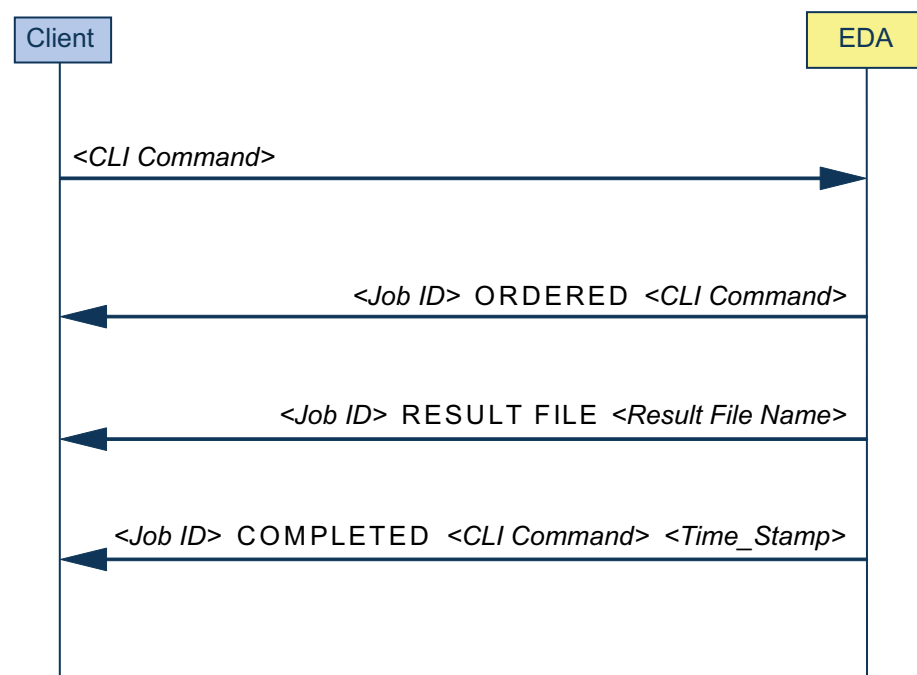


Figure 1 Message Flow for Commands with File Response

Example:

```

hostname>HEMSLIP:MSISDNS=494;

87 ORDERED HEMSLIP

87 RESULT FILE 87_HEMSLIP_20090311_141951.xml.zip

87 COMPLETED HEMSLIP (Started 2009-03-11 14:19:51)
  
```



Response File Format

The response filename has the following syntax:

```
<jobid>_<CLI command>_<yyyymmdd>_<hhmmss>.xml.zip
```

Example:

```
87_HEMSLIP_20090311_141951.xml.zip
```

The content of the response file is based on the following XML schema:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://schemas.ericsson.com/MA/"
  elementFormDefault="qualified"
  attributeFormDefault="qualified">
  <xs:complexType name="ResultType">
    <xs:sequence>
      <xs:element name="Request">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="command" type="xs:string" />
            <xs:element name="starttime" type="xs:string" />
            <xs:element name="jobid" type="xs:string" />
          </xs:sequence>
        </xs:complexType>
      </xs:element>

      <xs:any namespace="##other" processContents="skip"/>

      <xs:element name="Fault" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="code" type="xs:integer" />
            <xs:element name="message" type="xs:string" />
            <xs:element name="additionalinfo" type="xs:string"
              minOccurs="0" />
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="Result" type="ResultType" />
</xs:schema>
```

The `<xs:any namespace="##other" processContents="skip"/>` part of the generic schema, represents the application schema that is unique for each command.



The schema for each command can be found in their respective CLI Interface Description.

3.2.2 Command with Direct Client Response

Some commands echo their response directly back to the client, instead of writing to a file. The following types of direct client responses are used:

- Command response, Section 3.2.2.1 on page 11
- Notification only, Section 3.2.2.2 on page 12

3.2.2.1 Command Response

Command responses are typically used for control commands and commands with reasonably small response sizes. The request and response flow is as shown in Figure 2.

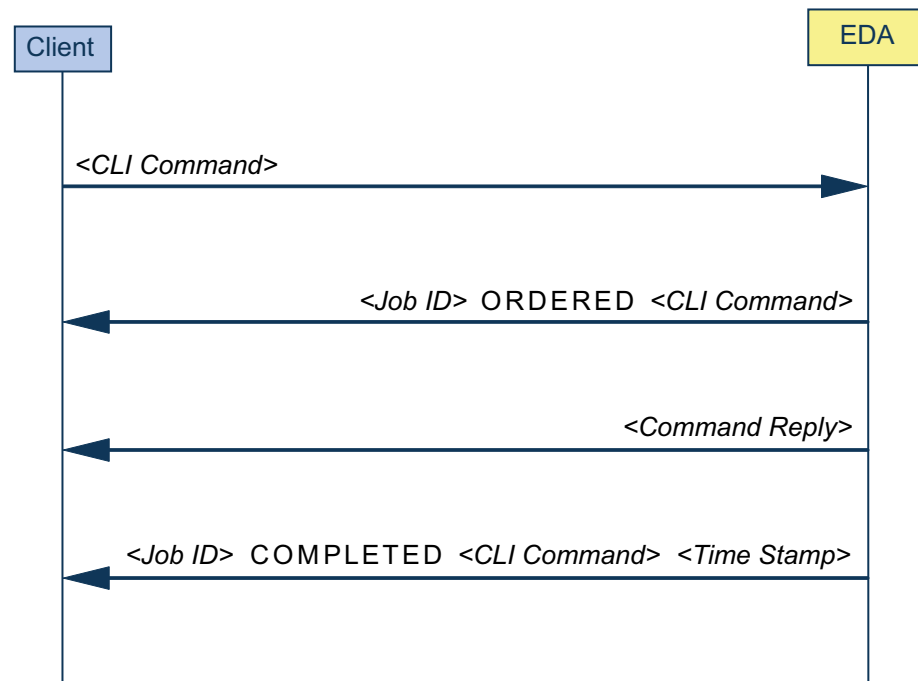


Figure 2 Message Flow for Commands with Direct Command Response

The following is an example of a command with a reasonably small response size, along with its response:

```

hostname>HECDAPP:APNID=ALL;

69 ORDERED HECDAPP

69 RESULT
  
```



```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes" ?>
<Result xmlns="http://schemas.ericsson.com/MA/">
  <Request>
    <command>HECDAPP:APNID=ALL</command>
    <starttime>2008-09-24 09:29:16</starttime>
    <jobid>69</jobid>
  </Request>
  <AccessPointNameData xmlns="http://schemas.ericsson.com/pg/hlr/13.5/">
    <AccessPointName>
      <apn>COM.ERICSSON</apn>
      <apnid>1</apnid>
    </AccessPointName>
    <AccessPointName>
      <apn>SE.ERICSSON</apn>
      <apnid>2</apnid>
    </AccessPointName>
    <AccessPointName>
      <apn>SERVICES.OPERATOR</apn>
      <apnid>3</apnid>
    </AccessPointName>
  </AccessPointNameData>
</Result>
```

69 COMPLETED (STARTED 2008-09-24 09:29:16)

3.2.2.2

Notification Only

Some CLI commands only return a notification to the client. The request and response flow is as shown in Figure 3.

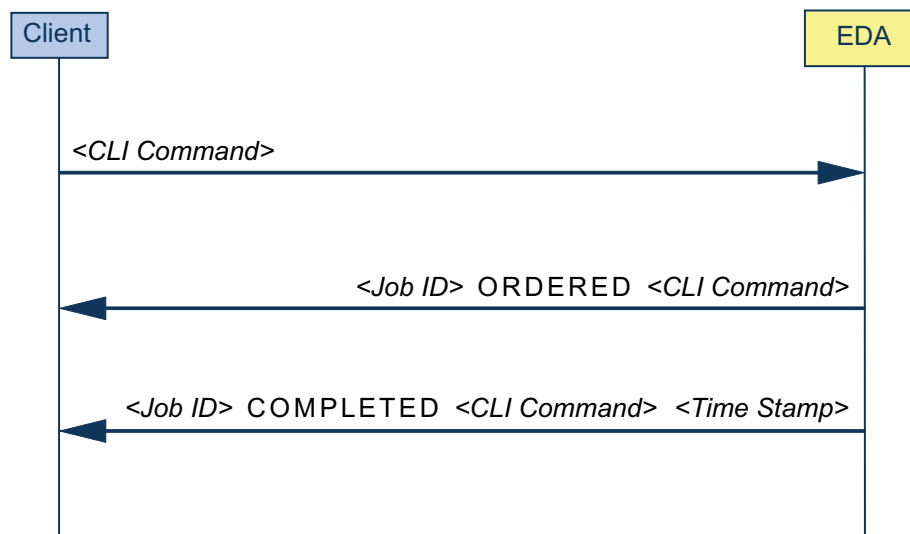


Figure 3 Message Flow for Commands with Notification Only

The following is an example of a command that returns only a notification:



```
hostname>HECDAPE:APNID=2;
```

```
55 ORDERED HECDAPE
```

```
55 COMPLETED HECDAPE (Started 2009-02-24 08:33)
```

3.2.3

Errors

The errors returned by the system can be divided in syntax errors and other errors.

Syntax Error

If there is a syntax error in the command, the request and response flow is as shown in Figure 4.

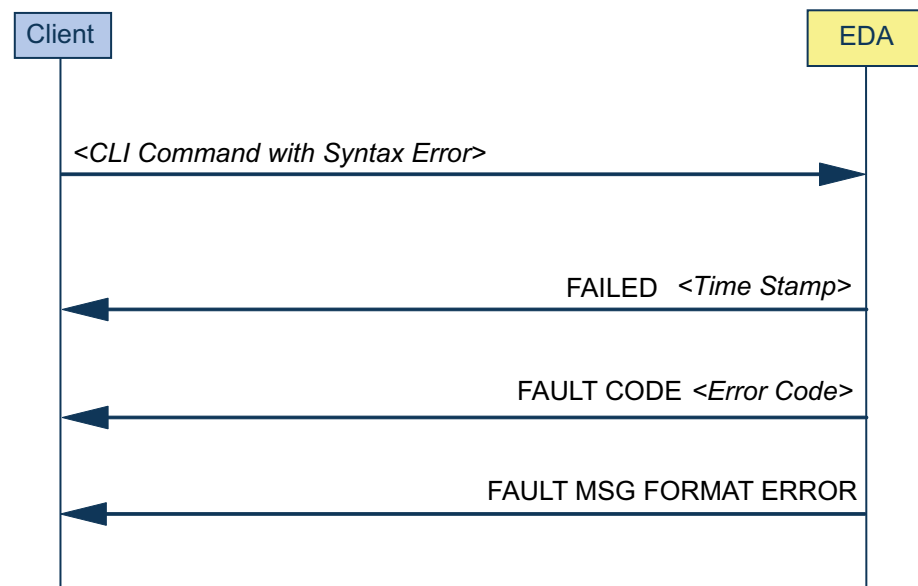


Figure 4 Message Flow for Message with Syntax Error

An example of a command containing a syntax error, and the response returned is shown as follows:

```
hostname>HECDBDI;
```

```
FAILED HECDBDI (Started 2008-11-21 09:20:53)
```

```
FAULT CODE 2001
```

```
FAULT MSG FORMAT ERROR
```



Other Error

If any other type of error occurs in the command, the request and response flow is as shown in Figure 5.

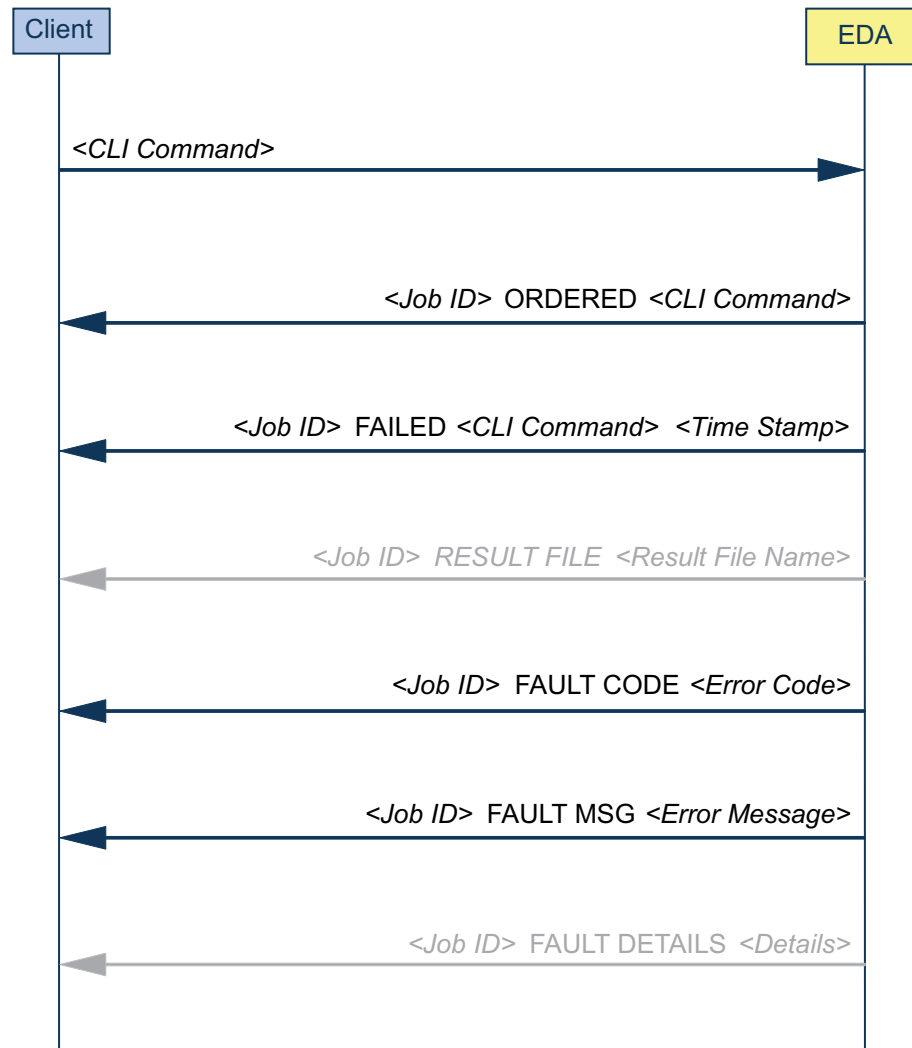


Figure 5 Message Flow for Any Errors Except Message Syntax Error

Note: The result filename is only returned if a result file has been created.

Not all failed requests return details.

An example of a command containing an error, and the response returned is shown as follows:

```
hostname>HEMSSDP:MSISDN=100000,ALL;
```

```
71 ORDERED HEMSSDP
```

```
71 FAILED HEMSSDP (Started 2009-03-16 10:06:54)
```



71 RESULT FILE 71_HEMSSDP_20090316_100654.xml.zip

71 FAULT CODE 1095

71 FAULT MSG Communication error while interacting with
a network element

71 FAULT DETAILS No functional connections for the network
element group [CUDB-MASSIVE-OPER-GROUP]





4 Control Commands

This section describes the CLI control commands. All these commands send their responses directly to the prompt as dynamic responses, see Section 3.2.2.1 on page 11.

4.1 Connect

If the connection between the client and Dynamic Activation is disrupted, the client can reconnect by using the `CONNECT` command. This can be done towards one or several jobs.

Request Syntax

Command Description:

```
CONNECT:JOBID=jobid...;
```

Note: Connecting multiple Job IDs can only be carried out using the “&” delimiter. Connecting Job ID ranges using “&&” is not possible.

The Job ID is an integer. It identifies a specific CLI job on the host.

Response Syntax

```
<jobid> ORDERED CONNECT
```

```
    <connected jobid> CONNECTED
```

```
<jobid> COMPLETED CONNECT
```

Example

```
hostname>CONNECT:JOBID=85;
```

```
86 ORDERED CONNECT
```

```
    85 CONNECTED
```

```
86 COMPLETED CONNECT
```



4.2 Cancel

The `CANCEL` command stops a job that is executing. The job cannot be resumed after this. The fault code 1100, `OPERATION CANCELLED` is inserted in the result file.

Note: The result file is not automatically deleted.

It is not necessary to be connected to a job to cancel it.

Request Syntax

Command Description:

```
CANCEL:JOBID=jobid...;
```

Note: Canceling multiple Job IDs can only be carried out using the “&” delimiter. Canceling Job ID ranges using “&&” is not possible.

Response Syntax

```
<jobid> ORDERED CANCEL
```

```
    <cancelled jobid> CANCELLED <command>
```

```
<jobid> COMPLETED CANCEL
```

Example

```
hostname>HEMSLIP:MSISDNS=546;
```

```
87 ORDERED HEMSLIP
```

```
hostname>CANCEL:JOBID=87;
```

```
88 ORDERED CANCEL
```

```
    87 CANCELLED HEMSLIP
```

```
88 COMPLETED CANCEL
```

4.3 Status

The command `STATUS` prints the status of the specified job. This command is applicable for completed and currently executing CLI commands.

Note: Canceled and failed jobs are shown as completed if their result files still exist. Completed jobs which have no existing result file are not saved. Thus, they do not show on a status request.



Request Syntax

Command Description:

$$\text{STATUS:JOBID}=\begin{bmatrix} \text{jobid...} \\ \text{ALL} \end{bmatrix};$$

Note: Checking the status of multiple Job IDs can only be carried out using the “&” delimiter. Checking the status of Job ID ranges using “&&” is not possible.

If ALL is entered, the status of all jobs is printed.

Response Syntax

<jobid> ORDERED STATUS

<jobid 1> <status>

<jobid 2> <status>

...

<jobid n> <status>

<jobid> COMPLETED STATUS

Note: The status can be one of the following:

- EXECUTING (Connected=<boolean>) (Started <YYYY>-<mm>-<dd> <hh>:<mm>:<ss>)
- COMPLETED

EXECUTING The job has started but not completed.

COMPLETED The job is finished.

Connected If true, the job is connected to the currently used client.
If false, use the Connect command to connect.

Example

hostname>STATUS:JOBID=ALL;

81 ORDERED STATUS

80 EXECUTING (Connected=true) (Started 2008-06-02
13:47:44)



```
69 COMPLETED
```

```
71 COMPLETED
```

```
78 COMPLETED
```

```
81 COMPLETED STATUS
```

4.4 Remove File

The command `RMFILE` removes the result file connected to a job. First fetch the file using `FTP`. Then use `RMFILE` to remove the file.

Request Syntax

Command Description:

```
RMFILE:JOBID=jobid...;
```

Response Syntax

```
<jobid> ORDERED RMFILE
```

```
<jobid> COMPLETED RMFILE
```

Example

```
hostname>RMFILE:JOBID=80;
```

```
89 ORDERED RMFILE
```

```
89 COMPLETED RMFILE
```

4.5 Exit

The `EXIT` command terminates the CLI session and disconnects the client from all jobs. The jobs are not canceled by the `EXIT` command.

Request Syntax

Command Description:

```
hostname>EXIT;
```




Response Syntax

LOGGED OFF





Reference List

Ericsson Documents

- [1] *Library Overview*, 18/1553-CSH 109 628 Uen
- [2] *System Administrators Guide for Native Deployment*, 1/1543-CSH 109 628 Uen

Protocol Specifications

- [3] *TELNET Protocol Specification*, RFC 854 ISI May 1983 J.Postel and J. Reynolds