Prepared (also subject responsible if other)		No.		
/ EEIPHAL		1/1553-APR 901 0253 Uen		
Approved	Checked	Date	Rev	Reference
LMI/ ROO/WB (Liam Dolan)	(EEIPHAL)	2009-01-20	L	

AMOS, Advanced MO Scripting, User Guide

Copyright

Disclaimer

Contents **Page**

1 About This Document

1.1 Purpose

This User Guide describes the basic functions of the Advanced MO Scripting (AMOS) tool, gives an overview of the concepts needed to use AMOS and also describes how to use the AMOS Command Line Interface (CLI) application. The AMOS CLI is for the administration of CPP Platform Network Elements.

1.2 Target Group

This document is intended for trained personnel responsible for the administration and management of Managed Object's on CPP Platform Network Element's (NE's).

1.3 Prerequisites

It is assumed that the user of this document:

- Is familiar with SUN workstations
- Has the correct permissions set in the Telecom Security Service (TSS) to perform AMOS operations, Refer to the TSS System Administrators Guide see Reference [3] and AMOS Command Authorization in the OSS for more information.
- Has the correct security privileges in order to access the Network Element refer to Network Element Authentication section for more information on this.
- Has a licensed AMOS product see AMOS Licensing in the OSS.
- Has proficient training in the usage of AMOS commands before attempting to execute any commands against a live Network Element.

It is assumed that the following components are functioning correctly in the OSS:

- Common Object Request Broker Architecture (CORBA) NameService
- Common Integration Framework (CIF) Configuration Service (CS)
- Telecom Security Service
- Self Management (SM) Log Service
- Sentinel License Manager

1.4 Typographic Conventions

The typographic conventions for all Customer Product Information (CPI) in OSS-RC are found in Reference [2].

2 AMOS Overview

Warning!

AMOS is an extremely powerful command line utility and if not used carefully may render a Network Element unusable. The responsibility is on the reader of this document to ensure that they read and fully understand all AMOS command descriptions and that they have proficient training in the usage of AMOS commands before attempting to run any commands on the Network Element.

2.1 Basic Functions of AMOS

AMOS is a text-based Operation and Maintenance (O&M) client providing access to the following services:

- Alarm Service (AS)
- Configuration Service (CS)
- File Transfer (FTP/HTTP)
- Inventory Service (IS)
- Log Service (LS)
- Notification Service (NS)
- OSE Shell (COLI)
- Performance Measurement Service (PM)

Access to all services is supported in both secure mode (secure CORBA, SSH, SFTP) and non secure mode (non secure CORBA, TELNET, FTP).

2.1.1 Alarm Service

The Alarm Service can be used to retrieve the list of alarms currently active on each Managed Object (MO). The list of active alarms can be retrieved with the commands **al** or **ala**.

Note: **al** shows an overview of all alarms whereas **ala** shows all the details. For a complete description of the **al** command, type **h al** at the AMOS prompt.

2.1.2 Configuration Service

The Configuration Service is used to read and change configuration data. Configuration data is stored in MO attributes. AMOS supports the following operations on the configuration service:

- *GetChildren* - To load all or parts of the MO-tree
- *GetAttribute* - To read the attributes of an MO
- *CallAction* - To perform an action on an MO
- *SetAttribute* - To set (change) the value of an MO attribute
- *CreateMO* - To create a new MO in the Network Element
- *DeleteMO* - To delete an MO from the Network Element

2.1.3 File Transfer

AMOS can download and upload files and directories to and from a Network Element (NE). Hyper Text Transfer Protocol (HTTP), File Transfer Protocol (FTP) or Secure File Transfer Protocol (SFTP) may be used.

2.1.4 Inventory Service

The Inventory Service allows AMOS to get a list of all Hardware (HW) and Software (SW) defined on the NE.

2.1.5 Log service

The Log Service allows AMOS to save a log of certain events such as changes in the configuration data, alarms raising and ceasing, NE or board restarts, Java Virtual Machine (JVM) events and Operation and Maintenance security events. AMOS supports fetching and parsing of the following logs:

- Availability log
- System log
- Event log
- Alarm log
- Command log
- Operation and Maintenance Security log
- Command Line log

- HW inventory log
- Java Virtual Machine (JVM) Event log (Upgrade log)

2.1.6 Notification Service

The Notification Service allows AMOS to subscribe and receive notifications from the Network Elements. This informs AMOS about parameter and alarm state changes in the MOs.

2.1.7 OSE shell

Any OSE shell command can be typed at the AMOS prompt and the output can be piped through external utilities if required.

The following is sample OSE shell input:

```
te log read
te log read | grep ERROR
```

2.1.8 Performance Measurement Service

The Performance Measurement Service is used to monitor statistics scanners or event filters. The statistics counters are stored in MO Performance Management (PM) attributes and are output to an XML file every fifteen minutes. The events are output into binary files every fifteen minutes. AMOS supports the following four operations from the performance management service:

- List Scanners and Event Filters
- Stop Scanner
- Resume Scanner
- Set Event Filter

2.2 Concepts

The following general concepts, listed in alphabetical order, are central for understanding and using AMOS:

- Command Line
- Command Piping
- Command Syntax, Regular Expression
- Managed Object
- Managed Object Model
- MO Tree And MO Naming Conventions

- Services And Protocols
- User Specific Settings

2.2.1 Command Line

The command line in AMOS uses the Readline library from bash. The following keys are supported:

Table 1 Supported function keys

Ctrl+F or right arrow	Move forward one character.
Ctrl+B or left arrow	Move backward one character.
up arrow	Previous command in history buffer.
down arrow	Next command in history buffer.
backspace	Delete one character backward.
Ctrl+D or delete	Delete one character forward.
Ctrl+A or home	Go to beginning of line.
Ctrl+E or end	Go to end of line.
Ctrl+U	Erase all characters backward.
Ctrl+K	Erase all characters forward.
Alt+F	Move forward one word.
Alt+B	Move backward one word.
Ctrl+INSERT	Copy to clipboard.
INSERT or shift-INSERT	Paste from clipboard.

Note: If you type the beginning of a command and then use the up/down arrow key, you will see all previous commands starting with this string.

2.2.2 Command Piping

It is possible to redirect the output of a command into the standard input of another command. The construct that does this is called the pipe, notated as **|**. Some OSE shell commands support piping, **lh, tg, str** being particular examples. This is usually indicated in the menu and the help for that command.

The following are examples of using a Pipe to search the Network Elements log-file for errors:

```
> te log read | grep ERROR
> lh mp te log read | grep ERROR
```

The following example uses a Pipe to search for a particular cell of a Network Element:

```
str | grep cell=30456
```

For other commands that don't support piping (like MO commands), the work around is to save the output to a log-file then run the UNIX command on that log-file by using the `l` command.

The following example, opens the log-file, an arbitrary name will be given, run the command and close the log-file and then run UNIX commands on the log-file. `$logfile` is a default variable that stores the name of the latest log created

```
l+
prod loadmodule
l-
l sort $logfile
l grep -i basic $logfile
```

Note: `$logfile` is automatically set by AMOS to contain the name of the latest log file created.

2.2.3 Command Syntax, Regular Expressions

2.2.3.1 How Managed Objects are identified

- **RDN - Relative Distinguished Name**

  This is used to identify an MO in relation to its nearest parent in the MO tree.

  The RDN contains MO Class (also called MO Type), the equal sign, and MO identity. Example:

  `AtmPort=MS-24-1`

  AtmPort is the *MO Class*, **MS-24-1** is the *identity*.

  For more information see Section 2.2.6 on page 17.

- **LDN - Local Distinguished Name**

  This is used to uniquely identify an MO within a Network Element.

  The LDN shows the hierarchy above the MO, within the Managed Element's MO tree. Example:

  `ManagedElement=1,TransportNetwork=1,AtmPort=MS-24-1`

  For more information see Section 2.2.6 on page 17.

- **FDN - Full Distinguished Name**

  This is used to uniquely identify an MO within a network. Example:

```
SubNetwork=AUS,SubNetwork=H2RG_0201,MeContext=St_Leon
ards_Station_2065010,ManagedElement=1,TransportNetwor
k=1,AtmPort=MS-24-1
```

For more information see Section 2.2.6 on page 17.

2.2.3.2 Addressing MOs in MO related Commands

The first argument in the MO-related commands is usually used to specify the MOs that should be affected by the command.

There are currently six different ways to specify the MO(s):

1 - all

All loaded MOs will be affected.

The follow example gets the attribute `UserLabel` on all MOs

```
get all userlabel
```

Note: Instead of **`all`**, it is also possible to use a wildcard such as **`.`** or **`*`**. This has the same effect.

2 - Proxy ID(s)

All MO(s) with the given proxy id(s) will be operated on.

To specify several MO proxies, there are two ways:

A. Specify each proxy id with a space in between. The following example prints MO proxies 0, 2 and 5:

```
pr 0 2 5
```

B. Give a range. The following example prints proxies from 4 to 10.

```
pr 4-10
```

The following example calls the action restart on MOs with proxy from 10 up to 20:

```
acc 10-20 restart
```

3 - Link handler (for PluginUnit and Spm MOs only)

The following example restarts the MO Subrack=MS,Slot=14,PlugInUnit=1:

```
acc 001400 restart
```

The following locks the first Spm on the Spb in slot 19 with LDN of Subrack=MS,Slot=19,PlugInUnit=1,Spu=1,Spm=1

```
bl 001900/sp0.lnh
```

Note: MOs start counting from 1 and link handlers start counting from 0.

4 - MO Group

MO Groups are user defined groups of MOs. All MO(s) belonging to the given MO group will be operated upon.

MO groups can be created using the commands **`ma/lma`**. MO groups can also be created with the commands **`hget/lhget`**, **`lk/llk`**, **`st/lst`**.

Note: In RNC, running the **`bo`** command will automatically create a number of MO groups containing the cc/dc/pdr device MOs for each module.

5 - Board Group

MOs (**PlugInUnit** or **Spm**) mapped onto the boards belonging to the given board group will be operated upon.

The following example places all boards with the software allocation matching “sccp” into a board group named “sccp”. Lock all PlugInUnit or Spm MOs connected to the boards of this board group “sccp”:

```
baw sccp sccp
bl sccp
```

In an RNC, use the default board groups created after running the bo command:

```
acc mod10 restart
pr dc10
acc dc10 restart #board group dc10 contains the Spm MOs
bl dc10dev #MO group containing the Device MOs
```

6 - MO Filter (Regular Expressions)

MO(s) whose LDN/RDN match the regular expression pattern will be affected.

If the command starts with **`l`** then the pattern will match against the LDN. If the command does not start with **`l`**, then the pattern will match against the RDN.

The following example gets MOs whose RDN contains ms-24-1:

```
pr ms-24-1
```

The following example gets MOs whose LDN contains ms-24-1:

```
lpr ms-24-1
```

Note: When using the mo-filter, it is a good idea to test the pattern with the **`pr`** or **`lpr`** command before issuing a **`get`**, **`set`**, **`acc`**, **`cr`** or **`del`** command, in order to see which MOs will be matched by the pattern.

Sometimes, a second or third argument can be given, which is usually a string matching the *attribute* or *attribute value* that you want to display.

2.2.3.3 Regular Expressions

Note: AMOS pattern matching is not case sensitive.

The search string that is used in the filters is a Unix Regular Expression (like the patterns used in **`grep -E`** command). Therefore, special meta-characters such as **`. * [ ] ^ $`** can be used.

Table 2 Short descriptions of the most common meta-characters.

.	Match any single character.
*****	Match 0 or more occurrences of the previous character.
[]	Match a character or range of characters inside the brackets.
[^]	Do NOT match character or character range inside the brackets.
|	OR.
^	Match from the beginning of the string.
$	Match from the end of the string.
!	Negation. Do not match.
%	Match in reverse order.
()	Group regular expressions.

Table 3 Sample meta-character usages.

a*	Match a or aa or aaa.
.*	Wildcard that matches 0 or more occurrences of any character.
[a-z]	Matches all letters from a to z.
[abe]	Matches letters a, b and e
[^3]	Matches any character but not 3.
3|5|6	Matches 3 or 5 or 6.
^a.*4$	Matches a string beginning with a and finishing with 4, with any character in the middle.
cell(11|23|45)	Group regular expressions together using brackets this matches cell11 or cell23 or cell45.

The following are examples of how Regular Expressions can be used:

```
Print all MOs that contain ms-24-1 and vp2 with any characters in between:
> lpr ms-24-1.*vp2

Print all MOs that contain ms-24-1 and vp2 with any characters in between in
reverse order:
> lpr %ms-24-1.*vp2

Print all MOs except those matching loadmodule or program:
> lpr !loadmodule|program

Print all MOs whose LDN contains 20, any number of characters followed by os:
> pr 20.*os

Print all MOS whose RDN contains cc and a numeric value of 1, 2, 3 or 4:
> pr cc[1-4]

Print all MOS whose RDN contains cc and a numeric value of 1, 3 or 5:
> pr cc[135]

Print all MOs that contain the value =6 followed by any character and contain
the value prog followed by any character and contain the value =1:
> lpr =6.*prog.*=1

Print all MOs that contain the value =6 followed by any character and contain
the value prog followed by any character and end in the value =1:
> lpr =6.*prog.*=1$

Print all MOs that contain the value ms-24-1:
> lpr ms-24-1

Print all MOs that contain the value ms-24-1 followed by any character that
contains =vc, but that does not contain vc3:
> lpr ms-24-1.*=vc[^3]
```

2.2.3.4 How to specify attribute values in set, cr and acc command

Struct

For attributes of type **Struct**, use the following syntax:

`attr1=val1[,attr2=val2[,attr3=val3]]]...`

The following example specifies Struct attributes in the set, cr and acc commands:

```
set sid sib11 sib11repperiod=128

set mtp3bspitu sppriority prioslt=2

set mtp3bspitu sppriority prioslt=2,prioco=2
```

MORef

For attributes of type **MoRef**, just type the MO LDN (without **ManagedElement=1**).

```
lset AtmPort=1221,VplTp=vp1 atmTrafficDescriptor,
transportnetwork=1,atmtrafficdescriptor=C1P4500
```

It is also possible to skip the first parent (for example **TransportNetwork** or **SwManagement**).

```
cr rncfunction=1,utrancell=30451,utranrelation=30451to30521
```

Array of MORef

For attributes of type **array of MoRefs**, separate each element of the array with spaces.

```
set jvm admclasspath  loadmodule=oms loadmodule=vbjorb \
 loadmodule=asms_sharp loadmodule=iaik \
loadmodule=cma \ loadmodule=roa

acc aal2pathdistributionunit=1 addPath
```

Array of Struct

For attributes of type **array of Struct**, separate each element of the array with semicolons.

```
set rncfunction aliasPlmnIdentities mcc=300,mnc=23,\
mnclength=2;mcc=345,mnc=32,mnclength=2; \
mcc=208,mnc=123,mnclength=3
```

Array of Integer, Float, String or Boolean

For attributes of type **array of integer/long/float/string/boolean**, separate each element of the array with commas.

```
set antfeedercable=6 ulattenuation 10,10,10,10,10,10,10,10,10

set jvm options  -Xms65000k,-Dhttp.root=/c/public_html,\
 -Dse.ericsson.security.PropertyFileLocation= \
/c/java/SecurityManagement.prp

set cell=3041 ActiveOverlaidCDMAChannelList \
true,true,false,true

acc managedelementdata addDhcpServerAddress
```

Empty value

To input an empty value in **`set`** command, just leave the value field blank.

```
set 0 userlabel

set reliableprogramuniter admpassiveslot
```

In the **`cr`** command, type **`null`** if it's an **MoRef** or **`d`** for other attributes. This is only supported for non-mandatory (restricted) attributes, because mandatory attributes must have a value.

In the **`acc`** command, type **`null`**. This is only supported for parameters of type **MoRef** or **String**.

2.2.4 Managed Object

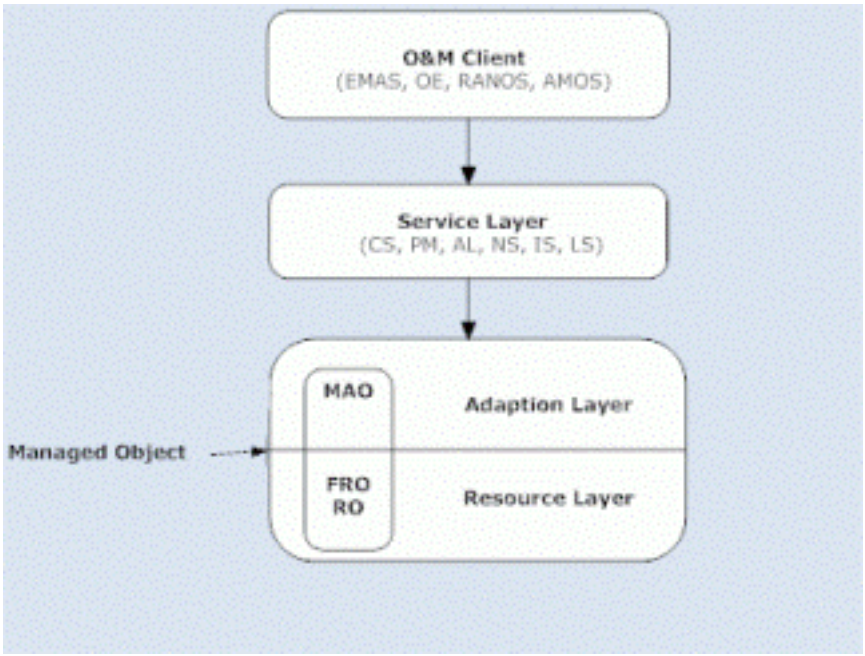

Figure 1 The relationship between Element Managers (such as AMOS, EMAS, RANOS), the Service Layer and the MOs, FROs and ROs

The Operation and Maintenance client can access the Managed Objects (MO) through a number of services:

- Configuration Service (CS): To read and change configuration data. Configuration data is stored in the MO attributes.
- Performance Measurement (PM): to setup statistics scanners or event filters. The statistics counters are stored in MO pm-attributes and output to an XML file every 15 minutes. The events are output into binary files every 15 minutes.
- Alarm Service (AS): to retrieve the list of alarms currently active on each MO.
- Notification Service (NS): to subscribe and receive notifications from the Network Element, informing about parameter/alarm changes in the MO's.
- Inventory Service (IS): to get a list of all HW and SW defined in the Network Element.
- Log Service (LS): to save a log of certain events such as changes in the configuration, alarms raising and ceasing, Network Element and board restarts and JVM events.

The MO is a way of modeling resources in a Cello Network Element. It consists of:

- A *Management Adaptation Layer* which is implemented in JAVA, in the MP running the JVM (the Operation and Maintenance MP). The purpose of the MAO (Management Adaptation Object) is to interface towards the various Operation and Maintenance services described above.
- A *Resource Layer* consisting of Facade Resource Object (FRO) and a Resource Object (RO) which are implemented in C and run on the various boards. The RO is the actual resource modeled by the MO. The purpose of the FRO is to act as an interface between the MAO and the RO, by handling the configuration transactions and storing configuration data for the RO.

2.2.5 Managed Object Model

Each MO class contains a number of attributes which are used to store *configuration data* or *performance measurement* data.

Each MO class can also support a number of defined *actions*. These represent certain operations which can be performed by the MO. A typical example is the action **restart** which will cause the MO to restart the resource it is responsible for (a board, a program, etc).

The *Managed Object Model* (MOM) is a reference document describing all the MO Classes that can exist in a Network Element, together with their *attributes* and *actions*.

The format of the MOM can be UML, XML, HTML, or MS-Word.

The XML version of the MOM is usually stored on the web server of the Network Element at the address: `http://<ipaddress>/cello/oe/xml/<filename>.xml`

2.2.6 MO Tree and MO Naming Conventions

2.2.6.1 LDN – Local Distinguished Name

The MOs are organized in a hierarchical structure.

Each MO instance is uniquely identified in the Network Element by its *Local Distinguished Name* (LDN).

The highest MO in a Network Element, the so called *root MO* is the *ManagedElement*. This MO represents the whole Network Element.

There is only one instance of the ManagedElement MO in the Network Element and it is referenced by the LDN: **ManagedElement=1**.

The string at the left of the equal sign is called the MO class (or MO type) and the string at the right of the equal sign is called the MO identity. In the case of the root MO, the *MO class* is **ManagedElement** and the *identity* is **1**.

If an MO is located further down in the MO tree, the LDN must contain the MO classes and identities of all the parents of that MO, in a sequence going from the root MO down to the MO in question. See example below:

```
ManagedElement=1
ManagedElement=1,Equipment=1
ManagedElement=1,Equipment=1,Subrack=MS
ManagedElement=1,Equipment=1,Subrack=MS,\
Slot=19
ManagedElement=1,Equipment=1,Subrack=MS,\
Slot=19,PlugInUnit=1
ManagedElement=1,Equipment=1,Subrack=MS,\
Slot=19,PlugInUnit=1,Program=DbmFpgaLoader
```

From this example, we can see that the ManagedElement has a child called **Equipment=1** which has a child called **Subrack=MS** (representing the main subrack of the Network Element), which has a child called **Slot=19** (representing the slot in position 19), which has a child called **PlugInUnit=1** (representing the board located in that slot), which has a child called **Program=DbmFpgaLoader** (representing one of the programs loaded in that board).

The LDN of the lowest MO (the one called **Program=DbmFpgaLoader**) contains the address of all successive parents of that MO all the way up to the **ManagedElement**.

2.2.6.2 RDN – Relative Distinguished Name

The string located at the far right of an LDN, just after the last comma, is called a *Relative Distinguished Name* (RDN).

It is a unique way of addressing an MO instance in relation to its closest parent.

This means that there is only one MO instance with the RDN **Program=DbmFpgaLoader** under the parent MO M**anagedEleme nt=1,Equipment=1,Subrack=MS,Slot=19,PlugInUnit=1**. However, there could be another MO instance with the same RDN under a different parent MO. For instance, there could be an MO instance with the RDN **Program=DbmFpgaLoader** under the parent MO M**anagedElement=1,Equipment=1,Subrack=MS,Slot=23,PlugInUnit=1**.

Therefore the RDN is a relative way of addressing an MO instance.

2.2.6.3 FDN – Full Distinguished Name

When a Network Element is connected to a Network Management System such as OSS-RC, there is a need to uniquely address each MO within the whole network. The *Full Distinguished Name* (FDN) adds a network element prefix in front of the LDN of each MO instance in order to specify which Network Element this MO belongs to.

2.2.7 Services and Protocols

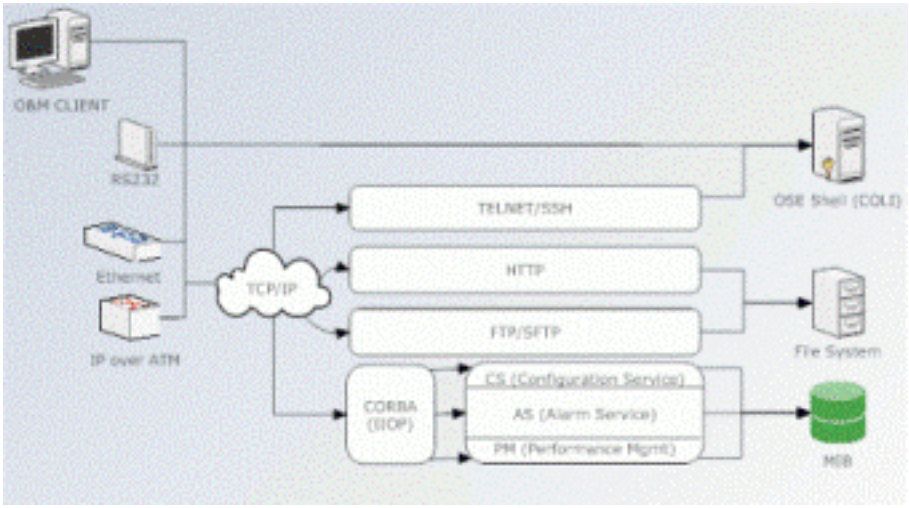

Figure 2 CPP nodes have various access methods for different services. For Managed Services like CS, CORBA is used. For the command shell, SSH or TELNET is used. To collect PM XML ROP files FTP is used.

The MO services described above (CS, AS, PM, IS, NS) are carried by the IIOP protocol (Internet Inter-ORB Protocol), also called CORBA (Common Object Request Broker Architecture). At startup, the Network Element generates its IOR (Interoperable Object Reference) and stores it in a **nameroot.ior** file which can be used by the Operation and Maintenance client to access the Network Element. The IOR file contains the Network Element's IP address and some specifications as to how the clients can send requests to the Network Element. The OSE-shell (also called COLI) can be accessed through telnet/ssh or locally, through the serial port (RS232). It gives direct access to the operating system, the file system, and parts of the FRO/RO layer.

3 Getting Started

This chapter is for new or infrequent users of the AMOS CLI. It describes the following:

- AMOS Administration
- Launching AMOS
- Stopping AMOS

3.1 AMOS Administration

3.1.1 AMOS Licensing in the OSS

The new AMOS feature is a licensed feature within the OSS. This means that AMOS cannot be used without a valid license being present in the OSS. The user can check to see if the AMOS feature is licensed by launching AMOS see Section 3.2 on page 21. If AMOS is not licensed the user will see the following message

```
error : license control - AMOS license check failed,
license server does not recognize the given feature
```

when they attempt to launch AMOS or when they attempt to run their first command.

3.1.2 AMOS Command Authorization in the OSS

In order for a user to be able to run an AMOS command in the OSS, they must have the correct authorization privileges configured in the OSS. AMOS will use three activity-sets in the OSS; these activity-sets have been defined as AMOS_READONLY, AMOS_WRITE and AMOS_TELNET. It is assumed that each activity-set is inclusive meaning that if a user has AMOS_WRITE access they also have AMOS_READONLY access. On each command description in this user guide a ACTIVITY_SET section is defined which shows what ACTIVITY_SET an operator must belong to in TSS in order to be in to run the command. If a user is not authorized to run a command they will see the following message

```
error : access control - user not authorized to execute command
```

3.1.3 Network Element Authentication

CORBA security is supported on AMOS and related utilities. To run the applications using CORBA Security, a valid credential file must be fetched in the OSS. There are two types of credentials that can be used:

- standalone credential `sam.pbe` file.
- single logon credential file `ssucredential.xml`.

both files are generated by the SLS server together with a standalone password. These credential files have a short validity period between one day and one week usually. The validity period is determined by the SLS server when generating the file. Depending on the properties of the credential file, it might be possible to communicate to any Network Element in the network or only certain Network Elements again this is determined by the SLS server when generating the credential. From the users point of view, there is no difference between running against a secure or unsecure Network element except for the fact that different user variables must be set when starting AMOS. This is discussed in depth in the Starting AMOS section. The discussion of the initialization of the `sa_credential` file and the `sa_password` are beyond the scope of this document, for a detailed step by step guide to fetching these security settings please see Reference [4]

3.2 Starting AMOS

Different Network Elements are configured differently in separate operator domains so therefore they need different security/connection variables to be set upon startup of AMOS to successfully connect and manage different Network Elements. Each of the following sections assumes that you have the main AMOS command shell started as is described in the following procedure:

Table 4 Launching AMOS

Step	Action	Comment
1	In **LTE-WCDMA Common Explorer**, select **<tools> <Advanced MO Shell>**	This opens the **AMOS** main command shell.

Note: AMOS cannot be started from the master server. If AMOS is started from the master server the user will get the following error:

```
error : user authentication - authentication check failed during logon (AMOS
cannot be run from the master server)
```

Note: In the following sections user variables are set by starting AMOS with the `-v` construct. This is to provide clarity to the user, any variable that can be set with this construct may also be set by placing it in the `~/moshellrc` file.

3.2.1 Starting AMOS against an unsecure Network Element

When running AMOS against an unsecure Network Element the user does not need to set any and user variables. These are set automatically by AMOS.

Table 5 Starting AMOS against an unsecure Network Element

Step	Action	Comment
1	Run the UNIX command **`moshell <Network Element name>`**	This will launch **AMOS** against the specified unsecure Network Element.
alternatively		
1	Run the UNIX command **`moshell <ip address>`**	This will launch **AMOS** against the specified IP Address.

```
> moshell 127.0.0.1
```

```
> moshell atrnc1
```

Note: In order to simplify each of the subsequent examples AMOS will be started with the command **`moshell <Network Element name>`**

please be aware that the command **`moshell <ipaddress>`** can also be used.

Note: When AMOS first starts up, it is neither connected to the Configuration Management Service or the Performance Management Service. AMOS uses lazy initialization in that it will only connect to the Configuration Management Service or the Performance Management Service on the Network Element when these services are first used. So to connect to Configuration Management Service, the `lt` command or the `re` command may be used. As soon as AMOS has connected to the Configuration Management Service the `userlabel` attribute of `ManagedElement` will be read and prompt will be set to this value. To connect to Performance Management Service, the `pst` command may be typed as this will list all scanners defined on the Network Element and in so doing AMOS will connect to the Performance Management Service.

3.2.2 Starting AMOS against an unsecure Network Element that has secure TELNET and secure FTP turned on

Depending on the security configuration settings defined by an operator the TELNET and FTP operations may be replaced by secure FTP (SFTP) and secure TELNET (SSH) operations. AMOS can interact with these Network Elements if the `secure_shell` and the `secure_ftp` variables are turned on as in the example below this is known as **secure-port** mode.

Table 6 Starting AMOS in secure-port mode

Step	Action	Comment
1	Run the UNIX command **`moshell -v secure_shell=1,secure_ftp=1 <Network Element name>`**	This will launch **AMOS** against a Network Element that has secure TELNET and secure FTP turned on.

```
> moshell -v secure_ftp=1,
secure_shell=1 atrnc2
```

3.2.3 Starting AMOS against a Network Element with CORBA security level 2 turned on

Note: In the following example a secure FTP and SSH environment on the Network Element is assumed. Although this is the most likely scenario some operators may not have secure FTP or SSH engaged in such a case remove the `secure_shell=1` and the `secure_ftp=1` setting on the command line. This will allow AMOS to connect to the FTP port and to the TELNET port on the Network Element.

An operator may have CORBA security level 2 engaged on the Network Element that AMOS wishes to contact. This means that all operations occurring via CORBA on the Network Element will be treated securely. When an operator has CORBA security enabled AMOS can be viewed as needing to be started in

sec2-mode. In order to start AMOS in this mode the user must supply both a `sa_credential` file location and a `sa_password` to AMOS, the credential file must be fetched from the Single Logon Server (SLS) in the OSS see Reference [4], AMOS can then be started as in the following example:

Table 7 Starting AMOS in sec2-mode

Step	Action	Comment
1	Run the UNIX command **`moshell -vcorba_class=3,secure_shell=1,secure_ftp=1,sa_credential=/home/big_admin/Ericsson/OMSec/sam.pbe,sa_password=amos09 <Network Element name>`**	This will launch **AMOS** against a Network Element that has secure TELNET and secure FTP and CORBA security level 2 turned on.

```
> moshell -v corba_class=3,secure_shell=1,secure_ftp=1,
sa_credential=/home/big_adm/Ericsson/OMSec/sam.pbe,
sa_password=somA&9 atrnc3
```

3.2.4 Starting AMOS against a Network Element with CORBA security level 3 turned on

An operator may have CORBA security level 3 engaged on the Network Element that AMOS wishes to contact. This means that all operations occurring via CORBA on the Network Element will be treated securely. When an operator has CORBA security enabled AMOS can be viewed as needing to be started in **sec3-mode**. In order to start AMOS in this mode the user must supply the following to AMOS: the `corba_class` variable must be set to a value of 4, a `secure_shell=1` setting, a `secure_ftp=1` setting, the `sls_username` and `sls_password` variables must be set to the user's username and password in the SLS server, and `username` must be set to the correct user name that will be used to login to the node for COLI and file transfers setting.

Note: When `corba_class` is set to 4 this also starts AMOS in **auto-fetch**. In this mode AMOS is configured to automatically fetch the security credential files from the Single Logon Server and to automatically configure itself with the users security setting.

AMOS can be started as in the following example:

Table 8 Starting AMOS in sec3-mode

Step	Action	Comment
1	Run the UNIX command **moshell -vcorba_class=4,secure_shell=1,secure_ftp=1,sls_username=bigadmin,sls_password=somA09,username=correctusername <Network Element name>**	This will launch **AMOS** against a Network Element that has CORBA security level 3 turned on.

```
> moshell -v corba_class=4,secure_shell=1,secure_ftp=1,
sls_username=bigadmin,sls_password=somA09,
username=correctusername atrnc3
```

3.2.5 Starting AMOS in already fetched mode

Security credential files operate on a time limited basis that usually ranges from 1 day to several weeks, once AMOS has automatically fetched the security credential files from the Single Logon Server if the credential file is still within its validity period AMOS may be started in **already-fetched** mode. In this mode the security credential file downloaded from the SLS server is still valid and the user needs only to supply two user variables when starting AMOS `corba_class` at a value of 5 and the `nm_credential` variable which states where the security credential file was downloaded to.

Table 9 Starting AMOS already-fetched mode

Step	Action	Comment
1	Run the UNIX command **moshell -v corba_class=5,nm_credential=/somefolder/ssucredentials.xml <Network Element name>**	This will launch **AMOS** in already-fetched mode in that AMOS will use the security credentials stored in the `nm_folder` directory for all subsequent security negotiations.

```
> moshell -v moshell -v corba_class=5,nm_credential=//ssucredentials.xml atrnc4
```

3.2.6 Starting AMOS in offline mode

AMOS may also be started in **offline-mode**. Offline mode means that AMOS is not connected to the Network Element.

Offline mode is usually used in three distinct circumstances:

- Browsing the Managed Object Model when used in conjunction with the **parsemom** command.
- Browsing the help when used in conjunction with the **help** command.

- Browsing the default user variables settings by entering the **uv** command.

The following command starts AMOS in offline mode:

```
> moshell
```

Offline mode can also be used against an MO dump. An MO dump is produced by running the commands **lt all; kegt**, it contains the printout of all MO attributes of a node.

The following starts AMOS against an MO dump.

```
> moshell /path/to/modump
```

3.3 Stopping AMOS

AMOS may be exited in the following manner.

Table 10 Stopping AMOS

Step	Action	Comment
1	In **AMOS** main command shell run the command **q**, **quit**, **exit** or **bye**	This stops **AMOS** running.

4 Customizing the AMOS Environment

AMOS allows the user the freedom to customize their environment as they require. The customizing options are discussed below. For more information on all AMOS commands type **h <command name>** see Chapter Accessing Helpfor more detail

4.1 Displaying or Changing AMOS User Settings

There are a number of AMOS configuration parameters (called *user variables*) which can be set either permanently or on a session basis. These settings have a default value which is defined in the **moshell** file. If one or more user variables need to be changed from the default value, it is recommended to store the new setting in the `~./moshellrc` file located at **~/.moshellrc**, this way, the new setting will be kept even after an AMOS upgrade.

To display or change AMOS configuration settings using the following command:

```
> uv
```

This command can also be used to change the value of variables within the AMOS session.

Note: For any user variable that begins with `amos_` they must be set with the `-v` command line setting for them to take effect in AMOS, this is due to the fact that AMOS is a process whose variables can only be defined during startup.

It is also possible to define user variables on a session basis by:

1. Using the command **`uv [var=value]`** from the AMOS prompt (type **`h uv`** for more information).

2. Use the **`-v`** option from the command line when starting AMOS (type **`moshell`** on its own for more information)

3. Each user can define their own settings and aliases and save them into the **~/.moshellrc** file in their home directory. If the file is not present, it will be created automatically and can be modified any time. All user variables can be given a new value in the **~/.moshellrc** file.

4. A global **moshellrc** file also exists at the location `/var/opt/ericsson/amos/jarxml/moshellrc`. In this file a user can specify all user variables that they wish to exist for all users of AMOS. This file exist so that each user in the OSS does not have to specify user variables that may be common to all users of AMOS on the OSS.

For a complete listing of AMOS user variables and their default values see section AMOS User Specific Variables.

4.2 Changing the AMOS Command Line Interface

The following sections outline how to customize the AMOS Window and the AMOS Command Line Interface.

4.2.1 Changing the AMOS prompt

By default the `userLabel` attribute of the `ManagedElement` Managed Object is displayed at the AMOS prompt. To change the AMOS prompt use the following command:

```
> p
```

4.2.2 Changing the AMOS window title

By default the AMOS window title displays descriptive text with the Network Element Name and the **Stopfile** location. To customize the AMOS window title use the following command:

```
> w
```

4.2.3 Changing the AMOS prompt and AMOS window title

To change both the AMOS prompt and the AMOS window title at the same time use the following command:

```
> pw
```

4.2.4 Making the AMOS prompt bold

To change the command prompt to appear as bold or to turn the bold font off use the following command:

```
>b
```

Note: There is a limitation when the prompt is bold that command lines that are longer than the screen width do not wrap correctly.

4.3 Toggling On and Off the Printing of the Proxy Identities

To toggle on and off the display of proxy identities in the printout of **get mo attribute** commands use the following command:

```
> prox
```

4.4 Toggling On and Off the Line Color of the Proxy Identities

To toggle on and off the display of colors when running the **te log read** or the **cabrd** printouts use the following command:

```
> col
```

4.5 Toggling On and Off User Label

To toggle on and off the display of a **UserLabel** column in the **st/lst** and **pget/lpget** printouts us the following command:

```
> ul
```

By default the **UserLabel** column is not shown.

4.6 Toggling On and Off Confirmation on MO Commands

Caution!

By default certain commands require **Are you Sure [y/n] ?** or **Please Confirm [y/n]:** confirmation in the system, although it is possible to remove this confirmation with the following commands it is not advisable to do this.

The user may wish to disable the default confirmation behavior in AMOS. Confirmation may be disabled separately for the following groups of commands:

- **bl**, **deb**, **set** and **acc** commands.
- **lt** and **lc** commands.
- **del** and **rdel** commands.

4.6.1 Disabling confirmation for bl, deb, set and acc commands

To disable or enable the confirmation message when running a **bl**, **deb**, **set** or a **acc** command use the following command:

```
> confb
```

4.6.2 Disabling confirmation for lt and lc commands

To disable or enable the confirmation message when running a **lt** or a **lc**, command use the following command:

```
> confl
```

4.6.3 Disabling confirmation for bl, deb, set, acc, lt and lc commands

To disable or enable the confirmation message when running a **lt**, **lc**, **bl**, **deb**, **set** or a **acc**command use the following command:

```
> confbl
```

4.6.4 Disabling confirmation for del and rdel commands

To disable or enable the confirmation message when running a **del** or a **rdel**, command use the following command:

```
> confd
```

4.7 Toggling Display of Old and New Values for Attributes

This section describes how to toggle on and off the displaying of old and new attribute values when running the **set**, **bl** and **deb** commands.

Note: The set will not take place if the new attribute value is the same as the old attribute value. The result **No Change** is printed in output. If the set is accepted but the final attribute value is still the same as the old attribute value, then the result **Fail** is printed.

4.7.1 Old value is displayed before setting the attribute

To enable or disable the displaying of the old attribute value before an attribute has been set use the following command:

```
> gs
```

4.7.2 Old value is displayed before setting the attribute and the new value is displayed after setting the attribute

To enable or disable the displaying of the old attribute value before setting an attribute value and to display the new attribute value after setting the attribute value use the following command:

```
> gsg
```

4.8 Printing and Defining Command Aliases

Aliases in AMOS allows the user to rename a command or to type something simple at the command line rather than half a dozen options. Aliases may be defined on the command line or in your **~/.moshellrc** file.

4.8.1 Listing all currently defined aliases

To list all defined aliases use the following command:

```
> alias
```

This command can also be used to define aliases on the command line.

4.8.2 Defining an alias on the command line

Aliases may be defined for the current AMOS session on the command line using the following syntax:

```
> alias gsus pst gpeh . susp
```

4.8.3 Defining an alias in the .moshellrc file

Aliases may be added permanently in your **~/.moshellrc** file, using the same syntax as in the previous chapter

4.8.4 Removing an alias on the command line

To undefine an alias use the following command:

```
> unalias
```

4.9 Adjusting the Load Speed

It is possible to adjust the speed of loading Managed Objects from the Network Element. This is performed by the speed . It is possible to use a value from 1 to 200 to define the speed. It is recommended to not use a speed higher than 100 since this takes more memory from the Network Element. Type **speed** on its own to see the current speed.

4.9.1 Problems reading Managed Object Attributes

When performing a **get mos** or **get mos all** one CORBA request is sent for each Managed Object, asking for all attributes of that Managed Object but when you are performing a **get mos attributes** one CORBA request is sent for up to `100` Managed Objects at a time asking for the specified attributes of those Managed Objects. This means the call is much faster but if one or more attributes cannot be read due to some problem for example a **fRO Not Accessible** fault, then all attributes will return the same exception. The work around for this problem is to find out which attribute is causing the problem by running the **sget** command which reads each attribute one by one therefore the attributes that are causing the exception will then be easy to spot. The user may then use the standard **get** command with the negative filter **!** to exclude the faulty attribute as in the following example which gets all attributes of the `NodeBFunction` Managed Object except those matching the faulty `overload` attribute:

Example 1

```
> get nodebfunction !overload
```

As a CORBA request is sent for up to `100` Managed Objects at a time another problem may occur if one or more of the Managed Objects contain some attributes that cannot be read, then all Managed Objects within that CORBA request will return the same exception even if they can be read, the solution to this problem is to lower the `speed` of reading the Managed Objects from the Network Element so that only `1` Managed Object instead of `100` is read per CORBA request. This is done using the following command:

```
> speed 1
```

The command will be slower but the exception will only affect the Managed Object(s) that are faulty and not the healthy ones. The **speed** command affects the **get**, **pget**, **kget**, **prod**, and **st** commands.

5 AMOS Command Syntax Overview

Note: All MO CLI command syntax options are case sensitive.

Note: In the command examples an example of the output of the command is shown but very often not all of the output of the command can be displayed, in this case the three dots " . . . <cut>..." is placed in the output to imply to the user that there is more output.

The following chapters provide a very high level overview of the syntax of AMOS commands and a high level description of their capabilities.

5.1 AMOS Basic Command Overview and Syntax

```
mom[tcd]              Print description of MO classes.
lt/ltc[1-9]           Load MO tree (full or partial) and build proxy table.
lc[1-9]/lcc           Load MO tree (full or partial) and build proxy table.
pr/lpr                Print MO LDNs and proxy ids currently loaded in amos.
ma/lma                Add MO(s) to an MO group.
mr/lmr                Remove an MO group or remove MOs from an MO group.
mp                    Print all defined MO groups.
get/lget              Read CM/FM attribute(s) from MO(s).
hget[c]/lhget[c]      Read CM/FM attribute(s) from MO(s), print horizontally one
                      line per MO (instead of one line per attribute).
kget/lkget            Display CM/FM attributes in exportable printout format.
* fro/lfro[m]           Read MAO data of an MO and/or froid of the MO.
st/lst                Print state of MOs.
prod                  Print productData of MO(s).
lk/llk                View all MO's linked to an MO, and their states.
lko/llko              The old lk.
set[m]/lset[m]        Set an attribute value on one or several MO's.
rset/lrset            Set attribute value on a restricted attribute.
bl[s]/lbl[s]          Lock or soft-lock MO(s).
deb/ldeb              Unlock MO(s).
acl/lacl              Lists available MO actions.
acc/lacc              Execute an MO action.
cr                    Create an MO.
del/ldel              Delete MO(s).
rdel/lrdel            Delete MO(s) together with children and reserving MOs.
u+[s]/u-/u?/u!        Handling of undo mode
                      (for undo of del/rdel/set commands).
run                   Run a command file in amos format.
trun[i]               Run a command file in EMAS/MoTester format.
ctrl-z                Abort an MO command or a "for" loop.
pol[shcu]             Poll the Network Element until the CM service is up.
re[i]                 Disconnect and reconnect to the CM/PM service.
getmom                Check the MOM version currently stored on the
                      Network Element.
parsemom              Parse an xml MOM file.
ld                    Load one MO from the tree and add to the proxy table.
sget/lsget            Read CM/FM attributes from MO(s), one by one ("Slow" get).
```

Note: ***** - These commands will not function correctly in P6 RBS's

5.2 AMOS Managed Object Command Overview and Syntax

```
cvls/cvmk/cvms/cvset/cvrm/cvcu/cvget/cvls1  ConfigurationVersion (CV) handling.
inv[hr]               Complete HW/SW inventory.
cab[slxradgtme]       Display of miscellaneous COLI printouts.
stc[p][r]             Display state and configuration of AtmCrossConnections.
std                   Display state and configuration of devices.
* stv[b][r]             Display state, user, and bandwidth usage for ATM ports.
stt[r]                Display state and user of Physical Ports and Ds0Bundles.
hc                    Run a general healthcheck on the Network Element.
diff/ldiff            Parameter auditing or MO dump comparisons.
str[12ft]             Print status of the IubLinks/AbisLinks(RNC/BSC only).
lki                   Resource usage and configuration of IubLinks (RNC only).
remod[u][2]           Remodule an IubLink (RNC only).
tg[r][c][d]           Print Resource Object information for MOs in LmCell.
uer[d][t]             Print UE context data for all active calls (RNC only).
ced[h][r]             Print dynamic cell data or capacity usage <RNC only>
al[atkc]              Print the list of active alarms.
lg[aevsmircdyuolhf]   Fetching and processing of logs (alarm, event, etc)
dcg[mesra]            Fetching data for TR's/CSR's
```

Note: ***** - These commands will not function correctly in P6 RBS's

5.3 AMOS Administration Command Overview and Syntax

```
uv                    Print or change amos configuration settings.
pv                    Print scripting variables.
!/l                   Execute a unix command on the PC/workstation.
l+[m][m][s][o]/l-/l?  Open/close amos logfiles.
ose/coli command      Send a COLI command to the Network Element's OSE shell.
bo[r]/ba[swdp]/br[wd]/bp  Manage board groups.
proglist              List programs on a board or board group
progkill              Restart a program on a board or board group
lh                    Run COLI commands on all boards of a board group.
mon/mon+/mon-/mon?    Start/stop/check the target monitor server on the.
* sql+/sql-/sql?        Start/stop/check the SQL client on the Network Element
                      (CXC1325608).
pgu[c][f][r]          Program Upgrade. For STP use only, eg, to load black LMs.
ftree                 Recursive listing on the Network Element's file system.
ftget[c]/ftput[c]     Transfer files or directories to/from the Network Element,
                      using ftp.
htget                 Transfer files from the Network Element using http.
edit                  Edit a file on the Network Element.
fclean[f|ff|d|e]      Removal of obsolete loadmodules on the Network Element.
hi                    Print history of amos commands entered.
lmid[c]               Print translation of loadmodule product number.
p/w/pw/b              Change amos prompt and/or window title.
prox                  Toggle display of proxy identities in printout.
col                   Toggle display of colors.
ul                    Toggle display of userlabel in st/lst and pget/lpget.
conf[bld]             Toggle confirmation on various MO commands.
gs/gsg                Toggle display of old/new attribute value in set/bl/deb.
ip2d/d2ip             Convert an IP address into the format used in the fRO.
h2d/d2h               Convert an integer to hexadecimal or viceversa.
h2b/b2h               Convert a binary to hexadecimal or viceversa.
wait                  Specify a delay in hrs, mins, secs, or rops.
return                Exit from a command file without exiting from amos.
print                 Print a line or variable (scripting).
alias/unalias         Print or define command aliases.
q/bye/exit/quit        Exit amos.
```

Note: ***** - These commands will not function correctly in P6 RBS's

5.4 AMOS Performance Management Command Overview and Syntax

```
pmom[acd]/lmom[c]     Print description of PM counters (pmom).
pget/lpget            Read PM attribute(s) from MO(s).
spget/lspget          Read PM attribute(s) from MO(s) one by one (slow pget/lspget)
hpget[c]/lhpget[c]    Read PM attribute(s) from MO(s).
pdiff/lpdiff          Print incrementation of PM attributes.
pmx[hfdn]             Display counter values.
pmr[ag]               Produce PM KPI reports.
pme[fd][cgu]          Fetch/decode event ROP files (RNC/RBS only).
pst                   List all PM scanners and their state.
pgets[n]              Print scanner contents.
emom                  Display list of events for event-based scanners.
```

6 Accessing Help

6.1 Help Command

6.1.1 Show all help for AMOS

To show all help topics for AMOS, use the following command:

```
> h
```

6.1.2 Show detailed help

To show help related to a particular AMOS command, use the following command:

```
> h <command name>
```

6.1.3 Show old MOSHELL user guide online

To show the old MOSHELL user guide in online mode, give the following command:

```
> h <chapter number>
```

6.1.4 Show help first menu for AMOS

To show the help first menu for AMOS, use the following command:

```
> m
```

6.1.5 Show help second menu for AMOS

To show help second menu for AMOS, use the following command:

```
> n
```

6.1.6 Show performance related help for AMOS

To show all performance related help for AMOS, use the following command:

```
> p
```

7 Viewing MOM Descriptions

The following sections outline the commands used to view MOM details and contents. For more information on these commands type **h <command name>** see Chapter Accessing Help for more detail.

7.1 Viewing the whole Managed Object tree

To view the whole Managed Object tree use the following command:

```
>momt
```

7.2 Viewing all possible parents and children of a Managed Object class

To view all possible parents and children of a Managed Object class, give the following command:

```
>momt  <moclass, struct or enum>
```

7.3 Viewing the description of a Managed Object class

To view the description of a Managed Object class, use the following command:

```
>mom  <moclass, struct or enum>
```

7.4 Viewing the description of a Managed Object class and all its children/grandchildren

To view the description of all attributes of a Managed Object class, give the following command:

```
>momc momc <moclass, struct or enum>
```

7.5 Viewing the description of all attributes of a Managed Object class

To view the description of all attributes of a Managed Object class, use the following command:

```
> mom <moclass, struct or enum> all
```

7.6 Viewing the description of an action

To view a description of an action, use the following command:

```
> mom mom <action>
```

7.7 Viewing the descriptions of all attributes of type enum:admstate

To view a description of all attributes of type **enum**, use the following command:

```
> mom all all enumref:adms
```

7.8 Viewing the descriptions of all members of a struct type

To view a description of all **struct** members contained in **struct AdminProductData**, use the following command:

```
> mom adminproductda all
```

7.9 Viewing all attributes of type sequence:moRef who have a flag restricted

To view all attributes of type **sequence:moRef** who have a flag restricted, use the following command:

```
> mom all all sequence:moref restricted
```

7.10 Viewing all attributes that contain a specified word in their descriptions

To view all attributes that contain a specified word in their descriptions, use the following command:

```
> mom all all all all <specified word>
```

7.11 Viewing all attributes that contain a specified phrase in their descriptions

To view all attributes that contain a specified phrase in their descriptions, give the following:

1 Open the logfile, an arbitrary name is automatically given:

l+

2 Run the appropriate **mom** command to display the description of an attribute.

3 Close the logfile:

l-

4 Find the specified phrase in the logfile:

l grep -i <"specified phase"> $logfile

7.12 Checking the MOM Version

To check the Managed Object Model (MOM) version on the Network Element use the following command:

```
> getmom
```

The check is done by reading the header of the MOM file stored under `http://NodeIPAddress/cello/oe/xml`. This command is normally not needed as this check and fetch of the MOM is handled automatically by AMOS.

7.13 Parsing an XML MOM File

To re-parse the currently stored MOM version use the following command:

```
> parsemom
```

To parse a new MOM version use the following command:

```
> parsemom <MOM Name>
```

When the above command is used AMOS will parse the MOM and use it rather than the one currently loaded from the Network Element. This command should be used in two circumstances:

- The MOM stored on the Network Element is incorrect or cannot be retrieved.
- The user wishes to browse a MOM offline via the AMOS **mom** command suite.

8 Loading Managed Objects

This chapter outlines the commands used to load Managed Objects and their children. For more information on these commands type **h <command name>** see Chapter Accessing Help for more detail.

There are two commands that can be used to load Managed Objects:

- **lt**

  lt has the following command syntax:

  lt/ltc[1-9] <motype-filter>|root|all [<attribute==value> AND/OR <attribute==value>]

  lt stands for Load Managed Object Types, **ltc** stands for Load Managed Object Types and their Children. The numeric option in **ltc** is for specifying the number of levels of children to load. Without the option, all levels of children are loaded.

  This command queries the Network Element to find out which Managed Objects it contains and creates a table with the Managed Object Local Distingushed Names (LDNs) and a proxy number.

- **lc**

  lc has the following command syntax:

  lc/lcc[1-9] <moGroup>|<moFilter>|<proxy(s)>|all

  The **lc** command is for loading the LDNs of the children Managed Objects lying under an Managed Object or group of Managed Objects. **lc** on its own or combined with the numeric option "1" will only load the direct children. With the numeric options 2 to 9 it is possible to specify the number of levels of children to be loaded. The "c" option (lcc command) is for loading all possible levels of children under the Managed Object.

8.1 Loading the Root Managed Object

To clear the proxy table and allocate a proxy for the root Managed Object, use the following command:

```
> lt root
```

8.2 Loading a proxy table with all Managed Objects contained in the Network Element

To build a proxy table with all the Managed Objects contained in the Network Element, use any of the following commands:

```
> lt all
```

```
> lc all
```

```
> lcc 0
```

8.3 Loading all Managed Objects that match a specific word

To load all Managed Objects that match a specified phase, use the following command:

```
> lt <specified word>
```

8.4 Loading all Managed Objects that match a specified word, and all the children of these Managed Objects

To load all Managed Objects that match a specified word, and all the children of these Managed Objects, use the following command:

```
> ltc <specified word>
```

8.5 Loading all Managed Objects that match a specified word, and only one level of their children

To load all Managed Objects that match a specified word, and only one level of their children, use the following command:

```
> ltc1 <specified word>
```

To load all Managed Objects that match a specified word, and only two levels of their children, give the following command:

```
> ltc2 <specified word>
```

8.6 Loading Managed Objects given the Managed Object's proxy identity

To load a Managed Object and only one of its children by using its Managed Object proxy identity, use the following command:

lc/lcc[1-9] <moGroup>|<moFilter>|<proxy(s)>|all

For example to load the MO specified in a proxy table as `MO 0` that is, the `ManagedElement` MO, use the following command:

```
> lc 0
```

To load two levels of children under the `ManagedElement` MO, use the following command:

```
> lc2 0
```

To load all levels of children of the `ManagedElement` MO (same as **lc all** or **lt all**) use the following command: .

```
> lcc 0
```

8.7 Loading a selection of Managed Objects

To load a selection of Managed Objects use a command with the following syntax:

lt/ltc[1-9] <motype-filter>|root|all [<attribute==value> AND/OR <attribute==value>]

For example to load utrancells, fach, rach, pch MOs, use the following command:

```
> lt ^utrancell|fach|rach|pch
```

This command example uses regular expressions. For more information on Regular Expressions see previous chapter on Regular Expressions.

8.8 Loading a selection of Managed Objects and their children given the Managed Objects' proxy ids

To load a selection of Managed Objects and their children given the Managed Objects' proxy ids, use the following command:

lt/ltc[1-9] <motype-filter>|root|all [<attribute==value> AND/OR <attribute==value>]

For example to load Managed Objects with proxy ids 6 and 7 and give the following command:

```
> lc 6 7
```

8.9 Loading all Managed Objects of a particular type

To load all Managed Objects of a particular type use the following command:

```
> lt  <motype>
```

8.10 Loading a subset of Managed Object types that all have a particular attribute value

Note: This type of command initiates a search through a large number of Managed Objects. This command may not display its results as quickly as other commands.

To load a subset of MO types that all have a particular attribute value, use a command with the following syntax:

lt/ltc[1-9] <motype-filter>[<attribute==value> AND/OR <attribute==value>

For example, to load all `UtranCell` whose `operationalstate` attribute has a value of `0` (that is all disabled `UtranCell`'s), give the following command:

```
> lt utrancell operationalstate==0
```

8.11 Loading all Managed Objects that all have particular attribute values

Note: This type of command initiates a search through a large number of Managed Objects. This command may not display its results as quickly as other commands.

The following example loads all Managed Objects that have `operationalstate` of `0` or an `administrativestate` of `0`:

```
> lt all operationalstate==0 OR administrativestate==0
```

8.12 Loading all the children of a Managed Object that have particular attribute value

Note: This type of command initiates a search through a large number of Managed Objects. This command may not display its results as quickly as other commands.

The following example loads all children of the Managed Object `RncFunction` that have `operationalstate` of `0`:

```
> ltc rncfunction operationalstate==0
```

8.13 Loading One Managed Object

The following command loads a proxy for a Managed Object given its Local Distinguished Name (LDN). An LDN does not require `ManagedElement=1` to be set as this is assumed.

```
> ld rncfunction=1,iublink=1002
```

9 Unloading Managed Objects

This chapter outlines the command used to unload Managed Objects. For more information on these commands type **h <command name>** see Chapter Accessing Help for more detail.

The purpose of this command is to reduce the size of the proxy table by unloading unwanted Managed Objects. This is useful on large Network

Elements that have greater than `50,000` Managed Objects present. By running this command the memory usage on the OSS will be reduced and Managed Object commands will be faster. The typical case is to unload all relation MOs in the RNC (`UtranRelation` and `GsmRelation`) which are very numerous but not used in most commands.

The syntax of the **lu** command is:

lu/llu <moGroup>|<moFilter>|<proxy(s)>

9.1 Unloading all MOs

To unload all MOs from the MO tree use the following command:

```
> lu all
```

9.2 Unloading a selection of MOs by moGroup

The following command unloads all `Relation` MOs from the proxy table. To unload a selection MOs from the MO tree use the following command:

```
> lu relation
```

10 Printing Managed Object data currently loaded from the Network Element

This chapter outlines the commands used to print Managed Object data that is currently loaded from the Network Element. For more information on these commands type **h <command name>** see Chapter Accessing Help for more detail.

10.1 Printing all MOs currently loaded

To print all MOs that are currently loaded in AMOS use the following command:

```
> lt all
>pr
```

10.2 Printing a selection of MO by Proxy Identity

To print MOs using proxy identity use the following command:

```
> pr <proxy identity>
```

10.3 Printing all MO's that have the specified RDN value

To print all MO's with a specified RDN value use the following command:

```
> pr <RDN>
```

10.4 Printing all MOs that do not have the RDN value

The following uses the **NOT** syntax to print all Managed Objects that are not a `UtranRelation` or `Fach` Managed Object.

```
> pr !utranrel|fach
```

10.5 Printing all defined MO Groups

To print all defined MO groups, use the following command:

```
> mp
```

See **ma** command in Section 12 on page 47 for more information about MO groups.

Note: To print the contents of a group, use the following command:**pr <mo_group>**.

10.6 Printing State of all Managed Objects

To print the state of all Managed Objects that have an operational and administrative state on the Network Element use the following command:

```
> st
```

10.7 Printing the state of all disabled Managed Objects

The following command will view all disabled Managed Objects on the Network Element.

```
> st all dis
```

10.8 Printing all disabled Managed Objects under a particular Managed Object

The following command prints all Managed Objects that are disabled under the Managed Object `E1PhysPathTerm`

```
> lst e1phys dis
```

10.9 Print all Managed Objects that are unlocked and disabled

The following prints all Managed Objects that are unlocked and disabled:

```
> st all 1.*0
```

10.10 Print all Managed Objects that are locked

The following command prints all Managed Objects that are locked.

```
> st all ^0
```

10.11 Printing the state of all channels in cells based upon RDN

The following prints the state of all channels in cells starting with `cell37`

```
> lst cell=cell37
```

10.12 Printing MOs that have a specified product data

The following example will print the attribute `productData` on all Managed Objects:

Note: Managed Object classes that have a `productdata` attribute can be found via mom command **mom all struct:.*productdata**. Typically this includes the `Slot`, `SubRack`, `PiuType` and `LoadModule` Managed Objects.

```
> prod loadmodule
```

10.13 Printing the states of a Managed Object and all its linked Managed Objects

The **lk** or **llk** command is used to view all Managed Objects that are linked to a Managed Object and its associated `administrative` or `operational` state. The following Managed Objects can currently be queried using the **lk** or **llk** command.

`IubLink`,`UtranCell`, `Ranap`, `Rnsap`, `Vmgw`, `Mtp3bSls`, `Mtp3bSrs`, `M3uAssociation`,`UniSaalTp`, `NniSaalTp`,`Aal5TpVccTp`, `Aal0TpVccTp`, `Aal1TpVccTp`, `Aal2PathVccTp`, `VclTp`, `VplTp`, `VpcTp`, `Aal2RoutingCase`, `Aal2Ap`,`AtmPort`, `ImaGroup`.

10.13.1 Print all the MO's linked to an MO and their states

To view all Managed Objects that are linked to a particular Managed Object matching a filter use the following command:

```
> lk <filter>
```

Or use the following command:

```
> llk <filter>
```

10.14 Print all Managed Objects linked to a Managed Object and its state in the old format

The **lko** command means the 'old' **lk**. It supports a wider range of Managed Object than `lk` but prints them in a different format.

Note:

1. Not all referenced Managed Objects are followed. Some Managed Objects like `AtmTrafficDescriptor` contain references to many Managed Objects that don't relate to the Managed Objects being looked at. The Managed Objects whose references shouldn't be followed are defined in the `dontfollowlist` variable.

2. Some Managed Objects are not linked via an attribute but rather via a parent/child relationship. Those Managed Objects are specified in the `followlist` user variable.

To print all MO's linked to a MO and their state in the old format use the following command:

```
> lko <filter>
```

10.15 Printing Status of Links and their associated Cells and Channels

Note: This command can only be performed on RNC and BSC type Network Elements.

To print the state of all `Cells`, `Iubs` and `Channels` on the Network Element, one line per site, use the following command:

```
> str
```

Note: The `str` printout uses an abbreviated naming of the cells where it is assumed that the last digit is identifying the sector. For networks where the sector is not identified by the last digit, it may be necessary to use the **str1** or **str2** commands since the whole cell name will then be shown for each sector.

The **strt** command uses an abbreviated naming of the cells where it is assumed that the last digit is identifying the sector.

For networks where the sector is not identified by the last digit, it can be handy to use **str1** or **str2** since the whole cell name will then be shown for each sector.

10.16 Printing Resource Usage and Configuration of IubLinks (RNC only)

To print the resource usage and configuration information for each `IubLink` use the following command:

```
> lki
```

10.17 Printing Resource Object information for RNC only

To print resource object information for all MOs in `LmCell` (RNC only) use the following command:

```
> tg
```

10.18 Printing UE Context Data for all Active Calls

To print UE context data for all active calls (RNC only) use the following command on Central MP:

```
> ueregprint
```

On module MP use:

```
> printUeCtxt
```

10.19 Printing all Scripting Values

AMOS scripting variables can be displayed by using the `pv` command.

```
> pv
```

For more information on scripting see the AMOS Scripting User Guide

10.20 Running History

AMOS keeps a history of all the commands that have been run during the current session. To view a history of all commands entered during the current AMOS session use the following command:

```
> hi
```

10.20.1 Running a local UNIX Command from AMOS

To run local UNIX or OSS commands from with the AMOS shell use the following command:

```
> l <command>
```

For example:

```
> l pwd
```

11 Alarm Handling

The following sections give an overview of the commands that can be used for Alarm Handling. For more information on these commands type **h <command name>** see Chapter Accessing Helpfor more detail.

11.1 Alarm Handling

The output from the following commands can be piped through external UNIX utilities like **sort**, **grep**, **less** or **more**.

Note: In the **alt** and **alk** commands, the severity field is shortened to one character:

- `C` - Meaning Critical.
- `M` - Meaning Major.
- `m` - Meaning Minor.
- `w` - Meaning Warning.

11.1.1 Printing an overview of all Alarms

To print an overview of all active alarms use the following command:

```
> al
```

11.1.2 Printing an overview of only critical Alarms

To pipe the output of all alarms to the UNIX **grep** and to only show the alarms containing the string `Crit`.

```
> al | grep "Crit "
```

11.1.3 Counting the number of Major Alarms

In the example below we will use the `grep` UNIX command to count the number of Major alarm occurring on the Network Element, this can be achieved in a variety of different ways but perhaps the simplest is the **grep -c** command.

```
> al | grep -c "Maj "
```

11.1.4 Printing details on all Alarms

To print the active alarm list with detailed information on the alarm use the following command:

```
> ala
```

11.1.5 Printing all Active Alarms sorted chronologically

To print all alarms chronologically use the following command:

```
> alt
```

To see more detailed information than provided by the previous command use the following command:

```
> alat
```

11.1.6 Printing all Active Alarms with acknowledged alarms and unacknowledged alarms printed separately

To print which alarms which alarms have been acknowledged and which alarms have not been acknowledged use the following command:

```
> alk
```

To see more detailed information than provided by the previous command use the following command:

```
> alak
```

12 Managed Object Group Handling

The following sections give an overview of the commands that can be used for MO Group handling. For more information on these commands type **h <command name>** see Chapter Accessing Help for more detail.

The commands to add MOs to an MO group have the following syntax:

ma/lma <moGroup> <moGroup>|<moFilter>|<proxy(s)>|all [<attribute-filter>] [<value-filter>]

The first argument (mandatory),**<moGroup>** indicates the name of the MO-Group.

The second argument (mandatory), **<moGroup>|<moFilter>|<proxy(s)>** indicates the Proxy Id's or MO-Filter of the MOs to match.

If no further arguments are given then the MOs whose RDN/LDN match the MO-filter (or who have the corresponding proxy) are put in the MO-Group.

If further arguments are given then a get or pget command is performed using the second/third(/fourth) argument of the ma/lma command.

The third argument is a string to match the attribute and the fourth (if it's used) matchs the value. If the attribute is of any other type than MoRef, then the MO(s) whose attribute match the fourth argument is put in the group.

If the attribute is of type MoRef, then the MO(s) contained in the attribute is put in the group (except if there is a fourth argument).

12.1 Managed Object Group Creation

12.1.1 Adding all MOs that match a specified RDN into a specified MO

To add all MOs that match a specified RDN into a specified MO group use the following command:

```
> ma <MO group name> <RDN>
```

12.1.2 Adding all MOs that match a specified LDN into a specified MO

To add all MOs that match a specified LDN into a specified MO group use the following command:

```
> lma <MO group name> <LDN>
```

12.1.3 Adding all MOs that match a specified RDN and attribute value into a specified MO

To add all MOs that match a specified RDN and which have a specified attribute set to a certain value to a specified MO group use the following command:

```
> ma <MO group name> <RDN> <attribute name>
<attribute value>
```

12.1.4 Adding all MOs into a specified MO that match a specified RDN and are referenced through an attribute

For example to add all MOs into the MO **test** that match the specified RDN, **atmport** and are referenced through the attribute, **physpatherm**:

ma test atmport physpathterm

physpatherm is an attribute of type `MoRef`

12.2 Managed Object Group Removal

This command may be used to remove a Managed Object from a group or to remove a list of Managed objects from an MO group

Note: Managed Objects will not be deleted only the group.

12.2.1 Remove an MO Group

To remove a specified MO Group use the following command:

```
> mr <MO Group Name>
```

Note: Managed Objects will not be deleted only the group.

12.3 List MO Groups

12.3.1 Printing All MO Groups

To print all MO Groups currently defined in AMOS use the following command:

```
> mp
```

12.3.2 Printing the contents of an MO Group

To print the Managed objects contained in a specified MO Group use the following command:

```
> pr <MO Group Name>
```

13 Managed Object Handling

This chapter outlines the commands that can be used for MO handing. For more information on these commands type **h <command name>** see Chapter Accessing Helpfor more detail.

The `get` command is used to read CM/FM attributes from a Managed Object.

13.1 Reading Attributes

13.1.1 Normal Get

To read CM/FM attributes from a MO use the following command:

```
> get
```

or the following command:

```
> lget
```

13.1.2 Horizontal Get

To read CM/FM attributes from a Managed Object and print then horizontally one line per Managed Object instead of one line per attribute use the following command: .

```
> hget
```

or the following command:

```
> hlget
```

13.1.3 Slow Get attributes from a Managed Object

To read attributes from a Managed Object one attribute at a time often referred to as a 'slow' get use the following command:

```
> sget
```

The above command is useful in the case where the standard **get** command is not working due to some attribute returning an exception.

13.1.4 Displaying Attributes in Exportable Printout Format

To read attribute values in Separated Value format, which allows for easy exportation to EXCEL and many other products use the following command:

```
> hgetc
```

13.1.5 Reading Managed Object data without using the Managed Object Service

Managed Object data is stored on the Network Element in an SQL table. Users of AMOS can choose to read this data directly from the SQL database instead of using the Managed Object Service. This command requires read access to the Network Element database using the Network Element's SQL client. If the SQL client is not started, the `fro` command will start it automatically using the `sql+` functionality in AMOS.

Note: The `fro` command queries the `motype_r1` and `modata_r2` tables in order to find the FROID. If the `motype_r1` table is not present on the Network Element it could mean that the CPP version is too old, in this case the `fro` command cannot work.

The `from` command may be specified to read all the attributes of a Managed Object including the FROID as in the example below:

```
> from 0
```

Print all FROIDS for all Managed Objects of type `PluginUnit`:

```
> fro plugin
```

Print just the `resourceId` for all `PluginUnit` Managed Objects that have a FROID containing the value `5`

```
> fro plugin . 5
>
```

Print just the `resourceId` for all `PluginUnit` Managed Objects that have a FROID of exactly `5`

```
> fro plugin . ^5$
```

13.2 Setting Attributes on one or more MOs

Warning!

Care needs to be taken in AMOS when modifying a `LocationArea`, `RoutingArea`, `ServiceArea`, `MbmsServiceArea`, `ExternalGsmCell`, `ExternalGsmNetwork`, `GsmRelation`, `UtranCell`, `ExternalUtranCell`, `RncFunction`, `WcdmaCarrier`, `Hsdsch`, `Eul` or `UtranRelation` Managed Object within the WRAN as these are viewed as holding special meaning within the WRAN on a SubNetwork level, although a change may be applied to a Network Element any modification to any one of these Managed Objects can cause inconsistencies within the subnetwork of the OSS.

The **set** command is used to set an attribute value on one or several Managed Objects. The command adheres to the following format **set[m]/lset[m] moGroup|moFilter|proxy(s) attribute [value]**. For more information on the **set** command type **h set** at the AMOS command prompt. Only attributes that do not have the flag `readOnly` or `restricted` can be set. Use the **mom** command to check the flags of an attribute.

The following example will set the `primarycpichpower` attribute to `250` for all cells on an RNC:

```
>set cell primarycpichpower 250
```

13.3 Locking and Unlocking Managed Objects

Locking a Managed Object works by setting the `administrativestate` attribute of a Managed Object.

Note: There is also an `OSE` shell command called **bl** for listing blocks on the Network Element. If you need to run the `OSE` shell command rather

than the AMOS command, just type a **** in front of the **bl** command as in **\bl**.

13.3.1 Soft Locking a Managed Object

To set the `administrativestate` of a MO to `2` which means that the resource will have around `30` seconds grace period to hand over all traffic to other resources before it gets locked, this is known as a soft lock use the follow command:

```
> bls <moFilter>
```

or the following command:

```
> bls <proxy>
```

13.3.2 Hard Locking a Managed Object

To set the `administrativestate` of a MO to `0` which means the resource will be locked immediately without any grace period to hand over ongoing traffic to other resources before it gets locked, this is known as a hard lock, use the following command:

```
> bl <proxy>
```

or the following command:

```
> bl <moFilter>
```

13.3.3 Unlocking Managed Objects

To set the `administrativestate` attribute of an MO to `1` which means the resource unlocked and enabled, use the following command:

```
> deb <proxy>
```

or the following command:

```
> deb <moFilter>
```

13.4 Creating an MO

Warning!

Care needs to be taken in AMOS when creating a `LocationArea`, `RoutingArea`, `ServiceArea`, `MbmsServiceArea`, `ExternalGsmCell`, `ExternalGsmNetwork`, `GsmRelation`, `UtranCell`, `ExternalUtranCell`, `RncFunction`, `WcdmaCarrier`, `Hsdsch`, `Eul` or `UtranRelation` Managed Object within the WRAN as these are viewed as holding special meaning within the WRAN on a SubNetwork level, although a change may be applied to a Network Element any modification to any one of these Managed Objects can cause inconsistencies within the sub-network of the OSS.

The **cr** command is used to create a Managed Object. The command adheres to the following format **cr LDN**. For more information on the **cr** command type **h cr** at the AMOS command prompt.

```
>cr rncfunction=1,iublink=1234
```

Note: If there are any mandatory attributes to fill in the **cr** function will prompt for them. If there are any optional restricted attributes to fill in, the function will prompt for them. For optional attributes to use the default value type **d** when prompted.

13.5 Deleting an MO

Warning!

Care needs to be taken in AMOS when deleting a `LocationArea`, `RoutingArea`, `ServiceArea`, `MbmsServiceArea`, `ExternalGsmCell`, `ExternalGsmNetwork`, `GsmRelation`, `UtranCell`, `ExternalUtranCell`, `RncFunction`, `WcdmaCarrier`, `Hsdsch`, `Eul` or `UtranRelation` Managed Object within the WRAN as these are viewed as holding special meaning on a SubNetwork level, although a change may be applied to a Network Element any modification to any one of these Managed Objects can cause inconsistenties within the subnetwork of the OSS.

The **del** command adopts the following format **del/ldel moGroup|moFilter|proxy(s)**. A Managed Object can only be deleted when its `reservedBy` list is empty and when it does not have any children. If the Managed Object does have children and/or a non-empty `reservedBy` attribute, it is possible to use the **rdel** or the **lrdel** command instead. The command first prints the Managed

Object(s) to be deleted, then asks for confirmation. Once the Managed Object(s) are deleted, they are also removed from the proxy list.

13.5.1 Deleting a Managed Object and all its children

To delete a Managed Object and all its children, just use the % sign in front of the LDN or RDN filter.

```
>ldel %ms,slot=20,plug
```

13.5.2 Deleting a Managed Object together with its children and reserving Managed Objects.

The **rdel/lrdel moGroup|moFilter|proxy(s)** may be used to delete Managed Object(s) together with their children and reserving Managed Objects. For more detailed information on the command type **h rdel** at the AMOS command prompt.

Note: This command currently works on `IubLink`, `UtranCell`, `Ranap`, `Rnsap`, `Vmgw`, `Mtp3bSls`, `Mtp3bSrs`, `M3uAssociation`, `UniSaalTp`, `NniSaalTp`, `Aal5TpVccTp`, `Aal0TpVccTp`, `Aal1TpVccTp`, `Aal2PathVccTp`, `VclTp`, `VplTp`, `VpcTp`, `Aal2RoutingCase`, `Aal2Ap`, `AtmPort`, `ImaGroup`.

The command will first fetch all the references, then present the list of Managed Objects to be deleted and will ask the user for confirmation.

13.6 Actioning a Managed Object

The **acc** command adopts the following format **acc/lacc moGroup|moFilter|proxy(s)|all action**. It is used to execute an action on a Managed Object. For more detailed information on the command type **h rdel** at the AMOS command prompt.

Note: If an action requires parameters these will be prompted for. If an action contains parameters, the value of each parameter has to be entered at each prompt. If no value is entered, the action is aborted.

The following example demonstrates an action that will list the IP routing table.

```
>acc ip listroutes
```

13.7 Comparing Managed Objects

The **diff/ldiff** command can be used for parameter auditing or MO dump comparisons, it may be used to compare two or three Managed Objects side by side. Managed Objects must be of same Managed Object class. All attribute values that are different between the Managed Objects will be printed. For a complete description of the **diff** command type **h diff** at the AMOS command prompt.

14 Command Administration

14.1 Undoing Operations

The undo mode may be used to undo the **del**, **rdel**, **set**, **bl** or **deb** commands, this is achieved by performing a **get** before running each of the aforementioned commands and storing the results in an undo file. Upon stopping undo mode undo file is used to recreate the Managed Objects and set them back to their old values.

14.1.1 Starting Undo Mode

To start undo mode use the following command:

```
> u+
```

14.1.2 Stopping Undo Mode

To stop undo mode use the following command:

```
> u-
```

14.1.3 Starting Simulated Undo Mode

When running in simulated undo mode all deletions are simulated.

Note: Simulated undo mode only works for **del** and **rdel** commands and not **set**, **bl** or **deb** commands.

To enter the simulated undo mode use the following command:

```
> u+s
```

14.1.4 Checking the Undo Mode Status

To check if AMOS is in undo mode use the following command:

```
> u?
```

14.2 Aborting an MO Command

AMOS as a process does not trap UNIX command signals, for example **Ctrl-c** being the **INT** (interrupt) signal in UNIX will not interrupt the currently executing command but rather it will kill the currently executing AMOS process. The following chapters discuss the mechanisms AMOS provides for interrupting a long running process.

14.2.1 Abort an MO command like get/st/acc

As AMOS current cannot trap aborting a MO command is a two step process. First type `Ctrl-Z`, to suspend AMOS. Then, at the UNIX prompt, create an empty file `/tmp/xxxx` (where xxxx is the process number indicated in AMOS menu and in the window's title bar) and resume AMOS by reanimating the process. This is done using the following command:

```
touch /tmp/xxxx ; fg
```

Note: If the AMOS prompt doesn't come back even after typing **enter** a number of times, try again suspending via **ctrl-z** and resuming with **fg**.

14.2.2 Abort a for loop

Suppose for some reason you decide to build a monitoring application to continuously poll a Network Element for its alarms and if it exceeds a certain amount to take some action, if you wish to cancel this operation without killing AMOS you must, first type **Ctrl-Z**, to suspend AMOS then, at the UNIX prompt, create an empty file **touch /tmp/xxxx** (where xxxx is the process number indicated in AMOS menu and in the window's title bar) and then resume AMOS by reanimating the process with the **fg** command. The following example shows how to abort a for loop.

```
> for ever
Enter commands (one per line), then type "done"
> wait 5
> al
> if $nr_of_alarms > 50
> #mail some technical staff
> break
> fi
> done

------------------------------------------------------------------
071211-17:26:08    Loop number 1
------------------------------------------------------------------

> wait 5

Waiting from [2007-12-11 17:26:09] to [2007-12-11 17:26:14]...Done.

> al

071211-17:26:15 159.107.180.36 7.0s RNC_NODE_MODEL_G_5_22 stopfile=/tmp/19164
^Z
Suspended
nmsadm@atrcus74> touch /tmp/19164 ; fg
moshell -vcorba_class=2,secure_ftp=1,secure_shell=1 159.107.180.36
Trying file=/var/opt/ericsson/amos/moshell_logfiles/nmsadm/logs_moshell \
/tempfiles/20071211-172449_19124/ior19124
...<cut>...
> if $nr_of_alarms > 50
Enter commands (one per line), then type "fi"
> #mail some technical staff
> break
> fi

>
```

14.2.3 Abort the pol command

The `pol` command is mostly used to poll the Network Element until the Configuration Management Service is up. The user may use the `Ctrl-Z`, to suspend AMOS. Then, at the UNIX prompt, create an empty file `touch`

`/tmp/xxxx` (where xxxx is the process number indicated in AMOS menu and in the window's title bar) and resume AMOS by reanimating the process with the `fg` command.

```
> poll
```

15 Network Element Administration

15.1 TELNET/COLI Handling

15.1.1 Running COLI commands from inside AMOS

Note: If the Network Element password has not been specified in the `ipdatabase` file the user will be prompted to enter the password.

AMOS may be used to access a Network Element to perform certain Command Line (COLI) commands on the OSE Shell. The command is sent to the Network Element via TELNET or SSH depending on the value of the `secure_shell` variable.

15.1.1.1 Listing all COLI commands

To list all COLI commands use the following command:

```
> ?
```

15.1.1.2 List a particular COLI command

To list information about a particular COLI command use the following command:

```
> ? <command name>
```

15.1.1.3 Running Multiple COLI commands

To execute multiple COLI commands at once use the following format:

```
> <command> ; <command> ; <command>
```

15.1.2 Printing Translation of Load Module Product Number

AMOS can be used to translate load module product numbers to trace and error codes.

Note: If some names are missing from the printout, the **lmid refresh** or the **bo** command can be run and it will update the AMOS Load Module (LM) reference file with all missing Load Module names. The refresh also happens automatically if no Load Module name is found for the pattern given.

15.1.2.1 Print the translation of Load Modules that match a particular filter

To print the translation of Load Modules that match a particular filter use the following command:

```
> lmid <filter>
```

15.1.2.2 Print detailed information on all load modules

To print detailed information on all load modules on the Network Element, use the following command:

```
> pglist
```

15.1.2.3 Print the translation of Trace and Error Log Error Codes

The **c** option is to print the error codes list (aal2/mtp3/sccp/utrancell) which is used to decode error codes from the `te log read` command.

```
> lmidc utrancell
```

15.1.2.4 Print Trace and Error Log Error Codes that match a particular filter

To show Trance and Error Log Error Codes that match a particular filter use the following command:

```
> lmidc <filter>
```

15.1.3 Displaying Miscellaneous COLI Printouts

The **cab[slxradgtme]** command may be used to display miscellaneous COLI printouts relating to hardware, software, restarts, leds, central processing unit load, errors, disk and random access memory usage. The **cab** command offers a number of options, it is possible to combine several options for example: **cabslxrdg**, **cablx** or **cabxs**. The command **cabslxrdgm** will give the maximum amount of information. The following is a list of the commands and an overview of what they will print.

- **cab** - Prints MP/BP HW info and LED status, MP temperature, and COREMGR status.
- **cabt** - Same as **cab** but without the temperature.
- **cabx** - Same as **cab** plus LED and HW info for the XP boards.
- **cabl** - Same as **cab** but plus MP/BP processor load.
- **cabs** - Same as **cab** plus list of programs running in all MP/BP.
- **cabr** - Prints all MP/BP restarts. Abnormal restarts are shown in red.
- **caba** - Prints only abnormal MP/BP restarts.
- **cabd** - Print disk usage. Disks that are getting over a certain limit will appear in color.

- **cabg** - Print MP/BP HW errors.
- **cabm** - Print MP/BP RAM memory usage.
- **cabe** - Print MP/BP added T&E trace conditions

For more information on the **cab** command type **h cab** at the AMOS command prompt.

Note: Regarding the CoreManager status: If a board has got a CoreManager status, it means that the board is running the Core Manager programs (EqmMgr, Database, LoaderServer). If the Network Element is configured with Fault Tolerant Core, there are two boards running the Core Manager functionality. One board is Active and the other one is Standby. When the Network Element is configured with Fault Tolerant Core, the **/c** drive is mirrored between the two Core Manager boards. If the status of the Standby board is **StandbyReady**, then it means that the **/c** drive is correctly mirrored and the standby board can take over the active role at any time, in case the active board fails or restarts. If the status of the Standby board is S**tandbyWriting**, then it means that the **/c** drive is performing a small update and the standby board can take over in a short while, as soon as the disks are updated. If the status of the Standby board is **StandbySync**, then it means that the **/c** drive is performing a complete update and the standby board will not be able to take over until this is completed. The progress is shown as a percentage value (eg: **StandbySync-56%**).

Note: When many commands are to be sent, the cab function will put them into a command file, transfer that file (via (s)ftp) to the Network Element and run that file from within the Network Element, using the shell **-f** command. This will save time instead of having to send each command one by one to the Network Element. There is a user variable called **fast_cab_threshold** which determines the number of boards in the Network Element above which a command file will be transferred to the Network Element.

Note: Regarding PMD Ids appearing in `cabr/caba`: a Post-Mortem Dump (PMD) may be associated with an abnormal board restart. In this case, the PMD Id is shown in **cabr/caba** commands. The path to the PMD should normally appear in the printout dump **list -a** or **ftree /c/pmd**. If not there, the PMD can be saved using command **pmdsave <pmdId><file>**. To find out which load module a process belongs to, use command: **ps -w <processName>**.

15.1.4 Network Element File System

The **ftree** command recursively lists the Network Elements file System. The **ftree** command is similar to the `ls -lR` UNIX command. The **ftree** command follows the following syntax **ftree[f] [directory] [lnh] [| unix-command]**. The first argument specifies from which directory the listing will start for example `/c`. The option **f** is for printing files only. For more information type **h ftree** at the AMOS command prompt.

Note: It is not possible to directly pipe the output of `ftree` directly into an external utility. The workaround is to log the output and then run it through the utility, see Section 7.11 on page 35.

15.1.4.1 File System

In order to list a file system type the following at the AMOS prompt

```
>cd /d/configuration
>ftree
```

Or to list all files located in the directory `/c/pmd` sorted by date and time perform the following:

```
>ftree /c/pmd
```

15.1.4.2 Boards Groups

It is possible to run **ftree** on several boards by using the **lh** command.

```
>ba bp 0-9999
>br bp gpb
>lh bp ftreee
```

15.2 Polling a Network Element

AMOS session and proxy table stays valid after a Network Element or Configuration Management Service JVM restart. If MO operations are performed during the actual restart, a CORBA error is will be printed (CORBA.TRANSIENT or CORBA.OBJECT_NOT_EXIST) and AMOS will automatically start to poll the Network Element to find out when the MO service is ready to start receiving requests again. It is also possible to manually start polling using the `pol[shcu] [interval] [waitTime]` command. The parameters are optional. The first one tells how long to wait between each polling (default 10 seconds) and the second one tells how long to wait before starting to poll (default 20 seconds). This gives the Network Element time to perform the actual restart. After the `pol` command has completed, it is possible to carry the session as before, without having to reload the proxies.

Note: If the Network Element database has changed after the restart for example due to a software upgrade then AMOS will automatically re-read the MOM information via the `getmom` and `parsemom` commands and will rebuild the MIB information via the `lt all`.

15.2.1 Poll the Network Element until the Configuration Service is up

The `pol` command is mostly used to poll the Network Element until the Configuration Management Service is up. See the following example that waits 9 seconds before starting to pol a Network Element at intervals of 7 seconds until the Configuration Management Service is up.

```
> pol 7 9
```

15.2.2 Poll the Network Element until the OSE Shell is up

The **pols** command may be used to check to see if the OSE Shell is available on the Network Element. Underneath the covers AMOS keeps running the OSE **vii** command until it returns the state `GREEN_LEDI_ON` . This indicates that all programs have finished loading in the main MP.

```
> pols 5 5
```

15.3 Disconnecting and Reconnecting to the CM and PM Servers

The **re** command may be used to disconnect and reconnect to the Configuration Management Service (known internally in AMOS as `Mo Browser`) and to connect to the Performance Management Service (known internally in AMOS as `pmtester`). It may be useful to enter this command if the Network Element security settings have been changed during the lifetime of the AMOS session.

15.3.1 Reconnect to the Configuration Management Server and the Performance Management Server on the Network Element

In order to reconnect to the Configuration Management Server and the Performance Management Server the user must give the `re` command to AMOS as in the following example:

```
> re
```

15.3.2 Re-fetch the CORBA Interoperable Object Reference (IOR) from the Network Element

The `i` option when passed to the `re` command is used to re-fetch the IOR file from the Network Element. This operation is necessary if the IOR has changed on the Network Element. This may happen for two reasons:

- If the Network Element has been upgraded from a pre CPP 5.1 Network Element to a post CPP 5.1 Network Element, the CORBA suppliers will have changed from VISIBROKER to JACORB and it is necessary to fetch the IOR file.
- If the Network Elements CORBA security level has changed.

Note: It is **not** necessary to type **re** after a Network Element restart or JVM restart as long as the CORBA definitions has stayed the same.

The fetching of the CORBA IOR from the Network Element may be achieved with the following command:

```
> rei
```

15.4 Transferring Files

Transferring files from a Network Element may be achieved by using either the File Transfer Protocol (FTP) or the Hyper Text Transfer Protocol (HTTP) whereas transferring files to the Network Element may only be achieved via the File Transfer Protocol. It is possible to get or put many files to many network elements in parallel by using the `mobatch` facility see section. For more information on MOBatch see later chaper on MoBatch.

15.4.1 Transferring files using the FTP protocol

Note: Wildcards (*) are currently not supported but it is possible to download/upload a whole directory from/to the Network Element by specifying a remote/local directory instead of a remote/local file.

15.4.1.1 Transferring files to the Network Element

The `ftput[c] localfile/localdir [remotefile/remotedir]` command may be used to place files from the OSS UNIX file system onto the Network Elements file system as in the example below:

```
> ftput /home/nmsadm/trouble_notes.log \
/c/tmp/my_trouble_notes.log
```

The c (conditional) option may be specified as a safeguard to ensure that if the file already exists on the Network Elements file system it will not be overwritten.

Other options are as follows:

- **-f** - Used to specify a regular expression to only transfer the files matching that expression.
- **-s** - Used to give the starting date.
- **-e** - Used to give the ending date
- **-m** - Used to give how long backward from today's date
- **-p** - Used to give how long forward from today's date.

For more information on these options type **h ftput** at the AMOS command prompt.

15.4.1.2 Transferring files from the Network Element

The `ftget[c] remotefile/remotedir [localfile/localdir]` command may be used to get files from the Network Elements file system and place them onto the OSS UNIX file system as in the example below:

```
> ftget
```

The c (conditional) option may be specified as a safeguard to ensure that if the file already exists on the Network Elements file system it will not be overwritten.

Other options are as follows:

- **-f** - Used to specify a regular expression to only transfer the files matching that expression.
- **-s** - Used to give the starting date.
- **-e** - Used to give the ending date
- **-m** - Used to give how long backward from today's date
- **-p** - Used to give how long forward from today's date.

For more information on these options type **h ftget** at the AMOS command prompt.

15.4.2 Transferring files using HTTP protocol

Files may also be transferred from a Network Element to the OSS UNIX file system using Hyper Text Transfer Protocol.

Note: The `http://` prefix is optional, if it is not given it will added by AMOS.

15.4.2.1 Transferring a File using HTTP from local Network Element

The `htget remotefile localfile/localdir` command may be used to get files from the Network Elements file system and place them onto the OSS UNIX file system via HTTP, if no Internet Protocol address is specified then the file is fetched from the current Network Element as in the current example:

```
> htget /cello/oe/xml/rnc_node_mim.xml ~/atrnc1_node_mim.xml
```

15.4.2.2 Transferring a File using HTTP from a different Network Element

If an IPADDRESS is specified the file can be fetched from a different server

The `htget remotefile localfile/localdir` command may be specified with a different Internet Protocol address, if this is the case then the file is transferred from that Network Elements file system to the OSS UNIX file system as in the current example:

```
> htget 159.107.180.33/cello/oe/xml/rnc_node_mim.xml
~/atrnc2_node_mim.xml
```

15.5 Editing Remote Files

Edits a file on the Network Element. AMOS will download the file, spawn your editor an then upload the file once you have finished.

The editor used is specified in the `editor` user variable, which by default is set to the environment variable `$EDITOR`, if this environment variable is not set the `vi` editor will be used.

15.5.1 Editing a file on the Network Element

The following example will create a file on the Network Element.

```
> cd /c/tmp/
> edit my_file
```

16 Checking the MOM Configuration Version

To check a MO CV version use the following command:

```
> cv
```

16.1 Other Configuration Version Handling Commands

The following are commands that are similar to the `cv` commands in OSE but operate through the MO interface instead of TELNE/SSH.

- **cvls**

  Displays both the current CV information (equivalent of **cv cu**) and CV list (equivalent of `cv ls`). It is possible to filter the output of `cvls`to only show CVs where the CV name or CV attributes match a certain string.

- **cvcu**

  Displays the current CV information only (equivalent of **cv cu**).

- **cvmk**

  Creates a CV. Operator name and comments (not longer than 40 characters) can be given as argument. The advantage of using **cvmk** instead of OSE command **cv mk** is that the Upgrade Package information is included in the CV attributes.

- **cvset**

  Set a CV as startable.

- **cvms**

  Create a CV and make it startable (combination of **cvmk** and **cvset**)

- **cvrm**

  Can remove many CVs in one go using pattern matching on the CV name (remove from rollback list is attempted before deletion).

- cvget

Make a remote backup of a CV to the workstation.

- **cvls1**

  Gives similar output to **cvls1** but accesses the CV information through OSE shell instead of the MO interface.

Note: If the OSE shell command **cv mk** is used to create a CV (instead of **cvmk** or **cvms**), the CV attributes will not contain the UpgradePackage reference which means that the CV will not be deleted when the UpgradePackage MO is removed. If there are CVs left on the Network Element without their corresponding UpgradePackage and the Network Element is restarted on one of these CVs, there is a high risk that the Network Element will start performing cyclic restarts. To avoid this situation it is recommended to always use the AMOS command **cvmk** or **cvms** when creating CVs since the UpgradePackage reference will then automatically be included in the CV attributes.

17 Displaying the Hardware/Software Inventory

The **inv[hr] <filter>** displays the complete HW/SW inventory. Including information about RPUs, licensing, JVM, devices, XPs, etc.

This command performs a HW/SW inventory via the MO interface. The output is similar to **cabsx** which provides a HW/SW inventory via the OSE shell interface. Some complementary information can be obtained such as Device SW (spm,dsp,fpga), Piu states, RET info (RBS), etc. Information about the JVM, the RPUs and the SW licenses (features and capacity) is also shown.

The **h** option is for fetching HW information only (equivalent to **cabx**). The first time the command is run, it takes a bit longer because the data has to be fetched from the Network Element before parsing. The following times the command is run, the existing data is parsed again, unless the **r** switch is used (refresh), in which case, the data is fetched again and parsed. The first table shows the MP/BP hardware, the second table shows the XP/EP hardware and software, and the third table shows the MP/BP and device (SP/DSP/FPGA) SW. Each table gets its information from the MOs. The state of the MO is abbreviated to one digit:

- 1 = unlocked enabled
- 0 = unlocked disabled
- L = locked (operationalState could be enabled or disabled)

The filter makes it possible to match only those rows which correspond to the filter string. The stateFilter allows to only shows the MOs matching the state filter.

18 Displaying State and Configuration

18.1 Displaying State and Configuration of AtmCrossConnections

The **stc[p][r] <Filter> [<stateFilter>]** command displays the state and configuration of AtmCrossConnections. The filter only show the rows matching the filter string. The stateFilter only shows the MOs matching the state filter. The first time the command is run, it takes a bit longer because the data has to be fetched from the Network Element before parsing. The following times the command is run, the existing data is parsed again, unless the r switch is used (refresh), in which case, the data is fetched again and parsed. The first field is the MO id of the AtmCrossConnection. The second field is the state information, consists of five digits:

- 1st digit: **operationalState** of the AtmCrossConnection MO (0=disabled, 1=enabled).
- 2nd and 3rd digits: **operationalState** of the VclTp MOs (A and B side).
- 4th and 5th digits: only applicable if p option was used. Shows the status of the **pget** on VclTp MOs (A and B side). Set to 0 if the transmittedCells counter has incremented but the receivedCells counter has not incremented within a given time period (by default 25 seconds, can be changed in the uservariable pm_wait): this indicates that there is no response from the remote end.

The third and fourth field are the MoId for VclTpA and VclTpB, abbreviated in the following way: AtmPortId/VplTpId/VclTpId The fifth and sixth fields are the actual Vpi/Vci value for VclTpA and VclTpB. Can be useful in case the MoId of the VclTp does not match the Vpi/Vci value. The seventh and eigth fields are the MoId for the traffic descriptor of VclTpA and VclTpB. The last field is the userLabel of the AtmCrossConnection MO.

18.2 Displaying State and Configuration of Devices

The **std** command displays state and configuration of RNC and MGW

18.2.1 std on RNC

This command shows the state and location of all the RNC traffic devices:

- **CC** - Common Channels.
- **DC** - Dedicated Channels.
- **PDR** - Packet Data Router.
- **IurCC** - Common Channels for Soft Handover.

The state is abbreviated in the following way:

- **1** = unlocked enabled.

- **0** = unlocked disabled.
- **L** = locked (enabled or disabled).

Each device has a corresponding MO (called **CcDevice**, **DcDevice**, **PdrDevice** or **IurCcDevice**) that is connected to a particular SPM. Therefore the state consists of two digits: one for the state of the Device MO and one for the state of the Spm MO. The MO identity of each of these two MOs is abbreviated in the following way:

- SubrackId/SlotId/DeviceType/DeviceId or SubrackId/SlotId/SpmId.

Also, the module number and SPM linkhandler is shown at the beginning of each line.

```
> std
```

18.2.2 std on MGW

This command shows the number of devices and usage status of the devices in the MGW pools. In MGW R2/R3, the usage is giving as a number, in MGW R4, it is given as a percentage.

18.3 Displaying State, User, and Bandwidth usage for ATM Ports

The **stv[b][r] <Filter> [<stateFilter>]** command displays the state, user, and bandwidth usage for ATM ports and channels.

18.4 Displaying State and User of Physical Ports and Ds0Bundles

The **stt[r] <Filter> [<stateFilter>]** command displays the state and user of `Physical Ports` and `Ds0Bundles`.

19 Miscellaneous Commands

19.1 Checking the health of a Network Element

The **hc** command runs a general health checkup on the Network Element. If no logfile is currently open, then a logfile will be automatically opened to capture the output of the **hc** command. When **hc** is run a number of different commands will be run by AMOS.

For more information on these commands use **h hc** at the AMOS command prompt.

19.2 Remodule an IubLink

To remodule an IubLink the following command may be used **remod[u][2] IubLink-MO|IubLink-Proxy NewModule**. This command is only applicable to RNC Network Elements. It is used to move cells from one module in an RNC to another. This command is used to move an IubLink from one RncModule to another. It is also possible to remodule to the same RncModule.

The following example will move the IubLink=11111 and its related UtranCells from its original RncModule to RncModule=7.

```
>remod iublink=11111 7
```

For more help on the **remod** command type **h remod** at the AMOS command prompt.

19.3 Fetching and Processing of Logs

The command **lg[aevsmircdyuolhf] [-l <logdirectory | logfile>] [-m <minustime>] [-p <plustime>] [-s <startdate>] [-e <enddate>] [| unixcmds]** is used for the fetching and processing of logs alarm, event, availability, system and many others from the Network Element. The output can be piped through external utilities like `sort`, `grep`, `less` and many others. All options can be combined together, except the `d` option which can only be combined with the `r` option as in: **lgae**, **lgvsmir**, **lgaevsuyol**, **lgd** or **lgdr**.

Note: The first time the command is run, the log is fetched from the Network Element and cached to a local area in the OSS. Next time the command is run the logfile isn't fetched again, unless the `r` switch is used.

The optional arguments are as follows:

- **startDate** - For example 2006-03-20, or 20060320, or 2006-03-20.09:45:30, or 20060320.094530): show all entries that happened since that date and time.
- **daysAgo** - For example 5: show all entries that happened in the past xx days.
- **hoursAgo** - For example 3h: show all entries that happened in the past yy hours.
- **minutesAgo** - For example. 2m: show all entries that happened in the past zz minutes.
- **csvFile** - For example. alarm_rnc1.csv: output the entries into the csvFile. Only valid together with the c option. The csvFile MUST have the extension `.csv`. No date filtering is possible.

The options are as follows:

- **a** - Parse alarm log `/c/logfiles/cello/CELLO_ALARM_LOG.xml`.
- **e** - Parse event log `/c/logfiles/cello/CELLO_EVENT_LOG.xml`.
- **v** - Parse availability log `/c/logfiles/cello/CELLO_AVAILABILITY_LOG.xml`
- **s** - Parse system log `/c/logfiles/systemlog/`.
- **u** - Parse upgrade log `/c/tmp/Trace.log`.
- **o** - Parse command log (cpp_version >= 5.1: located in `/c/logfiles/audit_trail/CORBA_AUDITTRAIL_LOG.xml` (cpp_version < 5.1 : located in `/c/commandlog` and in each cv directory, provided the command logging is active.
- **y** - Parse securityevent log `/c/logfiles/security/CELLO_SECURITYEVENT_LOG.xml`. CPP5 and above only.
- **l** - Parse COLI log `/c/logfiles/audit_trail/SHELL_AUDITTRAIL_LOG.xml`. CPP5.1 and above only.
- **h** - Parse HILI log `/c/logfiles/hw_inventory/CELLO_HWINVENTORY_LOG.xml`. CPP5 and above only. Must be generated with **hili mk** first.
- **m** - Merge the different logs together. (Example: **lgaevm** will merge alarm/event/availability logs). Do not use together with "h" option.
- **i** - Inverse chronological order.
- **r** - Refetch the logs from the Network Element
- **c** - Print the output into a file, in csv format. A default output file is chosen unless it was given on the command line (in which case it must have the ".csv" extension).
- **d** - Show Network Element downtime figures. This option can only be combined with the `r` option.

The following example displays the system log for the whole period stored which can be several months:

```
>lgs
```

The following example displays the last 5 hours of the alarm/event/availability/system logs.

```
>lgaevs 5h
```

For more information on the **lg** command type **h lg** at the AMOS command prompt.

19.4 Managing Board Groups

The **bo[r]/ba[swd]/br[wd]/be[0-50]/bp** suite of commands may be run to manage board groups on the Network Element.

For more information on getting an overview of all board groups currently defined type **h bo** at the AMOS command prompt.

For more information on adding boards to a board group type **h ba** at the AMOS command prompt.

For more information on removing boards from a board group type **h br** at the AMOS command prompt.

For more information on printing the contents of a board group type **h bp** at the AMOS command prompt.

The **lh boardGroup OSE-command|run commandfile [| unix commands]** command can be used to run COLI commands on all boards of a board group as in the following example:

```
>ba group1 1 2 4-8
>lh group1 te log read | grep ERROR
```

19.4.1 Listing or Restarting programs on a Board or Board Group

To List or restart programs on a Board or Board Group use the following commands:

```
> proglist/progkill [string] [|unix-cmds]
```

For more information on the **proglist/progkill** commands type **h proglist/progkill** at the AMOS command prompt.

19.5 Fetching Data for TRs/CSRs

To fetch data for TRs/CSRs run the follow command:

```
>dcg
```

The follow are the optional **dcg** commands:

- **m** - Mandatory data.
- **e** - Subset of the mandatory data which can be taken in case of emergency, before doing board/node restart.
- **a** - AAL2 printouts.
- **s** - Data for SS7 issues.
- **r** - RNC specific printouts.

20 Conversion Routines

20.1 Numeric Conversions

AMOS may be used for numeric conversions. Currently it is possible to convert the following from within AMOS:

- Convert a binary number to a hexadecimal number
- Convert a hexadecimal number to a binary number
- Convert a decimal number to a hexadecimal number
- Convert a hexadecimal number to a decimal number

20.1.1 Convert a binary number to a hexadecimal number

Conversion from a binary to a hexadecimal number may be achieved with the h2b command as shown in the following example:

```
> b2h 101010
```

20.1.2 Convert a hexadecimal number to a binary number

Conversion from a hexadecimal to a binary number may be achieved with the **h2b** command as shown in the following example:

```
> h2b 2a
```

20.1.3 Convert a decimal number to a hexadecimal number

Conversion from a decimal to a hexadecimal number may be achieved with the **d2h** command as shown in the following example:

```
> d2h 11
```

20.1.4 Convert a hexadecimal number to a decimal number

Conversion from a hexadecimal to a decimal number may be achieved with the **h2d** command as shown in the following example:

```
> h2d b
```

20.2 IP Address Conversion

AMOS may be used to convert an IP Address into the FRO format which is used by the SQL database on the Network Element. Currently it is possible to convert the following from within AMOS:

- Convert an IP Address to the FRO decimal format.
- Convert a FRO formatted address to an IP Address

20.2.1 Convert an IP address into the FRO format

AMOS may be used to convert an IP Address into the FRO format by using the **ip2d** command as in the example below:

```
> ip2d 159.107.180.35
```

20.2.2 Convert an FRO into an IP address

AMOS may be used to convert an FRO formatted address into the Internet Protocol format using the **d2ip** command as in the example below:

```
> d2ip -1620331485
```

21 Logging in the OSS

Logging in the OSS is implemented by the CIF Self-Management Log Service component.

Note: These log files are OSS log files so contain data that may not be specifically about AMOS faults but rather about OSS faults.

Note: All COLI Telnet operations which are performed against a Network Element are **not** logged in the OSS. These may be accessed by running the **lgl** AMOS command.

21.1 Command Logging

All commands that are run in AMOS are stored persistently in the OSS Command log. The command log may be accessed by entering the following command **smlog -type command -number number_of_entries** at the UNIX prompt or by entering the **l smlog -type command -number number_of_entries** command at the AMOS prompt as in the following example:

```
> l smlog -type command -number 1
```

21.2 Error Logging

Error logging will only be performed when the AMOS fails due to some internal error. We cannot report AMOS script failures as it presently has no knowledge if a command fails or succeeds and thus would not be able to inform AMOS of such a failure. All Error events will be logged at the MINOR log level as this severity level indicates that a fault condition has occurred that does not affect service. The error log may be access by entering the command **smlog -type error -number number_of_entries** at the UNIX prompt or by entering the **l smlog -type error -number number_of_entries** command at the AMOS prompt as in the example below.

```
> l smlog -type error -number 1
```

21.3 Security Logging

Security log messages will be recorded for all failed authorization attempt or for all usage of non-authorized commands within the OSS. The security log may be access by entering the command **smlog -type security -number number_of_entries** at the UNIX prompt or by entering the **l smlog -type security -number number_of_entries** command at the AMOS prompt as in the example below.

```
> l smlog -type security -number 2
```

21.4 Logging AMOS data

By default the data returned by AMOS after executing a command on a Network Element is not stored persistently. The storage of AMOS data persistently may be achieved by using the `l+[m][m][s][o]/l-/l? [logfile]` command construct.

Note: The Pipe symbol **|** does not work for some AMOS post-processed commands so the work around is to perform the following:

1. **l+**
2. **<run command>**
3. **l-**
4. **<run unix command on $logfile>**

For a reasonable example of this see Section 7.11 on page 35

21.4.1 Opening a default Log File

A user may open a log file by giving the **l+** command at the AMOS command prompt as the example below shows:

```
> l+
```

The path to the default log file is always **/var/opt/ericsson/amos/moshell_logfiles/<userid>/logs_moshell/sessionlog/<date>_<ne>.log**. Internally the variable **$logfile** is automatically set by AMOS to contain the name of the latest log file created.

21.4.2 Opening a named Log File

A user may open a named log file by giving the **l+ logfile** command at the AMOS command prompt as the example below shows:

```
> l+ /var/opt/ericsson/amos/moshell_logfiles/nmsadm/ \
my_own_log
```

21.4.3 Opening a muted Log File

If the user wishes for most of the AMOS output to be logged directly to a log file and not to be displayed on screen the **l+m** command can be given as in the example below.

```
> l+m
```

21.4.4 Opening an extra muted Log File

If the user wishes for all AMOS output to be logged directly to a log file and not to be displayed on screen the **l+mm** command can be given as in the example below.

```
> l+mm
```

21.4.5 Muting the Log File Header information

If the user wishes to suppress the printing of the **log open** or **log close** file header messages, this can be achieved by running the **l+s** command.

21.4.6 Overwriting a Log File

The user may decide to overwrite an existing log file by specifying the **l+o** command as in the example below:

```
> l+ /var/opt/ericsson/amos/my_log
```

21.4.7 Closing a Log File

Closing either the default or named log file is achieved by administering the **l-** command at the AMOS prompt as the example below shows:

```
> l-
```

21.4.8 Checking to see if a Log File is currently open

The user may use the **l?** command to check to see if a log file is currently open as the example below shows:

```
> l?
```

22 Performance Management Commands

22.1 Viewing All PM Counters for an MO

To view all PM Counters for **AtmPort** use either:

- **pmom atmp**

or

- **pget atmp**

The **pmom** command has the following syntax:

pmom[acd]/lmom[c] [<moclass>] [<attribute>] [<attr-description>]

The **pget** command has the following syntax:

pget/lpget [<moGroup>|<moFilter>|<proxy(s)>|all] [<attribute-filter>|all] [<value-filter>]

Note: The commands **pget/lget** do not work on RNC MOs (UtranCell, IubLink, etc).

22.2 Viewing Specified PM Counters for an MO

To view **Cell** PM Counters for **AtmPort** give the following command:

pmom atmp cell

22.3 Viewing the PM Configuration Attributes for an MO

To view the PM Configuration Attributes for **AtmPort**, use the **a** option on **pmom** to show configuration attributes that can be included in scanners:

pmoma atmp

The **pmom** command has the following syntax:

pmom[acd]/lmom[c] [<moclass>] [<attribute>] [<attr-description>]

22.4 Viewing Log Attributes for an MO

To view information about the **cdmachannel** log attributes give either of the following commands:

- **lmom cdmachan all**

  (**lmom** is for CDMA only)

Or

- **lget cdmachan all**

The **lmom** command has the following syntax:

lmom[c] [<moclass>] [<attribute>] [<attr-description>]

The **lget** command has the following syntax:

pget [<moGroup>|<moFilter>|<proxy(s)>|all] [<attribute-filter>|all] [<value-filter>]

Note: The commands **pget/lget** do not work on RNC MOs (UtranCell, IubLink, etc).

22.5 Printing PM attribute(s) from an MO horizontally, one line per MO (instead of one line per attribute)

To print PM attribute(s) from MO horizontally one line per MO (instead of one line per attribute) give the following command:

hpget[c]/lhpget[c] <moGroup>|<moFilter>|<proxy(s)> [<attribute-filter>] [<value1-filter>] [<value2-filter>] [<value3-filter>] etc...

For example to print the counter values for vclTP MOs (pmreceivedcells, pmtransmittedcells) give the following command:

hpget vcltp

The **c** option is to display the output in CSV (Comma Separated Value) format. CSV format makes it easier to export to for example excel.

22.6 Printing specified PM attribute(s) for an MO, one line per MO

Use regular expressions to specify which PM attributes of an MO are to be printed. For more information on Regular Expressions see previous on Regular expressions.

To print all **vcltps** that have 0 **receivedCells** and more than 0 **transmittedCells** give the following command:

hpget vcltp . ^0$!^0$

22.7 Printing the output in CSV (Comma Separated Value) Format

The following print commands print their output in CSV (Comma Separated Value) format when the letter **c** is added to the end of the command:

- **pmomc**

- **lmomc**
- **pgetc**
- **hpgetc**
- **lpgetc**
- **hlpgetc**

22.8 Printing PM Attributes whose Values have Changed

To print the PM attributes whose values have changed use the command:

pdiff/lpdiff [<moGroup>|<moFilter>|<proxy(s)>|all] [<attribute-filter>|all] [<value-filter>]

To check all **VclTp** MOs whose **transmittedcells** have incremented but whose **receivedcells** have not incremented give the following commands:

1. **pdiff vcltp= transmit !^0**

   This command stores all **VclTp** MOs whose **transmittedcells** have incremented in the **<moGroup>** called **pdiff_group**. All MOs displayed as a result of the command **pdiff** are stored in an MO group called **pdiff_group**.

2. **pdiff pdiff_group receive ^0**

   This instruction selects all MOs from the MO Group, **pdiff_group** whose **receivedcells** have not incremented and stores the result in the MO Group, **pdiff_group** thus overwriting the original contents of **pdiff_group**.

3. **acc pdiff_group eteloopback**

   A loopback test can now be performed on all **VclTp** MOs whose **transmittedcells** have incremented but whose **receivedcells** have not incremented. This could be used to check if there was for example a transmission problem.

22.9 Displaying Counter Values, Extracted from the Statistics ROP (Report Output Period) Files

To display counter values extracted from the statistics ROP (Report Output Period) files give the following command:

pmx[hfdn] [<mofilter>|<mogroup>] [<counter-filter>] [-l <PMfiles-directory>] [-m <minushours>] [-p <plushours>] [-s <startdate>[.<starttime>]] [-e <enddate>[.<endtime>]] [-a|-d|-h] [| <unix-command>]

22.9.1 Displaying Specified Counters for Last ROP Period

To display all counters matching **downtime** on all MOs matching **utrancell** give the following command:

pmx utrancell downtime

22.9.2 Displaying the Counters side-by-side

The **h** option in **pmxh** is for displaying the counters side-by-side (h as in horizontal). Otherwise, there is one line for each MO instance and counter.

To display the resultant counters side-by-side of the command issued in Section 22.9.1 on page 78 add in the **h** option:

pmxh utrancell downtime

22.9.3 Displaying Counter Values, Extracted from Aggregated ROP Files

Using the command described inSection 22.9.1 on page 78, to display the counters of this command, extracted from more than one ROP file use the following command:

pmxh utrancell downtime -m 3 -a

The **-m** is used to specify a time span.

The **3** is for 3 hours of ROP files. Each ROP file is created every 15 minutes, so 3 hours is a total of 12 ROP files.

The **-a** is used to specify aggregate the ROP files.

22.9.4 Displaying Counter Values Quickly

Counter values can be displayed quickly by using the **f** option:

pmxhf utrancell downtime

The **f** option means the command does not recheck for new ROP files.

22.9.5 Displaying and Sorting Counter Values Quickly, side-by-side from Aggregated ROP Files

To display and sort counter values quickly, side-by-side from Aggregated ROP Files give the following command:

pmxhf utrancell downtime -m 3 -a | sort +1

This command displays side-by-side all counters matching **downtime** on all MOs matching **utrancell**, using the aggregated information contained in 3 hours of ROP files. The result of the command is then sorted on the second field (**| sort +1**).

22.9.6 Storing the Counter Values in a File

If the first two arguments of the command, **pmx** are omitted then the ROP files are parsed and the results are stored in a file instead of being displayed on the screen. The results are stored in the same location on the workstation as where the ROP files are stored that is:**~/moshell_logfiles/logs_moshell/pmfiles/nodeipaddress**

The location can be changed with the user variable **pm_logdir**.

To fetch and store the counter values for the last 2 hours in a file give the following command:

pmx -m 2

22.9.7 Parsing all ROP files that Exist in the pmlog Directory

To parse all ROP files that currently exist in the **pmlog** directory in the workstation give the following command:

pmxf -s 19000101

22.9.8 Producing KPI Reports based on Counter Values in ROP Files and Formulas in CPI Documents

To produce KPI reports based on counter values in ROP files and formulas in CPI documents, give the following command:

pmr[ag] [-r <report(s)>] [-l <PMfiles-directory>] [-i <iubCellModule-file>] [-f <formulafile>] [-c <configfile>] [-m <minushours>] [-p <plushours>] [-s <startdate>[.<starttime>]] [-e <enddate>[.<endtime>]] [-o <outputFormat>]

To print all available reports give the following command:

pmrg

22.9.9 Showing PM Statistics from the Last Few Hours

To show PM statistics for the last 5 hours give the following command:

pmr -m 5

22.9.10 Showing PM Statistics from a Specified Time Period

To show PM statistics starting from 5 hours ago and from 3 hours from then on, that is ending 2 hours ago, give the following command:

pmr -m 5 -p 3

22.9.11 Show PM Statistics for the Last Few Minutes

To show PM statistics for the last 15 minutes give the following command:

pmr -m 0.25

22.9.12 Fetching Event ROP Files for CTR only

To fetch event ROP Files use the following command:

pme[fd][cgu] [<pm_logdir>] [-b <boardgroup>] [-m <minushours>] [-p <plushours>] [-s <startdate>[.<starttime>]] [-e <enddate>[.<endtime>]]

To fetch event ROP Files use for CTR only use the following command:

pmefc

22.9.13 Fetching and Decoding Event ROP Files for UETR and GPEH only

Fetching and Decoding Event ROP Files for UETR and GPEH only, give the following command:

pmefdug

22.9.14 Fetching Event ROP Files for all three event measurement types

Fetching and Decoding Event ROP Files for UETR, GPEH and CTR, give the following command:

pmef or **pmefcgu**

22.9.15 Decoding Event ROP Files for all three event measurement types (no fetching)

Fetching and Decoding Event ROP Files for UETR, GPEH and CTR, give the following command:

pmed or **pmedcgu**

22.9.16 Fetching ROP Files to /home/eric/eventrops

To fetch ROP files to **/home/eric/eventrops** give the following command:

pmef /home/eric/eventrops

22.9.17 Decoding the Stored ROP Files

To decode ROP files stored in **/home/eric/eventrops** with the last hour give the following command:

pmef /home/eric/eventrops -m 1

22.9.18 Fetching and Decoding the gpeh ROP Files for the Last 5 Hours

To fetch and decode all gpeh ROP files for the last 5 hours:

pmefgd -m 5

23 Scripting Guide

AMOS supports the use of variables and logical constructs. These can be used directly from the command line or within AMOS command files.

23.1 Preset Variables

The following variables are set immediately after AMOS startup:

- `$logdir` - path to the **moshell_logfiles/logs_moshell** directory
- `$moshelldir` - path to the AMOS directory
- `$gawk` - path to gawk
- `$ipaddress` - IP address of the node that AMOS is connected to
- `$moshell_version` - the MoShell version
- $logdir - path to the **moshell_logfiles/logs_moshell** directory
- `$tempdir` - path to the directory containing all temporary files for this AMOS session. Gets deleted at the end of the session.

The following variables are set after the MOM has been parsed:

- `$momversion` - the MOM version of the node (eg: RNC_NODE_MODEL_E_5_3, MGW_NODE_MODEL_R3_9_0)
- `$cellomomversion` - the Cello MOM version (3.3, 4.3, 5.1, etc) of the node
- `$momdocnumber` and `$momdocrevision` - the document number and revision of the MOM (for example: 15554-AXD10503/1 , rev: Z1)
- `$background_pid` - the process id of a process started into background from the AMOS command line, for example:

  `l $moncommand > $logfile &`

```
l $moncommand > $logfile &
```

These variables can be handy to have when a script needs to know what SW revision is running in the node or what kind of node it is.

The following variables are set after running certain MO commands:

- `$nr_of_mos` the number of MOs that were printed on screen by the last run of the `pr/st/get/prod/fro/set/del/acc` commands.
- `$nr_of_mos_ok` the number of MOs that were successfully operated upon by the last run of the `set/del/acc` commands.

- `$command_result` set after the `cr/pcr/pset/trun` commands. Possible values: 0 for success, 1 for failure.
- `$nr_of_alarms` the number of active alarms on the node. Set after the last run of the `al` command.
- `$nr_of_cvs` the number of CV:s that exist on the node, is set after the last run of the `cvls` command.
- `$nr_of_scanners` the number of scanners printed by the last run of the `pst/pgets/pdel/pbl/pdeb` commands.
- `$nr_of_counter_instances` the number of counter instances printed by the last run of the `pgetsn` command.
- `$moncommand` the command to start the monitor client after having run the `mon` command.

The following variables are set after running one of the **l+/u+/u-** commands:

- `$logfile` the logfile that is currently open. Set immediately after executing the **l+** command, stays set even after **l-** and will only be reset the next time a new logfile is open with **l+**
- `$undologfile` the logfile used by the undo command. Set immediately after executing the **u+/u+s** command, stays set even after **u-** and will only be reset the next time a new undo mode is started with **u+/u+s**.
- `$undocommandfile` the command file that can be used to undo the commands that were run between **u+/u+s** and **u-**. Set immediately after executing the **u-** command.
- `$undodelcommandfile` - the file containing the delete commands. Only applicable to simulated undo mode **u+s**.

The following variable is set after having logged on to the node via telnet/SSH or FTP/SFTP.

- `$password`

The contents of the variable can not be printed, it will only shown if it's empty or not. By setting this variable to empty (by doing: `$password =`), this will force MoShell to check the password again. Useful in case the password has changed on the node during the MoShell session.

The `$nr_of_vars` variable is set after running the **pv** command.

This variable indicates the number of scripting variables that were printed in the last **pv** printout. By using **pv** together with a filtering pattern (eg: **pv $table**), it is possible to find out the number of variables that had matched the pattern, for instance the number of elements in a hashtable.

The`$nr_of_lines` variable is set after using the functions "readfile" or "testfile".

After using the function "testfile", this variable is set to 0 if the file does not exist and to 1 if the file exists.

After using the function "readfile", this variable is set to 0 if the file does not exist and to the number of lines in the file if the file exists. The difference between testfile and readfile is that testfile won't actually read the file, it will just check if the file exists whereas readfile will test the file, then read it.

Example 2 Using TestFile

```
$lineContent = testfile(/path/to/myfile)
if $nr_of_lines = 0
l echo "File not found"
return
fi
```

Example 3 Using ReadFile

```
$lineContent = readfile(/path/to/myfile)

if $nr_of_lines = 0
    l echo "File not found"
    return
fi

for $lineNumber = 1 to $nr_of_lines
    if $lineContent[$lineNumber] ~ thispattern
        print We found it The line is $lineNumber.
        return
    fi
done
```

23.2 Variable assignment

A variable value can be assigned in seven ways, see below:

A variable can also be unassigned, using the **unset** command.

By using the command **unset small**, all variables are **unset**, except:

- the "system" variables (`$gawk`, `$ipaddress`, `$password`, `$moshell_version`, `$moshelldir`,`$logdir` , `$momversion`, `$cellomomversion`)
- the "global" variable(s) (ie: assigned with the ":=" sign, instead of "=").

By using the command **unset all** all variables are unset except the "system" variables.

Note: It is always good practice to unset a variable as soon as it is not needed anymore since having too many variables defined slows down the processing of the command line. It is also good to do **unset all** at the beginning and end of a command file (and before doing the `return` command) in order to avoid interference from un-needed variables. See the script examples in **moshell/commonjars/scripts**

To print all currently assigned variables, use the command **printvar**. To just print one variable, type:

```
pv pattern
(where the pattern matches the variable(s) to print)

print "$variable"
```

The variable value can be assigned in seven ways:

1. - From the command line.

The variable to be assigned is on the left side of the equal sign and the value is on the right side. Each element must be separated by spaces.

```
$i = 3
$node = RNC
$password =
```

Note: By running `password =` this sets the password to an empty value and will force AMOS to ask for the password again.

2. - At AMOS startup, using the **-v** option. In this case, the "$" sign should be omitted. (otherwise it gets interpreted by the Unix shell)

```
moshell -v upmo=CXP9011008_R1A03,ftpserv=10.1.0.16,
secure_shell=1,secure_ftp=1 rnc34
```

In this case, we can see that scripting variables (upmo and ftpserv) and user variables (secure_shell and secure_ftp) have been mixed in the same statement. This is OK because any variable that is not recognized as a user variable will be treated as a scripting variable.

3 - From the following commands: **get, fro ip2d/d2ip, h2d/d2h, uv**

The mo-filter and attribute-filter must be specified, then comes the redirection sign (>), then the variable name. If several attributes are printed, only the last attribute value gets assigned into the variable.

```
get 0 productName > $nodeType
(result: $nodeType = RBS3202)

get ethernetlink ipaddress > $ipaddress
(result: $ipaddress =10.1.128.17

get configurationversion currentupgrade > $currentUp
(result:$currentUp = UpgradePackage=CXP9011123_R12F)

get $currentUp administrativedata > $swRev
(result: $swRev = Struct{5} »> 1.productNumber = CXP9011123 »>
2.productRevision = R12F)

fro subrack=ms,slot=1,pluginunit=1$ ^r > $froid
(result: $froid =0)

d2ip -4127129085 > $ip_addr \
(result: $ip_addr = 10.1.2.3)

h2d 0xa > $res
(result: $res = 10)

uv ^credential > $credential
(result: /var/opt/ericsson/amos/moshell/commonjars/host.p12)
```

4 - Using an arithmetic operation

The following numeric operations are supported:

- + addition
- - substraction
- * muliplication
- / division
- % **modulo** (returns the remainder of an integer division operation)

Examples:

```
$i = 1 (result: $i=1)
$i = $i + 1 (result: $i=2)
$j = $i * 3 (result: $j=6)
$k = $i * $j (result: $k=12)
$l = $i / $j (result: $l=0.333)
$m = $k % 5 (result: $m=2)
```

Note: Only one operation per line is allowed. A space must exist between each element of the operation. There cannot be more than two members in the operation (ie: $i = $j + $k ==> OK. But $i = $j + $k + $l ===> NOTOK)

5 - Using the output from a Unix command:

The Unix command must be surrounded by back-quotes (‘). Variables can be used within the Unix command.

Examples:

```
$date = ‘date +%y%m%d-%H%M‘
(result: $date = 040930-1745)

get ethernetlink ipaddress > $ipaddress
(result: $ipaddress = 10.1.128.17)

$logfile = $ipaddress_$date.log
(result: $logfile = 10.1.128.17_040930-1745.log)

l+m $logfile
(open a logfile, don’t display anything on the screen)

te log read l-
(close logfile)

$errors = ‘grep -c ERROR $logfile‘
(result: $errors = the number of ERRORs found in the logfile)

l rm $logfile (remove the logfile)
```

The following is an example of making a cv that has the same name as the current startable cv but the last digit is incremented by 1

```
lt configurationversion get configuration startable > $startable
(result: $startable = RBS3045_P2.1.5_CU3_A_01)

$cvname = ‘$gawk -v cvname=$startable’BEGIN
{ print gensub(/..$/,"",1,cvname)
sprintf("%02s", substr(cvname,length(cvname)-1)+1)
}’‘ cvms $cvname
(result: $cvname = RBS3045_P2.1.5_CU3_A_02)
```

6 - Using String manipulation:

The following string operations are supported: *concatenation and substitution / replacement.*

The concatenation is performed by juxtaposing the strings. Syntax for concatenation: *var = string1string2string3* (thestringsareconcatenatedwithoutspaceinbetween) or *var = string1 string2 string3* (the strings are concatenated with spaces in between)

Syntax for concatenation:

```
$var = string1string2string3
(the strings are concatenated without space in between)

$var = string1 string2 string3
(the strings are concatenated with spaces in
between)
```

The substitution/replacement is performed using the **-s** switch to specify the string to substitute and the **-r** switch to specify the string it should be replaced with. If the **-r** switch is not used, then the string will be replaced by nothing. If the **-g** switch is specified, then all instances of the string to substituted, otherwise, only the first instance.

Syntax for substitution/replacement:

```
$var = origString -s strToSubstitute [-r strToReplaceItWith [-g]]
```

Regular expressions can be used in the string manipulations. Examples:

```
$var = abc_defabc ghi
$var1 = $var -s abc
(Result: $var1 = _defabc ghi, only first instance of abc was replaced)

$var2 = $var -s \x020
(Result: $var2 = abc_defabcghi, the space sign was removed)

$var3 = $var -s abc -g
(Result: $var3 = _def ghi, all instances of abc were replaced)

$var4 = $var -s abc -r xyz
(Result: $var4 = xyz_defabc ghi, first instance of abc was replaced)

$var5 = $var -s abc -r xyz -g
(Result: $var5 = xyz_defxyz ghi, all instances of abc were replaced with xyz)

$var6 = $var -s a.*c -r xyz
(Result: $var6 = xyz ghi, the regular expression a.*c was replaced with xyz)

$var7 = $varABC$var6
(Result: $var7 = abc_defabc ghiABCxyz ghi, the three strings $var, ABC and
$var6 have been concatenated)

$var8 = $var ABC $var6
(Result: $var8 = abc_defabc ghi ABCxyz ghi, there are spaces in between the
three strings)
```

Note: Note: if more advanced string manipulation is needed, it is always possible to use an external program such as gawk to do the string manipulation. See the example above about using Unix programs.

7 - Using output from a predefined function:

Currently, the following functions exist:

- `fdn(proxy)` input is the proxy id, output is the FDN
- `ldn(proxy)` input is the proxy id, output is the LDN
- `rdn(proxy)` input is the proxy id, output is the RDN
- `motype(proxy)` input is the proxy id, output is the MO type
- `proxy(string)` input is the LDN or FDN (NOT RDN), output is the proxy id
- `readinput(sring)` input is a prompt that should appear on the screen, so that the user can input an answer which will then be assigned to the variable.
- `readfile(file)` input is a filename. Each line of the file is assigned into an element of the hashtable into which we have assigned the result of the function. If the file is not found, the variable $nr_of_lines is set to 0, otherwise it is set to the number of lines in the file. Note, this should not be used on large files as it will slow down things very much.
- `testfile(file)` input is a filename. If the file is not found, the variable $nr_of_lines is set to 0, otherwise it is set to 1.
- `split(string)` The string is split into the array specified on the left side of the equal sign (see example below). The separator used to split the string can be specified in the variable "$split_separator". By default it is a

space. If the $split_separator has been changed and needs to be reset to the default value, just run the command "unset $split_separator". The number of elements in the array is stored in the variable $split_last

Examples:

```
Example 1:
lt iublink
ma iub iub

for $mo in iub
    $mordn = rdn($mo)
    if $mordn ~ 1023
         lcc $mordn
         lbl $mordn,
    fi
done

Example 2:
$var = readinput(Please confirm [y/n]: )
     if $var !~ ^y
        return
     fi

Example 3:
$table = readfile(/path/to/myfile)
for $lineNumber = 1 to $nr_of_lines
    print $table[$lineNumber]
    $word = split($table[$lineNumber])
    if $word[1] ~ ^#
       $nr_of_comments = $nr_of_comments + 1
    fi
    unset $word
    unset $table[$lineNumber]
 done
```

Note: By unsetting the entry we've just read - provided we don't need it anymore - will make things faster.

The following example shows an AMOS script which makes a customized CV name like: date_nodeType_swRev

```
cvls
$date = 'date +%y%m%d' (result: $date = 040930)
get 0 productName > $nodeType
(result: $nodeType = RBS3202)
$nodeType = $nodeType -s RBS
(result: $nodeType = 3202)
get configurationversion currentupgrade
> $currentUp
(result: $currentUp = UpgradePackage=\ CXP901913%2_R12N)
get $currentUp administrativedata
> $swRev (result: $swRev = Struct{5})
>>> 1.productNumber = CXP901913/2 >>>
2.productRevision = R12N >>>
3.\ productName = CXP901913%2_R12N ....)
$swRev='gawk -v currentsw="$swRev"
'BEGIN{ swrev=gensub(/\r|\n/,"","g",currentsw);
print gensub(/^.*Revision = |
>>> 3.product.*$/,"","g",swrev) }''
(result: $swRev = R12N)
cvms $date_$nodeType_$swRev
(result: cvms 040930_3202_R12N)
```

23.3 Hashtables (arrays)

The index and the value of the hashtable can be a variable, a constant, or a mix of both.

To print a hashtable, do: **`pv tablename`**

The following is an example of assigning constants into a hashtable:

```
>> $table[1] = hello
>> $table[2] = hej
>> $table[hoho] = 5
>> pv tab (result printout:)
$table[hoho] = 5
$table[1] = hello
$table[2] = hej
```

The following is an example of assigning variables into a hashtable:

```
>> $mo = AtmPort=MS-6-1
>> $proxy = proxy($mo)
>> $proxylist[$mo] = $proxy
>> $mo = AtmPort=MS-6-2
>> $proxy = proxy($mo)
>> $proxylist[$mo] = $proxy
>> pv proxylist (result printout:)
$proxylist[AtmPort=MS-6-1] = 103
$proxylist[AtmPort=MS-6-2] = 112
```

23.4 If/Else constructs

The `if` statement must be followed by a condition. The comparison operator of the condition must be surrounded by spaces. Zero or more `else if` statements can be used after the `if` statement. Zero or one `else` statements can be after the `if` or `else if` statements.

The end of the if/else structure must be specified with a `fi` statement. Each statement must be on its own line and can be followed by one or more commands. Several conditions can be combined, using the logical AND (&&), or the logical OR (||). Any number of AND/OR can be put on a line but NOT BOTH on the same line.

Grouping conditions with brackets is **NOT** supported.

The `return` command can be used to exit from the command file in case a certain condition is met. Type **`h return`** for more information on how to use this command.

Syntax examples:

```
1.
if condition
   command1
   command2
fi

2.
if condition1 || condition2
   command1
   command2
else
   command3
fi

3.
if condition1 && condition2 && condition3
      command1
else if condition4
      command2
else
      command3
fi
```

A condition can use the following comparison operators:

- = equals
- ~ matches (as in pattern match)
- != is not equal to
- !~ does not match
- > greater than
- < less than
- >= greater than or equal to
- <= less than or equal to

The words around the operator can be either a variable or a single word but NOT a string containing spaces or a concatenation of a variable and string. Following conditions are syntactically correct:

```
if $var1 = $var2
if mystring ~ $var
if 10 > 3
if $i < 2
```

The following conditions are NOT syntactically correct and will return unexpected results:

```
if mystring_$var1 ~ $var2
if mystring is this ~ your string
```

The follow example checks if a variable `$var` exists. If $ exists (that is has any value set) then it will do something.

```
if $var
    ...do something
fi
```

The follow example checks for node type and see attenuation accordingly

```
get 0 productname > $nodeType

if $nodeType ~ 3202 || $nodeType ~ 3104
    get feeder attenuation
    set feeder attenuation 4
else if $nodeType ~ 3101
    get feeder attenuation
    set feeder attenuation 16
else
    get feeder attenuation
fi
```

23.5 For constructs

The parameter to the `for` construct can be:

1. `ever` - to repeat the loop an infinite number of times

2. `numberOfIterations` - to repeat the loop a specific number of times

3. `$mo in mogroup` - to run the body of the loop on each MO of the specified moGroup. MO groups are created using **ma/lma**. Type **`h ma`** for more information.

4. `$board in boardgroup` - to run the body of the loop on each board of the specified board group. Board groups are created using **ba/ba**. Type **`h bo`** for more information.

5. `$var in $table` - for each iteration of the loop, `$var` will cycle through the index values of the hashtable `$table`

6. `$var = $start to $stop` `$var` is assigned every integer value between `$start` and `$stop` . `$start` and `$stop` can be variables or constants but must be an integer. If `$start` is smaller than `$stop` then the order will be ascending, otherwise it will be descending.

The end of the `for` structure must be specified with a `done` statement.

The `wait` command can be used in the body of the loop to specify a delay to wait in between each iteration. The delay can be in seconds, minutes, hours, or even ROP periods. Type **`h wait`**, for more information.

Note: Do not use the **sleep** command as this will result in hanging if the loop is aborted.

The loop can be aborted any time by typing **`ctrl-z`** , then **`touch stopfile`**, then **`fg`**. The stopfile path is shown in the window title bar. Type **`h ctrl-z`** for more information about aborting out of loops.

The `break` command can be used within the loop to exit from the loop.

Syntax examples:

```
Example 1:
for ever
    command1
    command2
done

Example 2:
for numberOfTimes
    command1
    wait numberOfSeconds
done

Example 3:
for $mo in moGroup
    get $mo attribute > $variable
    $variable1 = ....
    set $mo attribute $variable1
done

Example 4:
for $board in boardGroup
    bl $board
    acc $board restart 0 1
    wait 10
    deb $board
    lhsh $board vii
done

Example 5:
for $proxy in $proxytable
    bl $proxy
    st $proxy
    deb $proxy
    st $proxy
    get $proxy operational > $opstate
    if $opstate != 1
        break
    fi
done

Example 6;
for $var = $maxproxy to $minproxy
    del $var
done

for $var = 1 to 6
    te e trace$min process
done
```

The following example demonstrates hoe to check the progress of a UP installation, every 10 seconds. Break from the loop if the result is 1 (INSTALL_COMPLETED), and continue with upgrade action. Abort the command file if the result is 6 (INSTALL_NOT_COMPLETED)

```
lt upgrade
acc upgradepackage=xxx nonblockinginstall
for ever
    $return = 0
    wait 10
    get upgradepackage=xxx state > $upstate

    if $upstate ~ ^1
        break
    else if $upstate ~ ^6
        $return = 1
        break
    fi
done

if $return = 1
    return
fi

acc upgradepackage=xxx upgrade
```

The following is an example of running a test case 50 times.

```
for 50
    run testcase_3.1.1.cmd
    wait 2m
done
```

The following is an example of increasing the primaryCpichPower by 0.1 dBm on each UtranCell.

```
lt ^utrancell
ma cell ^utrancell

for $mo in cell
    get $mo primarycpichpower $pich
    $pich = $pich + 1
    set $mo primarycpichpower $pich
done
```

The follow example shows how to restart all boards in a board group:

```
ba spb spb
for $board in spb
    acc $board restart 0 1
done
```

The following is an example of saving the fRO values of all programs into a table and then restart every program:

```
lma programs_on_slot_19 subrack=ms,slot=19,.*program

for $prog in programs_on_slot_19
    $i = $i + 1
    fro $prog ^res > $frolist[$i]
done

for $fro in $frolist
    restartObj pgm $fro
done
```

The follow example shows how to restart some boards in a specific order:

```
for $var = 20 to 14
    $board = 00$var00
    acc $board restart 0 1
done
```

23.6 User-defined functions

Users can define their own functions, using the `func/endfunc` construct.

If the function is called with arguments, these are assigned to the variables $1, $2, $3, etc

Note: Two types of comments can be used in scripts. Visible comments start with the "#" sign. These comments are printed on the screen while the script is executing and invisible comments which start with the "//" sign. These comments are not printed on the screen.

The following is an example of defining a function (the function definitions can be run in a different command file):

```
#This function checks the state of the \
mirrored disks and returns
#once the disks are in sync

func check_disk_state
    #if $1 is undefined or different \
to an integer value
    #then we set it to 10 seconds
    if $1 ~ ^[0-9]+$
        $wait_interval = $1
    else
        $wait_interval = 10
    fi

    for ever
        wait $wait_interval
        l+om $tempdir/diskstate
        lh coremp mirror s
        l-
        $res = 'grep -c "Peer Disk \
        Status: *Valid" $tempdir/diskstate
        if $res > 0
            break
        fi
    done
endfunc


func waitforuser
    $date = 'date "+%Y-%m-%d %H:%M:%S"'
    for ever
        $reply = readinput(Waiting \
        from [$date]. Type "y" when ready: )
        if $reply ~ ^[yY]
            break
        fi
    done

    $date = 'date "+%Y-%m-%d %H:%M:%S"'
    print "Finished waiting at [$date]"
endfunc
```

The following is an example of calling our own user defined functions.

```
#First we are running a file that contains all
#the definitions for our user-defined functions.
#(Functions can also be defined within the
#same script, but by keeping all
#functions in a separate file means that
#several command files can use the same functions)
#We have called the function check_disk_state
#with an argument "5" which in this
#case will be used as the "$wait_interval"
#parameter in the function

run ~/myfunctions_define.mos
for ever
    check_disk_state 5
    waitforuser
    acc 0 restart
    pol
done
```

23.7 Nesting for and if statements

It is possible to nest one or more `if/else` statement within a loop statement and vice-versa. But it is currently not possible to nest an if/else statement within an if/else statement and a loop statement within another loop statement.

The current workaround is to put the for/if constructs into functions:

The following example starts an install, then checks the state of the install every 10 seconds. Once the upgradepackage is installed, it starts the upgrade. Then it checks the state again and once the upgrade is in state awaiting confirm, it confirms the upgrade

```
$UP = upgradepackage=CXP9011123_R12F
acc $UP nonblockinginstall
for ever
    wait 10
    get $UP state > $state
    if $state ~ ^1
        break
    fi
done

get $UP state > $state

if $state ~ ^1
    acc $UP upgrade
fi
wait 120
for ever
    wait 10
    get $UP state > $state
    if $state ~ ^3
        break
    fi
done

if $state ~ ^3
    acc $UP confirmupgrade
fi
```

The following example checks the mirror status of the node (that is, to check whether the passive FTC MP is ready to take over or not)

```
for ever
    board_status -d 00 10 -c \
"mirror stat" | tee tmpfile.tmp
    board_status -d 00 11 -c \
"mirror stat" | tee -a tmpfile.tmp
    $tmp = ‘grep -c \
"Peer Disk Status: Valid" tmpfile.tmp‘

if $tmp > 0
        break
    else
        wait 60
    fi
done
```

The following example checks if an upgrade is complete (that is, the upgradepackage is in state 3)

```
wait 300 #give it some time to run first
for ever
    pol 1 1
    get upgradepackage=mypkg state > $state

    if $state ~ ^1
        break #upgrade failed
    else if $state ~ ^3
        break #upgrade complete
    fi

    wait 60
done
```

23.8 Example Scripts

Example scripts can be found under `/opt/ericsson/amos/moshell/commonjars/scripts` and `/opt/ericsson/amos/moshell/examples/scripting/`

24 MO Batch

The purpose of the `mobatch` command mode is to send AMOS commands to several Network Elements in parallel. The `mobatch` command must adhere to the following syntax:

mobatch [options] sitefile command(s)|commandfile|commanddir [logdirectory]

The first argument is the **sitefile**. The **sitefile** contains the list of sites to connect to. Each line in the **sitefile** contains:

- The `IP/DNS` addresses and/or site names whose IP address are defined in the IP database file.
- Optionally: the user variables/scripting variables to input with the `-v` option.

Example 4 The contents of a simple sitefile

```
#Note: it is possible to use a mix of site names and IP/DNS addresses

3124100

#These are ip addresses and DNS addresses
#For certain sites, we are specifying some uservariables/scriptingvariables that
 will be input
#to the corresponding amos session with the "-v" option
10.1.10.234
10.1.10.242 corba_class=2
```

Example 5 A simple IP database file contents

```
#The first field is the Network Element name, the second field is the ip address
or DNS name, and the third field is the
#telnet password.
################
### RNCs  (in this case we are mapping the node name to its DNS name) and also
specifying its password
################
H2RG ndb2g rnc_h2rg
H2RH ndb2h rnc_h2rh

################
### RBS's (in this case we are mapping the node name to its ip address and also
specifying its password)
################
2000010 10.123.10.17            rbs_2000010
2000020 10.123.9.241            rbs_2000020
```

The second argument is the **command(s)**, **commandfile** or the **commanddir** to be run in parallel.

Example 6 A simple command file contents

```
lt ^(rncfunction|channelswitching|rrc|uerc|uemeascontrol|cchframesynch|utrancell)$
bl      utrancell
set     RncFunction=1           uetrFileSize                    275
set     RncFunction=1           ctrFileSize                     5000
set     RncFunction=1           recordingStorageSize            60000
set     ChannelSwitching=1      inactivitytimer                 1
set     ChannelSwitching=1      ulRlcBufUpswitch                512
set     ChannelSwitching=1      downswitchTimer                 10
set     ChannelSwitching=1      thpReportInterval               5
set     ChannelSwitching=1      downswitchTimerThreshold        16
set     ChannelSwitching=1      downswitchThreshold             8
set     Rrc=1                   packetEstMode                   0
set     UeRc=7                  dlAse                           1110
set     UeRc=7                  ulAse                           1110
set     UeRc=7                  dlPwrEstimation                 23
set     UeRc=6                  dlAse                           200
set     UeRc=6                  ulAse                           200
set     UeRc=5                  dlAse                           210
set     UeRc=5                  ulAse                           200
set     UeMeasControl=1         TimeToTrigger1a                 11
set     UeMeasControl=1         TimeToTrigger1b                 11
set     UeMeasControl=1         TimeToTrigger1c                 11
set     UeMeasControl=1         TimeToTrigger1d                 11
set     UeMeasControl=1         Hysteresis1d                    2
set     CchFrameSynch=1         dto                             10
set     UtranCell               pwrEstFact                      100
set     UtranCell               pwrHyst                         400
set     UtranCell               tmCongAction                    10000
set     UtranCell               maxTxpowerUl                    24
set     UtranCell               qRxLevMin                       -115
set     UtranCell               qQualMin                        -20
set     UtranCell               sRatSearch                      2
set     UtranCell               sf8Adm                          1
set     UtranCell               sf32Adm                         8
deb     utrancell
```

If a directory is given, then a different **commandfile** will be used for each Network Element, the name of each **commandfile** should be `network-element-name.cmd` or `network-element-name.mos`, the Network Element-name should be the same as given in the sitefile. if the Network Element-name is `rbs602` then the **commandfile** should be `rbs602.cmd` or `rbs602.mos` and it will be run exclusively against that Network Element.

The third argument **logdirectory** is optional. If no **logdirectory** is specified, a default one will be used. This is where the output of running the `mobatch` command is placed.

Example 7 Sample output of

```
Sitefile = /var/opt/ericsson/amos/rbs501.txt. Number of sites = 1
Command = hc
Timeout set to 20 minutes (default value)
Maximum number of parallel processes set to 10 (default value)

Logfiles stored in /var/opt/ericsson/amos/moshell/moshell_logfiles/logs_mobatch/
2007-08-27/rbs501.txt/10-53

################################################################################
>>>>> STARTED (pid)
** FINISHED (result)              STATUS (started, finished, queue, running,
sites running)
################################################################################
>>>>> 159.107.180.45 (12339)       1s  0f  0q  1r: 159.107.180.45
...<cut>...
```

Options may also be specified to the `mobatch` command these are:

- **-t (minutes)** - This allows the user to specify the number of minutes before the `mobatch` command times out. If this value is set to `0` the command will never time out. A default time out of `20` minutes is configured in AMOS.
- **-p (processes)** - This specifies the maximum number of AMOS sessions that will run in parallel, by default a maximum number of `10` processes can be started.
- **-v (uservariables)** - This specifies an AMOS user variable.

Example 8 Running multiple commands against all Network Elements listed in a sitefile

```
>mobatch -p 7 -t 60 ~/sitefiles/london-rbs 'lt ^utrancell ; \
st cell'
```

Example 9 Running multiple commands contained in a script against all Network Elements listed in a sitefile

```
>mobatch ~/sitefiles/all-rbs.txt \
~/cmdfiles/do_increase_power.mos
```

Example 10 Specifying a user variable while running commands against Network Elements listed in a sitefile

```
>mobatch -v corba_class=2 -p 20 ~/paris-rnc 'hc'
```

25 Appendix

25.1 AMOS Command Authorization

Certain AMOS commands can be viewed as being dangerous when they are run by inexperienced users, with this in mind AMOS can be used to ensure that an operator is authorized to run a particular AMOS command. The OSS currently defines five default roles within the OSS namely:

- APPLADM
- ASSOPE
- NWOPE
- OPE
- SYSADM

When a user is setup within the OSS they are usually assigned one of these default roles, AMOS will use three of these roles to check if a user has the correct command authorization within the system before executing the command against a Network Element.

The three roles are:

- ASSOPE
- OPE
- SYSADM

AMOS logically breaks down commands as being one of three types, a command may be READ, WRITE or TELNET. So in order for a user to be able to run a READ command they must have been assigned either the ASSOPE, OPE or the SYSADM role within the OSS. In order for a user to run a WRITE command they must have been assigned either the OPE or the SYSADM role within the OSS. In order for a user to run a TELNET based command the user must have been assigned the SYSADM role within the OSS.

The following table identifies all AMOS commands and what role a user needs to have been assigned in order to run that particular command.

Table 11 AMOS Command Authorization

Command	**Command Type**	**OSS Role Required**
mom[tcd]	READ	ASSOPE,OPE or SYSADM role.
lt/ltc[1-9]	READ	ASSOPE,OPE or SYSADM role.
lc[1-9]/lcc	READ	ASSOPE,OPE or SYSADM role.
pr/lpr	READ	ASSOPE,OPE or SYSADM role.
ma/lma	READ	ASSOPE,OPE or SYSADM role.
mr/lmr	READ	ASSOPE,OPE or SYSADM role.
mp	READ	ASSOPE,OPE or SYSADM role.
get/lget	READ	ASSOPE,OPE or SYSADM role.
hget[c]/lhget[c]	READ	ASSOPE,OPE or SYSADM role.
kget/lkget	READ	ASSOPE,OPE or SYSADM role.
st/lst	READ	ASSOPE,OPE or SYSADM role.
prod	READ	ASSOPE,OPE or SYSADM role.
lk/llk	READ	ASSOPE,OPE or SYSADM role.

lko/llko	READ	ASSOPE,OPE or SYSADM role.
acl/lacl	READ	ASSOPE,OPE or SYSADM role.
run	READ	ASSOPE,OPE or SYSADM role.
trun[i]	READ	ASSOPE,OPE or SYSADM role.
ctrl-z	READ	ASSOPE,OPE or SYSADM role.
pol[shcu]	READ	ASSOPE,OPE or SYSADM role.
re[i]	READ	ASSOPE,OPE or SYSADM role.
getmom	READ	ASSOPE,OPE or SYSADM role.
parsemom	READ	ASSOPE,OPE or SYSADM role.
flt/fltc	READ	ASSOPE,OPE or SYSADM role.
ld	READ	ASSOPE,OPE or SYSADM role.
sget/lsget	READ	ASSOPE,OPE or SYSADM role.
stc[p][r]	READ	ASSOPE,OPE or SYSADM role.
std	READ	ASSOPE,OPE or SYSADM role.
stv[b][r]	READ	ASSOPE,OPE or SYSADM role.
stt[r]	READ	ASSOPE,OPE or SYSADM role.
hc	READ	ASSOPE,OPE or SYSADM role.
diff/ldiff	READ	ASSOPE,OPE or SYSADM role.
str	READ	ASSOPE,OPE or SYSADM role.
lki	READ	ASSOPE,OPE or SYSADM role.
uer	READ	ASSOPE,OPE or SYSADM role.

ced	READ	ASSOPE,OPE or SYSADM role.
al[atkc]	READ	ASSOPE,OPE or SYSADM role.
lg[<options>]	READ	ASSOPE,OPE or SYSADM role.
uv	READ	ASSOPE,OPE or SYSADM role.
pv	READ	ASSOPE,OPE or SYSADM role.
!/l	READ	ASSOPE,OPE or SYSADM role.
l+[m][m][s][o]/l-/l?	READ	ASSOPE,OPE or SYSADM role.
mon/mon+/mon-/mon?/ monr	READ	ASSOPE,OPE or SYSADM role.
ftget[c]/ftput[c]	READ	ASSOPE,OPE or SYSADM role.
htget	READ	ASSOPE,OPE or SYSADM role.
edit	READ	ASSOPE,OPE or SYSADM role.
hi	READ	ASSOPE,OPE or SYSADM role.
p/w/pw/b	READ	ASSOPE,OPE or SYSADM role.
prox	READ	ASSOPE,OPE or SYSADM role.
col	READ	ASSOPE,OPE or SYSADM role.
ul	READ	ASSOPE,OPE or SYSADM role.
conf[bld]	READ	ASSOPE,OPE or SYSADM role.
gs/gsg	READ	ASSOPE,OPE or SYSADM role.
ip2d/d2ip	READ	ASSOPE,OPE or SYSADM role.
h2d/d2h	READ	ASSOPE,OPE or SYSADM role.
b2b/b2h	READ	ASSOPE,OPE or SYSADM role.

wait	READ	ASSOPE,OPE or SYSADM role.
return	READ	ASSOPE,OPE or SYSADM role.
print	READ	ASSOPE,OPE or SYSADM role.
alias/unalias	READ	ASSOPE,OPE or SYSADM role.
q/by//exit/quit	READ	ASSOPE,OPE or SYSADM role.
pmom[acd]	READ	ASSOPE,OPE or SYSADM role.
pget/lpget	READ	ASSOPE,OPE or SYSADM role.
spget/lspget	READ	ASSOPE,OPE or SYSADM role.
hpget[c]/lhpget[c]	READ	ASSOPE,OPE or SYSADM role.
pdiff/lpdiff	READ	ASSOPE,OPE or SYSADM role.
pmx[hfdn]	READ	ASSOPE,OPE or SYSADM role.
pmr[ag]	READ	ASSOPE,OPE or SYSADM role.
pme[fd][cgu]	READ	ASSOPE,OPE or SYSADM role.
pst	READ	ASSOPE,OPE or SYSADM role.
pgets[n]	READ	ASSOPE,OPE or SYSADM role.
cvls/cvcu/cvls1	READ	ASSOPE,OPE or SYSADM role.
emom	READ	ASSOPE,OPE or SYSADM role.
dcg[measr]	READ	ASSOPE,OPE or SYSADM role.
set/lset	WRITE	OPE or SYSADM role.
del/ldel	WRITE	OPE or SYSADM role.
u+/u-/u?/u!	WRITE	OPE or SYSADM role.
fclean	WRITE	OPE or SYSADM role.
remod	WRITE	OPE or SYSADM role.

bl[s]/lbl[s]	WRITE	OPE or SYSADM role.
acc/lacc	WRITE	OPE or SYSADM role.
deb/ldeb	WRITE	OPE or SYSADM role.
cr	WRITE	OPE or SYSADM role.
cvmk/cvms/cvset/cvrm/ cvget	WRITE	OPE or SYSADM role.
pbl/pdeb	WRITE	OPE or SYSADM role.
fro/lfro[m]	TELNET	SYSADM role.
lmid[c]	TELNET	SYSADM role.
ftree	TELNET	SYSADM role.
bo[r]/ba[swdp/br[wd]/bp	TELNET	SYSADM role.
proglist	TELNET	SYSADM role.
progkill	TELNET	SYSADM role.
lh	TELNET	SYSADM role.
te	TELNET	SYSADM role.
inv[hr]	TELNET	SYSADM role.
sql+/sql-/sql?	TELNET	SYSADM role.
tg[r][c][d]	TELNET	SYSADM role.
cab[slxradgtme]	TELNET	SYSADM role.

25.2 AMOS User Specific Variables

The following is a description of the AMOS variables and their default values that can be customized by each user.

Table 12 AMOS User Specific Variables

Variable Name	Description	Default Value
amos_debug	This value if set to `true` will show the OSS operations that AMOS is performing on each command.	false
ask_for_attribute_type	If `0`, attribute type will be automatically entered in the **cr** and **actc** commands. If `1`, attribute type must be entered manually.	0

bldebset_confirmation	Ask for confirmation in **bl**/**deb**/**set**/**actc** commands. `0` means no confirmation and `1` means confirmation. Confirmation is never required when running a command file, regardless of these two variables. Confirmation can be toggled online using command **conf[bl]**.	1
credential	This defines the path to a host credential file. This file is required to handle CORBA Security level 1. This file may be generated by PKS server or found on the Network Element (in /c/java/host.p12) or found in OSS (in /etc/opt/ericsson/nms_cif_security/certificates/host.p12). This file has an unlimited lifetime.	/opt/ericsson/amos/moshell/commonjars/host.p12
commandlog_path	The path to the command log for the **lgo** command.	/c/commandlog
csnotiflist	The list of attributes that are not displayed on screen by the CS notification client. This client may be run with the script `/opt/ericsson/amos/moshell/runClient.sh -c <ipaddress>`.	availabilitystatus
default_mom	Default xml MOM file (in case not stored on the Network Element.	/var/opt/ericsson/amos/jarxml/MGW_R2.xml
del_confirmation	Ask for confirmation in the **del** or **rdel** commands. `0` means no confirmation and `1` means confirmation. Confirmation is never required when running a command file, regardless of these two variables. Confirmation can be toggled online using command **confd**.	1

dontfollowlist	List of Managed Objects that should not be followed in the **lk** or **llk** functions. The list is not case sensitive. The Managed Objects must be separated by a comma.	rncmodule,unisaalprofile,nnisaalprofile,atmtrafficdescriptor,pluginunit,reliableprogramuniter,Ranap,LocationArea,RoutingArea,ServiceArea,UtranCell,TimDevice,Etm4,Etm1,Aal2ap,NbapCommon,NbapDedicated,NodeSynch,Mtp2Pr ofileItu,Mtp2ProfileAnsi,Aal2QosProfile,Aal2QosCodePointProfile,Mtp2ProfileChina,AiDevice,Subrack,NodeSynchTp,TdmTermGrp ,Slot
editor	The Editor to use for the **edit** command to edit a file on the Network Element.	vi
fast_lh_threshold	The **lh** command is made faster by sending the command file to the Network Element via FTP and running it locally using the **shell -f** construct. The command file is then removed from the Network Element. The **lh** command will only use FTP if more than `fast_lh_threshold` commands have to be sent.	5
fast_cab_threshold	The **cab** command is made faster by sending the command file to the Network Element via FTP and running it locally using the **shell -f** construct. The command file is then removed from the Network Element. The **cab** command will only use FTP if more than `fast_cab_threshold` commands have to be sent.	10
ftp_port	The FTP port to use.	21
followlist	List of Managed Objects that should be followed in the **lk** or **llk** commands via their parent/child relationship. The list is not case sensitive. The Managed Objects must be separated by a comma.	iublink,mtp3bsls,iub

ftp_timeout	The maximum time allowed for FTP or HTTP transfers	600
http_port	The HTTP port to use.	80
inactivity_timeout	AMOS will exit if no user input arrives within this amount of time in seconds.	43200
include_nonpm	The HTTP port to use.	0
ip_connection_timeout	Disconnects if a connection could not be established within this amount of seconds.	20
ip_database	The IPDATABASE file is used: to specify Network Element names and Network Element passwords. Several IPDATABASE files may be specified if they have been separated with a comma.	/opt/ericsson/amos/moshell/sitefiles/ipdatabase
ip_inactivity_timeout	Disconnects if no output is received from the server within this amount of time in seconds.	45
java_settings_high	This is java memory settings: ms is initial memory, mx is maximum memory. The more Managed Objects on the Network Element the more memory is necessary. The high setting is used for RNC Network Elements.	-mx512M -ms8M
java_settings_low	This is java memory settings: ms is initial memory, mx is maximum memory. The more Managed Objects on the Network Element the more memory is necessary. The high setting is used for RBS Network Elements.	-mx512M -ms4M
java_settings_medium	This is java memory settings: ms is initial memory, mx is maximum memory. The more Managed Objects on the Network Element the more memory is necessary. The high setting is used for RANAG Network Elements.	-mx512M -ms4M

keepLmList	Files that should not be cleaned up by the **fclean** command.	/d/loadmodules/CXC1326054_R1C01,/d/loadmodules/CXC1326054_R1D01,/c/java/host.p12,/c/java/TaskCo nfiguratio nFile.txt,/c/java/OLD_SecurityManagement.prp,/d/loadmodules/CXC33077_R5A08,/c/java/SecurityManagement.prp
lt_confirmation	Ask for confirmation in **lt** and **lc** commands. `0` means no confirmation and `1` means confirmation. Confirmation is never required when running a command file, regardless of these two variables. Confirmation can be toggled online using command **confl**.	1
loginfo_print	This variable enables or disables the printing of `logstart` or `logclose` information in the **l+** or the **l-** commands. A value of `1` means enabled where a value of `0` means disabled.	1
muteFactor	The default mute factor, this variable controls the amount of progress dots that may be shown when running the **l+m** or the **l+mm** command.	100
node_login	This value should always be `1`,	1
print_lmid	This value states whether AMOS should perform loadmodule name translation in COLI commands. A value of `1` means append loadmodule name translation in COLI commands.	1
PrintProxyLDN	This variable controls the output of **lt** or **lc** commands. This command only applies when the variable `PrintProxySilent` is set to `0`. A value of `0` means print only proxy number whereas a value of `1` means print the LDN and proxy number.	1

PrintProxySilent	Display progress during **lt** or **lc** command. A value of `0` means progress is printed with each LDN being loaded being printed on the screen whereas a value of `1` means that no progress is printed until the command is completed where a summary line will be printed.	1
prompt_highlight	A variable that specifies whether the prompt should appear in bold or not where a value of `1` means the prompt should appear in bold and a value of `0` means that the prompt should appear normal.	1
pm_wait	The default wait time in seconds when checking Performance Management counter increments.	25
pm_logdir	In this variable you can define a different path for the Performance Management ROP files. If the variable is not defined, the files are stored under the standard logfiles directory: ~/moshell_logfiles/logs_moshell	<empty>
sa_credential	The path to the STAND-ALONE credential file. This file is required to handle CORBA Security level 2. This file is usually generated by the SLS server. This file has a limited lifetime, set by SLS server.	/opt/ericsson/amos/moshell/commonjars/sam.pbe
sa_password	The STAND-ALONE credential password. This password is required to handle CORBA Security level 2.	oemas
secure_ftp	Set this variable to a value of `1` if you wish to securely FTP from a Network Element using the `sftp` protocol.	0
secure_port	The secure port to be used for `ssh` or `sftp` a different value can be used in case of RSG port forwardings.	22
secure_shell	Set this variable to a value of `1` if you wish to use `ssh` when contacting the Network Element.	0

corba_class	The type of CORBA communication to perform when contacting the Network Element. A value of 1 means we use `vbjorb` for both secure and unsecure Network Elements and a value of 2 means we use `jacorb` for unsecure communication with the Network Element and a value of 3 means we use `jacorb` for secure communication with the Network Element.	1
set_window_title	This variable specifies if we should set the Window Title at startup, a value of `0` means no Window Title is set during startup.	1
show_timestamp	This variable specifies if we should show a timestamp value. A value of `0` means no timestamp is displayed.	1
telnet_port	The port to use for TELNET operations.	23
transaction_timeout	The timeout in seconds before a transaction when performing a set, create, delete or action will be aborted.	3600
username	The TELNET, SSH, FTP and SFTP username that is specified if AMOS cannot correctly configure the user's username.	moshelluser
xmlmomlist	List of xml files to search for in `http://node/cello/oe/xml`. The files will be searched in the same order as displayed in the list.	rnc_node_mim.xml,RbsNode.xml,mmgwr4_node_mom_hidden.xml,mmgwr4_node_mom_open.xml,mgw_node_mom_ro.xml,BSCMOM.xml,CelloMOM.xml

26 References

[1] *Operations Support System (OSS) Glossary*
0033-AOM 901 017/2

[2] *OSS Library Typgraphic Conventions*
1/154 43-AOM 901 017/4

[3] *TSS System Administrators Guide*
1543-APR 901 0003

[4] *SLS, Single Logon Server, Installation Guide*
1531-APR 901 0128

[5] *AMOS, Advanced MO Scripting, Scripting User Guide*
2/1553-APR9010253