Oracle® Communications Convergent Charging Controller

SMS Email Interface Technical Guide

Release 15.0.0

October 2023

Copyright

Copyright © 2023, Oracle and/or its affiliates.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software" or "commercial computer software documentation" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government’s use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

About This Document

Scope

The scope of this document includes all the information required to install, configure and administer the SMS Email Interface application.

Audience

This guide was written primarily for system administrators and persons installing, configuring and administering the SEI application. However, sections of the document may be useful to anyone requiring an introduction to the application.

Prerequisites

A solid understanding of UNIX and a familiarity with IN concepts are an essential prerequisite for safely using the information contained in this technical guide. Attempting to install, remove, configure or otherwise alter the described system without the appropriate background skills, could cause damage to the system; including temporary or permanent incorrect operation, loss of service, and may render your system beyond recovery.

Although it is not a prerequisite to using this guide, familiarity with the target platform would be an advantage.

This manual describes system tasks that should only be carried out by suitably trained operators.

Related Documents

The following documents are related to this document:

- *Messaging Manager Technical Guide*
- *SMS Email Interface Alarms Guide*

Document Conventions

Typographical Conventions

The following terms and typographical conventions are used in the Oracle Communications Convergent Charging Controller documentation.

Formatting Convention	Type of Information
Special Bold	Items you must select, such as names of tabs. Names of database tables and fields.
Italics	Name of a document, chapter, topic or other publication. Emphasis within text.
Button	The name of a button to click or a key to press. **Example:** To close the window, either click **Close**, or press **Esc**.
Key+Key	Key combinations for which the user must press and hold down one key and then press another. Example: **Ctrl+P** or **Alt+F4**.
`Monospace`	Examples of code or standard output.
`Monospace Bold`	Text that you must enter.
variable	Used to indicate variables or text that should be replaced with an actual value.
menu option > menu option >	Used to indicate the cascading menu option to be selected. Example: **Operator Functions > Report Functions**
hypertext link	Used to indicate a hypertext link.

Specialized terms and acronyms are defined in the glossary at the end of this guide.

Chapter 1

System Overview

Overview

Introduction

This chapter provides a high-level overview of the application. It explains the basic functionality of the system and lists the main components.

It is not intended to advise on any specific Oracle Communications Convergent Charging Controller network or service implications of the product.

In this Chapter

This chapter contains the following topics.

What is SEI?

Introduction

The Short Message Service Email Interface (SEI) facilitates sending and receiving Internet email on a mobile telephone handset using Short Message Service (SMS) technology.

About the SEI Listen Port

The SEI uses Simple Mail Transfer Protocol (SMTP) for sending and receiving email. SMTP by default uses the standard TCP port 25. The SEI, however, listens on a non-standard port number for SMTP traffic that you configure in the `sei`, `server` section of the **eserv.config** configuration file. For more information about configuring the SEI listen port, see the discussion on SEI configuration, *server section parameters* (on page 14).

To enable the SEI to handle SMTP traffic, you must also configure your system to ensure that SMTP traffic arrives on the configured port.

SMS to Email

Format of SMS

Subscribers can send a specially formatted SMS to a Direct SMS-to-Email short code to send email from their SMS enabled cell phone.

- This number would typically be labeled “email” in their own phone's address book.
- The first word of the SMS will be the destination email address. The message body follows, for example:

```
sam@example.invalid.com How about lunch?”
```

- Optionally the subject may be provided by prefixing it with an 's', the subject continues until the first double space or newline in the SMS, for example:
  ```
  sam@example.invalid.com s Lunch today  How about lunch?
  ```
  The Subject is “Lunch today” and the message body is “How about lunch?”.
- The From:Email address will be in the form msisdn@domain. This facilitates replies back to the phone – see *Email to SMS* (on page 2).
- Subscribers may be charged, in the SMSC, for each email sent through the gateway.

Email to SMS

Format of email

Emails sent to an address that is known to be a mobile subscriber, for example msisdn@domain, will be relayed to the MSISDN as an SMS will be relayed to the MSISDN as an SMS through the Email_to_SMS control plan.

The From:email address and subject will be tightly packed into the SMS as follows: for example:

```
From: Sam Smith <sam@example.invalid.com>
To: 441632960001 <441632960001@example.invalid.com>
Date: Aug 5, 2005 2:27 PM
Subject: RE: Lunch
How about Rahzoo?.
```

This email becomes the SMS:

```
sam@example.invalid.com  RE: Lunch:  How about Rahzoo?
```

The From:Email address is looked up in the mobile subscriber's profile. If it is:

- found, the From:address is set to the special short code plus a digit representing the placement in the listing.
  For example, sam@example.invalid.com is found in subscriber 441632960001's Email Address 3. The From:address is set to 703, for example 70 is the special short code and 3 represents Email Address 3. The mobile subscriber would send a reply to 703 which will trigger the Enhanced_SMS_to_Email control plan which will replace the A party number with an email address in the form msisdn@domain - see *Enhanced SMS to Email* in *Messaging Manager User's Guide*.
- not found, the From:address is set to a special short code for unprovisioned email addresses, for example 710. The mobile subscriber would send a reply to 710 which will trigger the SMS_to_Email control plan plan which will replace the A party number with an email address in the form msisdn@domain - see *Direct SMS to Email* in *Messaging Manager User's Guide*.

Chapter 2

Configuration

Overview

Introduction

This chapter explains how to configure the Oracle Communications Convergent Charging Controller application.

In this chapter

This chapter contains the following topics.

eserv.config Configuration

Introduction

The **eserv.config** file is a shared configuration file, from which many Oracle Communications Convergent Charging Controller applications read their configuration. Each Convergent Charging Controller machine (SMS, SLC, and VWS) has its own version of this configuration file, containing configuration relevant to that machine. The **eserv.config** file contains different sections; each application reads the sections of the file that contains data relevant to it.

The **eserv.config** file is located in the **/IN/service_packages/** directory.

The **eserv.config** file format uses hierarchical groupings, and most applications make use of this to divide the options into logical groupings.

Configuration File Format

To organize the configuration data within the **eserv.config** file, some sections are nested within other sections. Configuration details are opened and closed using either { } or [ ].

- Groups of parameters are enclosed with curly brackets – { }
- An array of parameters is enclosed in square brackets – [ ]
- Comments are prefaced with a # at the beginning of the line

To list things within a group or an array, elements must be separated by at least one comma or at least one line break. Any of the following formats can be used, as in this example:

```
{ name="route6", id = 3, prefixes = [ "00000148", "0000473"] }
{ name="route7", id = 4, prefixes = [ "000001049" ] }
```

or

```
{ name="route6"
    id = 3
    prefixes = [
        "00000148"
        "0000473"
    ]
```

```
    }
    { name="route7"
        id = 4
        prefixes = [
            "000001049"
        ]
    }
```

or

```
    { name="route6"
        id = 3
        prefixes = [ "00000148", "0000473" ]
    }
    { name="route7", id = 4
        prefixes = [ "000001049" ]
    }
```

eserv.config Files Delivered

Most applications come with an example **eserv.config** configuration in a file called **eserv.config.example** in the root of the application directory, for example, **/IN/service_packages/eserv.config.example**.

Editing the File

Open the configuration file on your system using a standard text editor. Do not use text editors, such as Microsoft Word, that attach control characters. These can be, for example, Microsoft DOS or Windows line termination characters (for example, ^M), which are not visible to the user, at the end of each row. This causes file errors when the application tries to read the configuration file.

Always keep a backup of your file before making any changes to it. This ensures you have a working copy to which you can return.

Loading eserv.config Changes

If you change the configuration file, you must restart the appropriate parts of the service to enable the new options to take effect.

SEI Configuration

Introduction

Configure the SEI section in the **/IN/service_packages/eserv.config** file to enable the SMS Email Interface (SEI). An example of **eserv.config** file showing all the available configuration options is installed by the CCC package in:

/IN/services_packages/SEI/etc/eserv.config.example

Add the information in **eserv.config.example** to /IN/service_packages/**eserv.config** and update the required parameters. The configuration file is available on SLCs.

Note: All mandatory configuration in the configuration file is done at installation time by the configuration script.

SEI Section

The SEI is configured by the `sei` parameters within the `SEI` section of the **eserv.config** configuration file.

Example SEI Configuration

Here is an example `SEI` section of the **eserv.config** file. This example is the standard SMS to Email Interface setup for the MM PME package.

```
#
# The EMAIL_DOMAIN environment variable needs to be defined in order to run sei with
this config
#

SEI = {
    # Short message service internet Email Interface system

    # some config values may have $values expanded
    #  $HOSTNAME    the unix hostname
    #  $ENV         the value of an environment variable, for example. $HOME
    #  $link.to.var another value in this config file, for example $SEI.sefor
    examplemail.domain
    #               links may refer to other links
    sei = {

        usleep = 10000

        database = {
            user = "mmx_admin"
            password = "mmx_admin"
        }

        email = {

            domain = "$EMAIL_DOMAIN"
            errorMailbox = "error"

            numberRules = [
            ]

            postmasterAction = {
                action = "ignore"
            }

            errorNotificationAction = {
                action = "ignore"
                }

            deliveryStatusNoification = {

                directory = "/IN/service_packages/SEI/tmp/dsn"
                failDirectory = "/IN/service_packages/SEI/tmp/fail"
                retries = ["2 hours", "3 minutes"]
            }

            client = {

                connectTimeout = 300
                initalMessageTimeout = 300
                helloTimeout = 300
                mailTimeout = 300
                recipientTimeout = 300
                dataTimeout = 120
                dataChunkTimeout = 300
                quitTimeout = 300
                origHostname = "$SEI.sei.email.domain"
                port = 25
```

```
        } # client

        server = {

            helloTimeout= 300
            mailTimeout = 300
            recipientTimeout = 200
            dataTimeout = 120
            dataChunkTimeout = 600
            receivedEmail = 300
            quitTimeout = 300
            port = 2500
            greetHostname = "$SEI.sei.email.domain"
            host = "$HOSTNAME"

        } # server

        adapter = {

            config = {
                xmsTrigger = {
                pc = 55
                ssn = 10
                type = "itu"
                }
                xmsTimeout = 5
                tcapTimeout = 6
                xmsWrapper = {
                    interface = "xmsIf"
                    pc = 0
                    ssn = 40
                    type = "itu"
                }
            } # config
        } # adapter

    } # email

    SMS = {

        replyMsisdn= {

            file = "/IN?Service/tmp/sei-reply.addrMap"
            prefix = "642188"
            maxSuffixDigits = 6
        }

        newLine = "CR"

        numberRules = [
        ]

        protocol = "SMPP"


        SMPP = {
            remote = {
                host = "$HOSTNAME"
                port = 3003
            }

            username = "1234"
            password = "PASSWORD"
```

```
                preOpen = true
                version = "5.0"
                maxConcurrentTransactions = 1024
                outgoingTimeout = 10
                idleTimeout = 0
                heartbeatInterval = 10

                adapter = {

                    lib = "mmxiSMPP.so"
                    SSN = 0
                    adapterName = "SMPP1"

                    config = {

                        suppressPathInfoReport = true
                        displayZeroPathReport = false
                        PathReportingInterval = 60

                        smppDefaults = {
                        }

                    } # mmxiSMPP.so config
                } # adapter
            } # SMPP
        } # SMS
    } # sei


} # SEI
```

sei parameters

Here are the parameters for `SEI.sei` section.

`usleep`

Syntax:	`usleep = miSecs`
Description:	How may micro seconds to sleep if there is nothing to do.
Type:	Integer
Optionality:	Mandatory
Allowed:	
Default:	
Notes:	10,000 = 0.01 sec
Example:	`usleep = 10000`

database parameters

The database parameters provide access to the database.

Note: There is no tnsname, so the SEI will use $ORACLE_SID to find the local database.

`password`

Syntax:	`password= "pass"`
Description:	The user's password.
Type:	String

Optionality: Mandatory
Allowed:
Default:
Notes:
Example: `password = "mmx_admin"`

`user`

Syntax: `user = "username"`
Description: The user name.
Type: String
Optionality: Mandatory
Allowed:
Default:
Notes:
Example: `user = "mmx_admin"`

email section parameters

Here is a high level view of the parameters in the `email` section.

```
email = {
    domain = "$EMAIL_DOMAIN"
    errorMailbox = "error"

    numberRules = [
    ]

    postmasterAction = {
        action = "ignore"
    }

    errorNotificationAction = {
        action = "ignore"
    }

    deliveryStatusNoification = {
        deliveryStatusNoification_parameters
    }

    client = {
        clientSection_parameters
    }

    server = {
        serverSection_parameters
    }

    adapter = {
        adapterSection_parameters
    }

}
```

`domain`

Syntax: `domain= "name"`
Description: The domain to use for recipient email addresses.

Type: String
Optionality: Mandatory
Allowed:
Default:
Notes: This value matches the first and second levels of the domain, so if `domain = "example.com"`, then:

```
host.example.com
example.com
```

will be valid, but:

```
badexample.com
```

will not be.

You may use $values.

Examples:

```
domain = "example.com"
domain = "$EMAIL_DOMAIN"
```

errorMailbox

Syntax: `errorMailbox = "domain"`
Description: The mailbox where email relay failures will be delivered to.
Type: String
Optionality: Optional
Allowed:
Default:
Notes: If no domain is given (no @), then the full email address is `errorMailbox@domain`
Example:

```
errorMailbox = "error"
```

errorNotificationAction

Syntax: `errorNotificationAction = {action = "enaction"}`

Description: <*enaction*> is what to do with returned mail notification emails.
Type: String
Optionality:
Allowed: Actions available:

- "ignore" - silently ignore the emails
- "relay" - relay on to another email address
- "save" - save to disk

Default:
Notes:
Example:

```
errorNotificationAction = {
    action = "ignore"
}
```

numberRules

Syntax: `NumberRules= [rule]`
Description: The rules for how to transform the MSISDN in the database into the SMS world and back.
Type: Array

Optionality: Optional
Allowed:
Default:
Notes: Not used for PME
Example:

`postmasterAction`

Syntax: `postmasterAction = {action = "pmaction"}`
Description: What to do with emails directed to the postmaster.
Type: String
Optionality:
Allowed: Actions available:

- "ignore" - silently ignore the emails
- "relay" - relay on to another email address
- "save" - save to disk

Default:
Notes:
Example:

```
postmasterAction = {
    action = "ignore"
}
```

deliveryStatusNoification parameters

Here are the parameters for this section, which handles sending of message delivery failures.

```
deliveryStatusNoification = {

    directory = "/IN/service_packages/SEI/tmp/dsn"
    failDirectory = "/IN/service_packages/SEI/tmp/fail"
    retries = ["2 hours", "3 minutes"]
}
```

`directory`

Syntax: `directory = "path"`
Description: The directory to save emails while sending.
Type: String
Optionality: Mandatory
Allowed:
Default:
Notes:
Example: `directory = "/IN/service_packages/SEI/tmp/dsn"`

`failDirectory`

Syntax: `failDirectory = "path"`
Description: The directory for emails that failed to be sent.
Type: String
Optionality: Mandatory
Allowed:
Default:

Notes:	This is only used if `action = "relay"` and `saveFailed` is true. See `errorNotificationAction` (on page 9) for details.
Example:	`failDirectory = "/IN/service_packages/SEI/tmp/fail"`

`retries`

Syntax:	`retries = [`*`periods`*`]`
Description:	The list of when to retry sending the email after it fails.
Type:	Array of strings
Optionality:	Mandatory
Allowed:	Either periods, for example, "2 hours", "3 minutes" or "3 times *period*" to try 3 times every hour "after *period*" to try that long after the last attempt.
Default:	
Notes:	This is only used if `action = "relay"`. See `errorNotificationAction` (on page 9) for details.
Example:	`retries = ["2 hours", "3 minutes"]`

client section parameters

Here are the parameters.

```
client = {

    connectTimeout = 300
    initalMessageTimeout = 300
    helloTimeout = 300
    mailTimeout = 300
    recipientTimeout = 300
    dataTimeout = 120
    dataChunkTimeout = 300
    quitTimeout = 300
    origHostname = "$SEI.sei.email.domain"
    port = 25

} # client
```

`connectTimeout`

Syntax:	`connectTimeout = `*`seconds`*
Description:	How long to wait for the TCP connection to complete.
Type:	Integer
Optionality:	Mandatory
Allowed:	in seconds
Default:	
Notes:	
Example:	`connectTimeout = 300`

`dataChunkTimeout`

Syntax:	`dataChunkTimeout = `*`seconds`*
Description:	How long to wait for the data chunk response.

Type: Integer
Optionality: Mandatory
Allowed: In seconds
Default:
Notes:
Example: `dataChunkTimeout = 300`

dataTimeout

Syntax: `dataTimeout = seconds`
Description: How long to wait for the data command response.
Type: Integer
Optionality: Mandatory
Allowed: In seconds
Default:
Notes:
Example: `dataTimeout = 120`

helloTimeout

Syntax: `helloTimeout = seconds`
Description: How long to wait for the hello command response.
Type: Integer
Optionality: Mandatory
Allowed: In seconds
Default:
Notes:
Example: `helloTimeout = 300`

initalMessageTimeout

Syntax: `initalMessageTimeout = seconds`
Description: How long to wait for the initial SMTP message.
Type: Integer
Optionality: Mandatory
Allowed: In seconds
Default:
Notes:
Example: `initalMessageTimeout = 300`

mailTimeout

Syntax: `mailTimeout = seconds`
Description: How long to wait for the mail command response.
Type: Integer
Optionality: Mandatory
Allowed: In seconds
Default:
Notes:

Example: `mailTimeout = 300`

origHostname

Syntax: `origHostname = name`
Description: The hostname we give to SMTP servers.
Type: String
Optionality: Optional
Allowed:
Default:
Notes: No host value means listen for incoming SMTP connections on any interface. You may use $values
Example: `origHostname = "$SEI.sei.email.domain"`

port

Syntax: `port = num`
Description: The TCP port to connect to for SMTP.
Type: Integer
Optionality: Mandatory
Allowed:
Default:
Notes: Must be 25 in production - only change for testing
Example: `port = 25`

quitTimeout

Syntax: `quitTimeout = seconds`
Description: How long to wait for the quit command response.
Type: Integer
Optionality: Mandatory
Allowed: In seconds
Default:
Notes:
Example: `quitTimeout = 300`

recipientTimeout

Syntax: `recipientTimeout = seconds`
Description: How long to wait for the recipient command response.
Type: Integer
Optionality: Mandatory
Allowed: In seconds
Default:
Notes:
Example: `recipientTimeout = 300`

server section parameters

The following example configuration shows the parameters in the `sei`, `server` section of the configuration file.

```
server = {

    helloTimeout= 300
    mailTimeout = 300
    recipientTimeout = 200
    dataTimeout = 120
    dataChunkTimeout = 600
    receivedEmail = 300
    quitTimeout = 300
    port = 2500
    greetHostname = "$SEI.sei.email.domain"
    host = "$HOSTNAME"

} # server
```

dataChunkTimeout

Syntax:	`dataChunkTimeout = seconds`
Description:	How long to wait for the data chunks to be completed.
Type:	Integer
Optionality:	Mandatory
Allowed:	In seconds
Default:	
Notes:	
Example:	`dataChunkTimeout = 600`

dataTimeout

Syntax:	`dataTimeout = seconds`
Description:	How long to wait for the data command.
Type:	Integer
Optionality:	Mandatory
Allowed:	In seconds
Default:	
Notes:	
Example:	`dataTimeout = 120`

greetHostname

Syntax:	`greetHostname = name`
Description:	The SMTP initial greeting hostname.
Type:	String
Optionality:	Mandatory
Allowed:	
Default:	
Notes:	You may use $values
Example:	`greetHostname = "$SEI.sei.email.domain"`

helloTimeout

Syntax: `helloTimeout = seconds`
Description: How long to wait for the hello command.
Type: Integer
Optionality: Mandatory
Allowed: In seconds
Default:
Notes:
Example: `helloTimeout = 300`

host - server

Syntax: `host = name or IP address`
Description: The IP address or hostname on which SEI listens for incoming SMTP connections.
Type: String
Optionality: Mandatory
Allowed:
Default:
Notes:
Example: `host = "allan1-mmx30build"`

mailTimeout

Syntax: `mailTimeout = seconds`
Description: How long to wait for the mail command.
Type: Integer
Optionality: Mandatory
Allowed: In seconds
Default:
Notes:
Example: `mailTimeout = 300`

port

Syntax: `port = num`
Description: The port on which SEI listens for SMTP email messages.
Type: Integer
Optionality: Mandatory
Allowed:
Default: 2500
Notes: You should set the `port` parameter to the port on which SMTP traffic for SEI will arrive. The default value for the port serving the Internet is 2500.
Example: `port = 2500`

quitTimeout

Syntax: `quitTimeout = seconds`
Description: How long to wait for the quit command.

Type:	Integer
Optionality:	Mandatory
Allowed:	In seconds
Default:	
Notes:	
Example:	`quitTimeout = 300`

`receivedEmail`

Syntax:	`receivedEmail= seconds`
Description:	How long to wait for SEI to process the email.
Type:	Integer
Optionality:	Mandatory
Allowed:	In seconds
Default:	
Notes:	
Example:	`receivedEmail = 300`

`recipientTimeout`

Syntax:	`recipientTimeout = seconds`
Description:	How long to wait for the recipient command.
Type:	Integer
Optionality:	Mandatory
Allowed:	In seconds
Default:	
Notes:	
Example:	`recipientTimeout = 300`

adapter section parameters

The following example configuration shows the parameters in the `sei`, `adapter` section of the configuration file.

```
adapter = {

    config = {
        xmsTrigger = {
        pc = 55
        ssn = 10
        type = "itu"
        }
        xmsTimeout = 5
        tcapTimeout = 6
        xmsWrapper = {
            interface = "xmsIf"
            pc = 0
            ssn = 40
            type = "itu"
        }
    } # config
} # adapter
```

xmsTrigger

Following are the parameters in xmsTrigger sub-section used to determine the source address for outgoing messages.

`pc`

Syntax: `pc = point code`
Description: The point code.
Type: Integer
Example: `pc = 55`

`xmsTimeout`

Syntax: `xmsTimeout = seconds`
Description: This suggests the the time to wait for a response from XMS-RW.
Type: Integer
Allowed: In seconds
Notes: This entry is used in conjunction with the xmsWrapperTA on SLEE interface.
Example: `xmsTimeout = 5`

`tcapTimeout`

Syntax: `tcapTimeout = seconds`
Description: How long to wait for response from XMS remote wrapper.
Type: Integer
Allowed: In seconds
Default: 10
Notes: This entry is used in conjunction with the xmsWrapperTA SLEE interface.
Example: `tcapTimeout = 10`

`pc`

Syntax: `pc = code`
Description: Destination point code of messages to be handled by this adapter.
Type: Integer
Allowed: Defined by network administrator.
Notes: This would be zero if routing is done using subsystem number.
Example: `pc = 55`

`interface`

Syntax: `interface = "name"`
Description: interface name to establish dialog
Type: String
Optionality: Optional (default used if not set)
Default: xmsIf
Example: `paraMeter = "xmsIf"`

SMS section configuration

The `SMS` section provides the configuration for the interface to SMS functionality.

Here is a high level view of this section.

```
SMS = {

    replyMsisdn= {
        replyMsisdn_section_parameters
    }

    newLine = "CR"

    numberRules = [
    ]

    protocol = "SMPP"


    SMPP = {
        SMPP_section_parameters

        Adapter = {
            Adapter_section_parameters
        }
    }
}
```

SMS section parameters

Here are the parameters in this section.

newLine

Syntax: `newLine = "nl"`

Description: Defines how newlines are represented in text messages

Type: String

Optionality: Mandatory

Allowed: Options are:

- "CR"
- "LF"
- "CRLF"

Default:

Notes:

Example: `newLine = "CR"`

numberRules

Syntax: `NumberRules= [rule]`

Description: The rules for how to transform the MSISDN in the database into the SMS world and back.

Type: Array

Optionality: Optional

Allowed:

Default:

Notes: Not used for PME

Example:

`protocol`

Syntax: `protocol = "name"`
Description: The protocol to use.
Type: String
Optionality: Mandatory
Allowed:
Default:
Notes: This must have a corresponding configuration section.
Example: `protocol = "SMPP"`

replyMsisdn section parameters

The `replyMsisdn` configuration section is used to maintain the mappings from incoming emails to the origination address for SMSs and facilitates replies to these SMSs to be directed back to the original emailer.

Here is the configuration of this section.

```
replyMsisdn= {

    file = "/IN?Service/tmp/sei-reply.addrMap"
    prefix = "642188"
    maxSuffixDigits = 6
}
```

`file`

Syntax: `file = "path"`
Description: Specifies the file to store the mapping in.
Type: String
Optionality: Mandatory
Allowed:
Default:
Notes:
Example: `file = "/tmp/sei-reply.addrMap"`

`maxSuffixDigits`

Syntax: `maxSuffixDigits = num`
Description: The maximum number of digits to append to prefix when all of these are used but old ones will be reused.
Type: Integer
Optionality: Mandatory
Allowed:
Default:
Notes: The maximum number 6 = 1,111,111 numbers, 580MB
Example: `maxSuffixDigits = 6`

`prefix`

Syntax:	`prefix = "prefix"`
Description:	The prefix to use when generating reply SMS addresses.
Type:	Number String
Optionality:	Mandatory
Allowed:	
Default:	
Notes:	
Example:	`prefix = "642188"`

SMPP protocol parameters

The SEI uses the SMPP protocol, allowing an ASP to communicate with the SMSC, or an application, such as Messaging Manager, which has a configured SMPP adapter.

Here is high-level view of this section, showing the `SMPP` configuration required for SEI.

```
SMPP = {
    remote = {
        host = "$HOSTNAME"
        port = 3003
    }

    username = "1234"
    password = "PASSWORD"
    preOpen = true
    version = "5.0"
    maxConcurrentTransactions = 1024
    outgoingTimeout = 10
    idleTimeout = 0
    heartbeatInterval = 10

    adapters = {
        adapter_section_parameters
    }
}
```

`heartbeatInterval`

Syntax:	`heartbeatInterval = hbint`
Description:	How often to send enquire_link messages to check that the connection is up.
Type:	Integer
Optionality:	
Allowed:	
Default:	
Notes:	
Example:	`heartbeatInterval = 10`

`maxConcurrentTransactions`

Syntax:	`maxConcurrentTransactions = num`
Description:	The maximum number of unanswered outstanding messages.
Type:	Integer
Optionality:	
Allowed:	

Default:
Notes:
Example: `maxConcurrentTransactions = 1024`

outgoingTimeout

Syntax: `outgoingTimeout = seconds`
Description: The timeout period before shutting down if quiet for this long.
Type: Integer
Optionality: Mandatory
Allowed:
Default:
Notes:
Example: `outgoingTimeout = 10`

password

Syntax: `password = "passw"`
Description: The password for the user.
Type: String
Optionality: Mandatory
Allowed:
Default:
Notes:
Example: `password = "PASSWORD"`

preOpen

Syntax: `PreOpen = true|false`
Description: Whether or not to open before there are any messages to send.
Type: Boolean
Optionality: Mandatory
Allowed: true, false
Default:
Notes:
Example: `preOpen = true`

username

Syntax: `username = "name"`
Description: The user name.
Type: String
Optionality: Mandatory
Allowed:
Default:
Notes:
Example: `username = "1234"`

`version`

Syntax:	`version = "ver"`
Description:	The version of SMPP to use.
Type:	String
Optionality:	Mandatory
Allowed:	Available versions are: • "3.4" • "5.0"
Default:	
Notes:	
Example:	`version = "5.0"`

remote parameters

The remote section contains the parameters to identify the remote host.

Here is the configuration of the `remote` section.

```
remote = {
    host = "$HOSTNAME"
    port = 3003
}
```

`host`

Syntax:	`host = "host"`
Description:	Identifies the host.
Type:	String
Optionality:	Mandatory
Allowed:	hostname, IP address, or $HOSTNAME
Default:	
Notes:	
Example:	`host = "$HOSTNAME"`

`port`

Syntax:	`port= num`
Description:	The TCP port to connect to.
Type:	Integer
Optionality:	Mandatory
Allowed:	
Default:	
Notes:	
Example:	`port = 3003`

adapter parameters

Here is an example of the `adapter` section.

```
adapter = {

    lib = "mmxiSMPP.so"
```

```
                SSN = 0
                adapterName = "SMPP1"

                config = {

                    suppressPathInfoReport = true
                    displayZeroPathReport = false
                    PathReportingInterval = 60

                    smppDefaults = {
                    }

                } # mmxiSMPP.so config
            } # adapter
```

adapterName

Syntax: `adapterName = "adapter"`

Description: The identifier for the adapter.

Type: String

Optionality: Mandatory

Allowed: Any text string, but should be meaningful, i.e. include the protocol used.
For example "SMPP1" for SMPP

Default: No default.

Note: This name MUST also be in the configuration database before the application will run correctly.

Example: `adapterName = "SMPP1"`

lib

Syntax: `lib = "name"`

Description: The name of the file containing the adapter.

Type: String

Optionality: Mandatory

Allowed:

Default: No default

Notes:

Example: `lib = "mmxiSMPP.so"`

SSN

Syntax: `SSN = num`

Description: Destination subsystem number of messages to be handled by this adapter.

Allowed Valid subsystem number

Notes: Non-zero to handle incoming TCAP.

Example: `SSN = 18`

config

The parameters in this sub-section below this give the configuration for all messages for this adapter.

suppressPathInfoReport

Syntax: `suppressPathInfoReport = true|false`

Description: Whether or not to suppress path connection reports

Type:	boolean
Optionality:	Mandatory
Allowed:	true, false
Default:	false
Notes:	An ASP receives heartbeats from the SMSC when Messaging Manager is configured to operate as an SMSC, then it will respond to these heartbeats. These are logged in the xmsTrigger logfile. This can cause the logfile to fill up unnecessarily.
Example:	`suppressPathInfoReport = true`

smppDefaults section

The smppDefaults section of the **eserv.config** specifies the SMPP values that will be used for connections. Refer to *Messaging Manager Technical Guide* for details.

Chapter 3

Background Processes

Overview

Introduction

This chapter explains the processes that are started automatically by Service Logic Execution Environment (SLEE).

Note: This chapter also includes some plug-ins to background processes which do not run independently.

In this chapter

This chapter contains the following topics.

sei

Purpose

The sei process acts as a gateway between SMS and email.

Startup

You can run sei process as an SLEE interface capable of triggering IN applications such as ACS. The sei is started automatically by the SLEE.

Configure the SLEE at start-up. The default configuration file is **/IN/service_packages/SLEE/etc/SLEE.cfg**.

To run sei as a slee process, add the following line to SLC's SLEE.cfg file.

```
INTERFACE=sei sei.sh /IN/service_packages/SEI/bin EVENT
```

Note: Add a service key entry for this interface. For more information on configuring service key entries, see Service Logic Execution Environment Technical Guide.

Command line parameters

There are no command line parameters for the sei process.

Configuration

The configuration parameters for the sei process are automatically added to the `SEI` section of **eserv.config** at installation. For details, see *SEI Configuration* (on page 4).

Failure

If the sei fails, alarm is raised to the syslog and any incoming inbound emails and SMSs from xmsTrigger is not processed.

Chapter 4

About Installation and Removal

Overview

Introduction

This chapter provides information about the installed components for the Oracle Communications Convergent Charging Controller application described in this guide. It also lists the files installed by the application that you can check for, to ensure that the application installed successfully.

In this Chapter

This chapter contains the following topics.

Installation and Removal Overview

Introduction

For information about the following requirements and tasks, see *Installation Guide*:

- Convergent Charging Controller system requirements
- Pre-installation tasks
- Installing and removing Convergent Charging Controller packages

SEI packages

An installation of SMS Email Interface includes the following package, on the SLC:

- seiScp

Checking the Installation

Introduction

Refer to these check lists to ensure the package has been installed correctly.

SEI directories and files

The SEI installation creates the following directories:

- **/IN/service_packages/SEI/bin**
- **/IN/service_packages/SEI/etc**
- **/IN/service_packages/SEI/lib**
- **/IN/service_packages/SEI/tmp**

The SEI installation installs the following binaries and interfaces:

- **/IN/services_packages/SEI/bin/sei**

The SEI installation installs the following example configuration files:

- **/IN/service_packages/SEI/etc/eserv.config.example**
- **/IN/service_packages/SEI/etc/eserv.config.pme**

Error mailbox

Before the SEI application can be used, a valid address for error messages must be configured. Please update the *errorMailbox* (on page 9) parameter, in the **eserv.config.pme** file in **/IN/service_packages/SEI/etc**, with a valid mailbox then restart SEI through inittab.

Profile scp file

The **.profile-scp** file is created in **/IN/service_packages/SEI** when SEI is installed. Here is an example.

```
ORACLE_SID=SCP
export ORACLE_SID
ORACLE_HOME=/u01/app/oracle/product/10/2/0/db_1
export ORACLE_HOME
ORACLE_BASE=/u01/app/oracle
export ORACLE_BASE
ORACLE_TERM=vt100
export ORACLE_TERM
LD_LIBRARY_PATH=${LD_LIBRARY_PATH:+$LD_LIBRARY_PATH:}/u01/app/oracle/product/10/2/0/
db_1/lib32:/u01/app/oracle/product/10/2/0/db_1/lib:/usr/lib/secure:$ORACLE_HOME/lib3
2:$ORACLE_HOME/lib:/IN/service_packages/SEI/lib
export LD_LIBRARY_PATH
PATH=$PATH:$ORACLE_HOME/bin:/IN/service_packages/SEI/bin
export PATH
EMAIL_DOMAIN=mmx3tstscp11-zone03.oracle.com
ESERV_CONFIG_FILE=/IN/service_packages/SEI/etc/eserv.config
export EMAIL_DOMAIN ESERV_CONFIG_FILE
```