Using the Loopback Connector

V4.0
Document ID: TSGN-11-4.0-01
Date Published: 9.10.03

edocs, Inc., One Apple Hill Drive, Suite 301, Natick, MA 01760

Preface

In This Section

Using this Manual

Welcome to Using the Loopback Connector.

This manual covers installing and using the Loopback Connector.

Before You Get Started

You should be familiar with the following:

- The Synchronizer connector and SmartLink Framework
- Your application architecture
- Programming Java and Java Server pages
- System administration and management

Who Should Read this Manual

This manual is for developers and project managers who need to use this component to mimic backend systems.

- Administrators

  As this connector is especially for developers, you do not need to read this manual.
- Developers

  When developing your Account Management solution, you need to have a way of sending and eivigmsae without connection to other backend systems. The Loopback Connector responds to all of the standard requests just like a backend application would.
- Project Architect

  As this connector is especially for developers, you do not need to read this manual.
- Project Manager

  You will find information about the LoopBack connector and how it can be used to help develop Account Management solutions.

How this Manual is Organized

This manual covers the following:

- **Introducing the Loopback Connector**

  Covers the background information you need to understand and use this connector.

- **Configuring the Loopback Connector**

  Covers system settings and properties you need to change after installing this connector.

- **Managing the Loopback Connector**

  Covers running and shutting down this connector.

What Typographical Changes and Symbols Mean

This manual uses the following conventions:

TYPEFACE	MEANING	EXAMPLE
Italics	Manuals, topics or other important items	Refer to *Developing Connectors*.
Small Capitals	Software and Component names	Your application uses a database called the CID.
Fixed Width	File names, commands, paths, and on screen commands	Go to `//home/my file`

Finding the Information You Need

The product suite comes with comprehensive documentation set that covers all aspects of building Telco Service Manager (TSM) solutions. You should always read the release bulletin for late-breaking information.

Getting Started

If you are new to the edocs Telco Solutions, you should start by reading *Introducing Telco Service Manager*. This manual contains an overview of the various components along with a list of the available features. It introduces various concepts and components you must be familiar with before moving on to more specific documentation. Once you have finished, you can read the manual that covers different aspects of working with the application. At the beginning of each manual, you will find an introductory chapter that covers concepts and tasks.

Designing Your Solution

While reading *Introducing Telco Service Manager*, you should think about how the different components can address your TSM Solution's needs.

You can refer to *Developing Telco Service Manager* for information about extending the object model, application security, and other design issues. The *CID Reference Guide* also gives you the information about how the information in your solution is managed and stored.

Installing Your Telco Service Manager Application

You should start by reading the Release Bulletin. For detailed installation and configuring information, refer to *Installing Telco Service Manager*. This manual covers installing TSM on one or more computers. It also contains the information you need to configure the different components you install. You might also refer to *Developing Telco Service Manager* and *Developing Connectors for Telco Service Manager* as these manuals contain information on customizing applications and working with other software.

Building Telco Service Manager Solutions

If you are designing and programming *Telco Service Manager*, you have several different sources of information. If you are programming the user interface of the solution, you should read *Developing User Interfaces for Telco Service Manager*. You also refer to the BLM Specification for detailed information about programming the user interface. For configuring the various components, you refer to *Installing Telco Service Manager* and sections in other documents that deal with the component to configure.

If you are working with the business logic of your solution, you should read *Developing Telco Service Manager*. You can also refer to the *BLM Reference Guide* for more information about the design and structure of the BLM object model. For information about how this information is stored, you should refer to the *CID Reference Guide* along with the CID Reference documentation for your database. In order to develop your application, you most likely will need to install and run the Loopback Connector. This component mimics back-end applications for development purposes. For information about installing and running this component, refer to *Using the Loopback Connector with Telco Service Manager*.

Integrating Telco Service Manager Solutions

If you are involved in configuring your solution to work with Operation Support Software (OSS), you should read *Developing Connectors with Telco Service Manager*. This manual helps you understand the integration architecture and shows you how to build connectors to connect to today's market-leading OSS software. You can also read *Using the Loopback Connector with Telco Service Manager* for information about a connector built for development purposes. Other manuals you can refer to for information about configuring your application include *Introducing Telco Service Manager* and *Developing Telco Service Manager*.

Managing Telco Service Manager

If you are responsible for managing TSM, you should read the *Installing Telco Service Manager* for information about configuring various components and information about working with different application servers. *Administrating Telco Service Manager* covers what you need to know about managing your solution at runtime. For information about OSS systems, you should read *Developing Connectors with Telco Service Manager*.

If You Need Help

Technical support is available to customers who have valid maintenance and support contracts with edocs. Technical support engineers can help you install, configure, and maintain your edocs application.

To reach the U.S. Service Center, located in Natick, MA (Monday through Friday 8:00am to 8:00pm EST):

- Telephone: 508.652.8400
- Toll Free: 877.336.3362
- E-support: support.edocs.com (This requires a one-time online registration)
- E-mail: support@edocs.com

When you report a problem, please be prepared to provide us the following information:

- What is your name and role in your organization?
- What is your company's name?
- What is your phone number and best times to call you?
- What is your e-mail address?
- In which edocs product did a problem occur?
- What is your Operating System version?
- What were you doing when the problem occurred?
- How did the system respond to the error?
- If the system generated a screen message, please send us that screen message.
- If the system wrote information to a log file, please send us that log file.

If the system crashed or hung, please tell us.

Contents

CHAPTER 1

Introducing the Loopback Connector

In This Section

About the Loopback Connector

When developing and testing applications, you might not have access to an OSS that can process your messages. The Loopback Connector mimics an OSS and responds to the standard messages it receives from an application.

Before you start using the Loopback Connector, you need to make sure that you have the required software needed to run TSM and the connector. You also need to know some specific system settings to get started.

The Loopback Connector only replies to standard messages. Therefore it does not have built-in support for customized messages nor does the Loopback Connector have exception queue administration scripts.

The Loopback Connector should never be used in production environments.

Overview of the Loopback Connector Architecture

Overview of the SmartLink Connectors

The SmartLink Framework is not only a set of application integration services, but is a framework in which the providers of the services work together.

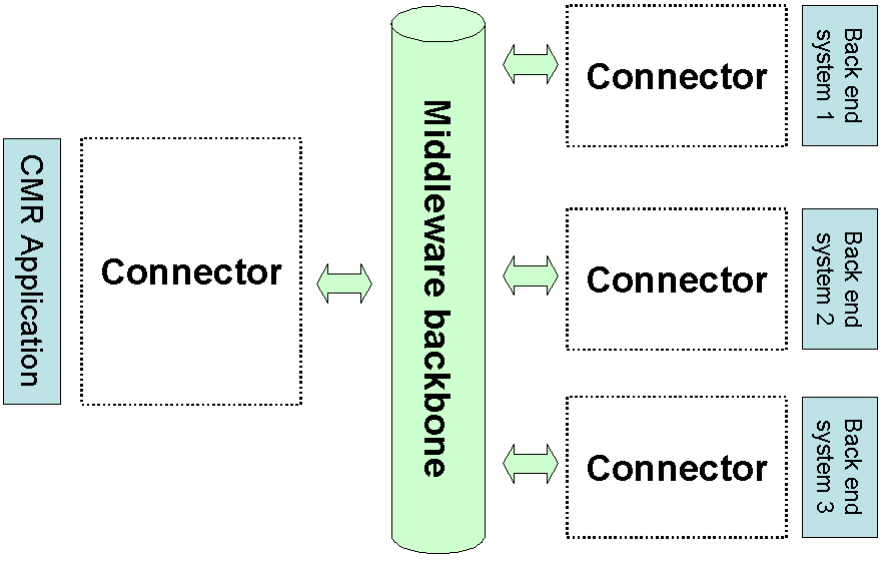
In order to integrate an application, these service providers, or SmartLink Framework components, are arranged into dataflows. This means that they are connected together and organized into a step-by-step sequence. A connector is part of the SmartLink Framework that is responsible for moving data between applications and transport layers.

There are two types of connectors:

- Synchronizers

  They are responsible for data exchanged between TSM and the transport layer
- OSS Connectors

  They are responsible for data exchanged between the BSS/OSS applications and the transport layer



For example, the Synchronizer connector is responsible for moving data between the Account Management Application and the transport layer. This connector uses different connector components to carry out the following tasks:

- **Send data to the middleware**
  1. Extract the requests from the request queue located in the CID
  2. Use these requests to generate XML messages
  3. Send this XML message to the middleware backbone
- **Receive data from the middleware**
  1. Receive the XML message from the middleware
  2. Use these messages to create CID objects
  3. Update the CID by running scripts which use the CID objects

The Loopback Connector

The loopback connector is an OSS connector that responds to the messages sent by the Synchronizer connector.

To do this, the Loopback Connector:

1 Recieves standard messages

2 Reads the information in the messages

3 Builds reply messages using this information

4 Sends the reply messages

Managing Message Queues

To efficiently manage the inbound and outbound messages, the Loopback Connector comes with a set of queues that help manage errors or transport problems.

If the connector cannot send a message, it:

1 Waits the specified interval then tries to send it again

2 If the message cannot be sent after the specified number of tries:
  1. Puts the message in a message retry queue
  2. Logs an error

Message queues can be divided into the following categories:

- Inbound message queues
- Outbound message queues
- Retry exception message queues

About Inbound Message Queues

The connector manages the following inbound message queues:

- Inbound queue

  This queue receives inbound messages.

- Inbound retry queue

  This queue holds messages that could not be sent to the adaptor after the specified number of tries and requires user intervention.

About Outbound Message Queues

The connector manages the following outbound message queues:

- Outbound queue

  This queue holds outbound messages.

- Outbound transport error queue

  This queue holds messages that could not be sent due to middleware problems. This queue is a file queue you can manage with the connector administration tools.

- Outbound retry queue

  This queue holds messages that could not be sent after the specified number of tries and requires user intervention. This queue is a file queue you can manage with the connector administration tools.

About Exception Message Queues

The connector manages the following exception message queues:

- Inbound OSS exception queue

  This queue stores OSS exception messages. OSS exceptions include exceptions generated by the OSS or by the adaptor itself.

- Inbound SmartLink Framework exception queue

  This queue stores OSS SmartLink Framework exception messages that occur when processing or generating inbound messages.

- Outbound SmartLink Framework exception queue

  This queue stores OSS SmartLink Framework exception messages that occur when processing or generating outbound messages.

Outbound and Inbound Message Types

The following table presents the available outbound message types and their corresponding inbound message types:

OUTBOUND MESSAGE TYPE	INBOUND MESSAGE TYPE
REQADDBILLINGACCOUNT	DOADDBILLINGACCOUNT
REQADDCONTRACT	DOADDCONTRACT
REQADDLEVEL	DOADDLEVEL
REQADDLOGIN	DOADDLOGIN
REQADDMEMBER	DOADDMEMBER
REQADDORG	DOADDORG
REQADDSERVICE	DOADDSERVICE
REQASSOCIATEDEDICATEDOFFER	DOASSOCIATEDEDICATEDOFFER
REQADDTROUBLETICKET	DOADDTROUBLETICKET
REQDECLAREPAYMENTRESP	DODECLAREPAYMENTRESP
REQDISSASSOCIATEDEDICATEDOFFER	REQDISSASSOCIATEDEDICATEDOFFER
REQLOSTDECLARATION	DOMODIFYCONTRACTSTATUS
REQMODIFYBILLINGCONTACT	DOMODIFYCONTACT
REQMODIFYBILLINGACCOUNT	DOMODIFYBILLINGACCOUNT
REQMODIFYCONTRACT	DOMODIFYCONTRACT
REQMODIFYCONTRACTSTATUS	DOMODIFYCONTRACTSTATUS
REQMIGRATECONTRACT	DOMIGRATECONTRACT
REQMODIFYCONTRACTLINE	DOMODIFYCONTRACTLINE
REQMODIFYCONTRACTOWNER	DOMODIFYCONTRACTOWNER
REQMODIFYLANGUAGE	DOMODIFYLANGUAGE
REQMODIFYLEGALCONTACT	DOMODIFYCONTACT
REQMODIFYLEVEL	DOMODIFYLEVEL
REQMODIFYMEMBER	DOMODIFYMEMBER
REQMODIFYORG	DOMODIFYORG
REQMODIFYPAYMENTINFO	DOMODIFYPAYMENTINFO
REQMODIFYRATEPLAN	DOMODIFYRATEPLAN
REQMODIFYSERVICE	DOMODIFYSERVICE
REQMODIFYTROUBLETICKET	DOMODIFYTROUBLETICKET
REQORDER	ACK for the REQORDER message For each REQ action contained in this message, the corresponding DO message is generated.
REQORDERDOC	ACK

OUTBOUND MESSAGE TYPE	INBOUND MESSAGE TYPE
REQOSSAPPROVAL	DOOSSAPPROVAL
REQRECHARGEPREPAID	ACK
REQREMOVESERVICE	DOREMOVESERVICE
REQREPLACESERVICE	DOREPLACESERVICE
REQSETPERSONALINFO	DOSETPERSONALINFO

CHAPTER 2

Configuring the Loopback Connector

In This Section

Configuring the Loopback Connector

When installed, the Loopback Connector is ready to run.

You can modify some of the default settings to reflect TSM and system architecture.

Configuring the connector involves:

- Specifying the retry settings
- Specifying the message queue settings

If you change the connector settings, you must stop and restart the connector.

Configuring the Connector

If an unforeseen error occurs during the transport of a message, the connector waits a specified amount of time and tries to send the message again. The connector continues to try and send the message the specified number of times before logging an error and placing the message in a message retry queue.

Configuring the connector involves:

- Specifying the length of time the connector waits before trying to send again
- Specifying the number of times the connector tries to send the message before logging an error

To specify the number of time between tries

1. Go to `<home_dir>/config/connectors/loopback/macros`.
2. Open `constants.properties`.
3. Modify the following:
   - For messages between the connector and the middleware, set `TRANSPORT_RETRY_DELAY` to the number of milliseconds to wait
   - For messages between the connector and the adaptor, set `ADAPTOR_RETRY_DELAY` to the number of milliseconds to wait
4. Save your changes.

To specify the number of tries

1. Go to `<home_dir>/config/connectors/loopback/macros`.
2. Open `constants.properties`.
3. Modify the following:
   - For messages between the connector and the middleware, set `MAX_TRANSPORT_RETRIES` to the number of times to try to send the message.
   - For messages between the connector and the adapter, set `MAX_ADAPTOR_RETRIES` to the number of times the connector tries to send the message.
4. Save your changes.

Configuring Message Queues

You can manage the messages retry queues that store messages that could not be sent. You can also manage exception queues where the connector places exception messages when an internal error occurs.

These message queues are file queues. This means that the connector stores the messages and all of the internal information as a file in a specified directory.

Configuring the message queues involves:

- Specifying the directory of each message queue

For more information on managing message queues and the logger, refer to *Developing Connectors*.

To specify the directory of a message queue

1. Go to `<home_dir>/config/connectors/loopback`.
2. Open `loopback.ilcr`.
3. Find the section corresponding to the queue to modify.
4. Under the message queue, modify the path to the queue. Use the syntax:

   `loopback.QUEUE_NAME="path"`

   where `QUEUE_NAME` corresponds to the `parametername` of the queue.
5. Save your changes.

Example of a queue declaration in the `loopback.ilcr`

```
#* @parametername           :SYNC2OSS
#* @connectorname           :loopback
#* @parameterdescription    :Directory to use for the sync2oss queue
#* @typedescription         :string
#* @multiplicitydescription :false
#* @valuedescription        :
loopback.SYNC2OSS="c:/connector_home/var/data/sync2oss"
```

For more information about modifying the other message queue parameters, refer to *Developing Connectors*.

CHAPTER 3

Managing the Loopback Connector

In This Section

About Managing the Loopback Connector

Managing the Loopback Connector involves:

- Starting the connector
- Stoping the connector

Starting and Stopping the Loopback Connector

After installing and configuring the Loopback Connector, you are ready to run the connector.

The connector comes with a set of administration tools to easily start and stop the connector. This set of tools includes:

- `ossstart`
- `ossadm`

To start the Loopback Connector

1. Go to `<home_dir>/bin`.
2. Run `ossstart loopback`

   The connector initializes all of the required components and is ready to process messages.

To stop the Loopback Connector

1. Go to `<home_dir>/bin`.
2. Run `ossadm <host> <port> shutdown`

   The connector ends all processing of incoming and outgoing messages and closes all inbound and outbound queues.

To kill the Loopback Connector

1. Go to `<home_dir>/bin`.
2. Run `ossadm <host> <port> kill`

   The connector process is forced to exit without properly closing used resources.

Only use the kill command when the connector has stopped responding.

Index