Framework 7.5

Network SIP Server

Deployment Guide

About Genesys

Genesys Telecommunications Laboratories, Inc., a subsidiary of Alcatel-Lucent, is 100% focused on software for call centers. Genesys recognizes that better interactions drive better business and build company reputations. Customer service solutions from Genesys deliver on this promise for Global 2000 enterprises, government organizations, and telecommunications service providers across 80 countries, directing more than 100 million customer interactions every day. Sophisticated routing and reporting across voice, e-mail, and Web channels ensure that customers are quickly connected to the best available resource—the first time. Genesys offers solutions for customer service, help desks, order desks, collections, outbound telesales and service, and workforce management. Visit www.genesyslab.com for more information.

Each product has its own documentation for online viewing at the Genesys Technical Support website or on the Documentation Library DVD, which is available from Genesys upon request. For more information, contact your sales representative.

Notice

Your Responsibility for Your System's Security

You are responsible for the security of your system. Product administration to prevent unauthorized use is your responsibility. Your system administrator should read all documents provided with this product to fully understand the features available that reduce your risk of incurring charges for unlicensed use of Genesys products.

Trademarks

Technical Support from VARs

If you have purchased support from a value-added reseller (VAR), please contact the VAR for technical support.

Technical Support from Genesys

If you have purchased support directly from Genesys, please contact Genesys Technical Support at the following regional numbers:

Region	Telephone	E-Mail
North and Latin America	+888-369-5555 or +506-674-6767	support@genesyslab.com
Europe, Middle East, and Africa	+44-(0)-118-974-7002	support@genesyslab.co.uk
Asia Pacific	+61-7-3368-6868	support@genesyslab.com.au
Japan	+81-3-5649-6871	support@genesyslab.co.jp

Prior to contacting technical support, please refer to the *Genesys Technical Support Guide* for complete contact information and procedures.

Ordering and Licensing Information

Complete information on ordering and licensing Genesys products can be found in the ***Genesys 7 Licensing Guide*.**

Released by

Genesys Telecommunications Laboratories, Inc. www.genesyslab.com

Document Version: 75fr_dep-nsip_03-2007_v.7.5.001.02

Table of Contents

Preface

Welcome to the *Framework 7.5 Network SIP Server Deployment Guide*. This document introduces you to the concepts, terminology, and procedures relevant to T-Servers® in general and provides detailed reference information about Network SIP Server. The reference information includes, but is not limited to, configuration options, limitations, and switch-specific functionality. You must configure the configuration objects and options described in this document in the Framework Configuration Layer.

Use this document only after you have read through the *Framework 7.5 Deployment Guide*, and the Release Note for your T-Server.

This document is valid only for the 7.5 release of this product.

Note: For releases of this document created for other releases of this product, please visit the Genesys Technical Support website, or request the Documentation Library CD, which you can order by e-mail from Genesys Order Management at `orderman@genesyslab.com`.

This preface provides an overview of this document, identifies the primary audience, introduces document conventions, and lists related reference information:

T-Server is the Genesys software component that provides an interface between your telephony hardware and the rest of the Genesys software components in your enterprise. It translates and keeps track of events and requests that come from, and are sent to, the CTI (computer-telephony integration) link in the telephony device. T-Server is a TCP/IP-based server that can also act as a messaging interface between T-Server clients. It is the critical point in allowing your Genesys solution to facilitate and track the contacts that flow through your enterprise.

Intended Audience

This guide is intended primarily for system administrators, both those who are new to T-Server and those who are familiar with it.

- If you are new to T-Server, read the *Framework 7.5 Deployment Guide* and the Release Note mentioned earlier, and then read all of the sections of this document that apply to your software and its accompanying components. Refer back to the *Framework 7.5 Deployment Guide* as needed.
- If you are an experienced T-Server user—someone with computer expertise, who is used to installing, configuring, testing, or maintaining Genesys software—you may find it more time efficient to go to the Index to see what is new or different in T-Server release 7.5. If you take that approach, please also read Release Notes and refer to other related resources, such as the *Genesys 7 Events and Models Reference Manual*.

In general, this document assumes that you have a basic understanding of, and familiarity with:

- Computer-telephony integration concepts, processes, terminology, and applications.
- Network design and operation.
- Your own network configurations.
- Your telephony hardware and software.
- Genesys Framework architecture and functions.
- Configuration Manager interface and object-managing operations.

Based on your specific contact center environment and your responsibilities in it, you may need to be familiar with a much wider range of issues as you deploy T-Server.

Reading Prerequisites

You must read the *Framework 7.5 Deployment Guide* before using this *T-Server Deployment Guide*. That book contains information about the Genesys software you must deploy before deploying T-Server.

Chapter Summaries

This *T-Server Deployment Guide* encompasses all information, including conceptual, procedural, and reference information, about Genesys T-Servers in general, and Network SIP Server in particular. Depending on the subject addressed in a particular section, the document style may move from narration, to instructions, to technical reference.

To distinguish between general T-Server sections and those chapters intended for your particular T-Server, this document is divided into two main parts

Part One—Common Functions and Procedures

Part One of this T-Server document, "Common Functions and Procedures," consists of Chapters 1 through 4. These chapters contain architectural, functional, and procedural information common to all T-Servers:

- Chapter 1, "T-Server Fundamentals," on page 17, describes T-Server, its place in the Framework 7 architecture, T-Server redundancy, and multi-site issues. It does not, however, provide configuration and installation information.
- Chapter 2, "T-Server Configuration and Installation," on page 33, presents Configuration and Installation procedures for all T-Servers.
- Chapter 3, "Multi-Site Support," on page 47, describes the variations available for T-Server implementations across geographical locations.
- Chapter 4, "Start and Stop T-Server Components," on page 97, describes how, and in what order, to start up T-Server among other Framework components. It also provides possible stopping commands.

Although you certainly would refer to these chapters if you have never before configured or installed T-Server, you might also use them, even if you are already familiar with T-Server, to discover any changes to functionality, configuration, and installation since you last deployed this component.

Genesys recommends that you use wizards to deploy T-Server. If you do, first read Chapter 1 to familiarize yourself with T-Server, and then proceed with the deployment process using Framework wizards.

Part Two—Reference Information

Part Two of this T-Server document, Reference Information consists of Chapters 5 through 10. These chapters contain reference information specific to Network SIP Server. However, they also contain information on all T-Server options, both those specific to your T-Server and those common to all T-Servers.

- Chapter 5, "Network SIP Server Specific Configuration," on page 109, presents switch-specific reference information for configuring Network T-Server for SIP
- Chapter 6, "Supported Functionality in Network SIP Server," on page 111 describes the features that are supported by this T-Server, including T-Library functionality, and error messages.
- Chapter 7, "HA Configuration for Network SIP Server," on page 127 describes how to implement a high-availability (HA) configuration for Network SIP Server.

- Chapter 8, “Common Log Options,” on page 135, describes log configuration options common to all Genesys server applications.
- Chapter 9, “T-Server Common Configuration Options,” on page 149, describes configuration options common to all T-Server types including options for multi-site configuration.
- Chapter 10, “Configuration Options in Network SIP Server,” on page 173, describes configuration options specific to this T-Server including the link-related options—those that address the interface between T-Server and the switch.

Document Conventions

This document uses certain stylistic and typographical conventions—introduced here—that serve as shorthands for particular kinds of information.

Document Version Number

A version number appears at the bottom of the inside front cover of this document. Version numbers change as new information is added to this document. Here is a sample version number:

75fr_nsip_03-2007_v7.5.001.00

You will need this number when you are talking with Genesys Technical Support about this product.

Type Styles

Italic

In this document, italic is used for emphasis, for documents’ titles, for definitions of (or first references to) unfamiliar terms, and for mathematical variables.

Examples:

- Please consult the *Genesys 7 Migration Guide* for more information.
- *A customary and usual practice* is one that is widely accepted and used within a particular industry or profession.
- Do *not* use this value for this option.
- The formula, $x + 1 = 7$ where x stands for . . .

Monospace Font

A monospace font, which looks like `teletype or typewriter text`, is used for all programming identifiers and GUI elements.

This convention includes the *names* of directories, files, folders, configuration objects, paths, scripts, dialog boxes, options, fields, text and list boxes, operational modes, all buttons (including radio buttons), check boxes, commands, tabs, CTI events, and error messages; the values of options; logical arguments and command syntax; and code samples.

Examples:

- Select the `Show variables on screen` check box.
- Click the `Summation` button.
- In the `Properties` dialog box, enter the value for the host server in your environment.
- In the `Operand` text box, enter your formula.
- Click `OK` to exit the `Properties` dialog box.
- The following table presents the complete set of error messages T-Server distributes in `EventError` events.
- If you select `true` for the `inbound-bsns-calls` option, all established inbound calls on a local agent are considered business calls.

Monospace is also used for any text that users must manually enter during a configuration or installation procedure, or on a command line:

Example:

- Enter `exit` on the command line.

Screen Captures Used in This Document

Screen captures from the product GUI (graphical user interface), as used in this document, may sometimes contain a minor spelling, capitalization, or grammatical error. The text accompanying and explaining the screen captures corrects such errors *except* when such a correction would prevent you from installing, configuring, or successfully using the product. For example, if the name of an option contains a usage error, the name would be presented exactly as it appears in the product GUI; the error would not be corrected in any accompanying text.

Square Brackets

Square brackets indicate that a particular parameter or value is optional within a logical argument, a command, or some programming syntax. That is, the parameter's or value's presence is not required to resolve the argument, command, or block of code. The user decides whether to include this optional information. Here is a sample:

```
smcp_server -host [/flags]
```

Angle Brackets

Angle brackets indicate a placeholder for a value that the user must specify. This might be a DN or port number specific to your enterprise. Here is a sample:

```
smcp_server -host <confighost>
```

Related Resources

Consult these additional resources as necessary:

- The *Framework 7.5 Deployment Guide*, which will help you configure, install, start, and stop Framework components.
- The *Framework 7.5 Configuration Options Reference Manual*, which will provide you with descriptions of configuration options for other Framework components.
- The *Framework 7.5 Configuration Manager Help*, which will help you use Configuration Manager.
- The *Genesys 7 Migration Guide*, also on the Genesys Documentation Library CD, which contains a documented migration strategy from Genesys product releases 5.x and later to all Genesys 7.x releases. Contact Genesys Technical Support for additional information.
- The *Genesys 7 Events and Models Reference Manual*, which contains the T-Library API, information on `TEvents`, and an extensive collection of call models.
- The *Genesys Technical Publications Glossary*, which ships on the Genesys Documentation Library CD and which provides a comprehensive list of the Genesys and CTI terminology and acronyms used in this document.
- The Release Notes and Product Advisories for this product, which are available on the Genesys Technical Support website at `http://genesyslab.com/support`.

Information on supported hardware and third-party software is available on the Genesys Technical Support website in the following documents:

- *Genesys 7 Supported Operating Systems and Databases*
- *Genesys 7 Supported Media Interfaces*

Genesys product documentation is available on the:

- Genesys Technical Support website at `http://genesyslab.com/support`.
- Genesys Documentation Library CD, which you can order by e-mail from Genesys Order Management at `orderman@genesyslab.com`.

Making Comments on This Document

If you especially like or dislike anything about this document, please feel free to e-mail your comments to `Techpubs.webadmin@genesyslab.com`.

You can comment on what you regard as specific errors or omissions, and on the accuracy, organization, subject matter, or completeness of this document. Please limit your comments to the information in this document only and to the way in which the information is presented. Speak to Genesys Technical Support if you have suggestions about the product itself.

When you send us comments, you grant Genesys a nonexclusive right to use or distribute your comments in any way it believes appropriate, without incurring any obligation to you.

Part

1

Part One: Common Functions and Procedures

Part One of this *Network T-Server Deployment Guide* familiarizes the reader with T-Server in general. It addresses architectural, functional, and procedural information common to all T-Servers.

The information in Part One is divided into the following chapters:

- Chapter 1, "T-Server Fundamentals," on page 17, describes T-Server, its place in the Framework 7 architecture, T-Server redundancy, and multi-site issues. It stops short of providing configuration and installation information.
- Chapter 2, "T-Server Configuration and Installation," on page 33, presents configuration and installation procedures for all T-Servers.
- Chapter 3, "Multi-Site Support," on page 47, details the variations available for T-Server implementations across geographical locations.
- Chapter 4, "Start and Stop T-Server Components," on page 97, describes how, and in what order, to start up T-Server among other Framework components. It also provides possible stopping commands.

New for All T-Servers in 7.5

Before looking at T-Server's place in Genesys solutions and in the architecture of the Genesys Framework, note the following general changes that have been implemented in the 7.5 release of T-Server:

- **Transport Layer Security (TLS) support.** T-Server can now be configured for secure data exchange with the other Genesys components that support this functionality. Refer to the *Genesys 7.5 Transport Layer Security Deployment Guide* for details.

  Warning! The Genesys TLS is not supported on all operating systems that T-Server itself supports. For information about the supported operating systems, see the *Genesys 7.5 Transport Layer Security Deployment Guide.*

- **Optimization of User Data distribution.** This release of T-Server supports an optimized distribution of user data where user data is communicated in a few selected events, as opposed to all DN events.
- **Enhanced ISCC routing support for T-Servers using load balancing.** This support applies to T-Servers that use the `dnis-pool` transaction type for call routing in a multi-site environment. See "dnis-pool" on page 58 for details.
- **Discontinued support for KPL.** Starting with release 7.5, the Keep-Alive Protocol (KPL) backward compatibility is no longer supported. If you have used the KPL with previous versions of Genesys, consider using ADDP after upgrading to 7.5. It provides the same functionality as KPL with fewer limitations. For more information on ADDP, see "Advanced Disconnect Detection Protocol" on page 24.

Note: For information about the new features that are available in your T-Server in the initial 7.5 release, see Part Two of this document.

Chapter

1

T-Server Fundamentals

This chapter provides general information about T-Server features and functionality and about its configuration and installation. For reference information about your specific T-Server and about options for all T-Servers, see “Part Two: Reference Information.”

This chapter has various levels of information, some of it intended for people who have configured, installed, and used previous releases of T-Server, and some of it aimed at those less familiar with such T-Server operations. That means some sections will not necessarily be relevant for you.

- If you are an experienced user of T-Server, start with “New for All T-Servers in 7.5” on page 16, and then move to the chapters comprising Part Two of this document, where specific information about your T-Server is available.
- If you are new to T-Server, begin with “Learning About T-Server.” Once you have read through that and subsequent sections, you are ready for the other chapters in Part One that go into detail about T-Server configuration and installation.

Generally, this chapter presents overview information that applies to all T-Servers (and Network T-Servers) and their deployment. This chapter is divided into the following sections:

Learning About T-Server

The *Framework 7.5 Deployment Guide* provides you with a high-level introduction to the role that T-Server plays in the Genesys Framework. If you have already looked through that guide, you may recall that T-Server is the most important component of the Framework Media Layer (the other two components are Load Distribution Server (LDS) and HA Proxy). The Media Layer enables Genesys solutions to communicate with various media, including traditional telephony systems, voice over IP (VoIP), e-mail, and the Web. This layer also provides the mechanism for distributing interaction-related business data, also referred to as *attached data*, within and across solutions.

Framework and Media Layer Architecture

Figure 1 illustrates the position Framework holds in a Genesys solution.

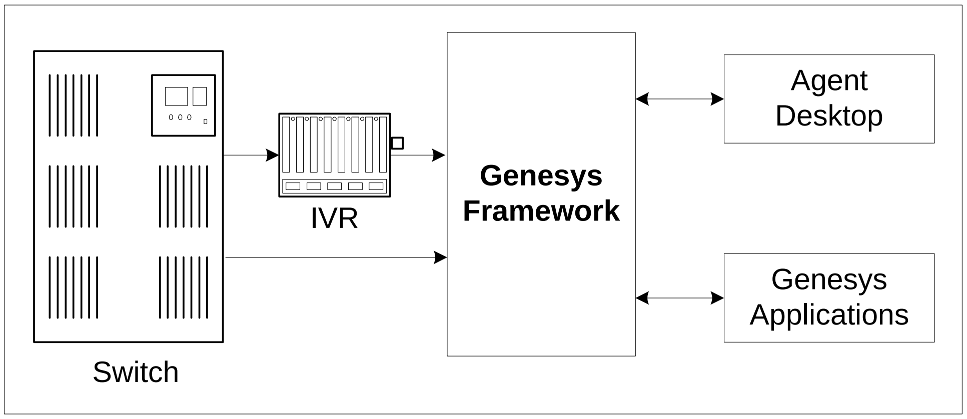


Figure 1: Framework in a Genesys Solution

Moving a bit deeper, Figure 2 presents the various layers of the Framework architecture.

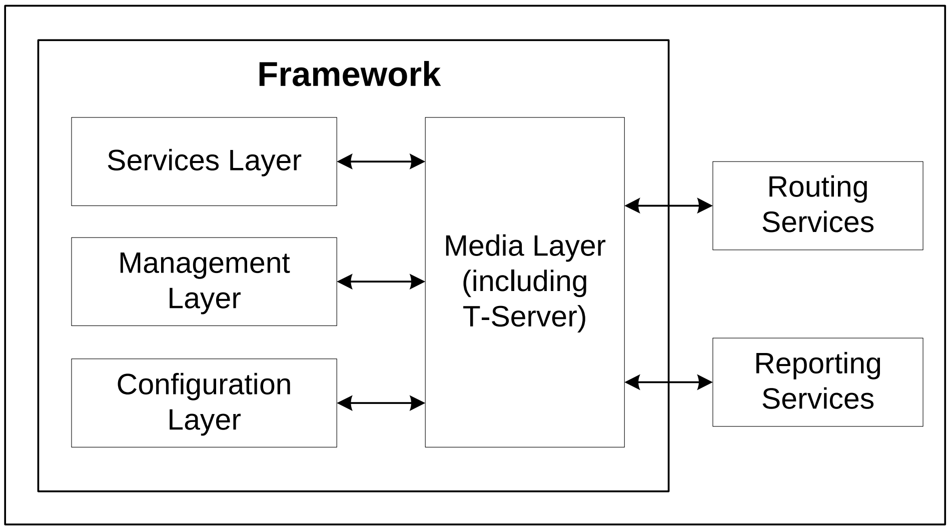


Figure 2: The Media Layer in the Framework Architecture

T-Server is the heart of the Media Layer—translating the information of the media-device realm into information that Genesys solutions can use. It enables your contact center to handle the computer-based form of the interactions that arrive and it translates the information surrounding a customer contact into reportable and actionable data.

Figure 3 presents the generalized architecture of the Media Layer.

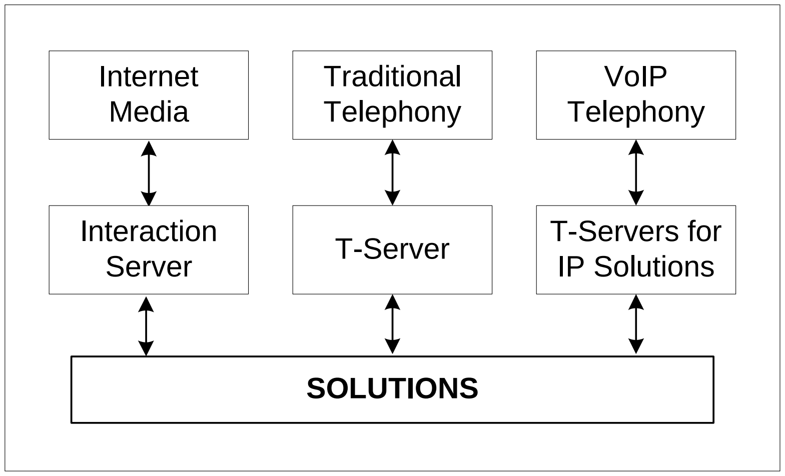


Figure 3: Media Layer Architecture

In addition to being the most important component of the Media Layer, T-Server plays the most significant role in making information about telephony traffic and its data available to Framework as a whole.

One or more components in practically every solution are T-Server clients. Solutions comprise a number of different Genesys software packages, from collections of components for various types of routing to those that allow for outbound dialing to still others. Framework in general, and T-Server in particular, enable these solutions to function in your enterprise.

T-Server has several typical clients: Stat Server, Call Concentrator, Universal Routing Server, and agent desktop applications. T-Server gets the information it needs about the enterprise from Configuration Server. Additionally, if you use the Management Layer, T-Server provides its ongoing status and various other log messages to server components of the Management Layer (for instance, allowing you to set alarms).

T-Server Requests and Events

This section outlines the roles that T-Server plays in a contact center. While it is possible to describe roles for all T-Servers, at a detailed level, T-Server’s functionality depends on the hardware to which it is connected. (For example, when connected to a traditional switch, it performs CTI functions, but when connected to a VOIP-based telephony device, it controls IP traffic.) The CTI connection is only for the switch.

Details of T-Server Functionality

T-Server is a TCP/IP server that enables intelligent communication between media-specific protocols (such as the various CTI protocols, including CSTA and ASAI) and TCP/IP-based clients of T-Server. Applications that are clients to T-Server use the T-Library format to transmit requests to T-Server through a TCP/IP socket. T-Server can then either translate those requests to CTI protocol for switch use or relay them directly to other TCP/IP clients.

T-Server performs three general functions in the contact center: Bridging, Messaging, and Interaction Tracking.

Bridging

T-Server acts as a platform-independent interface between media devices and business applications. In the case of a telephony device, for instance, it receives messages from and sends commands to the telephony equipment using either CTI links provided by the switch manufacturer or interface protocols provided by telephony network vendors.

On the client-application end, T-Server offers three models (call model, agent model, and device model) unified for all switches. The core functionality (such as processing an inbound call, an agent login, or a call-forwarding request) translates into a unified application programming interface (API) called T-Library, so that applications do not need to know what specific switch model they are dealing with. On the other hand, T-Library accommodates many functions that are unique to a specific switch, so that client applications are able to derive the maximum functionality offered by a particular switch.

Refer to the *Genesys 7 Events and Models Reference Manual* for complete information on all T-Server events and call models and to the `TServer.Requests` portion of the *Voice Platform SDK 7.5 .NET* (or *Java*) *API Reference* for technical details of T-Library functions.

Messaging

In addition to translating requests and events for the client application involved in an interaction, T-Server:

- Provides a subscription mechanism that applications can use to receive notifications about interaction-related and non-interaction-related events within the contact center.
- Broadcasts messages of major importance (such as a notification that the link is down) to all clients.
- Broadcasts messages originated by a T-Server client to other T-Server clients.

The subscription mechanism consists of two parts, the DN subscription and event-type masking. Applications must register for a DN or a set of DNs to receive notifications about all events that occur in association with each registered DN. For example, when two softphone applications are registered for the same DN, and the first application initiates a call from the DN, T-Server notifies both applications that the call is initiated from the DN.

Client applications can also specify one or more types of events, and T-Server will filter out events of the non-specified types and only send events of the requested types. For example, if agent supervisors are interested in receiving agent-related events, such as `AgentLogin` and `AgentLogout`, they have to mask `EventAgentLogin` and `EventAgentLogout`, provided that a particular T-Server supports these events.

The combination of each client’s subscription for DNs and masking of event types defines what messages T-Server distributes to what client.

Interaction Tracking

T-Server maintains call information for the life of the call (or other T-Server-supported media type) and enables client applications to attach user data to the call. Call information includes:

- A unique identifier, connection ID, that T-Server assigns when creating the call.
- Automatic Number Identification (`ANI`) and Dialed Number Identification Service (`DNIS`), if reported by the CTI link.
- User data that a client application (such as an Interactive Voice Response unit or Genesys Universal Routing Server) provides.

Difference and Likeness Across T-Servers

Although Figure 3 on page 19 (and other figures) depicts T-Server that works with telephony systems as a single product, this is a simplification. Because almost every traditional telephony device has its own characteristics and communication protocols, Genesys makes different T-Servers for different telephony systems. (That means T-Server you have will not work with another switch.) Thus, all T-Servers play a common role in the architecture, but their specific features differ from implementation to implementation, based on the media device in use.

Despite their switch-based differences, T-Servers for telephony systems are similar to one another in at least one important respect: they are all built with a certain amount of shared software code. This shared code is rolled into a single unit and is called T-Server Common Part (TSCP). TSCP is the central, common component for all T-Servers and has its own Release Note, which is accessible via a hyperlink from your T-Server's Release Note.

Note: This document separates common-code features based on TSCP into separate sections and chapters, such as the "T-Server Common Options" chapter. These are the options for all T-Servers that TSCP makes available for configuration.

T-Server Functional Steps During a Sample Call

The following example, Figure 4, outlines some basic steps that T-Server might take when a call arrives from outside the contact center. In this scenario, T-Server starts tracking the call even before it is delivered to the agent. T-Server then informs the selected agent that a call has arrived. When the switch delivers the call to the agent's extension, T-Server presents account information, collected at an Interactive Voice Response (IVR) unit, to the agent at the agent desktop application.

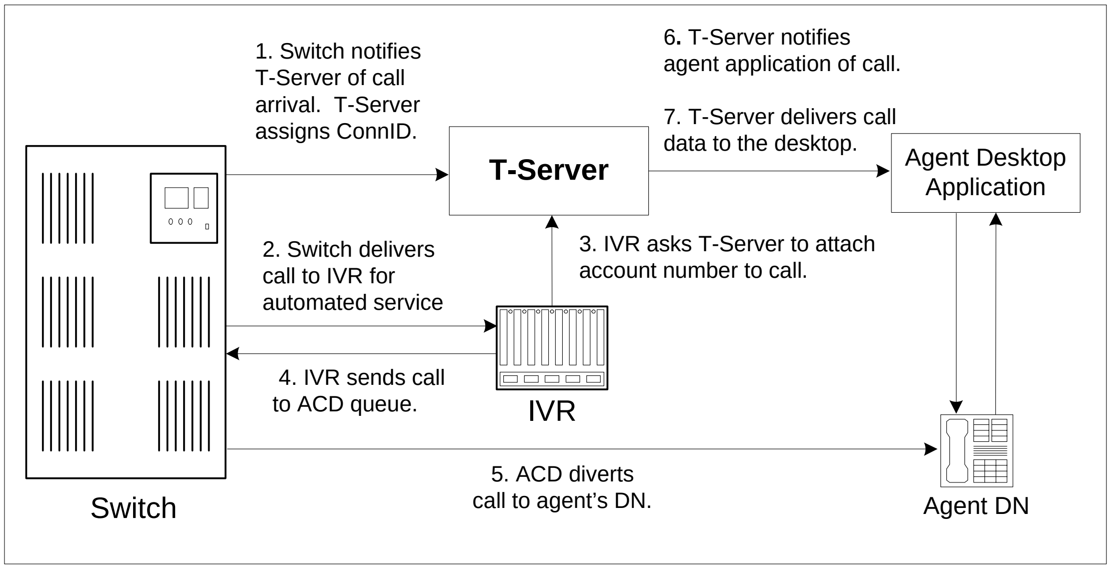

1. Switch notifies T-Server of call arrival. T-Server assigns ConnID.
6. T-Server notifies agent application of call.
7. T-Server delivers call data to the desktop.
T-Server
Agent Desktop Application
2. Switch delivers call to IVR for automated service
3. IVR asks T-Server to attach account number to call.
4. IVR sends call to ACD queue.
IVR
5. ACD diverts call to agent's DN.
Switch
Agent DN

Figure 4: Functional T-Server Steps

Step 1

When the call arrives at the switch, T-Server creates a call in its internal structure. T-Server assigns the call a unique identifier, connection ID.

Step 2

The switch delivers the call to an Interactive Voice Response (IVR) unit, which begins automated interactions with the caller.

Step 3

IVR acquires user information from the caller through prompts and requests T-Server to attach that information to the call. T-Server updates the call with the user information.

Step 4

IVR sends the call to an ACD (Automated Call Distribution) queue.

Step 5

The ACD unit distributes the call to an available agent logged in to a particular DN (directory number).

Step 6

T-Server notifies the agent desktop application that the call is ringing on the agent's DN. The notification event contains call data including `ANI`, `DNIS`, and account information that the IVR has collected.

Step 7

The agent desktop application presents the account information, including the name of the person whose account this is, on the agent's screen, so that the agent answering the call has all the relevant information.

These seven steps illustrate just a small part of T-Server's bridging, messaging, and interaction-processing capabilities.

Advanced Disconnect Detection Protocol

Since the 6.0 release of T-Server, the Advanced Disconnect Detection Protocol (ADDP) has replaced the Keep-Alive Protocol (KPL) as the method to detect failures for certain T-Server connections, including connections between two T-Servers and between a T-Server and its clients.

Notes:

- Starting with release 7.5, the KPL backward compatibility feature is no longer supported.
- ADDP applies only to connections between Genesys software components.

With ADDP, protocol activation and initialization is made on the client's side and you can change these parameters. No additional messages are sent when there is existing activity over the connection. T-Server client applications and the remote T-Server (if any) must be listening to the socket and respond promptly to the polling signal for the connection to be preserved.

If you are going to enable ADDP, you must do it using the `protocol`, `addp-timeout`, `addp-remote-timeout`, and `addp-trace` configuration options. When configuring a timeout, consider the following issues:

- The configured timeout must be at least twice as long as the maximum network latency.
- There may be an interval when T-Server does not check for network activity.
- If the link connection fails but the client is not notified (for example, because the host is turned off, or because a network cable is unplugged), the maximum reaction time to a link-connection failure is equal to double the configured timeout plus the established network latency.

Also keep in mind that the T-Server receiving the polling signal may not respond immediately, and that a delay occurs between the polling signal and the response to travel from one T-Server to another. If you don't account for these contingencies when configuring a timeout, the connection that ADDP is monitoring will be dropped periodically.

Redundant T-Servers

T-Servers can operate in a high-availability (HA) configuration, providing you with redundant systems. The basics of each T-Server's redundant capabilities differ from T-Server to T-Server. One basic principle of redundant T-Servers is the standby redundancy type, which dictates how quickly a backup T-Server steps in when the primary T-Server goes down.

The Framework Management Layer currently supports two types of redundant configurations: `warm standby` and `hot standby`. All T-Servers offer the `warm standby` redundancy type and, starting with release 7.1, the `hot standby` redundancy type is implemented in T-Servers for most types of switches. (See Table 1.)

Instructions for configuring T-Server redundancy are available in Chapter 3, "High-Availability Configuration and Installation." Specifics on your T-Server's HA capabilities are outlined in Part Two of this document.

Notes:

- Network T-Servers use a load-sharing redundancy schema instead of `warm` or `hot standby`. Specifics on your T-Server's HA capabilities are discussed in Part Two of this document.
- IVR Server does not support simultaneous configuration of both Load Balancing functionality and `warm standby`. Only one of these is supported at a time.

Support for Hot Standby Redundancy in Various T-Servers

Use Table 1 to determine whether your T-Server supports the `hot standby` redundancy type. The table also indicates whether HA Proxy components are required for this support, and, if so, how many are required per pair of redundant T-Servers (or per link if so noted).

Table 1 only summarizes `hot standby` redundancy support in various T-Servers. For detailed, up-to-date information on the subject, see the *Genesys 7 Supported Media Interfaces* white paper located on the Technical Support website at `http://genesyslab.com/support/dl/retrieve/default.asp?item=A9CB309AF4DEB8127C5640A3C32445A7&view=item`.

Table 1: T-Server Support of the Hot Standby Redundancy Type

T-Server Type	Hot Standby Supported	HA Proxy Required	Number of HA Proxy Components
Aastra Matra Nexpan 50	Yes	No	—
Alcatel A4200/OXO	Yes	No	—
Alcatel A4400/OXE	Yes	No	—
Aspect ACD	Yes	No	1
Avaya Communication Manager	Yes	No[a]	—
Avaya INDeX	Yes	No	—
Cisco CallManager	Yes	No	—
DataVoice Dharma	Yes	No	—
Digitro AXS/20	Yes	No	—
EADS Intecom M6880	Yes	No	—
eOn eQueue	Yes	No	—
Ericsson MD110	Yes	No	—
Fujitsu F9600	Yes	No	—
Huawei C&C08	Yes	No	—
Mitel SX-2000/MN-3300	Yes	No	—
NEC NEAX/APEX	Yes	No	—
Nortel Communication Server 2000/2100	Yes	Yes[b], No[c]	1 per link
Nortel Communication Server 1000 with SCCS/MLS	Yes	No	—
Philips Sopho iS3000	Yes	No[d]	1
Radvision iContact	No	—	—
Rockwell Spectrum	Yes	No	—
Samsung IP-PCX IAP	Yes	No	—
Siemens Hicom 300/HiPath 4000 CSTA I	Yes	No	—

Table 1: T-Server Support of the Hot Standby Redundancy Type (Continued)

T-Server Type	Hot Standby Supported	HA Proxy Required	Number of HA Proxy Components
Siemens HiPath 3000	Yes	No	—
Siemens HiPath 4000 CSTA III	Yes	No	—
Siemens HiPath DX	Yes	No	—
SIP Server	Yes	No	—
Tadiran Coral	Yes	No	—
Teltronics 20-20	Yes	Yes	1
Tenovis Integral 33/55	Yes	No	—
Network T-Servers[e]			
AT&T	No	—	—
Concert	No	—	—
CRSP	No	—	—
DTAG	No	—	—
GenSpec	No	—	—
ISCP	No	—	—
IVR Server, using network configuration	No	—	—
KPN	No	—	—
MCI	No	—	—
NGSN	No	—	—
Network SIP Server	No	—	—
Sprint	No	—	—
SR3511	No	—	—
Stentor	No	—	—

a. With release 7.1, T-Server for Avaya Communication Manager no longer uses HA Proxy for its support of `hot standby`. Earlier releases of this T-Server require two HA Proxies (for which there is a Configuration Wizard) to support `hot standby`.

b. For T-Server for Nortel Communication Server 2000/2100 in high-availability (`hot standby`) configuration, Genesys recommends that you use link version SCAI14 or above with call-progress messages enabled. See the switch-specific information in Part 2 of this *Deployment Guide* for additional information on HA configurations.

c. Starting with release 7.5, T-Server for Nortel Communication Server 2000/2100 supports HA without HA Proxy when operating in Dual CTI Links mode. See the switch-specific information in Part 2 of this *Deployment Guide* for additional information on HA configurations.

d. Starting with release 6.5.3, T-Server for Philips Sopho iS3000 supports HA both with and without HA Proxy.

e. Although they do not support high availability per se, Network T-Servers do support a load-sharing schema.

Multi-Site Support

Multi-site configuration implies the existence of two or more switches that belong to the same enterprise or service provider, and that share the Genesys Configuration Database. (In some cases this may include isolated partitions on a given switch served by different T-Servers.) The main goal of T-Server support for multi-site operations is to maintain critical information about a call as it travels from one switch to another.

For instructions on installing and configuring a multi-site environment, including information on the Inter Server Call Control (ISCC) features, please see Chapter 3, “Multi-Site Support,” on page 47.

Agent Reservation

T-Server provides support for clients to invoke the agent reservation function, `TReserveAgent()`. This function allows a server application that is a client of T-Server to reserve a DN along with an agent, a `Place`, or both, so that no other T-Server client can route calls to it during a specified reservation interval. Alternatively, when clients use the ISCC feature (see “ISCC Call Data Transfer Service” on page 49), they can use an agent reservation embedded in an ISCC request. (To do so, clients have to specify a certain `Extensions` attribute in an ISCC request when initiating an ISCC transaction. See page 54 for the list of ISCC requests.)

The reservation does not currently prevent the reserved objects from receiving direct calls or calls distributed from ACD Queues; agent reservation is intended as a way of synchronizing the operation of several clients. See `RequestReserveAgent` in the *Voice Platform SDK 7.5 .NET* (or *Java) API Reference* for more details on this function from the client’s point of view.

To reserve an agent, in addition to invoking the `TReserveAgent` function on the client side, you must also configure options in the Configuration Layer. This is also necessary in order to coordinate multiple possible reservation requests.

See “Agent Reservation” in the “T-Server Common Options” chapter in Part Two for more details.

Licensing Requirements

Starting with release 7.2, the licensing requirements for T-Server have changed from previous releases. Please read this section carefully and refer to the *Genesys 7 Licensing Guide* for complete licensing information.

Licensing the Media Layer

All Genesys software is licensed—that is, it is not shareware. Genesys products are protected through legal license conditions as part of your purchase contract. However, the level of technical license-control enforcement varies across different solutions and components.

Before you begin to install T-Server, remember that, although you may not have had to use technical licenses for your software when you deployed the Configuration and Management Layers in their basic configurations, this is not the case with the Media Layer.

T-Server requires seat-related DN technical licenses to operate even in its most basic configuration. Without appropriate licenses, you cannot install and start T-Server. If you have not already done so, Genesys recommends that you install License Manager and configure a license file at this point. For complete information on which products require what types of licenses, and on the installation procedure for License Manager, refer to the *Genesys 7 Licensing Guide* available on the Genesys Documentation Library CD.

The sections that follow briefly describe the T-Server license types.

Licensing Basic Implementations

A stand-alone T-Server serving a single site requires licenses to register all DNs it monitors. DNs that agents use in day-to-day contact center operations, such as Extensions and ACD Positions, have to be registered using licenses that control agent seats.

Note: Configure all seat DNs that agents use (Extensions and ACD Positions) in the Configuration Layer. This enables detailed call monitoring through Genesys reporting, and generally allows you to control access to individual DNs.

Licensing HA Implementations

T-Servers operating with the `hot standby` redundancy type require a special CTI HA technical license, which allows for high-availability implementations, in addition to regular T-Server licenses. Neither T-Server in a redundant pair configured for `hot standby` starts if this license is unavailable. Moreover, the primary and backup T-Servers must use the same licenses to control the same pool of DNs. If your T-Servers are configured with the `hot standby` redundancy type, order licenses for CTI HA support.

Licensing Multi-Site Implementations

T-Servers performing multi-site operations require licenses that allow for such operations, in addition to regular T-Server licenses. If some of your T-Servers are configured for multi-site routing while others are not, either order licenses for multi-site support for all T-Servers or install an additional License Manager to handle the T-Servers involved in multi-site routing.

Note: You do not need licenses for multi-site support if some T-Server clients include the local location as the `location` attribute value in their requests for routing within the same site.

Client Connections

The number of connections T-Server can accept from its clients depend on the operating system that T-Server runs. Table 2 illustrates the number of client connections that T-Server support.

Table 2: The Number of T-Server's Client Connections

Operating System	Number of Connections
AIX 32-bit and 64-bit modes (versions 4.3.3, 5.1, 5.2, 5.3)	32767
HP-UX 32-bit and 64-bit modes (versions 11.0, 11.11, 11i v2)	2048
Linux 32-bit mode (versions RHEL 3.0, RHEL 4.0)	1024
Solaris 2.6 32-bit mode (versions 2.6, 2.7, 8, 9, 10)	1024

Table 2: The Number of T-Server's Client Connections (Continued)

Operating System	Number of Connections
Solaris 7 64-bit mode (versions 2.7, 8, 9, 10)	65536
Tru64 UNIX (versions 4.0F, 5.1, 5.1B)	4096
Windows Server 2003	4096

Next Steps

Now that you have gained a general understanding of the roles and features available with T-Servers, you're ready to learn how T-Servers are installed and configured. That information is presented in the next few chapters of this *Deployment Guide*. So unless you are already familiar with T-Server deployment and operation procedures, continue with Chapter 2, "T-Server Configuration and Installation," on page 33. Otherwise, you may want to jump to Part Two of this *Deployment Guide*, where you will find information about your specific T-Server.

Chapter

2

T-Server Configuration and Installation

This chapter contains general information for the deployment, configuration, and installation of your T-Server. You may have to complete additional configuration and installation steps specific to your T-Server and switch. You will find these steps in Part Two of this document.

This chapter contains these sections:

Note: You *must* read the *Framework 7.5 Deployment Guide* before proceeding with this T-Server guide. That book contains information about the Genesys software you must deploy before deploying T-Server.

Environment Prerequisites for T-Server

T-Server has a number of prerequisites for deployment. Read through the following sections before deploying your T-Server.

Software Requirements

Framework Components

You can only configure T-Server after you have deployed the Configuration Layer of Genesys Framework. This layer contains DB Server, Configuration Server, Configuration Manager, and, at your option, Deployment Wizards. If you intend to monitor or control T-Server through the Management Layer, you must also install and configure components of this Framework layer, such as Local Control Agent (LCA), Message Server, Solution Control Server (SCS), and Solution Control Interface (SCI), before deploying T-Server.

Refer to the *Framework 7.5 Deployment Guide* for information about and deployment instructions for, these Framework components.

Supported Platforms

Refer to the *Genesys 7 Supported Operating Systems and Databases* white paper for the list of operating systems and database systems supported in Genesys releases 7.x. You can find this document on the Genesys Technical Support website at `http://genesyslab.com/support/dl/retrieve/default.asp?item=B6C52FB62DB42BB229B02755A3D92054&view=item`.

For UNIX-based (UNIX) operating systems, also review the list of patches Genesys uses for software product builds, and upgrade your patch configuration if necessary. A description of patch configuration is linked to installation `read_me.html` files for the Genesys applications that operate on UNIX, and is available within the installation packages.

Notes: Starting with release 7.5, T-Server supports the Genesys Transport Layer Security (TLS) and can be configured for secure data exchange with the other Genesys components that support this functionality.

The Genesys TLS is not supported on all operating systems that T-Server itself supports. For information about the supported operating systems, see the *Genesys 7.5 Transport Layer Security Deployment Guide*.

Hardware and Network Environment Requirements

Hosting

Genesys recommends that you or your IT specialist assign host computers to Genesys software before you start Genesys installation. Keep in mind the following restrictions:

- Do not install all the Genesys server applications on the same host computer.

- When installing a few server applications on the same host computer, prevent them (except for Configuration Server) from using the swap area.

Installation Privileges

During deployment, be sure to log in with an account that will permit you to perform administrative functions—that is, one that has root privileges.

Server Locations

Refer to the “Network Locations for Framework Components” chapter of the *Framework 7.5 Deployment Guide* for recommendations on server locations.

Supported Platforms

Refer to the *Genesys Supported Media Interfaces* white paper for the list of supported switch and PABX versions. You can find this document on the Genesys Technical Support website at `http://genesyslab.com/support/dl/retrieve/default.asp?item=A9CB309AF4DEB8127C5640A3C32445A7&view=item`.

Media Layer Requires Licensing

You need a license to configure and install Media Layer components. Genesys recommends that, if you have not already done so, at this point you:

1. Install License Manager.
2. Configure license files.

Note: If you use the `<port>@<server>` format when entering the name of the license server during installation, remember that some operating systems use `@` as a special character. In this case, the installation routine is unable to write license information for T-Server to the Configuration Layer or the `run.sh` file. Therefore, when you use the `<port>@<server>` format, you must manually modify the command-line license parameter after installing T-Server.

For information about which products require what types of licenses and for the installation procedure for License Manager, refer to the *Genesys 7 Licensing Guide* available on the Genesys Documentation Library CD.

Note: Starting with release 7.0, T-Server has new licensing requirements. Be sure to check the appropriate information in the *Genesys 7 Licensing Guide* and in “Licensing Requirements” on page 29.

The Media Layer and LCA

To monitor the status of components in the Media Layer through the Management Layer, you must load an instance of LCA on every host running Media Layer components. Without LCA, Management Layer cannot monitor the status of any of these components. If you do not use the Management Layer, LCA is not required.

About Configuration Options

Configuring T-Server is not a onetime operation. It is something you do at the time of installation and then in an ongoing way to ensure the continued optimal performance of your software. You must enter values for T-Server configuration options in the relevant Wizard screens or on the `Options` tab of your T-Server Application object in Configuration Manager. The instructions for configuring and installing T-Server that you see here are only the most rudimentary parts of the process. You must refer extensively to the configuration options chapters located in Part Two of this book. Pay particular attention to the configuration options specific to your own T-Server.

Configuration options common to all T-Servers, independent of switch type, are described in the “T-Server Common Options” chapter of this guide. *Switch-specific* configuration options are described in a separate chapter. T-Server also supports unified Genesys log options, as described in the “Common Log Options” chapter.

Options that configure values for the TSCP software in your T-Server are common to all T-Servers. Options based on the custom features of your switch apply to your T-Server only. Familiarize yourself with both types of options. You will want to adjust them to accommodate your production environment and the business rules that you want implemented there.

T-Server Deployment Methods

Genesys recommends using the T-Server Configuration Wizard to deploy T-Server. However, if for some reason you must manually deploy T-Server, you will also find instructions for doing that in this section.

Note: Genesys does not recommend installation of its components via a Microsoft Remote Desktop connection. The installation should be performed locally.

General Order of Deployment

The recommended sequence to follow before deploying T-Server is described below. Steps 1 through 3 apply for both Wizard-based and manual deployment. For Wizard deployment, Steps 4 and 5 take place within the Wizard deployment process itself.

1. Deploy Configuration Layer objects and ensure Configuration Manager is running (see the *Framework 7.5 Deployment Guide*).
2. Deploy Network objects (such as Host objects).
3. Deploy the Management Layer (see the *Framework 7.5 Deployment Guide*).

When manually deploying T-Server, you must continue with the next two steps. If you are deploying T-Server with the Configuration Wizard, the next two steps take place within the Wizard deployment process itself, where you can create and configure all the necessary objects for T-Server deployment.

4. Configure Telephony objects (see “Manual Configuration of Telephony Objects” on page 40):
   - Switching Offices
   - Switches
   - Agent Logins
   - DNs
5. Deploy the Media Layer:
   - T-Server (beginning with “Manual Configuration of T-Server” on page 42).

If, during the installation procedure for any of the Genesys applications, the script warns you that Configuration Server is unavailable and that the configuration cannot be updated, continue with the installation. When installation is complete, you must complete the information on the `Start Info` tab to ensure that T-Server will run. See “After Completing the Manual Installation” on page 45.

Wizard Deployment of T-Server

Configuration Wizards facilitate component deployment. T-Server configuration and installation involves many steps, and Genesys strongly recommends that you set up T-Server using the Wizard rather than manually. T-Server Wizard guides you through a series of steps and options to customize your deployment of T-Server.

Wizard Configuration of T-Server

The first step to take for a Wizard-based configuration is to install and launch Genesys Wizard Manager. (Refer to the *Framework 7.5 Deployment Guide* for instructions.) When you first launch Genesys Wizard Manager, it suggests that you set up the Management Layer and then the Framework. The Framework setup begins with configuring and creating the objects related to T-Server, starting with the `Switch` and `Switching Office` objects, and the T-Server's `Application` object itself.

Note: With the Wizard, you create your T-Server `Application` object in the course of creating your `Switch` object.

During creation of the `Switch` object, you also have an opportunity to run the Log Wizard to set up T-Server logging. Then, you can specify values for the most important T-Server options. Finally, you can create contact center objects related to T-Server, such as DNs, Agent Logins, and some others.

Note: During configuration of a `Switch` object, the Wizard prompts you to copy a T-Server installation package to an assigned computer. After that package is copied to the destination directory on the T-Server host, complete the last steps of the T-Server configuration. Then, install T-Server on its host.

After you complete the Framework configuration, the Genesys Wizard Manager screen no longer prompts you to set up the Framework. Instead, it suggests that you set up your solutions or add various contact center objects to the Framework configuration, including the Switch, DNs and Places, Agent Logins, Agent Groups, Place Groups, and, in a multi-tenant environment, a Tenant. In each case, click the link for the object you wish to create. Again, you create a new T-Server `Application` object in the course of creating a new `Switch` object.

Wizard Installation of T-Server

After creating and configuring your T-Server and its related components with the Wizard, you proceed to T-Server installation. That installation process closely mimics that of previously installed components (for example, Message Server).

Note: Certain Wizard-related procedures are not described in this document. Refer to the *Framework 7.5 Deployment Guide* for general instructions.

To install T-Server on its host computer, perform the following steps:

On UNIX

1. In the directory to which the T-Server installation package was copied during Wizard configuration, locate a shell script called `install.sh`.
2. Run this script from the command prompt by typing `sh` and the file name. For example: `sh install.sh`.
3. When prompted, confirm the host name of the computer on which T-Server is to be installed.
4. When prompted, confirm the application name of the T-Server that is to be installed.
5. Specify the destination directory into which T-Server is to be installed, with the full path to it.
6. If asked which version of the product to install, the 32-bit or the 64-bit, choose the one appropriate to your environment.
7. Specify the license information that T-Server is to use.

As soon as the installation process is finished, a message appears announcing that installation was successful. The process places T-Server in the directory with the name specified during the installation.

On Windows

1. Open the directory to which the T-Server installation package was copied during Wizard configuration.
2. Locate and double-click `Setup.exe` to start the installation. The `Welcome` screen launches.
3. When prompted, specify the connection parameters to the Configuration Server associated with this T-Server.
4. Identify the T-Server Application object in the Configuration Layer to be used by this T-Server.
5. Specify the license information that T-Server is to use.
6. Specify the destination directory into which T-Server is to be installed.
7. Click `Install` to begin the installation.
8. Click `Finish` to complete the installation.

By default, T-Server is installed as a Genesys service (Windows `Services)` with `Automatic` startup type.

Manual Deployment of T-Server

Deploying T-Server manually requires that you configure a number of different objects in the Configuration Layer prior to setting up your T-Server objects and then install T-Server. This section describes the manual deployment process.

Manual Configuration of Telephony Objects

This section describes how to manually configure T-Server Telephony objects if you are using Configuration Manager.

Recommendations

Genesys recommends registering (configuring) only those entities you plan to use in the current configuration. The more data there is in the Configuration Database, the longer it takes for the CTI setup to start, and the longer it will take to process configuration data. Remember that adding configuration objects to the Genesys Configuration Database does not cause any interruption in contact center operation.

Depending on how much work is required to manually configure all applications and objects, consider registering more Person objects first, with a set of privileges that lets them perform configuration tasks.

Switching Offices

Your telephony network may contain many switching offices, but you should only configure those that are involved with customer interactions.

Using Configuration Manager, be sure to register a `Switching Office` object that accommodates your `Switch` object under `Environment`. Until you have done this, you cannot register a `Switch` object under `Resources` (single-tenant environment) or a `Tenant` (multi-tenant environment).

Note: The value for the switching office name must not have spaces in it.

Switches

1. Configure a `Switch` object for each switch on your telephony network. Assign each `Switch` object to the appropriate `T-Server` object.
2. If implementing the multi-site configuration, specify access codes for all switches on the network so that the call-processing applications can route and transfer calls between switches.

Two types of access codes exist in a Genesys configuration:

- Default access codes that specify how to reach this switch from any other switch in the Genesys environment.
- Switch-to-switch access codes that specify how to reach a particular switch from any other switch. Use this type when either a nondefault dial number or routing type is required between any two locations. When a switch-to-switch access code is configured, its value has a higher priority than that of a default access code.

See Chapter 3, “Multi-Site Support,” on page 47, for step-by-step instructions.

Note: When the numbering plan uses unique directory number (DN) assignment across sites and multi-site routing is not used, you do not have to configure access codes.

DNs and Agent Logins

Note: Starting with release 7.2, the requirements for configuring DNs in the Configuration Layer have changed. Refer to Part Two of this guide for information about the requirements on configuring specific DN types for your T-Server.

For each T-Server for which you are configuring DNs, you must configure all DNs that agents and their supervisors use in day-to-day contact center operation—so-called *seat-related DNs*—such as Extensions and ACD Positions. Otherwise, T-Server does not register such DNs.

1. To configure Telephony objects within each switch, consult the switch documentation. Information specific to your T-Server in Part Two of this document contains tables that indicate how to set DN types in the Genesys Configuration Database depending on the switch DN types and configuration.
2. Check the numbering plan for different types of DNs, to see if you can save time by registering Ranges of DNs. Usually, DNs of the same type have consecutive numbers, which will make an otherwise tedious configuration task easy. Agent Login objects almost always have consecutive numbers, which means you can register them through the Range of Agent Logins feature as well.
3. If you plan to use Virtual Queues and Virtual Routing Points in the contact center operation, Genesys recommends registering them after you have outlined the call-processing algorithms and identified your reporting needs.

   Note: Remember that CTI applications, not the switch, generate telephony events for DNs of these types.

Warning! DNs with the `Register` flag set to `false` may not be processed at T-Server startup; therefore, associations on the switch will be created only when T-Server client applications require DN registration.

Multi-Site Operations

See the section, “Configuring Multi-Site Support” on page 83, for information on setting up DNs for multi-site operations.

Manual Configuration of T-Server

Note: Use the *Framework 7.5 Deployment Guide* to prepare accurate configuration information. You may also want to consult *Configuration Manager Help,* which contains detailed information on configuring objects.

Recommendations

Genesys recommends using an Application Template when you are configuring your T-Server application. The Application Template for your particular T-Server contains the most important configuration options set to the values recommended for the majority of environments. When modifying configuration options for your T-Server application later in the process, you can change the values inherited from the template rather than create all the options by yourself.

Step-By-Step T-Server Configuration

To manually configure T-Server:

1. Follow the standard procedure for configuring all Application objects to begin configuring your T-Server `Application` object. Refer to the *Framework 7.5 Deployment Guide* for instructions.

2. In a `Multi-Tenant` environment, specify the `Tenant` to which this T-Server belongs on the `General` tab of the `Properties` dialog box.

3. On the `Connections` tab:

- Add all Genesys applications to which T-Server must connect.

Note: For multi-site deployments you should also specify T-Server connections on the `Connections` tab for any T-Servers that may transfer calls directly to each other.

4. On the `Options` tab, specify values for configuration options as appropriate for your environment.

Note: For T-Server option descriptions, see Part Two of this document. The configuration options common to all T-Servers are described in the "T-Server Common Options" chapter. The switch-specific configuration options are described in a separate chapter. T-Server also uses common Genesys log options, described in the "Common Log Options" chapter.

5. In a multi-site environment, you must complete additional T-Server configuration steps to support multi-site operations; see Chapter 3, "Multi-Site Support," on page 47.

Multiple Ports Configuration

Starting with release 7.5, T-Server can communicate with its clients via multiple ports. In order to configure additional listening ports:

1. Open the T-Server `Application Properties` dialog box.
2. Click the `Server Info` tab.
3. In the `Ports` section, click `Add Port`.
4. In the `Port Properties` dialog box, on the `Port Info` tab:
   a. In the `Port ID` text box, enter the port ID.
   b. In the `Communication Port` text box, enter the number of the new port.
   c. In the `Connection Protocol` box, select the connection protocol, if necessary.
   d. Select the `Listening Mode` option.

   Note: For more information on configuring secure connections between Framework components, see *Genesys 7.5 Transport Layer Security Deployment Guide*.

   e. Click `OK`.
5. Click `OK` to save the new configuration.

Manual Installation of T-Server

The following directories on the Genesys 7.5 Media product CD contain T-Server installation packages:

- `media_layer/<switch>/<platform>` for UNIX installations, where `<switch>` is your switch name and `<platform>` is your operating system.
- `media_layer\<switch>\windows` for Windows installations, where `<switch>` is your switch name.

On UNIX

Note: During installation on UNIX, all files are copied into the directory you specify. No additional directories are created within this directory. Therefore, do not install different products into the same directory.

1. In the directory to which the T-Server installation package was copied, locate a shell script called `install.sh`.
2. Run this script from the command prompt by typing `sh` and the file name. For example: `sh install.sh`.
3. When prompted, confirm the host name of the computer on which T-Server is to be installed.
4. When prompted, specify the host and port of Configuration Server.
5. When prompted, enter the user name and password to access Configuration Server.
6. When prompted, select the T-Server application you configured in "Step-By-Step T-Server Configuration" on page 42 from the list of applications.
7. Specify the destination directory into which T-Server is to be installed, with the full path to it.
8. If asked which version of the product to install, the 32-bit or the 64-bit, choose the one appropriate to your environment.
9. Specify the license information that T-Server is to use: either the full path to, and the name of, the license file, or the license server parameters.

As soon as the installation process is finished, a message appears announcing that installation was successful. The process places T-Server in the directory with the name specified during the installation.

On Windows

1. In the directory to which the T-Server installation package was copied, locate and double-click `Setup.exe` to start the installation.
2. When prompted, specify the connection parameters to the Configuration Server associated with this T-Server.
3. When prompted, select the T-Server `Application` you configured in "Step-By-Step T-Server Configuration" on page 42 from the list of applications.
4. Specify the license information that T-Server is to use: either the full path to, and the name of, the license file, or the license server parameters.
5. Specify the destination directory into which T-Server is to be installed.
6. Click `Install` to begin the installation.
7. Click `Finish` to complete the installation.

By default, T-Server is installed as a Genesys service (Windows `Services`) with `Automatic` startup type.

After Completing the Manual Installation

1. Open the `Properties` dialog box for a corresponding `Application` object in Configuration Manager.
2. Verify that the `State Enabled` check box on the `General` tab is selected.
3. Verify that the `Working Directory`, `command-line`, and `Command-Line Arguments` are specified correctly on the `Start Info` tab.
4. Click `Apply` and `OK` to save any configuration updates.

Next Steps

At this point, you have either used the Wizard to configure and install T-Server, or you have done it manually, using Configuration Manager. In either case, if you want to test your configuration and installation, go to Chapter 4, "Start and Stop T-Server Components," on page 97, and try it out. Otherwise, if you want to install T-Servers for a multi-site environment, proceed to Chapter 3, "Multi-Site Support," on page 47.

Chapter

3

Multi-Site Support

This chapter contains general information about multi-site environments, as well as information on deploying a multi-site environment for your T-Server.

This chapter is divided into the following sections:

Note: Each switch/T-Server combination offers different multi-site options. For details describing your specific switch/T-Server environment, refer to Chapter 9, "T-Server Common Configuration Options," on page 149.

The following instructions apply to both local and remote switches and T-Servers. Because different vendor switches can be installed at the local and remote locations, this chapter covers several, but not all, possible configurations. To help determine which sections of this chapter apply to your situation, refer to Table 3 on page 63 and Table 4 on page 67.

For more information on your specific switch/T-Server environment, refer to the multi-site topics in Part Two of this guide.

Multi-Site Fundamentals

A multi-site configuration has two or more switches that belong to the same enterprise or service provider and that share the Genesys Configuration Database. (In some cases, this may include isolated partitions on a given switch served by different T-Servers.) The main goal of T-Server support for multi-site operations is to maintain critical information about a call as it travels from one switch to another.

T-Server supports multi-site operations using its *Inter Server Call Control* (ISCC; formerly called External Routing), which supports the following functions:

- **Call matching—**To link instances of a call distributed across multiple sites and to re-attach essential data associated with the call (`ConnID`, `UserData`, call history). The following T-Server features support this capability:
  - ISCC Call Data Transfer Service (active external routing)—when requested by a T-Server client by specifying the desired destination in the `Location` parameter, and also with various ISCC strategies performed by direct dial or by using the Transfer Connect Service. See "ISCC Transaction Types" on page 54 and "Transfer Connect Service Feature" on page 66.
  - Inter Server Call Control/Call Overflow (ISCC/COF) feature (passive external routing)—applicable when calls are overflowed to another site either directly or manually (see page 67).
  - Number Translation feature (see page 71).
  - Network Attended Transfer/Conference (NAT/C) feature (see page 79).

Note: When ISCC detects call instance reappearance on a given site, the call is assigned a unique `ConnID` and the user data is synchronized with the previous call instances. This ensures that `ConnID`s assigned to different instances of the same call on a given site are unique.

- **Call data synchronization between associated call instances (ISCC Event Propagation)—**To provide the most current data to call instances residing on remote T-Servers. The following T-Server features support this capability:
  - User Data propagation (see page 81)
  - Party Events propagation (see page 82)

Note: ISCC automatically detects topology loops and prevents continuous updates.

Note: In distributed networks, Genesys recommends using call flows that prevent multiple call instance reappearance and call topology loops. This approach ensures that all T-Servers involved with the call report the same `ConnID`, and also optimizes telephony trunk allocation (that is, it prevents trunk tromboning).

The T-Server configuration contains information about other T-Servers with which it will communicate. T-Server uses this information to connect with the other T-Servers. During this “handshake” process, T-Servers exchange information about the following parameters:

- Protocol type
- Switch type
- Server name
- Location name (switch name)
- T-Server role (backup or primary)

To complete the handshake process, T-Servers exchange messages about the current condition of the links to their switches. After the handshake process is complete, T-Server is ready to support a multi-site operation.

ISCC Call Data Transfer Service

Because ISCC supports active external routing, T-Servers that serve different switches (usually on different sites) can exchange call data when a call is passed from one switch to another. With this functionality, T-Server provides its clients with the following additional information about each call received from another switch:

- The `ConnID` of the call
- Updates to user data attached to the call at the previous site
- Call history

Note: Load-sharing IVR Servers and Network T-Servers cannot be designated as the destination location for ISCC.

Figure 5 shows the steps that occur during a typical external routing (ISCC) transaction. Note that the location where a call is initially processed is called the *origination location,* and the location to which the call is passed is called the *destination location.*

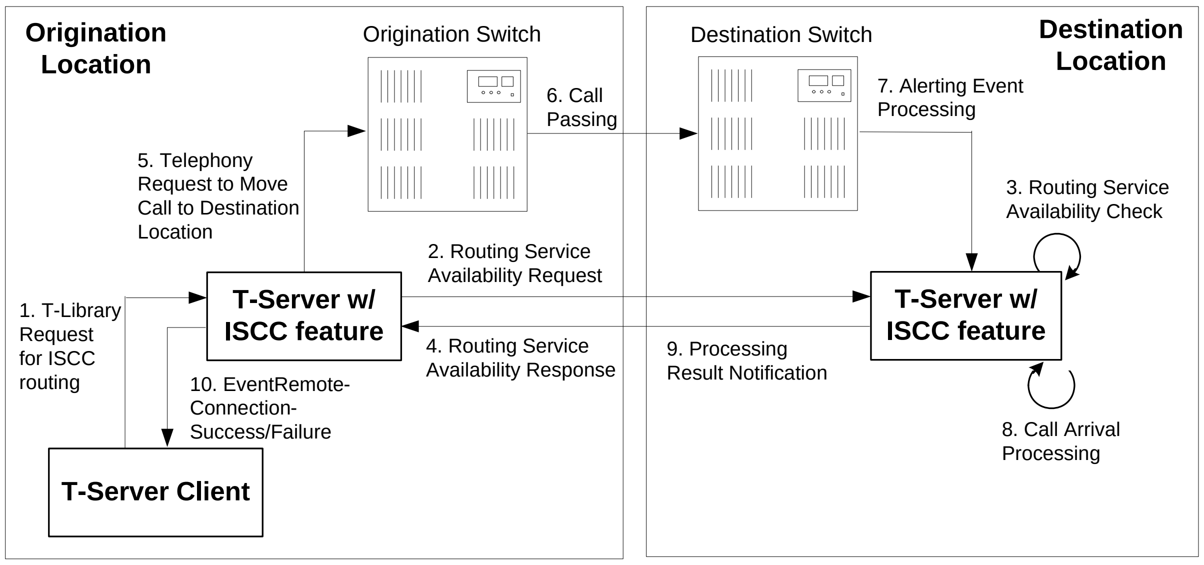

Origination Location
Origination Switch
Destination Switch
Destination Location
6. Call Passing
7. Alerting Event Processing
5. Telephony Request to Move Call to Destination Location
3. Routing Service Availability Check
2. Routing Service Availability Request
1. T-Library Request for ISCC routing
T-Server w/ ISCC feature
T-Server w/ ISCC feature
4. Routing Service Availability Response
9. Processing Result Notification
10. EventRemote-Connection-Success/Failure
8. Call Arrival Processing
T-Server Client

Figure 5: Steps in the ISCC Process

ISCC Call Flow

The following section identifies the steps (shown in Figure 5) that occur during an ISCC transfer of a call.

Step 1

A client connected to the T-Server at the origination location requests this T-Server to pass a call with call data to another location. For this purpose, the client must specify the `location` parameter (Attribute `Location`) when calling a corresponding T-Library function. ISCC processes the following T-Library requests:

- `TInitiateConference`
- `TInitiateTransfer`
- `TMakeCall`
- `TMuteTransfer`
- `TRouteCall`
- `TSingleStepTransfer`

Step 2

Upon receiving a client's request, the origination T-Server checks that the:

a. Connection to the destination T-Server is configured in the origination T-Server `Properties` dialog box.

b. Connection to the destination T-Server is active.

c. Destination T-Server is connected to its link.

d. Origination T-Server is connected to its link.

If these four conditions are met, the origination T-Server determines the transaction type that will be used for passing call data to another location in this transaction. The following possibilities exist:

- The client can request what *ISCC transaction type* (or simply *transaction type*) to use by specifying an appropriate key-value pair in the `Extensions` attribute of the request. The key-value pair must have a key equal to `iscc-xaction-type` and either an integer value as specified in the `TXRouteType` enumeration (see the *Voice Platform SDK 7.5 .NET (*or *Java) API Reference*) or a string value equal to one of the following: `default`, `route`, `direct` (or `direct-callid`), `direct-network-callid`, `direct-notoken`, `direct-ani`, `direct-uui`, `direct-digits`, `reroute`, `dnis-pool`, `pullback`, or `route-uui`.
- If the client does not specify the transaction type in the request or specifies the `default` transaction type, T-Server checks the Switch configuration for the transaction type configured in the Access Code (or Default Access Code) properties:
  - If the `Route Type` property of the Access Code is set to any value other than `default`, T-Server uses the specified value as the transaction type.
  - If the `Route Type` property of the Access Code is set to the `default` value, T-Server uses the first value from the list specified in the `cast-type` configuration option configured for the destination T-Server. If no value has been specified for the `cast-type` option, the default value of `route` is used as the transaction type.

Note: See “Switches” on page 85 for more information on Access Codes and Default Access Codes.

After the origination T-Server determines the requested transaction type, it determines if the destination T-Server supports this transaction type.

You must list the transaction types T-Server supports in the `cast-type` configuration option.

The origination T-Server issues a request for routing service availability and sends it to the destination T-Server. The T-Server request contains data that should be passed along with the call to the destination location. This data includes the transaction type, `ConnID`, `UserData,` and `CallHistory`.

The timer specified by the `request-tout` configuration option is set when the origination T-Server sends the request. If either the specified timeout expires or the call is abandoned before the origination T-Server receives a response from the destination T-Server, the operation is considered failed. In this scenario, the origination T-Server:

a. Generates a request to the destination T-Server to cancel the request for routing service.

b. Sends `EventError` to the client that requested the service.

c. Deletes information about the request.

Step 3

The destination T-Server receives the request for routing service availability and checks the requested type of routing. Depending on the ISCC transaction type, it stores the request information and, when appropriate, allocates access resources for the coming call. For example, an External Routing Point is allocated when the transaction type is `route`, and a DNIS number is allocated when the transaction type is `dnis-pool`.

Note: The `resource-allocation-mode` and `resource-load-maximum` configuration options determine how resources are allocated. Refer to Chapter 9, "T-Server Common Configuration Options," on page 149 for option descriptions.

If resources are unavailable, the request is queued at the destination location until a resource is free or the origination T-Server cancels the request. If the request is canceled, the destination T-Server deletes all information about the request.

If resources are unavailable because of incorrect configuration, the destination T-Server returns an error event to the origination T-Server.

Step 4

If resources are available, the destination T-Server generates a positive response and the timer is started for the interval specified by the `timeout` configuration option of the destination T-Server.

Step 5

If the origination T-Server receives a negative response, it sends an `EventError` to the client and clears all data about the request.

If the origination T-Server receives the confirmation about routing service availability, it processes the client's request and sends a corresponding message to the switch. The timer on the origination T-Server is also started for the interval specified by the `timeout` configuration option of the destination T-Server.

Step 6

The origination switch processes the T-Server request and passes the call to the destination switch.

Step 7

If the call arrives at the destination switch, the switch generates an alerting event.

The destination T-Server waits for the call no longer than the interval specified by the timeout configured on the destination T-Server. If the call is not received at the destination location within this interval, the destination T-Server issues a failure notification to the origination T-Server, deletes all data about the request, and, when appropriate, frees the resources previously allocated for the request.

If either the specified timeout expires or the call is abandoned before the origination T-Server receives a response from the destination T-Server, the operation is considered failed. In this case, the origination T-Server:

a. Generates a request to the destination T-Server to cancel the request for routing service.

b. Responds to the client that requested the service in one of the following ways:

- If the origination T-Server has already sent a response to the request the client sent in Step 1, the origination T-Server supplements its response with `EventRemoteConnectionFailed`.
- If the origination T-Server has not yet sent a response to the client, the origination T-Server sends `EventError`.

c. Deletes information about the request.

Step 8

If the destination T-Server matches the arrived call, it updates the `ConnID`, `UserData`, and `CallHistory` attributes with the data received in the request for routing service availability. The connection ID is updated as follows:

The arrived call is assigned the `ConnID` that is specified in the request for routing service availability, but only if this `ConnID` does not coincide with the `ConnID` of a call that has existed at the destination site. If two such `ConnID`s are identical, the arrived call is assigned a new unique `ConnID`.

For `direct-*` transaction types (where the asterisk stands for a `callid`, `uui`, `ani`, or `digits` extension), the call reaches the destination DN directly.

For the transaction types `route` and `route-uui`, the call first arrives at an External Routing Point from which it is routed to the destination DN. The call info is updated when the call reaches the External Routing Point. An External Routing Point is considered free when the first alerting event (`EventQueued` or `EventRouteRequest`) is distributed.

Please keep the following issues in mind when using the ISCC feature:

- If routing from a dedicated External Routing Point to the destination DN fails, T-Server considers the transaction failed. However, the `ConnID`, `UserData`, and `CallHistory` attributes are updated. Then, T-Server attempts to route the call to one of the Default DNs configured for this External Routing Point.
- If the destination T-Server did not receive a request for routing service availability, but a call arrives at an External Routing Point, T-Server considers the call to be unexpected and routes the call to the DN specified

by the `dn-for-unexpected-calls` configuration option. When no alternative targets are defined, the call remains at the External Routing Point until diverted by the switch or abandoned by the caller.

For `reroute` and `pullback` transaction types, the call returns to the network location. For the `dnis-pool` transaction type, the call reaches the destination DN directly.

Step 9

If, in Step 8, the call does not arrive within the configured timeout, or the transaction fails, the destination T-Server sends a notification of failure to the origination T-Server.

Otherwise, the destination T-Server notifies the origination T-Server that the routing service was successful and deletes all information about the request.

Step 10

The origination T-Server notifies the client that the routing service was successful (or failed) and deletes all information about the request.

ISCC Transaction Types

As switches of different types provide calls with different sets of information parameters, a single mechanism for passing call data between the switches is not feasible in some cases. Therefore, the ISCC feature supports a number of mechanisms for passing call data along with calls between locations. This section describes ISCC transaction type principles, identifies which transaction types are supported for each T-Server, and defines each transaction type (beginning with “direct-ani” on page 55).

It is important to distinguish the two roles that T-Servers play in an external routing (ISCC) transaction—namely *origination T-Server* and *destination T-Server.*

- The origination T-Server initiates an ISCC transaction. It prepares to send the call to another T-Server and coordinates the process.
- The destination T-Server receives call data from an origination T-Server and matches this data to a call that will arrive at some time in the future.

The distinction between these roles is important because the range of telephony-hardware functionality often requires T-Servers to support two entirely different sets of ISCC transactions based on which of the two roles they play. For instance, it is very common for a particular T-Server to support many types of ISCC transactions when it takes on the origination role, but fewer when it takes on the role of a destination T-Server.

The ISCC transaction type `Reroute` is a good example. Most T-Servers support `Reroute` as origination T-Servers, but very few support `Reroute` as destination T-Servers.

Determining and Configuring Transaction Type Support

You can find descriptions of these transaction types starting on page 55. Use Table 3 on page 63 to identify the transaction types your destination T-Server supports. A blank table cell indicates that T-Server does not support a certain transaction type.

You can configure the transaction types specific to your T-Server as values of the `cast-type` configuration option specified in the ISCC configuration section `extrouter`. Refer to Chapter 9, "T-Server Common Configuration Options," on page 149 for the option description.

ISCC Transaction Type General Principles

Generally, since most of the ISCC implementation is done at the T-Server Common Part (TSCP) code level, all T-Servers support certain ISCC transaction types. Any T-Server can act as the origination T-Server for the following transaction types:

- `direct-ani`, page 55
- `direct-notoken`, page 58
- `dnis-pool`, page 58
- `pullback`, page 60
- `reroute`, page 60
- `route` (aliased as `route-notoken`), the default transaction type, page 61

The following transaction types are unevenly supported for both the origination and destination T-Server roles:

- `direct-callid` (aliased as `direct`), page 56
- `direct-digits` (reserved for Genesys Engineering)
- `direct-network-callid`, page 57
- `direct-uui`, page 57
- `route-uui`, page 62

The `reroute` and `pullback` transaction types are supported only for selected T-Servers in the *destination* role. However, if you implement this support, other transaction types require additional configuration and testing—even those that would normally be supported by default.

direct-ani

With the transaction type `direct-ani`, the `ANI` network attribute is taken as the parameter for call matching. Properly configured switches and trunks can keep the `ANI` attribute when a call is transferred over the network. T-Server is capable of using this network feature for call matching.

Warnings!

- Depending on the switch platform, it is possible to inherit the `ANI` attribute after routing a call to a remote destination, and after performing a Single-Step Transfer and other telephone actions. However, ISCC only works properly in scenarios where the `ANI` attribute on the destination T-Server is represented by exactly the same digit string as on the origination T-Server.
- Typically, the `ANI` attribute represents the original call identifier (customer phone number), which guarantees that the attribute remains unique. However, you can use the `non-unique-ani` resource type to block ISCC from matching calls based on an ANI that is known to be non unique. (See "Access Resources for Non-Unique ANI" on page 91 for details.)

Notes:

- Some switches, such as Nortel Communication Server 2000/2100 (formerly DMS-100) and Avaya Communication Manager (formerly DEFINITY ECS (MV), may omit the `ANI` attribute for internal calls—that is, for calls whose origination and destination DNs belong to the same switch. If this is the case, do not use the `direct-ani` transaction type when making, routing, or transferring internal calls with the ISCC feature.
- When the `direct-ani` transaction type is in use, the Number Translation feature becomes active. See "Number Translation Feature" on page 71 for more information on the feature configuration.
- With respect to the `direct` transaction types, Network T-Servers and load-sharing IVR Servers are not meant to play the role of destination T-Servers for call routing. Using Network T-Server with these transaction types requires special architecture.

direct-callid

With the transaction type `direct-callid`, the call reaches the destination DN directly from another location, and the `CallID` of the call is taken as the attribute for call matching. When a call arrives at the final destination, the destination T-Server identifies its `CallID`, and updates the call info if the `CallID` matches.

Use this transaction type when the destination switch has the capability to assign to an incoming call the same network-wide unique `CallID` that the origination switch has already assigned to that call.

Notes:

- The `direct-callid` transaction type is used only in conjunction with the `TRouteCall` and `TSingleStepTransfer` function calls. They are applied only to the call that is in progress, and do not apply to functions that involve in the creation of a new call (for example, `TMakeCall`.)
- For T-Server for Nortel Communication Server 2000/2100, the `direct-callid` transaction type is also applied to the `TMuteTransfer` function.

direct-network-callid

With the transaction type `direct-network-callid`, the call reaches the destination DN directly from another location, and the `NetworkCallID` of the call is taken as the attribute for call matching. When a call arrives at the final destination, the destination T-Server identifies its `NetworkCallID`, and updates the call info if the `NetworkCallID` matches.

Use this transaction type when the destination switch has the capability to assign to an incoming call the same network-wide unique `NetworkCallID` that the origination switch has already assigned to that call.

Note: To support this transaction type, you must configure `Target Type` and `ISCC Protocol Parameters` fields of the corresponding `Switch Access Code` in the Configuration Layer. Refer to Part Two of this document for information about settings specific for your T-Server type.

direct-uui

With the transaction type `direct-uui`, so-called user-to-user information (UUI) is taken as the attribute for call matching. Some switches make it possible to send a small data packet along with a call. T-Server can use this data to recognize a call passed from one switch to another. The destination T-Server generates a local unique value for `UUI`, and then notifies the origination T-Server. The origination T-Server uses a provided value to mark the call coming from the origination location. The destination T-Server receives a call and checks whether it is marked with an exact `UUI` value. If so, the call is considered as matched.

On the Avaya Communication Manager and the Aspect ACD, UUI is referred to as “user-to-user information.” On the Siemens Hicom 300 switch with CallBridge, UUI is referred to as “Private User Data.” On the Alcatel A4400/OXE switch, UUI is referred to as “correlator data.”

Note: To support this transaction type, you must configure your switches to pass the UUI provided by your T-Server. Moreover, the trunks involved must not drop this data.

direct-notoken

With the transaction type `direct-notoken`, T-Server expects a call to arrive from another location to the destination DN specified in the request for routing service availability. When a call reaches the specified DN, T-Server processes the call as the expected externally routed call.

Notes:

- This matching criterion is weak because any call that reaches the specified DN is considered to be the expected call. Genesys recommends that you use this transaction type only in a contact center subdivision that can be reached from within the contact center only (for example, the second line of support, which customers cannot contact directly).
- With respect to the `direct` transaction types, Network T-Servers and load-sharing IVR Servers are not meant to play the role of destination T-Servers for call routing. Using Network T-Server with these transaction types requires special architecture.

dnis-pool

With the `dnis-pool` transaction type, T-Server reserves one of its DNIS access resources and waits for the call that has the same `DNIS` attribute as the name of the reserved DNIS access resource.

If the arrived call is matched successfully, the destination T-Server may update the value of the `DNIS` attribute of the call (along with `ConnID`, `UserData`, and `CallHistory`) with the value of the `DNIS` attribute of the original call. This occurs when the value of the `DNIS` attribute of the original call is specified as a value of the key-value pair `_ISCC_TRACKING_NUMBER_` in the `Extensions` attribute of the original client request.

The DNIS matching can be based on any number of digits out of all the digits that comprise the `DNIS` attribute. The number of digits that T-Server should use for DNIS matching is specified for the destination switch as the `ISCC Protocol Parameters` property of the Switch Access Code. The value syntax should be as follows:

```
dnis-tail=<number-of-digits>
```

For example, if this property is set to the `dnis-tail=7` value, ISCC matches only the last seven digits of a `DNIS`.

You must configure DNIS access resources in the switch; otherwise, ISCC fails to use this transaction type and sends `EventError` in response to the client application request.

Note: The `dnis-pool` transaction type is typically used for networks employing a "behind the SCP" architecture—network IVR. Network T-Server for GenSpec and IServer are two examples of this, but other Network T-Servers might also be used in this architecture.

In Load-Balancing Mode

When T-Server uses load balancing for call routing with the `dnis-pool` transaction type, the following processes occur:

1. A client of the origination T-Server sends a request to pass a call to the location with a DNIS access resource specified in the key-value pair `iscc-selected-dnis`.
2. The origination T-Server distributes the request for a routing service to all destination T-Servers.
3. The destination T-Servers receive the request and check that the specified DNIS is not being used by another routing service request.
4. The origination T-Server expects to receive a positive response from each destination T-Server. If the origination T-Server receives a negative response from at least one T-Server, it sends an `EventError` to the client and clears all data about the request. If the origination T-Server receives the confirmation about routing service availability from all destination T-Servers, it processes the client's request and sends a corresponding message to the switch.
5. The origination switch processes the T-Server request and passes the call to the destination switch.
6. The call arrives at the destination switch, which generates an alerting event to one of the corresponding load-balanced destination T-Servers.
7. That destination T-Server processes the call and notifies the origination T-Server that the routing service was successful and deletes all information about the request.
8. The origination T-Server sends a routing service request cancellation to all other destination T-Servers.
9. The origination T-Server notifies the client that the routing service has been successful and deletes all information about the request.

pullback

`Pullback` is used in the following scenario, for those T-Servers that support it:

1. A call arrives at Site A served by a Network T-Server.
2. At Site A, a Network T-Server client requests to pass the call by means of ISCC routing to Site B served by a premise T-Server. Any transaction type except `reroute` or `pullback` can be specified in this request.
3. The call arrives at Site B and is either answered by an agent or delivered to a routing point.
4. A client of the premise T-Server at Site B sends a `TRouteCall`, `TSingleStepTransfer`, or `TGetAccessNumber` request to transfer the call to the network.
5. The Site B premise T-Server notifies the Network T-Server about this request.
6. The network T-Server receives the notification and issues an `EventRouteRequest` to obtain a new destination.
7. After receiving the new destination information, the Network T-Server disconnects the call from its current premise location at Site B and attempts to route the call to the new destination.
8. The Site B premise T-Server stops tracking the call, which has disconnected from the premise's agent DN or routing point and is delivered to the network.
9. The network T-Server completes routing the call to its new destination.

Note: The transaction type `pullback` can be used only to return a call from a premise T-Server to the Network T-Server that serves the site from which the call was previously transferred.

reroute

Only Network T-Servers use the transaction type `reroute`, and only in the following scenario:

1. A call arrives at Site A served by a Network T-Server.
2. At site A, a Network T-Server client requests to pass the call by means of ISCC to Site B served by a premise T-Server. Any transaction type except `reroute` or `pullback` can be specified in this request.
3. An agent at Site B answers the call.
4. A client of the premise T-Server at Site B sends a `TSingleStepTransfer` or `TRouteCall` request to transfer the call elsewhere (to a PSTN, to an agent, or to a routing point).

5. The Site B premise T-Server notifies the Network T-Server about this request and releases the call leg that resides at the agent's phone (using `TReleaseCall`) or at the Routing Point (using `TRouteCall` with the parameter `RouteTypeCallDisconnect`).
6. The Network T-Server receives the notification and reroutes the call to the requested destination—that is, it sends `EventRouteRequest` and attaches the call's user data.

Notes:

- The transaction type `reroute` can be used only to return a call from a premise T-Server to the Network T-Server that serves the site from which the call was previously transferred.
- To perform multi-site operations that are initiated with `TRouteCall` and for which the `reroute` transaction type is requested, the origination T-Server must support the `RouteTypeCallDisconnect` subtype of `TRouteCall`.

route

With the transaction type `route` (aliased as route-notoken), a call from the origination location reaches a dedicated External Routing Point, and from there, it is routed to a destination DN.

To control configured External Routing Points, T-Server must register these DNs with the switch. Failure to register implies that the External Routing Point is not available for ISCC purposes. Client applications can register External Routing Points via T-Server for monitoring purposes only.

Point-to-Point (One-to-One)

In the Point-to-Point access mode, only one trunk line is used to access an External Routing Point (for example, VDN, CDN) at the destination site. See Figure 6.

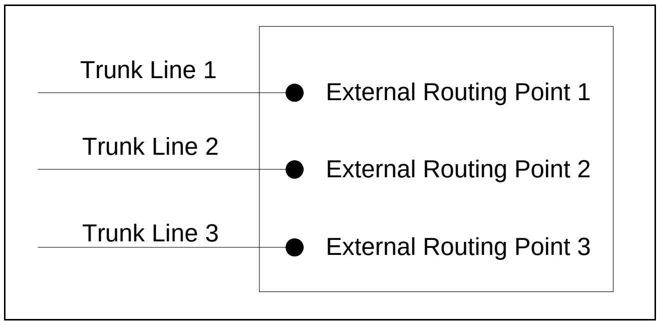


Figure 6: Point-to-Point Trunk Configuration

Note: Dedicated DNs of the `External Routing Point` type must be configured in a switch. See “Configuring Multi-Site Support” on page 83.

Multiple-to-Point (Multiple-to-One)

In the Multiple-to-Point access mode, trunk lines are assigned to the destination switch’s trunk group, from which calls are routed to the final destination. See Figure 7.

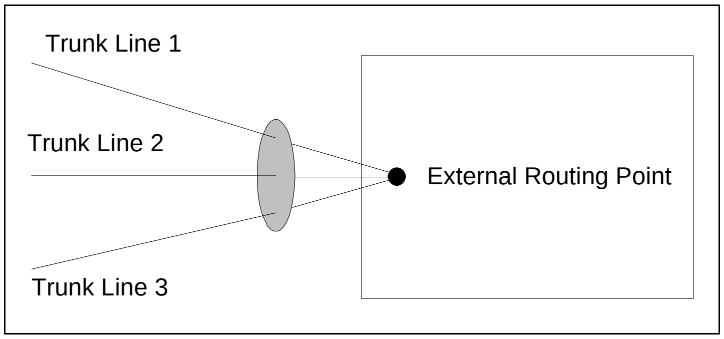


Figure 7: Multiple-to-Point Trunk Configuration

With this configuration, all calls reach the same External Routing Point. The `DNIS` attribute of a specific call differs from that of other calls and uniquely identifies the trunk from which the call arrived.

Note: To switch to this operating mode, you must configure the `route-dn` configuration option for T-Server.

route-uui

The `route-uui` transaction type employs the dedicated External Routing Point feature of the `route` transaction type (page 61) and the UUI matching feature of the `direct-uui` transaction type (page 57). This transaction type accommodates those switches that require a designated External Routing Point even though they use UUI for tracking.

Note: To support this transaction type, you must configure your switches to pass the UUI provided by your T-Server. Moreover, the trunks involved must not drop this data.

T-Server Transaction Type Support

Table 3 shows which transaction types are supported by a specific T-Server. Use this table to determine the transaction types that are available for use with your T-Server. This applies both to the `cast-type` you specify in the configuration options for your T-Server, and to any client-designated route-type requests specified for transfers of calls. A blank table cell indicates that T-Server does not support a certain transaction type.

Table 3: T-Server Support of Transaction Types

T-Server Type	Transaction Type										
	route		re-route	direct-callid	direct-uui / route-uui	direct-no-token	direct-ani	direct-digits	direct-network-callid	dnis-pool	pull-back
	one-to-one	multiple-to-one									
Aastra Matra Nexpan 50	Yes			Yes		Yes	Yes				
Alcatel A4200/OXO	Yes			Yes		Yes	Yes				
Alcatel A4400/OXE	Yes			Yes[a,b,c]	Yes[d]	Yes	Yes[a]		Yes[e]		
Aspect ACD	Yes	Yes		Yes		Yes[f]	Yes[f]				
Avaya Communication Manager	Yes			No	Yes	Yes	Yes	Yes[g]			
Avaya INDeX	Yes			Yes		Yes	Yes				
Cisco CallManager	Yes			Yes		Yes	Yes				
DataVoice Dharma	Yes			Yes		Yes	Yes				
Digitro AXS/20	Yes			Yes		Yes					
EADS Intecom M6880	Yes			Yes		Yes	Yes				
eOn eQueue	Yes			Yes		Yes					

Table 3: T-Server Support of Transaction Types (Continued)

T-Server Type	Transaction Type										
	route		re-route	direct-callid	direct-uui / route-uui	direct-no-token	direct-ani	direct-digits	direct-network-callid	dnis-pool	pull-back
	one-to-one	multiple-to-one									
Ericsson MD110	Yes			Yes[a]		Yes	Yes[a]				
Fujitsu F9600	Yes					Yes					
Huawei C&C08	Yes			Yes							
Mitel SX-2000/MN3300	Yes			Yes		Yes	Yes				
NEC NEAX/APEX	Yes			Yes		Yes	Yes				
Nortel Communication Server 2000/2100	Yes			Yes[f]		Yes[f]	Yes[f]				
Nortel Communication Server 1000 with SCCS/MLS	Yes			Yes		Yes	Yes		Yes		
Philips Sopho iS3000	Yes			Yes		Yes	Yes				
Radvision iContact	Yes		Yes								Yes
Rockwell Spectrum	Yes	Yes		Yes		Yes[f]	Yes[f]				
Samsung IP-PCX IAP	Yes			Yes		Yes					
Siemens Hicom 300/HiPath 4000 CSTA I	Yes			Yes	Yes[b]	Yes	Yes				

Table 3: T-Server Support of Transaction Types (Continued)

T-Server Type	Transaction Type										
	route		re-route	direct-callid	direct-uui / route-uui	direct-no-token	direct-ani	direct-digits	direct-network-callid	dnis-pool	pull-back
	one-to-one	multiple-to-one									
Siemens HiPath 3000	Yes			Yes		Yes					
Siemens HiPath 4000 CSTA III	Yes			Yes	Yes[b]	Yes	Yes				
Siemens HiPath DX	Yes			Yes	Yes	Yes	Yes				
SIP Server	Yes		Yes		Yes	Yes					Yes
Tadiran Coral	Yes			Yes		Yes	Yes				
Teltronics 20-20	Yes			Yes		Yes	Yes				
Tenovis Integral 33/55	Yes			Yes		Yes	Yes				
Network T-Servers											
AT&T											
Concert											
CRSP											Yes
DTAG			Yes								
GenSpec	Yes	Yes	Yes							Yes	
IVR Server, using network configuration	Yes	Yes	Yes							Yes	Yes
KPN			Yes								
ISCP											
MCI											
NGSN	Yes										Yes

Table 3: T-Server Support of Transaction Types (Continued)

T-Server Type	Transaction Type										
	route		re-route	direct-callid	direct-uui / route-uui	direct-no-token	direct-ani	direct-digits	direct-network-callid	dnis-pool	pull-back
	one-to-one	multiple-to-one									
Network SIP Server	Yes					Yes	Yes			Yes	
Sprint	Yes										
SR-3511											
Stentor											

a. Not supported in the case of function `TRequestRouteCall` on a virtual routing point: a routing point can be simulated using a hunt group with calls being deflected or transferred from the hunt-group member when routing. When a two-step (typically mute) transfer is used on such a hunt-group member, `CallID` and `ANI` usually change; thus, the `direct-callid` and `direct-ani` types do not work.

b. Not supported in the case of function `TSingleStepTransfer` when the T-Server service is simulated using a two-step transfer to the switch. In this case, `CallID` and `ANI` change; thus, the `direct-callid` and `direct-ani` types do not work.

c. Not supported if two T-Servers are connected to different nodes.

d. There are some switch-specific limitations when assigning CSTA correlator data UUI to a call.

e. Supported only on ABCF trunks (Alcatel internal network).

f. To use this transaction type, you must select the `Use Override` check box on the `Advanced` tab of the DN `Properties` dialog box.

g. Supported only for the DEFINITY 5ESS edition.

Transfer Connect Service Feature

The Transfer Connect Service (TCS) feature supports transfer connect services available on some telephony networks. When this feature is enabled, ISCC passes user data to remote locations to which calls are transferred or conferenced using transfer connect services.

To activate the TCS feature, set the `tcs-use` configuration option to `always`, and set the `tcs-queue` configuration option to the number of a DN on the origination switch. ISCC uses this DN as an intermediate step when sending calls to the remote location. The DN that is configured as `tcs-queue` receives attached data indicating the Feature Access Code (FAC) needed to reach the remote site. After a call is directed to the DN with data, a monitoring application takes the data and generates the required DTMF (dual-tone multifrequency) tones to redirect the call through the network to the remote location.

Note: With T-Server for Avaya Communication Manager, you can use `RequestRouteCall` with `RouteTypeOverwriteDNIS` to initiate the playing of DTMF tones. This is done through the use of another intermediate DN (typically, an announcement port configured to give the silence treatment), to which the call is routed. When the call is established on this DN, T-Server requests that the digits sent in the `DNIS` field of the `TRequestRouteCall` be played via the `ASAI-send-DTMF-single` procedure.

ISCC/COF Feature

The Inter Server Call Control/Call Overflow (ISCC/COF) feature of T-Server, that supports passive external routing, is specifically designed to handle calls delivered between sites by means other than ISCC. Such scenarios include contact center overflows and manual call transfers.

An *overflow situation* occurs when a call comes into a contact center where all agents are currently busy. In this situation, the switch can transfer (overflow) the incoming call to another site where there is an available agent.

T-Server uses two methods to handle call overflow and manual transfer scenarios. The first method is based on `NetworkCallID` matching and the second method is based on `ANI/OtherDN` matching.

When connected to each other via switch-specific networks, switches of some types can pass additional information along with transferred calls. This information may contain the `NetworkCallID` of a call, which is a networkwide unique identifier of the call.

When connected via a regular PSTN, switches of all types can send the `ANI` and/or `OtherDN` attributes to the destination switch during any call transfer operation.

While all T-Servers support the ISCC/COF feature using the `ANI` and/or `OtherDN` attributes, only a few support this feature using the `NetworkCallID` attribute. Table 4 shows the switches that provide the `NetworkCallID` of a call.

Table 4: T-Server Support of NetworkCallID for ISCC/COF Feature

T-Server Type	Supported NetworkCallID Attribute
Alcatel A4400/OXE	Yes
Aspect ACD	Yes

Table 4: T-Server Support of NetworkCallID for ISCC/COF Feature (Continued)

T-Server Type	Supported NetworkCallID Attribute
Avaya Communication Manager	Yes
Nortel Communication Server 2000/2100	Yes
Nortel Communication Server 1000 with SCCS/MLS	Yes
Rockwell Spectrum	Yes

The ISCC/COF feature can use any of the three attributes (`NetworkCallID`, `ANI`, or `OtherDN`) as criteria for matching the arriving call with an existing call at another location. Consequently, the attribute that is used determines what `ConnID`, `UserData`, and `CallHistory` are received for the matched call from the call's previous location.

Warning! Depending on the switch platform, it is possible to inherit the ANI attribute after routing a call to a remote destination, and after performing a Single-Step Transfer and other telephone actions. However, ISCC/COF works properly only in scenarios where the `ANI` attribute on the destination T-Server is represented by exactly the same unique digit string as on the origination T-Server. Typically the `ANI` attribute represents the original call identifier (customer phone number), which guarantees that the attribute remains unique.

Note: When the ISCC/COF feature is in use, the Number Translation feature becomes active. See "Number Translation Feature" on page 71 for more information on the feature configuration.

ISCC/COF Call Flow

Figure 8 shows the sequence of steps that occur in an ISCC/COF scenario when a call is made or transferred by an agent at Site A to a DN at Site B, or when a call is overflowed from Site A to Site B.

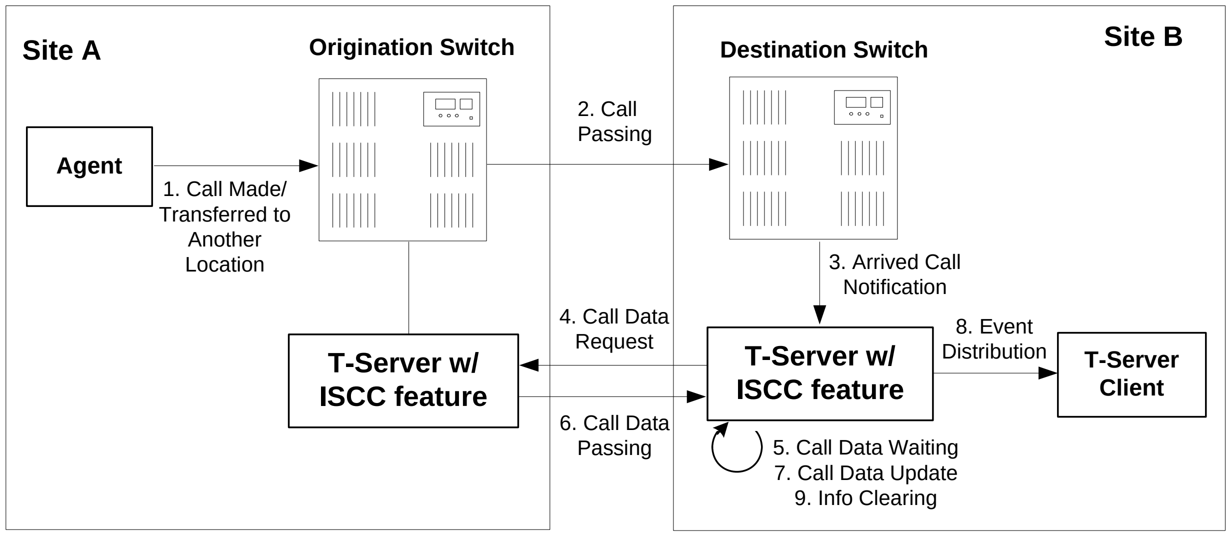


Figure 8: Steps in the ISCC/COF Process

Step 1

An agent makes or transfers a call manually to another location or a call is overflowed from Site A (origination location) to Site B (destination location).

Step 2

Switch A (the origination switch) passes the call to Switch B (the destination switch).

Step 3

The call reaches the destination switch, which notifies the destination T-Server about the arrived call.

Step 4

The destination T-Server verifies with remote locations whether the call was overflowed from any of them.

To determine which calls to check as possibly overflowed, T-Server relies on the `Switch` object configuration:

- If no COF DNs (that is, DNs of the `Access Resources` type with the `Resource Type` set to `cof-in` or `cof-not-in`) are configured for the destination switch, the ISCC/COF feature of the destination T-Server checks all arriving calls.
- If a number of COF DNs are configured for the destination switch, one of three scenarios occurs:

  - If the COF DNs with the `cof-in` setting for the `Resource Type` property are configured, the ISCC/COF checks for overflow only those calls that arrive to those `cof-in` DNs that are `Enabled`.
  - If no DNs with the `cof-in` setting for the `Resource Type` property are configured, but some DNs have the `cof-not-in` setting for the `Resource Type` property, the ISCC/COF checks for overflow only those calls that arrive to those `cof-not-in` DNs that are `Disabled`.
  - If no DNs with the `cof-in` setting for the `Resource Type` property are configured, some DNs have the `cof-not-in` setting for the `Resource Type` property, and some other DNs do not have any setting for the `Resource Type` property, the ISCC/COF checks for overflow only those calls that arrive to the DNs without any setting for the `Resource Type` property.
- In all other cases, no calls are checked for overflow.

To determine which location the call arrived from, T-Server checks the call type and checks whether the call has the `NetworkCallID`, `ANI`, or `OtherDN` attribute:

- If the call is not an inbound call, the request for call data is sent to all remote locations *except* those whose Switch Access Code has the ISCC Call Overflow `Parameters` property set to `inbound-only=true`.
- If the call of any type has the `NetworkCallID` attribute, the destination T-Server sends a request for call data to the remote locations of the same switch type as the destination location if their Switch Access Codes have the ISCC Call Overflow `Parameters` property set to `match-callid`.
- If the call of any type has the `ANI` or `OtherDN` attribute, the request for call data is sent to remote locations whose Switch Access Code has the ISCC Call Overflow `Parameters` property set to `match-ani`.

Step 5

The destination T-Server waits (suspending events related to that call) for the call data from the remote T-Server for the time interval specified in the `cof-ci-req-tout` configuration option. Within this interval, T-Server holds any events related to the call. In addition, the `cof-ci-defer-delete` option on the origination T-Server establishes the time interval only after which that T-Server deletes the call information. And the `cof-ci-wait-all`, if set to `true`, forces the origination T-Server to wait for responses related to possible call overflow situations before updating call data.

Step 6

The T-Server at the location from which the call was transferred or overflowed sends call data to the requesting T-Server.

Step 7

If a positive response to the call-data request is received, T-Server updates `ConnID`, `UserData`, and `CallHistory`, distributes all suspended events related to that call and deletes all information regarding the transaction (Step 9).

Step 8

If the timeout set by `cof-ci-req-tout` expires, T-Server distributes all suspended events, and starts the timeout specified by the `cof-rci-tout` option. If a positive response is received within the timeout set by `cof-rci-tout`, T-Server updates the `ConnID`, `UserData`, and `CallHistory` and notifies client applications by distributing `EventPartyChanged`.

Step 9

T-Server deletes all information regarding the transaction when one of these results occurs:

- The first positive response to the call-data request is received.
- Negative responses from all queried locations are received.
- The timeout specified by the `cof-rci-tout` option expires.

Number Translation Feature

The Number Translation feature of T-Server extends the ISCC/COF and `direct-ani` transaction type functions to provide more flexibility for handling calls distributed across multiple sites. T-Server translates the input string (ANI string) into a number defined by the translation rules. This processing is called number translation. T-Servers participating in handling calls at multiple sites exchange the translated numbers in order to match the call instances.

The translation process involves two algorithms, one for rule selection and the other for the actual translation. Through the first algorithm, T-Server selects a rule that will be used for number translation. Through the second algorithm, T-Server translates the number according to the selected rule definition. See "Number Translation Rules" on page 72 for more information on configuring rules for your environment.

Number translation occurs as follows:

1. The switch reports a number, typically via `AttributeANI`.
2. T-Server evaluates all configured inbound rules to determine which one is the best fit for the received number. The best fit is determined by comparing the length of, and the specific digits in, the input number with the inbound pattern of each configured rule. See "Rule Examples" on page 76 for specific examples.
3. T-Server translates the number according to the selected rule.

To enable T-Server to translate numbers, you must perform specific configuration tasks that are associated with translation. See “Configuration Procedure” on page 78.

Number Translation Rules

T-Server uses the number translation rules that you define in the T-Server configuration object in two ways:

- **Rule selection—**To determine which rule should be used for number translation
- **Number translation—**To transform the number according to the selected rule

Using ABNF for Rules

The number translation rules must conform to the following syntax, represented using Augmented Backus-Naur Form (ABNF) notation. For more information about ABNF, see RFC 2234, “Augmented BNF for Syntax Specifications: ABNF.”

Note: The notations are explained starting at the highest level, with the name of a component notation and a basic definition of each component that comprises it. Some components require more detailed definitions, which are included later in this section.

Common Syntax Notations

Syntax notations common to many of these rules include:

- `*`—Indicates that 0 to an infinite number of the item following this symbol are acceptable.
- `1*`—Indicates that one repetition is required. For T-Server, only one instance is acceptable.
- `/`—Indicates that any of the items mentioned, or a combination of those items, is acceptable.

Component Notations

Component notations include:

- `dialing-plan = *dialing-plan-rule`

  where:

  - `dialing-plan-rule` represents the name of the rule. Each rule must have a unique name. There are no other naming restrictions, and you do not need to model your names according to the examples in this chapter.

The rules are represented as separate options in the configuration. Also, fields from a rule are represented as parameters in a single option string.

- `rule = [name] in-pattern [out-pattern]`

  where:

  - `[name]` is the name for the rule option, for example, `rule-01`. In ABNF notation, the brackets `[]` indicate that 0 or 1 instance of the component is required. However, for T-Server, a name is required.
  - `in-pattern` is the part of the rule to which T-Server looks when attempting to match the input number.
  - `[out-pattern]` is the part of the rule that instructs T-Server on how to translate the input number into the required format. The brackets indicate that either 0 or 1 instance is required. You must create an `out-pattern` for number translation rules.
- `name = *( ALPHA / DIGIT / "-")`

  where:

  - `ALPHA` indicates that letters can be used in the name for the rule option.
  - `DIGIT` indicates that numbers can be used in the name for the rule option.
  - `"-"` indicates that a dash (-) can also be used in the option name, for example, `rule-01`.
- `in-pattern = 1*(digit-part / abstract-group)`

  where:

  - `digit-part` represents numbers. T-Server uses this when selecting the most appropriate rule from the entire dialing plan.
  - `abstract-group` represents one or more letters with each letter representing one or more numbers. T-Server uses this when transforming a dial string.

  For example, `[1-9]` is the `digit-part` (representing a range of numbers) and `ABBB` is the `abstract-group` for `in-pattern=[1-9]ABBB`.
- `out-pattern = 1*(symbol-part / group-identifier) *param-part`

  where:

  - `symbol-part` represents digits, symbols, or a combination. Symbols are rarely used. They are not used in the United States.
  - `group-identifier` are letters that represent groups of numbers. A letter in the `out-pattern` represents one or more digits, based on the number of times the letter is used in the `in-pattern`.
  - `*param-part` represents an additional parameter, such as `phone-context`. Reminder: an asterisk means that 0 to an infinite number of these are acceptable.

  For example, in `rule-04; in-pattern=1AAABBBCCC;out-pattern=91ABC, 91` is the `symbol-part`; `A, B,` and `C` are `group-identifiers` in the `out-pattern`, each representing three digits, since there are three instances of each in the `in-pattern`.

Note: Prefix an `out-pattern` value with a plus sign (+) for the inbound rule when the output must be in a global form (E.164 format).

- `digit-part = digits / range  / sequence`

  where:

  - `digits` are numbers 0 through 9.
  - `range` is a series of digits, for example, 1-3.
  - `sequence` is a set of digits.

- `symbol-part = digits / symbols`

  where:

  - `digits` are numbers 0 through 9.
  - `symbols` include such characters as `+, -,` and so on.

- `range = "[" digits "-"  digits "]" group-identifier`

  where:

  - `"[" digits "-"  digits "]"` represents the numeric range, for example, `[1-2]`.
  - `group-identifier` represents the group to which the number range is applied.

    For example, `[1-2]` applies to group identifier `A` for `in-pattern=[1-2]ABBB`. When T-Server evaluates the rule to determine if it matches the number, it examines whether the first digit of the number, identified as `group-identifier A`, is `1` or `2`.

- `sequence = "[" 1*(digits [","] ) "]" group-identifier`

  where:

  - `"[" 1*(digits [","] ) "]"` represents a sequence of digits, separated by commas, and bracketed. T-Server requires that each digit set have the same number of digits. For example, in `[415,650]` the sets have three digits.
  - `group-identifier` represents the group to which the number sequence is applied.

    For example, in `in-pattern=1[415,650]A*B, [415,650]` applies to `group-identifier A`. When T-Server evaluates the rule to determine if it matches the number, it examines whether the three digits (`group-identifier A`) following the 1 in the number are `415` or `650`.

- `abstract-group = fixed-length-group / flexible-length-group / entity`

  where:

  - `fixed-length-group` specifies a group composed of a specific number of digits and determined by how many times the group identifier is included in the `in-pattern`. For example, for `in-pattern=1AAABBBCCCC`, there are three digits in group `A` and `B` but four in group `C`.

    When you create an `out-pattern`, you include the group identifier only once because the `in-pattern` tells T-Server how many digits belong in

  that group. For example, `rule-04` (see page 77) is `in-pattern=1AAABBBCCCC; out-pattern=91ABC`.
  - `flexible-length-group` specifies a group composed of 0 or more digits in the group represented by the `group-identifier`. For example, in `in-pattern=1[415,650]A*B`, `*B` represents the flexible length group containing the remaining digits in the number.
  - `entity` represents digits defined for a specific purpose, for example, country code.

  The component `abstract-group` is used only for the `in-pattern`.

- `fixed-length-group = 1*group-identifier`

  See the earlier explanation under `abstract-group`.

- `flexible-length-group = "*"  group-identifier`

  See the earlier explanation under `abstract-group`.

- `entity = "#" entity-identifier  group-identifier`

  where:
  - `"#"` indicates the start of a Country Code `entity-identifier`.
  - `entity-identifier` must be the letter `C` which represents Country Code when preceded by a pound symbol (`#`). Any other letter following the # causes an error.
  - `group-identifier` represents the Country Code group when preceded by `#C`.

  The `entity` component is a special group that assumes some kind of predefined processing, such as the Country Code detection.

- `param-part = ";" param-name "=" param-value`

  where:
  - `";"` is a required separator element.
  - `param-name` is the name of the parameter.
  - `"="` is the next required element.
  - `param-value` represents the value for `param-name`.

- `param-name = "ext" / "phone-context" / "dn"`

  where:
  - `"ext"` refers to extension.
  - `"phone-context"` represents the value of the `phone-context` option configured on the switch.
  - `"dn"` represents the directory number.

- `param-value = 1*ANYSYMBOL`

  where:
  - `ANYSYMBOL` represents any number, letter, or symbol with no restrictions.

- `group-identifier = ALPHA`
- `entity-identifier = ALPHA`
- `digits = 1*DIGIT`
- `symbols = 1*("-" / "+" / ")" / "(" / ".")`

Recommendations for Rule Configuration

The configuration of rules for inbound numbers usually depends on the settings in the corresponding PBX. These settings often define the form in which the PBX notifies its client applications about the number from which an inbound call is coming.

As a general guideline, configure rules that define how to process calls from:

- Internal numbers.
- External numbers within the same local dialing area.
- External numbers within the same country.
- International numbers.

Rules for inbound numbers, typically for North American locations, might look like this:

1. Two rules to transform internal numbers (extensions):
   ```
   name=rule-01;in-pattern=[1-9]ABBB;out-pattern=AB
   name=rule-02;in-pattern=[1-9]ABBBB;out-pattern=AB
   ```
2. A rule to transform local area code numbers (in `333-1234` format in this example):
   ```
   name=rule-03;in-pattern=[1-9]ABBBBBB;out-pattern=+1222AB
   ```
3. A rule to transform U.S. numbers (in `+1(222)333-4444` format):
   ```
   name=rule-04;in-pattern=1AAAAAAAAAA;out-pattern=+1A
   ```
4. A rule to transform U.S. numbers without the `+1` prefix (in `(222)333-4444` format):
   ```
   name=rule-05;in-pattern=[2-9]ABBBBBBBBB;out-pattern=+1AB
   ```
5. A rule to transform U.S. numbers with an outside prefix (in `9 +1(222)333-4444` format):
   ```
   name=rule-06;in-pattern=91AAAAAAAAAA;out-pattern=+1A
   ```
6. A rule to transform international numbers with an IDD (international dialing digits) prefix (in `011 +44(111)222-3333` format):
   ```
   name=rule-07;in-pattern=011*A;out-pattern=+A
   ```
7. A rule to transform international numbers without an IDD prefix (in `+44(111)222-3333` format)
   ```
   name=rule-08;in-pattern=[2-9]A*B;out-pattern=+AB
   ```

Rule Examples

This section provides examples of six rules that are configured as options in the Genesys Configuration Database. It also provides examples of how T-Server applies rules to various input numbers.

Rules

rule-01 `in-pattern=[1-8]ABBB;out-pattern=AB`

rule-02 `in-pattern=AAAA;out-pattern=A`

rule-03 `in-pattern=1[415,650]A*B;out-pattern=B`

rule-04 `in-pattern=1AAABBBCCCC;out-pattern=91ABC`

rule-05 `in-pattern=*A913BBBB;out-pattern=80407913B`

rule-06 `in-pattern=011#CA*B;out-pattern=9011AB`

Example 1 T-Server receives input number `2326`.

As a result of the rule selection process, T-Server determines that the matching rule is `rule-01`:

```
name=rule-01;in-pattern=[1-8]ABBB;out-pattern=AB
```

The matching count for this rule is `1`, because Group `A` matches the digit `2`.

As a result of the parsing process, T-Server detects two groups: Group `A = 2` and Group `B = 326`.

T-Server formats the output string as `2326`.

Example 2 T-Server receives input number `9122`.

As a result of the rule selection process, T-Server determines that the matching rule is `rule-02`:

```
name=rule-02;in-pattern=AAAA;out-pattern=A
```

The matching count for this rule is `0`; however, the overall length of the input number matches that of the `in-pattern` configuration.

As a result of the parsing process, T-Server detects one group: Group `A = 9122`.

T-Server formats the output string as `9122`.

Example 3 T-Server receives input number `16503222332`.

As a result of the rule selection process, T-Server determines that the matching rule is `rule-03`:

```
name=rule-03;in-pattern=1[415,650]A*B;out-pattern=B
```

The matching count for this rule is `4`, because the first digit matches and all three digits in Group `A` match.

As a result of the parsing process, T-Server detects two groups: Group `A = 650` and Group `B = 3222332`.

T-Server formats the output string as `3222332`.

Example 4 T-Server receives input number `19253227676`.

As a result of the rule selection process, T-Server determines that the matching rule is `rule-04`:

```
name=rule-04;in-pattern=1AAABBBCCCC;out-pattern=91ABC
```

The matching count for this rule is `1`, because the first digit matches.

As a result of parsing process, T-Server detects three groups: Group `A = 925`, Group `B = 322`, and Group `C = 7676`.

T-Server formats the output string as `919253227676`.

Example 5 T-Server receives input number `4089137676`.

As a result of rule selection process, T-Server determines that the matching rule is `rule-05`:

`name=rule-05;in-pattern=*A913BBBB;out-pattern=80407913B`

The matching count for this rule is `3`, because three digits match.

As a result of the parsing process, T-Server detects two groups: Group `A = 408` and Group `B = 7676`.

T-Server formats the output string as `804079137676`.

Example 6 T-Server receives input number `011441112223333`.

As a result of the rule selection process, T-Server determines that the matching rule is `rule-06`:

`name=rule-06;in-pattern=011#CA*B;out-pattern=9011AB`

The matching count for this rule is `3`, because three digits match.

As a result of the parsing process, T-Server detects two groups: Group `A = 44` and Group `B = 1112223333`.

T-Server formats the output string as `9011441112223333`.

Configuration Procedure

The Number Translation feature becomes active when the ISCC/COF feature and/or the `direct-ani` transaction type are used.

The following configuration procedure must be completed within the T-Server `Application` object corresponding to your T-Server:

1. Open the T-Server `Application's Properties` dialog box.
2. Click the `Options` tab. Create a new section called `extrouter` or open an existing section with this name.
3. Create a new option called `inbound-translator-<n>`. This option points to another section that describes the translation rules for inbound numbers.
4. In this section, create one configuration option for each rule. Specify the rule name as the option name. The values of these options are the rules for the number translation. For the option description and its valid values, see Chapter 9, “T-Server Common Configuration Options,” on page 149.
5. When you are finished, click `Apply`.
6. Click `OK` to save your changes and exit the `Properties` dialog box.

Network Attended Transfer/ Conference Feature

The Network Attended Transfer/Conference (NAT/C) feature is designed to enable agents working in multi-site contact centers to consult with each other before making call transfers or conferences, regardless of whether both agents work at the same or different sites. It also enables the agent who requests a consultation to maintain his or her conversation with the customer while the system is looking for an available agent and setting up the consultation call.

The NAT/C feature does not rely on the call transfer capabilities of the local switch.

There are two modes in which the network attended transfer/conference can be performed: *direct* and *URS-controlled*. Figure 9 shows the sequence of steps that occur in *URS-controlled* mode, when Agent A, who is handling a customer call, requests a consultation with another agent, and URS (Universal Routing Server) selects Agent B, who is working at another site. The *direct* mode is similar to the *URS-controlled* mode, with the difference that URS is not involved in the process (Step 2 and Step 3 are omitted).

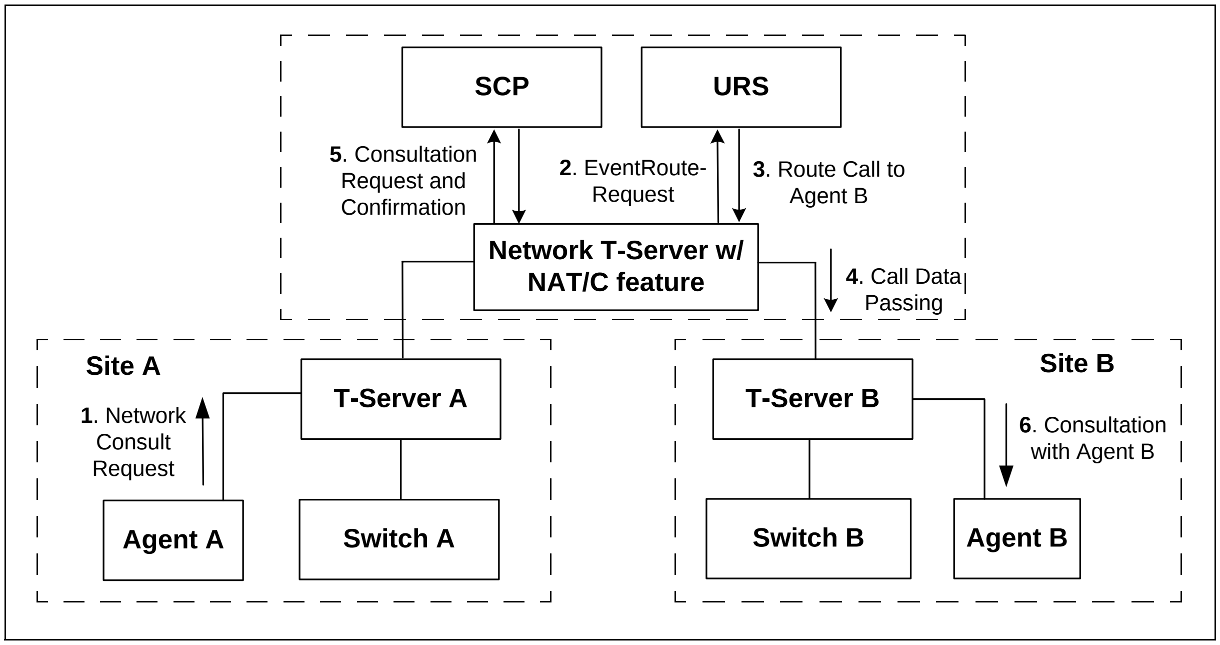


Figure 9: Steps in the NAT/C Process in URS-Controlled Mode

Step 1

Agent A makes a request for a consultation with another agent. A `TNetworkConsult` request is relayed to the Network T-Server. Depending on the parameter settings of the `TNetworkConsult` request, the NAT/C feature will operate in either *direct* or *URS-controlled* mode. For more information, see the *Voice Platform SDK 7.5 .NET* (or *Java) API Reference.*

Step 2

(*URS-controlled* mode only.) The Network T-Server sends `EventRouteRequest` to URS.

Step 3

(*URS-controlled* mode only.) URS locates an available agent at Site B and instructs the Network T-Server to route the call to Agent B. The Network T-Server confirms the initiation of the network transfer by sending `EventNetworkCallStatus` to T-Server A, which then relays it to Agent A.

Step 4

The Network T-Server proceeds to obtain the access number from T-Server B, and passes the call data to T-Server B. (See "ISCC Call Data Transfer Service" on page 49 for details.)

Step 5

The Network T-Server instructs the Service Control Point (SCP) to initiate a new voice path with Agent B. Once the connection is confirmed, the Network T-Server distributes `EventNetworkCallStatus` to both T-Server A and T-Server B, which then relay it to Agent A and Agent B respectively, to indicate that the consultation call is being established.

The Network T-Server also distributes `EventRouteUsed` to URS to confirm successful routing of the call to the selected agent.

Step 6

At this point, the customer is on hold, and Agent A is consulting with Agent B. Agent A can do one of the following:

- End the consultation and retrieve the original customer call
- Alternate between Agent B and the customer
- Set up a conference call with Agent B and the customer
- Transfer the customer call to Agent B

Note: All T-Servers support NAT/C requests with `AttributeHomeLocation` provided that this attribute identifies a network location that is capable of processing such requests. Refer to the *Network T-Server Deployment Guides* to determine whether a specific Network T-Server can process these requests.

Event Propagation Feature

The Event Propagation feature complements the ISCC and ISCC/COF features by distributing updated user data and party-related events to remote T-Servers. This feature is used when a call is being made, transferred, or conferenced to another location, and when, as a result, one or more instances of the call reside at one location while other call instances reside at another location. In this scenario, when a client at one location makes changes to user data, updated user data is passed *(propagated)* to T-Servers at other locations.

The Event Propagation feature consists of User Data update propagation and Party Events propagation.

To enable the Event Propagation feature of your T-Server, you must set the `event-propagation` configuration option to the `list` value. To enable the Event Propagation feature to also distribute party events, you must set the `use-data-from` configuration option to the `consult-user-data` value. (See “Activating Event Propagation” on page 92 and “T-Server Common Configuration Options” on page 149.)

If one of the T-Servers along the call distribution path has the Event Propagation feature disabled, that T-Server does not distribute events to remote locations.

User Data Propagation

User data propagation takes place when a client at one location makes changes to user data associated with a call that was made, transferred, conferenced, or routed to other locations. The remote clients involved with the call are notified about the changes with `EventAttachedDataChanged`.

When T-Server receives a local update to user data (that is, when a client of this T-Server has changed the call’s user data), T-Server determines if parties at remote locations are involved with the call and, if so, sends (propagates) the updated user data to the T-Servers at remote locations.

When T-Server receives a remote update to user data (that is, when a client of a remote T-Server has changed the call’s user data and the remote T-Server has used the Event Propagation feature to send the updated user data), T-Server:

1. Updates the user data of the corresponding local call.
2. Determines if parties at other remote locations are involved with the call and, if so, propagates the updated user data to T-Servers at other remote locations.

The locations to which user data is propagated are selected based on a call distribution topology. That is, the updated user data is passed directly to the location to which a call was sent and to the location from which the call was received, excluding the location from which the update was received.

For example, consider a call made from location A to location B, and then conferenced from location B to location C. The three instances of the call reside at different locations: the first instance is at location A, the second instance is at location B, and the third instance is at location C. The Event Propagation feature is employed in the following scenarios:

- When T-Server at location A receives a local update to user data, it notifies T-Server at location B (to which it sent the call) about changes to the call's user data. Thus, T-Server at location B receives a remote update to user data and, in turn, notifies T-Server at location C (to which it sent the call) about these changes.

  Although T-Server at location C receives a remote update to user data, it does not pass the notification to any other T-Servers, because it did not send the call to any other locations. As mentioned earlier, T-Servers at locations B and C update the user data of the corresponding local calls and notify their clients about the changes with `EventAttachedDataChanged`.
- When T-Server at location B receives a local update to user data, it notifies T-Server at location C (to which it sent the call) and T-Server at location A (from which it received the call) about changes to the call's user data. Thus, T-Servers at locations C and A receive a remote update to user data.

  Because T-Server at location C did not send the call to any other locations, and T-Server at location A originated the call, neither of these T-Servers passes the notification to any other T-Servers. T-Servers at locations C and A update the user data of the corresponding local calls and notify their clients about the changes with `EventAttachedDataChanged`.
- When T-Server at location C receives a local update to user data, it notifies T-Server at location B (from which it received the call) about changes to the call's user data. Thus, T-Server at location B receives a remote update to user data and, in turn, notifies T-Server at location A (from which it received the call) about these changes.

  Although T-Server at location A receives a remote update to user data, it does not pass the notification to any other T-Servers, because it originated the call. T-Servers at locations B and A update the user data of the corresponding local calls and notify their clients about the changes with `EventAttachedDataChanged`.

When a call is distributed between location A and location C using location B, and is then deleted on location B, propagation between locations A and C still occurs through the transit node at location B.

Party Events Propagation

Party events propagation takes place when a transfer or a conference is completed for a call that was made to or from one or more remote locations, or when a conference party is removed from the conference.

In these cases, the Event Propagation feature distributes party events, such as `EventPartyChanged`, `EventPartyAdded`, and `EventPartyDeleted`, to remote locations involved with the call, according to appropriate call model scenarios.

For example, consider a call made from DN 1 to DN 2 on location A. A `TInitiateConference` request is then issued for DN 2 to transfer the call to external DN 3 on location B. That transfer is made by means of ISCC routing. When this conference is completed on location A, the Event Propagation feature sends `EventPartyChanged` to location B and distributes this event to involved client applications that are connected to location B and registered for DN 3. After that, if a party of the conference is removed from the conference (for example, a party on DN 2), the Event Propagation feature sends `EventPartyDeleted` to location B and distributes this event to client applications registered for DN 3.

Warnings!

- The `OtherDN` and `ThirdPartyDN` attributes might not be present in the events distributed via the Event Propagation feature.
- The Event Propagation feature will not work properly with installations that use switch partitioning.

If a call involved in the propagation has no local parties but has two or more remote parties, the party events propagation is processed in the same manner as the propagation of user data updates.

For a complete event flow in such scenarios, refer to the *Genesys 7 Events and Models Reference Manual*.

Configuring Multi-Site Support

Prior to configuring T-Server to support multi-site operation, you must read the “Licensing Requirements” on page 29, as well as previous sections of this chapter on multi-site deployment. In particular, Table 3 on page 63 shows which transaction types are supported by a specific T-Server, while Table 4 on page 67 shows whether your T-Server supports the `NetworkCallID` attribute for the ISCC/COF feature. Use this information as you follow the instructions in this chapter.

Note: Before attempting to configure a multi-site environment, Genesys recommends that you plan the changes you want to make to your existing contact centers. You should then gather the configuration information you will need (such as the names of each T-Server application, port assignments, switch names, and so on), and use Configuration Manager to create and partially configure each T-Server object. Review multi-site option values in the "Multi-Site Support Section" on page 157 and determine what these values need to be, based on your network topology.

For T-Server to support multi-site operation, you must create and configure three types of objects in the Configuration Layer: Applications, Switches, including Access Codes, and DNs. You must configure these objects for origination and destination locations. Multi-site support features activate automatically at T-Server startup. See "Configuring DNs" on page 89 for details.

Applications

Ensure that T-Server Application objects, and their corresponding Host objects, exist and are configured for origination and destination locations.

Once you've done that, use Configuration Manager to add this configuration to a T-Server Application:

1. Open the T-Server Application's Properties dialog box.
2. Click the Connections tab and click Add to add a connection to the appropriate T-Server. The Connection Info Properties dialog box displays.
3. Use the Browse button to search for the T-Server you want to connect to, and fill in the following values:
   - Port ID
   - Connection Protocol
   - Local Timeout
   - Remote Timeout
   - Trace Mode
4. Click the Options tab. Create a new section called extrouter or open an existing section with this name.

   Note: If you do not create the extrouter section, T-Server works according to the default values of the corresponding configuration options.

5. Open the extrouter section. Configure the options used for multi-site support.

Note: For a list of options and valid values, see the "Multi-Site Support" section of the "T-Server Common Options" chapter in Part Two of this document.

6. When you are finished, click `Apply`.

Repeat this procedure for all T-Servers for origination and destination locations that are used for multi-site operations.

Switches

Ensure that `Switching Office` and `Switch` objects are configured for both origination and destination locations. You configure `Access Codes` to a destination switch in the origination `Switch's Properties` dialog box. The only exception is the `Default Access Code`, which is configured at the destination `Switch's Properties` dialog box.

You can configure two types of switch `Access Codes` in the `Switch's Properties` dialog box:

- A `Default Access Code` (for inbound calls)—Specifies the access code that other switches can use to access this switch when they originate a multi-site transaction.
- An `Access Code` (for outbound calls)—Specifies the access code that this switch can use when it originates a multi-site transaction to access another switch.

When the origination T-Server processes a multi-site transaction, it looks for an access code to the destination switch. First, T-Server checks the `Access Code` of the origination `Switch`:

- If an access code to the destination switch is configured with the target type `Target ISCC` and with any transaction type except `Forbidden`, T-Server uses this access code to dial the destination switch.
- If the access code to the destination switch is not configured on the `Access Code` tab of the origination switch, the origination T-Server checks the `Default Access Code` tab of the destination switch. If an access code is configured there with the target type `Target ISCC` and with any transaction type except `Forbidden`, T-Server uses this access code to dial the destination switch.
- If no access code with the required properties is found, T-Server rejects the transaction.

Configuring Default Access Codes

After you have configured `Switching Office` and `Switch` objects, follow this procedure to configure the `Default Access Codes` (one per `Switch` object):

1. Among configured `Switches`, select the `Switch` that the configured T-Server relates to.
2. Open the `Switch Properties` dialog box and click the `Default Access Codes` tab.
3. Click `Add` to open the `Access Code Properties` dialog box.
4. In the `Code` field, specify the access code used by remote switches to reach a DN at this switch. An access code is used as a prefix to the remote switch numbers.

> Note: If no prefix is needed to dial to the configured Switch, you can leave the `Code` field blank.

5. In the `Target Type` field, select `Target ISCC`.
6. In the `Route Type` field, select a value corresponding to the transaction type you want to use (given that it is supported for your switch type).

Configuring Access Codes

After you have configured `Switching Office` and `Switch` objects, follow this procedure to configure one or more `Access Codes`:

1. Among configured `Switches`, select the `Switch` that the configured T-Server relates to.
2. Open the `Switch Properties` dialog box and click the `Access Codes` tab.
3. Click `Add` to open the `Access Code Properties` dialog box.
4. In the `Switch` field, specify the switch that this switch can reach using this access code. Use the `Browse` button to locate the remote switch.
5. In the `Code` field, specify the access code used to reach a DN at the remote switch from this switch. An access code is used as a prefix to the remote switch numbers.

> Note: If no prefix is needed to dial from one switch to another, you can leave the `Code` field blank.

6. In the `Target Type` field, select `Target ISCC`.

7. In the `Route Type` field, select a value corresponding to the transaction type you want to use (given that it is supported for your switch type). Table 5 contains cross-reference information on transaction types that the Configuration Layer and T-Server use.

Table 5: Transaction Types

Route Type Field Value	ISCC Transaction Type
Default	The first value from the list of values specified in the `cast-type` option for the T-Server at the destination site
Direct	`direct-callid`
Direct ANI	`direct-ani`
Direct Digits	`direct-digits`
Direct DNIS and ANI	Reserved
Direct Network Call ID	`direct-network-callid`
Direct No Token	`direct-notoken`
Direct UUI	`direct-uui`
DNIS Pooling	`dnis-pooling`
Forbidden	External routing to this destination is not allowed
ISCC defined protocol	Reserved
PullBack	`pullback`
Re-Route	`reroute`
Route	`route`

After configuring a switch for multi-site support, proceed with the configuration of DNs assigned to this switch.

Configuring Extended Parameters in Access Codes

If you select `Target ISCC` as your target type, as specified in Step 6 above, the `Properties` dialog box changes its lower pane to the `Source` pane. It is here that you enter the extended parameters for your access codes, by specifying the `ISCC Protocol` and `ISCC Call Overflow Parameters`. To set these parameters, locate the two drop-down boxes that appear below the `Target Type` field in the `Source` pane of that `Properties` dialog box.

1. In the `ISCC Protocol Parameters` drop-down box, enter the appropriate ISCC Protocol parameter, as a comma-separated list of one or more of the following items:

`dnis-tail=<number-of-digits>`	Where `number of digits` is the number of significant DNIS digits used for call matching. `0` (zero) matches all digits.
`propagate=<yes, udata, party, no>`	Default is `yes`. For more information, see "Activating Event Propagation" on page 92.
`direct-network-callid=<>`	For configuration information, see Part Two of this document. (Use Table 3 on page 63 to determine if your T-Server supports the `direct-network-callid` transaction type.)

2. In the `ISCC Call Overflow Parameters` drop-down box, enter call overflow parameters, as a comma-separated list of one or more of the following items:

`match-callid`	Matches calls using network `CallID`.
`match-ani`	Matches calls using ANI.
`inbound-only=<boolean>`	Default is `true`. Setting `inbound-only` to `true` disables COF on consultation and outbound calls.

Compatibility Notes

When migrating from previous releases of T-Servers to 7.5, or when using T-Servers of different releases (including 7.5) in the same environment, keep in mind the following compatibility issues:

- The `Target External Routing Point` value of the `Target Type` field is obsolete and provided only for backward compatibility with T-Servers of releases 5.1 and 6.0. When two access codes for the same switch are configured, one with the `Target ISCC` target type and the other with the `Target External Routing Point` target type, T-Servers of releases 7.x, 6.5, and 6.1:
    - Use the `Target ISCC` access code for transactions with T-Servers of releases 7.x, 6.5, and 6.1.
    - Use the `Target External Routing Point` access code for transactions with T-Servers of releases 5.1 and 6.0.

  When the only access code configured for a switch has the `Target External Routing Point` target type, T-Server uses this access code for all transactions.
- When the `Target External Routing Point` value of the `Target Type` field is configured, you must set the `Route Type` field to one of the following:

- `Default` to enable the `route` transaction type
- `Label` to enable the `direct-ani` transaction type
- `Direct` to enable the `direct` transaction type

Note: The `direct` transaction type in releases 5.1 and 6.0 corresponds to the `direct-callid` transaction type in releases 6.1, 6.5, and 7.x.

- `UseExtProtocol` to enable the `direct-uui` transaction type
- `PostFeature` to enable the `reroute` transaction type

These values are fully compatible with the transaction types supported in T-Server release 5.1.

- For successful multi-site operations between any two locations served by release 5.1 T-Servers, identical `Route Type` values must be set in the `Switch's Access Code Properties` dialog boxes for both the origination and destination switches.

Configuring DNs

Use the procedures from this section to configure access resources for various transaction types.

Access Resources for the route Transaction Type

To use the transaction type `route`, you must configure dedicated DNs as follows:

1. Under a configured `Switch`, select the `DNs` folder. From the main menu, select `File > New > DN` to create a new `DN` object.
2. On the `General` tab of the DN's `Properties` dialog box, specify the number of the configured DN as the value of the `Number` field. This value must correspond to the Routing Point number on the switch.
3. Select `External Routing Point` as the value of the `Type` field.
4. If a dialable number for that Routing Point is different from its DN name, specify the number in the `Association` field.
5. Click the `Access Numbers` tab. Click `Add` and specify these access number parameters:
   - Origination switch.
   - Access number that must be dialed to reach this DN from the origination switch.

In determining an access number for the Routing Point, T-Server composes it of the values of the following properties (in the order listed):

1. Access number (if specified).

2. Switch access code from the switch of the origination party to the switch to which the Routing Point belongs, concatenated with its `Association` (if the `Association` value is specified).
3. Switch access code from the switch of the origination party to the switch to which the Routing Point belongs, concatenated with the number for the DN.
4. Default access code of the switch to which the Routing Point belongs, concatenated with its `Association` (if the `Association` value is specified).
5. Default access code of the switch to which the Routing Point belongs, concatenated with the number for the DN.

Note: If option `use-implicit-access-numbers` is set to `true`, the access number composed of switch access code and DN can be used for external transfers of calls originating at switches for which an access number is not specified.

Access Resources for the dnis-pool Transaction Type

To use the transaction type `dnis-pool`, you must configure dedicated DNs as follows:

1. Under a configured `Switch`, select the `DNs` folder. From the main menu, select `File > New > DN` to create a new `DN` object.
2. On the `General` tab of the DN's `Properties` dialog box, specify the number of the configured DN as the value of the `Number` field. This value must be a dialable number on the switch.
3. Select `Access Resource` as the `Type` field and type `dnis` as the value of the `Resource Type` field on the `Advanced` tab.
4. Click the `Access Numbers` tab. Click `Add` and specify these `Access Number` parameters:
   - Origination switch.
   - Access number that must be dialed to reach this DN from the origination switch.

An access number for the access resource is determined in the same manner as for the `route` access resource.

Access Resources for direct-* Transaction Types

You can use any configured DN as an access resource for the `direct-*` transaction types. (The `*` symbol stands for any of the following: `callid`, `uui`, `notoken`, `ani`, or `digits`.)

You can select the `Use Override` check box on the `Advanced` tab to indicate whether the override value should be used instead of the number value to dial to the DN. You must specify this value if the DN has a different DN name and dialable number. In fact, this value is required for T-Servers for some switch types—for example, Aspect ACD, Nortel Communication Server 2000/2100, and Spectrum.

Access Resources for ISCC/COF

Note: Use Table 4 on page 67 to determine if your T-Server supports the ISCC/COF feature.

To use the ISCC/COF feature, you must configure DNs as follows:

1. Under a configured `Switch`, select the `DNs` folder. From the main menu, select `File` > `New` > `DN` to create a new `DN` object.

   Note: The number of the access resource must match the name of a DN configured on the switch (usually, an ACD Queue) so that T-Server can determine if the calls arriving to this DN are overflowed calls.

2. On the `General` tab of the DN `Properties` dialog box, specify the number of the configured DN as the value for the `Number` field.
3. Select `Access Resource` as the value for the `Type` field.
4. Click `Apply`.
5. On the `Advanced` tab, type `cof-in` or `cof-not-in` as the value for the `Resource Type` field.

   Note: Calls coming to DNs with the `cof-not-in` value for the `Resource Type` are never considered to be overflowed.

Access Resources for Non-Unique ANI

The `non-unique-ani` resource type is used to block `direct-ani` and `COF/ani` from relaying on ANI when it matches configured/enabled resource digits. Using `non-unique-ani`, T-Server checks every ANI against a list of `non-unique-ani` resources.

To use the ISCC/COF feature, you must configure dedicated DNs as follows:

1. Under a configured `Switch`, select the `DNs` folder. From the main menu, select `File` > `New` > `DN` to create a new `DN` object.
2. On the `General` tab of the DN `Properties` dialog box, specify the ANI digits that need to be excluded from normal processing.
3. `Access Resource` as the value for the `Type` field.
4. On the `Advanced` tab, specify the `Resource Type` field as `non-unique-ani`.

Additional DN Configuration for Isolated Switch Partitioning

When using switch partitioning, identify DNs that belong to a particular partition and modify their configuration as follows:

1. Under a `Switch` object, select the `DNs` folder.
2. Open the `Properties` dialog box of a particular `DN`.
3. Click the `Annex` tab.
4. Create a new section named `TServer`.
5. Within that section, create a new option named `epn`. Set the option value to the partition name to which the DN belongs.
6. Repeat Steps 1–5 for all DNs, including DNs of the `External Routing Point` type, that belong to the same switch partition.

This configuration instructs T-Server to select an External Routing Point that has the same partition as the requested destination DN.

Note: When a target DN is not configured or has no configured partition name, T-Server allocates a DN of the `External Routing Point` type that belongs to any partition.

Activating Event Propagation

To activate the Event Propagation feature during ISCC transactions, modify the configuration of the `Switch` at the location where a T-Server client changes user data, as described in the following section.

When determining whether to notify other T-Servers of changes to user data, or to distribute party events, T-Server checks:

1. Call topology (what location a call came from and to what location the call was then transferred or conferenced).
2. Outbound parameters of the `Switch` this T-Server relates to (whether propagation parameters are configured for the access codes this switch uses to reach the switch at the location a call came from and the switch at the location to which the call was then transferred or conferenced).

Warning! The direction of user-data or party-events propagation does not necessarily match the direction of call distribution. Therefore, the access code used to deliver the call can differ from the access code used for the purpose of Event Propagation.

You can set Event Propagation parameters using:

- The `Default Access Code` properties of the `Switch` that receives an ISCC-routed call (the destination switch).

- The `Access Code` properties of the `Switch` that passes an ISCC-routed call (the origination switch).

Note: You can also use the value of the `event-propagation` configuration option in the `extrouter` section in T-Server `Application` object to enable Event Propagation. The option value has a higher priority than the `Switch` settings.

If you do not set up Event Propagation parameters for a given `Access Code`, T-Server uses corresponding settings configured for the `Default Access Code` of the destination switch.

The procedures for modifying `Default Access Codes` and `Access Codes` are very similar to each other:

1. Among configured `Switches`, select the `Switch` that the configured T-Server relates to.
2. Open the `Switch's Properties` dialog box and click either the `Default Access Codes` tab or the `Access Codes` tab.
3. Select a configured `Default Access Code` or configured `Access Code` and click `Edit`.

   Note: If no `Default Access Code` is configured, see page 86 for instructions. If no `Access Codes` are configured, see page 86 for instructions.

   ---

4. In the `Switch Access Code Properties` dialog box that opens, specify a value for the `ISCC Protocol Parameters` field as follows:
   - To enable distribution of both user data associated with the call and call-party–associated events[1], type:

     `propagate=yes`

     which is the default value.
   - To enable distribution of user data associated with the call and disable distribution of call-party–associated events, type:

     `propagate=udata`
   - To disable distribution of user data associated with the call and enable distribution of call-party–associated events, type:

     `propagate=party`
   - To disable distribution of both user data associated with the call and call-party–associated events, type:

     `propagate=no`

1. The following are call-party–associated events: `EventPartyChanged`, `EventPartyDeleted`, and `EventPartyAdded`.

5. Click `OK` to save configuration updates and close the `Switch Access Code Properties` dialog box.
6. Click `Apply` and `OK` to save configuration updates and close the `Switch Properties` dialog box.

Example 1

This section demonstrates the difference in how ISCC directs a call when you specify two different transaction types (`route` and `direct-ani`).

In this example, you configure an origination and a destination switch for Nortel Communication Server 2000/2100 (formerly DMS-100) as described in "Switches" on page 85. Set the `Access Code` field to `9`. Under the destination switch, configure a DN as described in "Access Resources for the route Transaction Type" on page 89. Set the DN `Number` field to `5001234567`. In addition, select the `Use Override` check box on the `Advanced` tab of this DN's `Properties` dialog box and enter `1234567` in the `Use Override` field.

Then, use a softphone application to register for this new DN with the destination T-Server and, therefore, with the switch. Finally, request to route a call from any DN at the origination switch to the destination DN you have just configured:

- If you are using the `route` ISCC transaction type, the client requests that T-Server deliver a call to a destination location using the DN number `5001234567`. ISCC requests that the switch dial one of the external routing points at the destination location, using the value either of the `Access Number` field or of the `Access Code` field, which is `9`, concatenated with the external routing point at the destination location. The call is routed to the DN number `5001234567`.
- If you are using the `direct-ani` ISCC transaction type, the client requests that T-Server deliver a call to a destination location using the DN number `1234567`, which is the `Use Override` value. ISCC requests that the switch dial `91234567`, which is a combination of the `Switch Access Code` value and the `Use Override` value. The destination T-Server is waiting for the call to directly arrive at DN number `5001234567`.

Example 2

This section demonstrates how to indicate which overflow methods a switch supports.

In this example, for T-Server to use ANI/OtherDN matching in call overflow and manual transfer scenarios, set the `ISCC Call Overflow Parameters` to

```
match-ani, inbound-only=true
```

when configuring `Switch Access Codes` as described on page 86.

With this setting, the switch's location is queried for call data each time the destination T-Server receives an inbound call with the `ANI` or `OtherDN` attribute.

For T-Server to use `NetworkCallID` matching in call overflow and manual transfer scenarios, set the `ISCC Call Overflow Parameters` to (for example)

```
match-callid, inbound-only=false
```

when configuring `Switch Access Codes` as described on page 86.

With this setting, the switch's location is queried for call data each time the destination T-Server receives a call of any type (including inbound) with the `NetworkCallID` attribute.

Next Steps

Continue with Chapter 4, "Start and Stop T-Server Components," on page 97 to test your configuration and installation.

GENESYS
AN ALCATEL-LUCENT COMPANY

Chapter

4 Start and Stop T-Server Components

This chapter describes methods for stopping and starting T-Server, focusing on manual startup for T-Server and HA Proxy for all switches. It includes these sections:

Introduction

You can start and stop Framework components using the Management Layer, a startup file, a manual procedure, or the Windows Services Manager.

With all these methods, command-line parameters are usually required for a server application in addition to an executable file name.

Common command-line parameters are as follows:

`-host`	The name of the host on which Configuration Server is running.
`-port`	The communication port that client applications must use to connect to Configuration Server.
`-app`	The exact name of an Application object as configured in the Configuration Database.

`-l`	The license address. Use for the server applications that check out technical licenses. Can be either of the following: • The full path to, and the exact name of, the license file used by an application. For example, `-l /opt/mlink/license/license.dat`. • The host name and port of the license server, as specified in the `SERVER` line of the license file, in the `port@host` format. For example, `-l 7260@ctiserver`. Note: Specifying the License Manager's host and port parameter eliminates the need to store a copy of a license file on all computers running licensed applications.
`-V`	The version of a Framework component. Note that specifying this parameter does not start an application, but returns its version number instead. You can use either uppercase or lowercase.
`-nco X/Y`	The Nonstop Operation feature is activated; `X` exceptions occurring within `Y` seconds do not cause an application to exit. If the specified number of exceptions is exceeded within the specified number of seconds, the application exits or, if so configured, the Management Layer restarts the application. If the `-nco` parameter is not specified, the default value of `6` exceptions handled in `10` seconds applies. To disable the Nonstop Operation feature, specify `-nco 0` when starting the application.
`-lmspath`	The full path to log messages files (the common file named `common.lms` and the application-specific file with the extension `*.lms`) that an application uses to generate log events. This parameter is used when the common and application-specific log message files are located in a directory other than the application's working directory, such as when the application's working directory differs from the directory to which the application is originally installed. Note that if the full path to the executable file is specified in the startup command-line (for instance, `c:\gcti\multiserver.exe`), the path specified for the executable file is used for locating the `*.lms` files, and the value of the `lmspath` parameter is ignored.

Note: In the command-line examples in this document, angle brackets indicate variables that must be replaced with appropriate values.

Starting and Stopping with the Management Layer

Before starting an application with the Management Layer, make sure the startup parameters of the application are correctly specified in the application's `Properties` dialog box in Configuration Manager. On the `Start Info` tab of the Application `Properties` dialog box:

- Specify the directory where the application is installed and/or is to run as the `Working Directory`.
- Specify the name of the executable file as the `command-line`.
- Specify command-line parameters as the `Command–Line Arguments`.

The command-line parameters common to Framework server components are described on page 97.

After its command-line parameters are correctly specified in the `Properties` dialog box, you can start and stop T-Server from Solution Control Interface (SCI), which is the graphical interface component of the Management Layer. (The starting procedure for SCI is described in the *Framework 7.5 Deployment Guide*.) *Framework 7.5 Solution Control Interface Help* provides complete instructions on starting and stopping applications.

You can also use the Management Layer to start a T-Server that has failed. To enable T-Server's autorestart functionality, select the corresponding check box in the `Application's Properties` dialog box.

Note that when you start (or restart) an application via the Management Layer, the application inherits environment variables from Local Control Agent (LCA), which executes the startup command. Therefore, you must also set the environment variables required by the application for the account that runs LCA.

Warning! *Stopping* an application via the Management Layer is not considered an application failure. Therefore, the Management Layer does not restart applications that it has stopped unless an appropriate alarm condition and alarm reaction are configured for these applications.

Starting with Startup Files

Startup files are files with the extension `run.sh` (on UNIX) or `startServer.bat` (on Windows), which installation scripts create and place into the applications' directories during the installations. These files are created for all Framework server applications except:

- Configuration Server (primary or backup) running on Windows.

- Backup Configuration Server running on UNIX.
- DB Server running on Windows.
- LCA running on either Windows or UNIX.

When using a startup file, verify that the startup parameters the installation script inserted in the startup file are correct. Use the following instructions for UNIX and Windows to start those application for which startup files are created. See the appropriate sections in “Starting Manually” on page 100 to identify which applications should be running for a particular application to start.

On UNIX

Go to the directory where an application is installed and type the following command line:

```
sh run.sh
```

On Windows

Double-click the `startServer.bat` icon in the directory where the application is installed. Or from the `MS-DOS` window, go to the directory where the application is installed and type the following command-line:

```
startServer.bat
```

Starting Manually

When starting an application manually, you must specify the startup parameters at the command prompt, whether you are starting on UNIX or Windows. At the command prompt, command-line parameters must follow the name of the executable file. On the `Shortcut` tab of the Program `Properties` dialog box, command-line parameters must also follow the name of the executable file.

The command-line parameters common to Framework server components are described on page 97.

If an `Application` object name, as configured in the Configuration Database, contains spaces (for example, `T-Server Nortel`), the `Application` name must be surrounded by quotation marks in the command-line:

```
-app "T-Server Nortel"
```

You must specify the rest of the command-line parameters as for any other application.

The following sections provide general instructions for starting HA Proxy and T-Server manually. Along with these instructions, refer to Table 6, which lists T-Servers and HA Proxy executable file names for supported switches for Windows and UNIX operating systems.

Table 6: T-Server and HA Proxy Executable Names

Switch Type	T-Server Executable File Name		HA Proxy Executable File Name	
	UNIX	Windows	UNIX	Windows
Aastra Matra Nexpan 50	m6500_server	m6500_server.exe	Not Applicable	
Alcatel A4200/OXO	a4200_server	a4200_server.exe	Not Applicable	
Alcatel A4400/OXE	a4400_server	a4400_server.exe	Not Applicable	
Aspect ACD	aspect_server	aspect_server.exe	Not Applicable	
Avaya Communication Manager	avayacm_server	avayacm_server.exe	Not Applicable[a]	
Avaya INDeX	Not Applicable	index_server.exe	Not Applicable	
Cisco CallManager	ciscocm_server	ciscocm_server.exe	Not Applicable	
DataVoice Dharma	Dharma_server	Dharma_server.exe	Not Applicable	
Digitro AXS/20	digitro_server	digitro_server.exe	Not Applicable	
EADS Intecom M6880	intecom_server	intecom_server.exe	Not Applicable	
eOn eQueue	eon_server	eon_server.exe	Not Applicable	
Ericsson MD110	md110_server	md110_server.exe	Not Applicable	
Fujitsu F9600	Not Applicable	F9600_server.exe	Not Applicable	
Huawei C&C08	cc08_server	cc08_server.exe	Not Applicable	
Mitel SX-2000/ MN 3300	SX2000_server	SX2000_server.exe	Not Applicable	
NEC NEAX/APEX	neax_server	neax_server.exe	Not Applicable	
Nortel Communication Server 2000/2100	dms_server	dms_server.exe	ha_proxy_dms	ha_proxy_dms.exe

Table 6: T-Server and HA Proxy Executable Names (Continued)

Switch Type	T-Server Executable File Name		HA Proxy Executable File Name	
	UNIX	Windows	UNIX	Windows
Nortel Communication Server 1000 with SCSS/MLS	succession_server	succession_server.exe	Not Applicable	
Philips Sopho iS3000	iS3000_server	iS3000_server.exe	ha_proxy_ iS3000	ha_proxy_ iS3000.exe
Radvision iContact	nts_server	nts_server.exe	Not Applicable	
Rockwell Spectrum	spectrum_server	spectrum_server.exe	Not Applicable	
Samsung IP-PCX IAP	samsung_server	samsung_server.exe	Not Applicable	
Siemens Hicom 300/ HiPath 400 CSTA I	rolmcb4_server	rolmcb4_server.exe	Not Applicable	
Siemens HiPath 3000	HiPath3000_server	HiPath3000_server.exe	Not Applicable	
Siemens HiPath 4000 CSTA III	HiPath4000_server	HiPath4000_server.exe	Not Applicable	
Siemens HiPath DX iCCL	RealitisDX-iCCL_server	RealitisDX-iCCL_ server.exe	Not Applicable	
SIP Server	sip_server	sip_server.exe	Not Applicable	
Tadiran Coral	Coral_server	Coral_server.exe	Not Applicable	
Teltronics 20-20	Teltronics2020_server	Teltronics2020_ server.exe	ha_proxy_ teltronics 2020	ha_proxy_ teltronics 2020.exe
Tenovis Integral 33/55	Tenovis_server	Tenovis_server.exe	Not Applicable	
Network T-Servers				
AT&T	nts_server	nts_server.exe	Not Applicable	
Concert	nts_server	nts_server.exe	Not Applicable	
CRSP	nts_server	nts_server.exe	Not Applicable	
DTAG	dtag_server	dtag_server.exe	Not Applicable	

Table 6: T-Server and HA Proxy Executable Names (Continued)

Switch Type	T-Server Executable File Name		HA Proxy Executable File Name	
	UNIX	Windows	UNIX	Windows
GenSpec	nts_server	nts_server.exe	Not Applicable	
ISCP	nts_server	nts_server.exe	Not Applicable	
IVR Server, using network configuration	nts_server	nts_server.exe	Not Applicable	
KPN	kpn_server	kpn_server.exe	Not Applicable	
MCI	mci800_server	mci800_server.exe	Not Applicable	
NGSN	nts_server	nts_server.exe	Not Applicable	
Network SIP Server	tsip_server	tsip_server.exe	Not Applicable	
Sprint	sprint_server	sprint_server.exe	Not Applicable	
SR3511	sr3511_server	sr3511_server.exe	Not Applicable	
Stentor	stentor_server	stentor_server.exe	Not Applicable	

a. For releases prior to 7.1, this T-Server has an HA Proxy available: `ha_proxy_g3tcp` (UNIX) or `ha_proxy_g3tcp.exe` (Windows).

HA Proxy

If you do not use HA Proxy in your Genesys implementation, proceed to "T-Server" on page 104.

If one or more HA Proxy components are required for the T-Server connection, start HA Proxy before starting T-Server.

Before starting HA Proxy, be sure that the following components are running:

- DB Server that provides access to the Configuration Database
- Configuration Server

The command-line parameters common to Framework server components are described on page 97.

On UNIX

Go to the directory where HA Proxy is installed and type the following command-line:

```
ha_proxy_<switch> -host <Configuration Server host>
-port <Configuration Server port> -app <HA Proxy Application>
```

Replace `ha_proxy_<switch>` with the correct HA Proxy executable name, which depends on the type of the switch used.

Table 6 on page 101 lists HA Proxy executable names for supported switches.

On Windows

Start HA Proxy from either the `Start` menu or the `MS-DOS` window. If using the `MS-DOS` window, go to the directory where HA Proxy is installed and type the following command-line:

```
ha_proxy_<switch>.exe -host <Configuration Server host> -port
<Configuration Server port> -app <HA Proxy Application>
```

Replace `ha_proxy_<switch>.exe` with the correct HA Proxy executable name, which depends on the type of the switch used. Table 6 on page 101 lists HA Proxy executable names for supported switches.

T-Server

Before starting T-Server, be sure that the following components are running:

- DB Server that provides access to the Configuration Database
- Configuration Server
- License Manager

Note: If an HA Proxy component is required for the T-Server connection, HA Proxy must be started before T-Server.

The command-line parameters common to Framework server components are described on page 97.

On UNIX

Go to the directory where T-Server is installed and type the following command-line:

```
<switch>_server -host <Configuration Server host>
-port <Configuration Server port> -app <T-Server Application>
-l <license address> -nco [X]/[Y]
```

Replace `<switch>_server` with the correct T-Server executable name, which depends on the type of the switch used. Table 6 on page 101 lists T-Server executable names for supported switches.

On Windows

Start T-Server from either the `Start` menu or the `MS-DOS` window. If using the `MS-DOS` window, go to the directory where T-Server is installed and type the following command-line:

```
<switch>_server.exe -host <Configuration Server host>
-port <Configuration Server port> -app <T-Server Application>
-l <license address> -nco [X]/[Y]
```

Replace `<switch>_server.exe` with the correct T-Server executable name, which depends on the type of the switch used. Table 6 on page 101 lists T-Server executable names for supported switches.

Verifying Successful Startup

After executing the startup command, you might want to check whether it was successful.

If you used the Management Layer to start either T-Server or HA Proxy, check whether Solution Control Interface displays `Started` or `Service Unavailable` status for the corresponding application. Refer to the “Troubleshooting” section of the *Framework 7.5 Management Layer User’s Guide* if the startup command does not result in either `Started` or `Service Unavailable` status for some period of time.

If you start your T-Server or HA Proxy with startup files or manually, and if you have configured logging to console or a log file, check the log for messages similar to the following:

- T-Server log file: `Link connected`
- HA Proxy log file: `Link connected`

Stopping Manually

The following stopping procedures apply to Genesys server applications, such as DB Server, Configuration Server, Message Server, Local Control Agent, Solution Control Server, HA Proxy, T-Server, and Stat Server.

On UNIX

To stop a server application from its console window on UNIX, use either of these commands:

- `Ctrl+C`
- `kill <process number>`

On Windows

- To stop a server application from its console window on Windows, use the `Ctrl+C` command.
- To stop a server application on Windows, use the `End Task` button on the Windows `Task Manager`.

Starting and Stopping with Windows Services Manager

When starting an application installed as a Windows Service, make sure the startup parameters of the application are correctly specified in the `ImagePath` in the `Application` folder in the `Registry Editor`. The `ImagePath` must have the following value data:

```
<full path>\<executable file name> -service <Application Name as
Service> -host <Configuration Server host>
-port <Configuration Server port> -app <Application Name>
-l <license address>
```

where the command-line parameters common to Framework server components are described on page 97 and

`-service`	The name of the Application running as a Windows Service; typically, it matches the Application name specified in the `-app` command-line parameter.

Framework components installed as Windows Services with the autostart capability are automatically started each time a computer on which they are installed is rebooted.

You can start Framework components installed as Windows Services with the manual start capability with the `Start` button in `Services Manager`.

Note: Use the `Windows Services` window to change the startup mode from `Automatic` to `Manual` and vice versa.

Regardless of a component’s start capability, you can stop Framework components installed as Windows Services with the `Stop` button in `Services Manager`.

Next Steps

This chapter concludes Part One of this document—the set of general instructions for deploying any T-Server. Refer to subsequent chapters in this guide for detailed reference information and any special procedural instructions that pertain to your particular T-Server.

Part

2

Part Two: Reference Information

Part Two of this *Network SIP Server Deployment Guide* contains reference information specific to your Network SIP Server. However, it also contains information on *all* T-Server options, both those specific to your Network SIP Server and those common to all T-Servers. The information is divided among these chapters:

- Chapter 5, "Network SIP Server Specific Configuration," on page 109, describes compatibility and configuration information specific to this T-Server, including known limitations and how to configure routing points for this T-Server.
- Chapter 6, "Supported Functionality in Network SIP Server," on page 111, describes which features are supported by this T-Server, including T-Library functionality, error messages and how to map the SIP parameters to Network SIP Server events.
- Chapter 7, "Overview of HA for Network SIP Server," on page 127 describes how to implement a high-availability (HA) configuration for Network SIP Server.
- Chapter 8, "Common Log Options," on page 135, describes log configuration options common to all Genesys server applications.
- Chapter 9, "T-Server Common Configuration Options," on page 149, describes configuration options common to all T-Server types.
- Chapter 10, "Configuration Options in Network SIP Server," on page 173, describes configuration options specific to this T-Server.

New for T-Server in Release 7.5

Before looking at T-Server's place in Genesys solutions and in the architecture of the Genesys Framework, note the following general changes that have been implemented in the 7.2 release of Network SIP Server:

- Network SIP Server supports the `OPTIONS` SIP request. Network SIP Server returns a `200 OK` SIP message as a response to that request.

Note: For a list of new features common to all T-Servers, see Part One of this document. Configuration option changes that apply to SIP Server are described in "Changes from 7.2 to 7.5" on page 177.

Chapter

5 Network SIP Server Specific Configuration

This chapter presents switch-specific reference information for configuring Network SIP Server and includes these sections:

Known Limitations

Several known limitations exist in the current Network SIP Server environment:

- Network SIP Server supports only SIP redirect functionality.
- If you start Local Control Agent as a Windows Service or run it as a Unix background process, do not use console output for the Application log.

Routing Points and Routers

- You can configure routing points for Network SIP Server to balance call load over several Genesys routers. T-Server uses all configured routing points to make calls.
- To make a new call, T-Server selects the first available routing point in round-robin fashion.
- If you configure no routing points, T-Server sends the error message —`503 Service Unavailable`.
- When a routing point is disabled, all calls currently active on that routing point remain active, but no new calls will be placed on it.

Chapter

6

Supported Functionality in Network SIP Server

This chapter describes the telephony functionality that Network SIP Server supports and includes these sections:

T-Library Functionality

The tables in this chapter present T-Library functionality supported in the Network SIP Server. The table entries use these notations:

- **N**—Not supported
- **Y**—Supported

In Table 7 on page 112, when a set of events is sent in response to a single request, the events are listed in an arbitrary order. An asterisk (*) indicates the event that contains the same `Reference ID` as the request. For more information, refer to the *Genesys 7 Events and Models Reference Manual*.

Table 7 reflects only that switch functionality used by Genesys software and might not include the complete set of events offered by the switch.

Certain requests listed in Table 7 are reserved for Genesys Engineering and are listed here merely for completeness of information.

Notes describing specific functionalities may appear at the end of a table.

Table 7: Supported Functionality

Feature Request	Request Subtype	Corresponding Event(s)	Supported
General Requests			
TOpenServer		EventServerConnected	Y
TOpenServerEx		EventServerConnected	Y
TCloseServer		EventServerDisconnected	Y
TSetInputMask		EventACK	Y
TDispatch		Not Applicable	Y
TScanServer		Not Applicable	Y
TScanServerEx		Not Applicable	Y
Registration Requests			
TRegisterAddress[a]		EventRegistered	Y
TUnregisterAddress[a]		EventUnregistered	Y
Call-Handling Requests			
TMakeCall[b]	Regular	EventDialing	N
	DirectAgent		N
	SupervisorAssist		N
	Priority		N
	DirectPriority		N
TAnswerCall		EventEstablished	N
TReleaseCall		EventReleased	N
TClearCall		EventReleased	N
THoldCall		EventHeld	N
TRetrieveCall		EventRetrieved	N
TRedirectCall		EventReleased	N
TMakePredictiveCall		EventDialing*, EventQueued	N

Table 7: Supported Functionality (Continued)

Feature Request	Request Subtype	Corresponding Event(s)	Supported
Transfer/Conference Requests			
TInitiateTransfer[b]		EventHeld, EventDialing*	N
TCompleteTransfer		First arriving EventReleased*, EventPartyChanged	N
TInitiateConference[b]		EventHeld, EventDialing*	N
TCompleteConference		EventReleased*, EventRetrieved, EventPartyChanged, EventPartyAdded	N
TDeleteFromConference		EventPartyDeleted*, EventReleased	N
TReconnectCall[c]		EventReleased, EventRetrieved*	N
TAlternateCall		EventHeld*, EventRetrieved	N
TMergeCalls	ForTransfer	EventReleased*, EventPartyChanged	N
	ForConference	EventReleased*, EventRetrieved, EventPartyChanged, EventPartyAdded	N
TMuteTransfer[b]		EventHeld, EventDialing*, EventReleased, EventPartyChanged	N
TSingleStepTransfer[b]		EventReleased*, EventPartyChanged	N
TSingleStepConference		EventPartyAdded* or EventRinging*, EventEstablished	N
Call-Routing Requests			
TRouteCall[b]	Unknown	EventRouteUsed	Y
	Label		N

Table 7: Supported Functionality (Continued)

Feature Request	Request Subtype	Corresponding Event(s)	Supported
TRouteCall[b]	Default	EventRouteUsed	Y
	OverwriteDNIS		N
	DDD		N
	IDDD		N
	Direct		Y
	Reject		Y
	Announcement		N
	PostFeature		N
	DirectAgent		Y
	Priority		N
	DirectPriority		N
	AgentID		N
	CallDisconnect		N
Call-Treatment Requests			
TApplyTreatment	Unknown	(EventTreatmentApplied + EventTreatmentEnd)/ EventTreatmentNotApplied	N
	IVR		N
	Music		N
	RingBack		N
	Silence		N
	Busy		N
	CollectDigits		N
	PlayAnnouncement		N
	PlayAnnouncementAnd-Digits		N
	RecordUserAnnouncement		N

Table 7: Supported Functionality (Continued)

Feature Request	Request Subtype	Corresponding Event(s)	Supported
TApplyTreatment	VerifyDigits	(EventTreatmentApplied + EventTreatmentEnd)/ EventTreatmentNotApplied	N
	DeleteUserAnnouncement		N
	CancelCall		N
	PlayApplication		N
	SetDefaultRoute		N
	TextToSpeech		N
	TextToSpeechAndDigits		N
	FastBusy		N
	RAN		N
TGiveMusicTreatment		EventTreatmentApplied	N
TGiveRingBackTreatment		EventTreatmentApplied	N
TGiveSilenceTreatment		EventTreatmentApplied	N
DTMF (Dual-Tone Multi-Frequency) Requests			
TCollectDigits		EventDigitsCollected	N
TSendDTMF		EventDTMFSent	N
Voice-Mail Requests			
TOpenVoiceFile		EventVoiceFileOpened	N
TCloseVoiceFile		EventVoiceFileClosed	N
TLoginMailBox		EventMailBoxLogin	N
TLogoutMailBox		EventMailBoxLogout	N
TPlayVoice		EventVoiceFileEndPlay	N
Agent & DN Feature Requests			
TAgentLogin		EventAgentLogin	N
TAgentLogout		EventAgentLogout	N

Table 7: Supported Functionality (Continued)

Feature Request	Request Subtype	Corresponding Event(s)	Supported
TAgentSetIdleReason		EventAgentIdleReasonSet	N
TAgentSetReady		EventAgentReady	N
TAgentSetNotReady		EventAgentNotReady	N
TMonitorNextCall	OneCall	EventMonitoringNextCall	N
	AllCalls		N
TCancelMonitoring		EventMonitoringCanceled	N
TCallSetForward	None	EventForwardSet	N
	Unconditional		N
	OnBusy		N
	OnNoAnswer		N
	OnBusyAndNoAnswer		N
	SendAllCalls		N
TCallCancelForward		EventForwardCancel	N
TSetMuteOff		EventMuteOff	N
TSetMuteOn		EventMuteOn	N
TListenDisconnect		EventListenDisconnected	N
TListenReconnect		EventListenReconnected	N
TSetDNDOn		EventDNDOn	N
TSetDNDOff		EventDNDOff	N
TSetMessageWaitingOn		EventMessageWaitingOn	N
TSetMessageWaitingOff		EventMessageWaitingOff	N
Query Requests			
TQuerySwitch[a]	DateTime	EventSwitchInfo	N
	ClassifierStat		N
TQueryCall[a]	PartiesQuery	EventPartyInfo	N

Table 7: Supported Functionality (Continued)

Feature Request	Request Subtype	Corresponding Event(s)	Supported
TQueryCall[a]	StatusQuery	EventPartyInfo	N
TQueryAddress[a]	AddressStatus	EventAddressInfo	N
	MessageWaitingStatus		N
	AssociationStatus		N
	CallForwardingStatus		N
	AgentStatus		N
	NumberOfAgentsInQueue		N
	NumberOfAvailableAgents-InQueue		N
	NumberOfCallsInQueue		N
	AddressType		N
	CallsQuery		N
	SendAllCallsStatus		N
	QueueLoginAudit		N
	NumberOfIdleTrunks		N
	NumberOfTrunksInUse		N
	DatabaseValue		N
	DNStatus		N
	QueueStatus		N
TQueryLocation[a]	AllLocations	EventLocationInfo	N
	LocationData		N
	MonitorLocation		N
	CancelMonitorLocation		N
	MonitorAllLocations		N
	CancelMonitorAllLocations		N
	LocationMonitorCanceled		N

Table 7: Supported Functionality (Continued)

Feature Request	Request Subtype	Corresponding Event(s)	Supported
TQueryLocation[a]	AllLocationsMonitor-Canceled	EventLocationInfo	N
TQueryServer[a]		EventServerInfo	Y
User-Data Requests			
TAttachUserData (Obsolete)		EventAttachedDataChanged	Y
TUpdateUserData		EventAttachedDataChanged	Y
TDeleteUserData		EventAttachedDataChanged	Y
TDeleteAllUserData		EventAttachedDataChanged	Y
ISCC (Inter Server Call Control) Requests			
TGetAccessNumber[b]		EventAnswerAccessNumber	Y
TCancelReqGetAccess-Number		EventReqGetAccess-NumberCanceled	Y
Special Requests			
TReserveAgent		EventAgentReserved	Y
TSendEvent		EventACK	Y
TSendEventEx		EventACK	Y
TSetCallAttributes		EventCallInfoChanged	N
TSendUserEvent		EventACK	Y
TPrivateService		EventPrivateInfo	N

a. Only the requestor receives a notification of the event associated with this request.

b. This feature request may be made across locations in a multi-site environment, if the `location` attribute of the request contains a value relating to any location other than the local site. However, when the response to this request is `EventError`, a second event response containing the same `reference ID` as the first event is sent. This second event is either `EventRemoteConnectionSuccess` or `EventRemoteConnectionFailed`.

c. Supported only for T-Library clients using version 5.0 or later.

Mapping SIP Parameters to T-Server Event Parameters

Table 8 indicates how Network SIP Server maps parameters from SIP messages to attributes of T-Library events.

Table 8: Mapping SIP Parameters to Network SIP Server Events

SIP INVITE Parameters[a]	EventRouteRequest Attributes		
	Attribute Name	Value Type	Value Description
To Uniform Resource Indicator (URI)	AttributeDNIS	String	Free dial number
From URI	AttributeANI	String	`From URI` is mapped to `AttributeANI` if `P-Asserted-Identity` header is absent inside the SIP `INVITE` request.
Contact URI	AttributeOtherDN	String	`Contact URI` is mapped to `AttributeOtherDN` if `P-Asserted-Identity` header is absent inside the SIP `INVITE` request.
P-Asserted-Identity first URI	AttributeOtherDN	String	If present, overrides `Contact URI`
P-Asserted-Identity second URI	AttributeANI	String	If present, overrides `From URI`
Request URI—cic parameter	AttributeExtensions: kv-pair with key = `INVITE:cic`[b]	String	Carrier identification code
Request URI—ca parameter	AttributeExtensions: kv-pair with key = `INVITE:ca`[b]	String	Charging area
From header—cpc parameter	AttributeExtensions: kv-pair with key = `From:cpc`[b]	String	Calling party category
Request URI—intl parameter	AttributeExtensions: kv-pair with key = `INVITE:intl`[b]	String	International flag

Table 8: Mapping SIP Parameters to Network SIP Server Events (Continued)

SIP INVITE Parameters[a]	EventRouteRequest Attributes		
	Attribute Name	**Value Type**	**Value Description**
Request URI—attempt parameter	AttributeExtensions: kv-pair with key = `INVITE:attempt`	String	Indicates the number of attempts to route a call. This number starts with `1` and is incremented for each unsuccessful attempt (such as busy or no answer). If this parameter is missing, the first attempt is assumed.
Request URI—lastattemptstatus parameter	AttributeExtensions: kv-pair with key = `INVITE:lastattempt status`	String	This parameter is returned for second and subsequent attempt to route a call. It indicates why the previous attempt has failed. The valid value of this parameter is a 3-digit SIP error code.
Request URI—Privacy	AttributeExtensions: kv-pair with key = `Privacy`	String	Specifies the contents of the privacy header Default Values: `none, id` Valid Value: Any positive integer If you do not set a value, the default value is applied.
Request URI—Local number portability lookup flag	AttributeExtensions: kv-pair with key = `INVITE:npdi`	String	Local number portability lookup flag

a. Additional SIP parameters can be passed as extension key-value pairs of T-Server events. You can configure the exact set of parameters. The naming convention for the keys is `header:parameter`. If the entire header must to be passed, the key is defined as `header`. Parameters of the `Request URI` line are passed as `request:parameter`, where `request` is the SIP request name, for example, `INVITE`.

b. All parameters that are to be mapped must be configured in the Extensions section. For further information, see Chapter 10, “Extensions Section,” page 175.

Table 9 indicates how Network SIP Server maps Request Attributes from SIP messages to T-Library event attributes.

Table 9: Mapping T-Library Request Attributes to SIP Parameters

	RequestRouteCall Attributes		
SIP 302 Response Parameters	**Attribute Name**	**Value Type**	**Value Description**
Contact header, expires parameter	AttributeOtherDN	String	It specifies timeout in seconds for ring-no-answer situation. If a call is not answered within “expires” seconds, ringing on the destination device should stop and a new `INVITE` with an incremented “attempt” issued to T-Server. If no parameter is specified the system default should be used.
Contact header, charge parameter	AttributeOtherDN	String	Charge indicator can be used to specify whether a call should be charged or not. Valid values: `yes`, `no` If you do not set a value, the default value, `yes`, is applied.

Examples of Expires and Charge Parameters

AttributeOtherDN of the RequestRouteCall is mapped into the value of SIP Contact header without any changes.

No parameters:

```
AttributeOtherDN= sip:012022222@announcement.genesyslab.com; is mapped
into the Contact header
Contact: sip:0120222222@announcement.genesyslab.com
```

Expires parameter:

```
AttributeOtherDN= sip:012022222@announcement.genesyslab.com; expires=20
is mapped into the Contact header
Contact: sip:0120222222@announcement.genesyslab.com; expires=20
```

Charge parameter:

```
AttributeOtherDN= sip:012022222@announcement.genesyslab.com; charge=yes
is mapped into the Contact header
Contact: sip:0120222222@announcement.genesyslab.com; charge=yes
```

Expires and Charge parameters:

```
AttributeOtherDN= sip:012022222@announcement.genesyslab.com;
expires=20;charge=yes is mapped into the Contact header
Contact: sip:0120222222@announcement.genesyslab.com; expires=2;
charge=yes
```

Error Messages

Table 10 and Table 11 on page 124 present the complete set of error messages Network SIP Server distributes with `EventError`. While some guidelines are given on how to handle T-Server Common Part (TSCP) error messages, refer to the switch documentation for resolution of switch-related errors.

Table 10: T-Server Common Part (TSCP) Error Messages

T-Library Error Code	Symbolic Name	Description	Recommendations
40	TERR_NOMORE_LICENSE	No more licenses are available	Ensure that the license file contains enough licenses or increase the number of licenses in T-Server configuration.
41	TERR_NOT_REGISTERED	Client has not registered for the DN	T-Server reports unauthorized access. Make sure that the T-client successfully registers the DN before sending other requests involving the DN.
42	TERR_RESOURCE_SEIZED	Resource is already seized	The resource (DN) cannot be registered because another application has registered for it in `Private` mode. Select another DN for registration or have that application unregister the DN first.
43	TERR_IN_SAME_STATE	Object is already in requested state.	
50	TERR_UNKNOWN_ERROR	Unrecognized error	T-Server could not identify the reason for the error. Check the error message text for possible explanation of the error.
51	TERR_UNSUP_OPER	Unsupported operation	The combination of this T-Server release with this switch version does not support the requested operation. Contact Genesys Technical Support for a possible workaround.

Table 10: T-Server Common Part (TSCP) Error Messages (Continued)

T-Library Error Code	Symbolic Name	Description	Recommendations
52	TERR_INTERNAL	Internal error	Contact Genesys Technical Support.
53	TERR_INVALID_ATTR	Invalid attribute value	Check the content of the client request for correctness.
54	TERR_NO_SWITCH	No connection to the switch	Ensure that the connection to the switch exists.
55	TERR_PROTO_VERS	Incorrect protocol version	T-Server cannot recognize the client version. Make sure that the client uses the T-Library protocol. Check the T-Server and client version compatibility.
56	TERR_INV_CONNID	Invalid Connection ID	At the time when T-Server received the request, the `Connection ID` was invalid. Ensure that the specified `Connection ID` is associated with a live call.
57	TERR_TIMEOUT	Timeout expired	The request processing was canceled because of a timeout. Resubmit the request or check that the request is valid in association with the subject of the request.
58	TERR_OUT_OF_SERVICE	Out of service	The referenced resource (for example, `DN`) is out of service.
59	TERR_NOT_CONFIGURED	DN is not configured in the Configuration Database	A client attempts to register for a DN that must be configured in the Configuration Database. Verify that the specified DN is valid, and if so, add the DN to the switch configuration in the Configuration Database.

Switch-Specific Error Messages

Table 11: Switch-Specific Error Messages

Error Code	Symbolic Name	Description	Recommendations
4526	GCTI_SERVER_PORT_UNABLE_TO_OPEN	T-Server cannot create port for accepting switch datagram.	Determine if this port number is already occupied by another Application. Release this port or change the `sip-port` option inside the SIP T-Server configuration.
52501	EXT_OPT_NOT_VALID	Invalid option name in `Extensions` section.	Change the name of the option. It should have the form `parameter-`*`n`*, where *`n`* is decimal number.
52502	DN_TYPE_NOT_SUP	The switch does not support DN objects of this type.	Delete this DN object from the configuration.
52550	NO_ROUTING_POINT	No configured routing points are available.	Determine if at least one routing point persists inside the switch configuration and is enabled. Verify that Genesys Universal Routing Server is running and connected to T-Server.
52551	ATTEMPTS_EXHAUSTED	Switch exceeded the configured number (`10`) of rerouting attempts.	Determine if the routing strategy has provided a valid reroute destination.
52552	ROUTER_TIMEOUT	No responses received from the router during configured timeout.	Check router strategy for accuracy or increase the value of `router-timeout` option.
52601	SIP_INCOMPLETE	Message is not coming from the SIP switch.	Be sure that datagram is coming from the SIP switch rather than a different application.

Table 11: Switch-Specific Error Messages (Continued)

Error Code	Symbolic Name	Description	Recommendations
52602	SIP_PARSE_ERR	Error occurred during parsing of SIP messages.	Check the switch configuration. The SIP switch may not be configured properly to work with T-Server.
52603	SIP_PROCESSING_ERR	T-Server cannot produce right response based on the switch request.	Check the switch configuration. The SIP switch may not be configured properly to work with T-Server.
52604	SIP_UNEXPECTED	T-Server received SIP message that does not conform with SIP call model.	If the error persists, adjust the configuration of T-Server timeout options `t1-timeout` and `t2-timeout` to conform with the SIP switch property.
52605	RDR_FSM_ERR	Error in processing T-Server finite state machine.	If the error persists, adjust the configuration of T-Server timeout options `t1-timeout` and `t2-timeout` to conform with the SIP switch property.
52606	NO_ACK	No `ACK` message received from the switch.	If the error persists, adjust the configuration of T-Server timeout options `t1-timeout` and `t2-timeout` to conform with the SIP switch property.

Chapter

7 HA Configuration for Network SIP Server

This chapter describes how to implement a high-availability (HA) configuration for Network SIP Server. It contains these sections:

Overview of HA for Network SIP Server

Network SIP Server differs from most Genesys T-Servers in the role it performs in the SIP network. It is not a switch client, because it is a switch itself.

Genesys T-Servers are mostly TCP/IP clients of a PABX, and maintain the TCP/IP connection with a switch on continual basis. Network SIP Server does *not* establish a TCP/IP connection to the SIP switch at all. Network SIP Server uses a connectionless protocol, User Datagram Protocol (UDP) by default.

UDP makes it difficult for a client (SIP phone, gateway, soft switch) to detect the failure of T-Server, and even more difficult to determine when a T-Server has recovered. But it also makes it easy to substitute another T-Server for one that has failed—without the switch even noticing it, since there is no connection to establish, or registration procedure to go through.

Warm Standby for Network SIP Server

You can achieve warm standby for Network SIP Server by deploying a cluster of SIP Servers with virtual IP address support, and preserving the IP address during the switchover. This method effectively hides the switchover (and switchback) from the endpoints and gateways.

Note: Hot-standby HA is not currently supported for SIP Communications Server.

Implementing Warm Standby on Solaris

Warm standby on Solaris is achieved by hiding two hosts, on which primary and backup servers are running, behind the same IP address. This method effectively hides the switchover (and switchback) from the endpoints and gateways.

Two Sun hosts on the same network subnet (“host1” and “host2”) are used. Each host has two logical IP interfaces assigned to a single Ethernet interface.

The first IP interface (called “unique” in this example) has the unique IP address of the host on the subnet.

The second IP interface (called “common” in this example, though it is sometimes referred to as “virtual”) has an address on the same subnet and this address is shared between two hosts.

The unique interface should always be activated on each host. T-Library clients, and management and configuration software also use the unique interface of the hosts.

The common interface should be activated only if Network SIP Server is working in primary role on that host. Otherwise, the common interface should be deactivated. SIP User Agents (endpoints) are only aware of the common interface.

The address and hostname of the common interface, as well as addresses and hostnames of the unique interfaces, should be known to the DNS server.

On each host, the `/etc/hosts` file should contain the record about the common interface, in the form:

```
 <IP_address> <common_name_of_the_host>
```

For example:

```
192.168.22.180 sipcs_common
```

On each host, inside the `/etc` directory you should create the file

```
hostname.name_of_ethernet_interface:1
```

Where `name_of_ethernet_interface` is the actual name of the Ethernet interface on that machine—for example;

```
hostname.dmfe0:1
```

This file should contain the hostname of the common interface as it is known to the DNS server and is recorded inside the `/etc/hosts` file.

Management of the state of the common interface is provided by the Sun administrative command `ifconfig`.

To bring up the common interface, you must issue the command:

```
ifconfig name_of_ethernet_interface:1 up
```

To bring it down, you must issue the command:

```
ifconfig name_of_ethernet_interface:1 down
```

This command should be wrapped by shell batch files. Each host must contain two such shell files, one to bring up the common interface, and one to bring it down.

The Configuration Layer contains four applications, which are referenced by these shell files as third-party servers.

Four corresponding Alarm Reaction Scripts should be created. These scripts are responsible for the start of each of the above-mentioned Third Party Servers in order to execute these shell files:

1. Virtual IP Down on host1
2. Virtual IP Up on host1
3. Virtual IP Down on host2
4. Virtual IP Up on host2

You should also create four corresponding Alarm Conditions:

1. Primary for host1
   Alarm Condition 1 is linked with Log Event `00-04562 Warm Standby (Primary) mode activated` for the server that is running on host1. It starts Alarm Reaction Scripts 2 and 3.
2. Primary for host2
   Alarm Condition 2 is linked with Log Event `00-04562 Warm Standby (Primary) mode activated` for the server that is running on host2. It starts Alarm Reaction Scripts 1 and 4.
3. Backup for host1
   Alarm Condition 3 is linked with Log Event `00-04560 Warm Standby (backup) mode activated` for the server that is running on host1. It starts Alarm Reaction Script 1.
4. Backup for host2
   Alarm Condition 4 is linked with Log Event `00-04560 Warm Standby (backup) mode activated` for the server that is running on host2. It starts Alarm Reaction Script 3.

Notes: The value of `Cancel timeout` for all Alarm Conditions should be set to `1` in Configuration Manager.

If hosts have more than one Ethernet interface, Genesys server clients should use the unique IP interface of the same Ethernet interface where the common IP interface is assigned.

Implementing Warm Standby on Windows

This section describes how to implement warm-standby high availability by using the Network Load-Balancing technology included in the Microsoft Windows 2000 Advanced Server.

The unique and fully distributed architecture of Network Load-Balancing enables it to deliver failover protection without requiring any special hardware.

Network Load Balancing

The HA approach for Network SIP Server on Windows uses a Network Load-Balancing solution to achieve a warm-standby level of HA. A Network Load-Balancing cluster uses the concept of a *Virtual IP address*. The SIP switch and/or all endpoints are configured to send all SIP requests to Network SIP Server using this single address. The cluster software delivers the requests to only one of the servers in the cluster, and switches over to another Network SIP Server if it detects a Network SIP Server failure.

The advantages of a Network Load-Balancing solution are:

- It requires no special configuration on the SIP switch or SIP endpoints—you need to configure only one IP address.
- The switch or endpoints do not require any special actions during switchover.
- Orderly primary/backup switchover can be invoked from the Genesys Management Layer (for maintenance purposes).
- No special hardware is required on the Network Load-Balancing cluster machines.

Configuring Warm Standby

You must use two hosts as shown in Figure 10 on page 131.

1. On both hosts, install the Windows Advanced Server and enable the Network Load-Balancing feature.

Note: Genesys recommends that you install the Management and Configuration Layers outside the Network Load-Balancing cluster. Genesys Universal Router Servers can work outside or inside the cluster.

2. On each cluster host, install Network SIP Servers and LCAs.

3. In the Configuration Database, for the host name (TCP/IP addresses) for each Network SIP Server, do *not* use the virtual TCP/IP address of the cluster, but the actual TCP/IP address of the host on which SIP Server will be running.

4. Configure Network SIP Servers in `warm standby` mode (that is, in the primary/backup relationship). You must set the same value for the `sip-port` option for both Network SIP Servers.

Note: If you run Network SIP Server in `warm-standby` mode make sure that the cluster feature is enabled on the host on which Network SIP Server is running. If the cluster feature on the host is disabled, warm standby will not work properly.

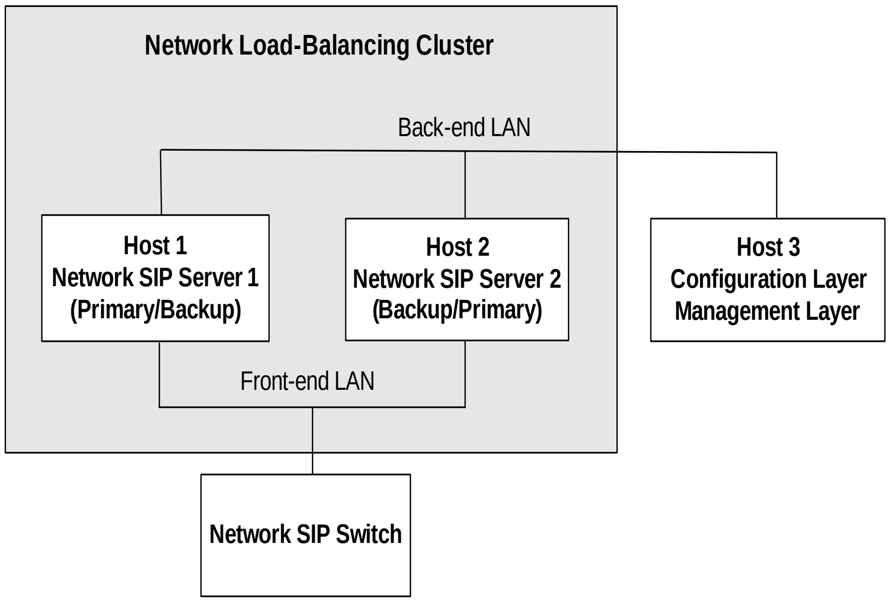


Figure 10: HA Architecture for Network SIP Server

You must configure the soft switch or endpoints to send datagrams to the virtual address of the Network Load-Balancing cluster—that is, to the port as it is configured in the `sip-port` option (see page 173).

Configuring Network Load-Balancing Parameters

On cluster hosts, you must configure the Network Load-Balancing parameters as follows:

- `Port Range`: Must include a port as configured in the `sip-port` option.
- `Protocols`: `UDP` and `TCP`, if required.
- `Filtering mode`: `Multiple` (for multiple hosts).
- `Affinity`: `None` or `Single`.
- `Load weight`: `Equal`.

For more information on how to administer Network Load-Balancing technology, see the Windows Advanced Server documentation or online help.

A datagram from the SIP switch arrives at both hosts of the cluster, but the Network Load-Balancing Driver lets traffic go to the Application Layer of only one cluster host. To ensure that this is the host where the primary Network SIP Server is running, the Management Layer issues the following commands:

- `wlbs enable` *XXXX* `<cluster_name>:<ID_Primary>`
- `wlbs disable` *XXXX* `<cluster_name>:<ID_Backup>`

Where:

- `wlbs` is name of the Windows utility used to control Network Load-Balancing.
- `enable` is the command to enable traffic handling for the host on which the primary Network SIP Server is running.
- *`XXXX`* is the decimal value number of the port as it is configured in the `sip-port` option.
- `cluster_name` is the virtual IP address of the cluster.
- `ID_Primary` is the unique `host ID` of the host where the primary Network SIP Server is currently running.
- `disable` is the command to disable traffic handling for the host on which the backup Network SIP Server is running.
- `ID_Backup` is the unique `host ID` of the host where the backup Network SIP Server is currently running.

If these commands are issued at the moment when Network SIP Server changes its roles, the primary Network SIP Server always handles the traffic.

Configuring Primary/Backup Network SIP Servers

In order to do this for both the primary and backup Network SIP Servers, follow these steps in the Configuration Layer:

1. On the host where Network SIP Server is installed, create a batch file containing the commands described in the previous section.
2. In Configuration Manager, create a new `Application` object that uses the Third Party Server application template.
3. On the `Server Info` tab of the new `Application` object, set the host to the name of the host on which Network SIP Server is installed. Set a valid value for the communication port. Use the default values for all other parameters on this tab.
4. On the `Start Info` tab, set the working directory to the location of the batch file you created in Step 1. For the command-line parameter, use the name of that batch file. Use the default values for all other parameters on this tab.

Creating and Setting Alarm Conditions

After configuring the new `Application` object, use the Solution Control Interface (SCI) to create alarm conditions for both the primary and backup Network SIP Servers. For each Network SIP Server two alarm conditions should be created.

To create and set alarm conditions, complete the following steps:

1. Create a new alarm condition.

2. Set the `Alarm Detection` to `Detection Log Event 00-04562 (Warm-Standby Primary mode activated)`.
3. Set the `Alarm Source` to the `Application` object corresponding to the relevant Network SIP Server.
4. On the `Assigning Alarm Reactions` tab, click `Add`. Right-click to create a new script.
5. On the `Alarm Reaction` tab, select `Start Specified Application`.
6. For the `application to start`, choose the third-party server you created in "Configuring Primary/Backup Network SIP Servers" on page 132.
7. Create a new alarm condition.
8. Set the `Alarm Detection` to `Detection Log Event 00-04560` (Warm-Standby Backup mode activated).
9. Set the `Alarm Source` to the specific `Application` object corresponding to the relevant Network SIP Server.
10. On the `Assigning Alarm Reactions` tab, click `Add`. Right-click to create a new script.
11. On the `Alarm Reaction` tab, select `Stop Specified Application`.
12. For the `application to stop`, choose the third-party server you created in "Configuring Primary/Backup Network SIP Servers" on page 132.

Recommended Sample Batch File

This section contains a sample of the batch file you should use to monitor the state of the host as a member of the load-balancing cluster. This script should be configured as a third-party server on host 1 of the cluster.

```
title Tserver and cluster monitor
wlbs enable 5060 clustername:1
wlbs disable 5060 clustername:2
:alive
@sleep 1
@pulist |findstr /C:"tsip_server.exe" >dump.log
@if errorlevel 1 goto ex
@wlbs query testcluster:2 |findstr /C:"Host 1 converg" >dump.log
@if not errorlevel 1 goto alive
@date /T>>monitor.log
@TIME /T>>monitor.log
@echo Host not in cluster anymore,t-server will be killed>>
monitor.log
```

```
tskill tsip_server*
exit
:ex
@date /T>>monitor.log
@TIME /T>>monitor.log
@echo t-server is not running anymore>> monitor.log
exit
```

For host 2 of the cluster, use the same script but with the host numbers reversed.

For a technical overview of Network Load Balancing and Windows 2003 clustering technologies, go to http://www.microsoft.com/.

Chapter

8 Common Log Options

Unless otherwise noted, the log configuration options that this chapter describes are common to all Genesys server applications and applicable to any Framework server component. This chapter includes the following sections:

Note: Some server applications also support log options that are unique to them. For descriptions of a particular application's unique log options, refer to the chapter/document about that application.

Mandatory Options

You do not have to configure any common log options in order to start Server applications.

Log Section

This section must be called `log`.

verbose

Default Value: `all`

Valid Values:

`all`	All log events (that is, log events of the Standard, Trace, Interaction, and Debug levels) are generated.
`debug`	The same as `all`.
`trace`	Log events of the Trace level and higher (that is, log events of the Standard, Interaction, and Trace levels) are generated, but log events of the Debug level are not generated.
`interaction`	Log events of the Interaction level and higher (that is, log events of the Standard and Interaction levels) are generated, but log events of the Trace and Debug levels are not generated.
`standard`	Log events of the Standard level are generated, but log events of the Interaction, Trace, and Debug levels are not generated.
`none`	No output is produced.

Changes Take Effect: Immediately

Determines whether a log output is created. If it is, specifies the minimum level of log events generated. The log events levels, starting with the highest priority level, are Standard, Interaction, Trace, and Debug. See also "Log Output Options" on page 142.

Note: For definitions of the Standard, Interaction, Trace, and Debug log levels, refer to the *Framework 7.5 Deployment Guide* or to *Framework 7.5 Solution Control Interface Help.*

buffering

Default Value: `true`
Valid Values:

`true`	Enables buffering.
`false`	Disables buffering.

Changes Take Effect: Immediately

Turns on/off operating system file buffering. The option is applicable only to the `stderr` and `stdout` output (see page 142). Setting this option to `true` increases the output performance.

Note: When buffering is enabled, there might be a delay before log messages appear at the console.

segment

Default Value: `false`
Valid Values:

`false`	No segmentation is allowed.
`<number> KB` or `<number>`	Sets the maximum segment size, in kilobytes. The minimum segment size is `100 KB`.
`<number> MB`	Sets the maximum segment size, in megabytes.
`<number> hr`	Sets the number of hours for the segment to stay open. The minimum number is `1` hour.

Changes Take Effect: Immediately

Specifies whether there is a segmentation limit for a log file. If there is, sets the mode of measurement, along with the maximum size. If the current log segment exceeds the size set by this option, the file is closed and a new one is created.

expire

Default Value: `false`
Valid Values:

`false`	No expiration; all generated segments are stored.
`<number> file` or `<number>`	Sets the maximum number of log files to store. Specify a number from `1–100`.
`<number> day`	Sets the maximum number of days before log files are deleted. Specify a number from `1–100`.

Changes Take Effect: Immediately

Determines whether log files expire. If they do, sets the measurement for determining when they expire, along with the maximum number of files (segments) or days before the files are removed.

Note: If an option's value is set incorrectly—out of the range of valid values— it will be automatically reset to `10`.

keep-startup-file

Default Value: `false`
Valid Values:

`false`	No startup segment of the log is kept.
`true`	A startup segment of the log is kept. The size of the segment equals the value of the segment option.
`<number> KB`	Sets the maximum size, in kilobytes, for a startup segment of the log.
`<number> MB`	Sets the maximum size, in megabytes, for a startup segment of the log.

Changes Take Effect: After restart

Specifies whether a startup segment of the log, containing the initial T-Server configuration, is to be kept. If it is, this option can be set to `true` or to a specific size. If set to `true`, the size of the initial segment will be equal to the size of the regular log segment defined by the `segment` option. The value of this option will be ignored if segmentation is turned off (that is, if the `segment` option set to `false`).

Note: This option applies only to T-Servers.

messagefile

Default Value: As specified by a particular application
Valid Values: `<string>.lms` (message file name)
Changes Take Effect: Immediately, if an application cannot find its `*.lms` file at startup

Specifies the file name for application-specific log events. The name must be valid for the operating system on which the application is running. The option value can also contain the absolute path to the application-specific `*.lms` file. Otherwise, an application looks for the file in its working directory.

Warning! An application that does not find its `*.lms` file at startup cannot generate application-specific log events and send them to Message Server.

message_format

Default Value: `short`
Valid Values:

`short`	An application uses compressed headers when writing log records in its log file.
`full`	An application uses complete headers when writing log records in its log file.

Changes Take Effect: Immediately

Specifies the format of log record headers that an application uses when writing logs in the log file. Using compressed log record headers improves application performance and reduces the log file's size.

With the value set to `short`:

- A header of the log file or the log file segment contains information about the application (such as the application name, application type, host type, and time zone), whereas single log records within the file or segment omit this information.
- A log message priority is abbreviated to `Std`, `Int`, `Trc`, or `Dbg`, for Standard, Interaction, Trace, or Debug messages, respectively.
- The message ID does not contain the prefix `GCTI` or the application type `ID`.

A log record in the full format looks like this:

```
2002-05-07T18:11:38.196 Standard localhost cfg_dbserver GCTI-00-05060
Application started
```

A log record in the short format looks like this:

```
2002-05-07T18:15:33.952 Std 05060 Application started
```

Note: Whether the full or short format is used, time is printed in the format specified by the `time_format` option.

time_convert

Default Value: `Local`
Valid Values:

`local`	The time of log record generation is expressed as a local time, based on the time zone and any seasonal adjustments. Time zone information of the application's host computer is used.
`utc`	The time of log record generation is expressed as Coordinated Universal Time (UTC).

Changes Take Effect: Immediately

Specifies the system in which an application calculates the log record time when generating a log file. The time is converted from the time in seconds since the Epoch (00:00:00 UTC, January 1, 1970).

time_format

Default Value: `time`
Valid Values:

`time`	The time string is formatted according to the `HH:MM:SS.sss` (hours, minutes, seconds, and milliseconds) format.
`locale`	The time string is formatted according to the system's locale.
`ISO8601`	The date in the time string is formatted according to the ISO 8601 format. Fractional seconds are given in milliseconds.

Changes Take Effect: Immediately

Specifies how to represent, in a log file, the time when an application generates log records.

A log record's time field in the ISO 8601 format looks like this:

```
2001-07-24T04:58:10.123
```

print-attributes

Default Value: `false`
Valid Values:

`true`	Attaches extended attributes, if any exist, to a log event sent to log output.
`false`	Does not attach extended attributes to a log event sent to log output.

Changes Take Effect: Immediately

Specifies whether the application attaches extended attributes, if any exist, to a log event that it sends to log output. Typically, log events of the Interaction log level and Audit-related log events contain extended attributes. Setting this option to `true` enables audit capabilities, but negatively affects performance. Genesys recommends enabling this option for Solution Control Server and Configuration Server when using audit tracking. For other applications, refer to *Genesys 7.5 Combined Log Events Help* to find out whether an application generates Interaction-level and Audit-related log events; if it does, enable the option only when testing new interaction scenarios.

check-point

Default Value: `1`
Valid Values: `0–24`
Changes Take Effect: Immediately

Specifies, in hours, how often the application generates a `check point` log event, to divide the log into sections of equal time. By default, the application generates this log event every hour. Setting the option to `0` prevents the generation of `check-point` events.

memory

Default Value: No default value
Valid Values: `<string>` (memory file name)
Changes Take Effect: Immediately

Specifies the name of the file to which the application regularly prints a snapshot of the memory output, if it is configured to do this (see "Log Output Options" on page 142). The new snapshot overwrites the previously written data. If the application terminates abnormally, this file will contain the latest log messages. Memory output is not recommended for processors with a CPU frequency lower than 600 MHz.

Note: If the file specified as the `memory` file is located on a network drive, an application does not create a snapshot file (with the extension `*.memory.log`).

memory-storage-size

Default Value: `2 MB`
Valid Values:

`<number> KB` or `<number>`	The size of the memory output, in kilobytes. The minimum value is `128 KB`.
`<number> MB`	The size of the memory output, in megabytes. The maximum value is `64 MB`.

Changes Take Effect: When memory output is created

Specifies the buffer size for log output to the memory, if configured. See also "Log Output Options" on page 142.

spool

Default Value: The application's working directory
Valid Values: `<path>` (the folder, with the full path to it)
Changes Take Effect: Immediately

Specifies the folder, including full path to it, in which an application creates temporary files related to network log output. If you change the option value while the application is running, the change does not affect the currently open network output.

compatible-output-priority

Default Value: `false`
Valid Values:

`true`	The log of the level specified by "Log Output Options" is sent to the specified output.
`false`	The log of the level specified by "Log Output Options" and higher levels is sent to the specified output.

Changes Take Effect: Immediately

Specifies whether the application uses 6.x output logic. For example, you configure the following options in the `log` section for a 6.x application and for a 7.x application:

```
[log]
verbose = all
debug = file1
standard = file2
```

The log file content of a 6.x application is as follows:

- `file1` contains Debug messages only.
- `file2` contains Standard messages only.

The log file content of a 7.x application is as follows:

- `file1` contains Debug, Trace, Interaction, and Standard messages.
- `file2` contains Standard messages only.

If you set `compatible-output-priority` to `true` in the 7.x application, its log file content will be the same as for the 6.x application.

Warning! Genesys does not recommend changing the default value of the `compatible-output-priority` option, unless you have specific reasons to use the 6.x log output logic—that is, to mimic the output priority as implemented in releases 6.x. Setting this option to `true` affects log consistency.

Log Output Options

To configure log outputs, set log level options (`all`, `standard`, `interaction`, `trace`, and/or `debug`) to the desired types of log output (`stdout`, `stderr`, `network`, `memory`, and/or `[filename]`, for log file output).

You can use:

- One log level option to specify different log outputs.
- One log output type for different log levels.
- Several log output types simultaneously, to log events of the same or different log levels.

You must separate the log output types by a comma when you are configuring more than one output for the same log level. See "Examples" on page 145.

Note: The log output options are activated according to the setting of the `verbose` configuration option.

Warnings! If you direct log output to a file on the network drive, an application does not create a snapshot log file (with the extension `*.snapshot.log`) in case it terminates abnormally.

Directing log output to the console (by using the `stdout` or `stderr` settings) can affect application performance. Avoid using these log output settings in a production environment.

all

Default Value: No default value
Valid Values (log output types):

`stdout`	Log events are sent to the Standard output (`stdout`).
`stderr`	Log events are sent to the Standard error output (`stderr`).
`network`	Log events are sent to Message Server, which can reside anywhere on the network. Message Server stores the log events in the Log Database. Setting the `all` log level option to the network output enables an application to send log events of the `Standard`, `Interaction`, and `Trace` levels to Message Server. `Debug`-level log events are neither sent to Message Server nor stored in the Log Database.
`memory`	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
`[filename]`	Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory.

Changes Take Effect: Immediately

Specifies the outputs to which an application sends all log events. The log output types must be separated by a comma when more than one output is configured. For example:

```
all = stdout, logfile
```

Note: To ease the troubleshooting process, consider using unique names for log files that different applications generate.

standard

Default Value: No default value
Valid Values (log output types):

`stdout`	Log events are sent to the Standard output (`stdout`).
`stderr`	Log events are sent to the Standard error output (`stderr`).
`network`	Log events are sent to Message Server, which can reside anywhere on the network. Message Server stores the log events in the Log Database.
`memory`	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
`[filename]`	Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory.

Changes Take Effect: Immediately

Specifies the outputs to which an application sends the log events of the `Standard` level. The log output types must be separated by a comma when more than one output is configured. For example:

```
standard = stderr, network
```

interaction

Default Value: No default value
Valid Values (log output types):

`stdout`	Log events are sent to the Standard output (`stdout`).
`stderr`	Log events are sent to the Standard error output (`stderr`).
`network`	Log events are sent to Message Server, which can reside anywhere on the network. Message Server stores the log events in the Log Database.
`memory`	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
`[filename]`	Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory.

Changes Take Effect: Immediately

Specifies the outputs to which an application sends the log events of the `Interaction` level and higher (that is, log events of the `Standard` and

`Interaction` levels). The log outputs must be separated by a comma when more than one output is configured. For example:

```
interaction = stderr, network
```

trace

Default Value: No default value
Valid Values (log output types):

`stdout`	Log events are sent to the Standard output (`stdout`).
`stderr`	Log events are sent to the Standard error output (`stderr`).
`network`	Log events are sent to Message Server, which can reside anywhere on the network. Message Server stores the log events in the Log Database.
`memory`	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
`[filename]`	Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory.

Changes Take Effect: Immediately

Specifies the outputs to which an application sends the log events of the `Trace` level and higher (that is, log events of the `Standard`, `Interaction`, and `Trace` levels). The log outputs must be separated by a comma when more than one output is configured. For example:

```
trace = stderr, network
```

debug

Default Value: No default value
Valid Values (log output types):

`stdout`	Log events are sent to the Standard output (`stdout`).
`stderr`	Log events are sent to the Standard error output (`stderr`).
`memory`	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
`[filename]`	Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory.

Changes Take Effect: Immediately

Specifies the outputs to which an application sends the log events of the `Debug` level and higher (that is, log events of the `Standard`, `Interaction`, `Trace`, and `Debug` levels). The log output types must be separated by a comma when more than one output is configured. For example:

```
debug = stderr, /usr/local/genesys/logfile
```

Note: `Debug`-level log events are never sent to Message Server or stored in the Log Database.

Log File Extensions

You can use the following file extensions to identify log files that an application creates for various types of output:

- `*.log`—Assigned to log files when you configure output to a log file. For example, if you set `standard = confservlog` for Configuration Server, it prints log messages into a text file called `confservlog.<time_stamp>.log`.
- `*.qsp`—Assigned to temporary (spool) files when you configure output to the network but the network is temporarily unavailable. For example, if you set `standard = network` for Configuration Server, it prints log messages into a file called `confserv.<time_stamp>.qsp` during the time the network is not available.
- `*.snapshot.log`—Assigned to files that contain the output snapshot when you configure output to a log file. The file contains the last log messages that an application generates before it terminates abnormally. For example, if you set `standard = confservlog` for Configuration Server, it prints the last log message into a file called `confserv.<time_stamp>.snapshot.log` in case of failure.

Note: Provide `*.snapshot.log` files to Genesys Technical Support when reporting a problem.

- `*.memory.log`—Assigned to log files that contain the memory output snapshot when you configure output to memory and redirect the most recent memory output to a file. For example, if you set `standard = memory` and `memory = confserv` for Configuration Server, it prints the latest memory output to a file called `confserv.<time_stamp>.memory.log`.

Examples

This section presents examples of a `log` section that you might configure for an application when that application is operating in production mode and in two lab modes, debugging and troubleshooting.

Production Mode Log Section

```
[log]
verbose = standard
standard = network, logfile
```

With this configuration, an application only generates the log events of the `Standard` level and sends them to Message Server, and to a file named `logfile`, which the application creates in its working directory. Genesys recommends that you use this or a similar configuration in a production environment.

Warning! Directing log output to the console (by using the `stdout` or `stderr` settings) can affect application performance. Avoid using these log output settings in a production environment.

Lab Mode Log Section

```
[log]
verbose = all
all = stdout, /usr/local/genesys/logfile
trace = network
```

With this configuration, an application generates log events of the `Standard`, `Interaction`, `Trace`, and `Debug` levels, and sends them to the standard output and to a file named `logfile`, which the application creates in the `/usr/local/genesys/` directory. In addition, the application sends log events of the `Standard`, `Interaction`, and `Trace` levels to Message Server. Use this configuration to test new interaction scenarios in a lab environment.

Failure-Troubleshooting Log Section

```
[log]
verbose = all
standard = network
all = memory
memory = logfile
memory-storage-size = 32 MB
```

With this configuration, an application generates log events of the `Standard` level and sends them to Message Server. It also generates log events of the `Standard`, `Interaction`, `Trace`, and `Debug` levels, and sends them to the memory output. The most current log is stored to a file named `logfile`, which the application creates in its working directory. Increased memory storage allows an application to save more of the log information generated before a failure. Use this configuration when trying to reproduce an application's failure. The memory log file will contain a snapshot of the application's log at the moment of failure; this should help you and Genesys Technical Support identify the reason for the failure.

Note: If you are running an application on UNIX, and you do not specify any files in which to store the memory output snapshot, a core file that the application produces before terminating contains the most current application log. Provide the application's core file to Genesys Technical Support when reporting a problem.

Log-Filter Section

This section must be called `log-filter`.

default-filter-type

Default Value: `copy`
Valid Values:

`copy`	The keys and values of the KVList pairs are copied to the log.
`hide`	The keys of the KVList pairs are copied to the log; the values are replaced with strings of asterisks.
`skip`	The KVList pairs are not copied to the log.

Changes Take Effect: Immediately

Specifies the default way of presenting KVList information (including `UserData`, `Extensions`, and `Reasons`) in the log. The selected option will be applied to the attributes of all KVList pairs except the ones that are explicitly defined in the `log-filter-data` section.

Example

```
[log-filter]
default-filter-type=copy
```

Here is an example of a log using the default log filter settings:

```
message RequestSetCallInfo
    AttributeConsultType        3
    AttributeOriginalConnID     008b012ece62c8be
    AttributeUpdateRevision     2752651
    AttributeUserData           [111] 00 27 01 00
            'DNIS'              '8410'
            'PASSWORD'          '111111111'
            'RECORD_ID'         '8313427'
    AttributeConnID             008b012ece62c922
```

Log-Filter-Data Section

This section must be called `log-filter-data`.

<key name>

Default Value: `copy`

Valid Values:

`copy`	The key and value of the given KVList pair are copied to the log.
`hide`	The key of the given KVList pair is copied to the log; the value is replaced with a string of asterisks.
`skip`	The KVList pair is not copied to the log.

Changes Take Effect: Immediately

Specifies the way of presenting the KVList pair defined by the key name in the log. Specification of this option supersedes the default way of KVList presentation as defined in the `log-filter` section for the given KVList pair.

Note: If the T-Server common configuration option `log-trace-flag` is set to `-udata`, it will disable writing of user data to the log regardless of settings of any options in the `log-filter-data` section.

Example

```
[log-filter-data]
PASSWORD=hide
```

Here is an example of the log with option `PASSWORD` set to `hide`:

```
message RequestSetCallInfo
   AttributeConsultType        3
   AttributeOriginalConnID     008b012ece62c8be
   AttributeUpdateRevision     2752651
   AttributeUserData           [111] 00 27 01 00
         'DNIS'                '8410'
         'PASSWORD'            '****'
         'RECORD_ID'           '8313427'
   AttributeConnID             008b012ece62c922
```

Changes from Release 7.2 to 7.5

There are no changes in common log configuration options between release 7.2 and the latest release 7.5.

Chapter

9

T-Server Common Configuration Options

This chapter describes the configuration options that are common to all T-Server types. It contains the following sections:

T-Server also supports common log options described in Chapter 8, “Common Log Options,” on page 135.

You set configuration options in Configuration Manager in the corresponding sections on the `Options` tab for the T-Server Application object.

Mandatory Options

Except as noted for certain environments, the configuration of common options is not required for basic T-Server operation.

T-Server Section

The T-Server section contains the configuration options that are used to support the core features common to all T-Servers.

This section must be called `TServer`.

user-data-limit

Default Value: `16000`
Valid Values: `0–65535`
Changes Take Effect: Immediately

Specifies the maximum size (in bytes) of user data in a packed format.

Note: When T-Server works in mixed 7.x/6.x environment, the value of this option must not exceed the default value of `16000` bytes; otherwise, 6.x T-Server clients might fail.

server-id

Default Value: An integer equal to the `ApplicationDBID` as reported by Configuration Server
Valid Values: Any integer from `0–16383`
Changes Take Effect: Immediately

Specifies the Server ID that T-Server uses to generate Connection IDs and other unique identifiers. In a multi-site environment, you must assign each T-Server a unique Server ID, in order to avoid confusion in reporting applications and T-Server behavior.

Configuration of this option is necessary for Framework environments in which there are two or more instances of the Configuration Database.

Note: If you do not specify a value for this option, T-Server populates it with the `ApplicationDBID` as reported by Configuration Server. Each data object in the Configuration Database is assigned a separate DBID that maintains a unique Server ID for each T-Server configured in the database.

Warning! Genesys does not recommend using multiple instances of the Configuration Database.

compatibility-port

Default Value: `0`
Valid Values: `0` or any valid TCP/IP port
Changes Take Effect: After T-Server has reconnected to the link

Specifies the TCP/IP port that 3.x clients use to establish connections with T-Server. Connections to this port are accepted only if T-Server has a connection with the switch. If set to `0` (zero), this port is not used.

management-port

Default Value: `0`
Valid Values: `0` or any valid TCP/IP port
Changes Take Effect: After T-Server is restarted

Specifies the TCP/IP port that management agents use to communicate with T-Server. If set to `0` (zero), this port is not used.

check-tenant-profile

Default Value: `false`
Valid Values: `true`, `false`
Changes Take Effect: For the next connected client

When set to `true`, T-Server checks whether a client provides the correct name and password of a tenant. If it does, T-Server allows that client to register DNs that are included in the switch configuration in the Configuration Database, but it does not allow the client to register DNs that are *not* included in the switch configuration.

Note: To make T-Server compatible with 3.x and 5.x clients, set the `check-tenant-profile` option to `false`.

customer-id

Default Value: No default value. (A value must be specified for a multi-tenant environment.)
Valid Values: Any character string
Changes Take Effect: Immediately

Identifies the T-Server customer. You must set this option to the name of the tenant that is using this T-Server. You must specify a value for this option if you are working in a multi-tenant environment.

Note: Do not configure the `customer-id option` for single-tenant environments.

background-timeout

Default Value: `60 msec`
Valid Values: See "Timeout Value Format" on page 170.
Changes Take Effect: Immediately

Specifies the time interval that T-Server waits before processing client requests in background mode. You must set the `background-processing` option to `true` in order for this option to take effect.

background-processing

Default Value: `false`
Valid Values: `true`, `false`
Changes Take Effect: Immediately

When set to `true`, T-Server processes all client requests in the background, giving higher priority to the rest of the messages. This ensures that it processes these messages without any significant delay.

With Background Processing functionality enabled, T-Server processes all switch messages immediately and wait until there are no switch messages before processing the message queue associated with T-Server client requests. T-Server reads all connection sockets immediately and places client requests in the input buffer, which prevents T-Server clients from disconnecting because of configured timeouts.

When T-Server processes client requests from the message queue, requests are processed in the order in which T-Server received them.

When set to `false`, T-Server processes multiple requests from one T-Server client before proceeding to the requests from another T-Server client, and so on.

Note: Use of this option can negatively impact T-Server processing speed.

log-trace-flags

Default Value: `+iscc, +cfg$dn, -cfgserv, +passwd, +udata, -devlink, -sw, -req, -callops, -conn, -client`

Valid Values (in any combination):

`+/-iscc`	Turns on/off the writing of information about Inter Server Call Control (ISCC) transactions.
`+/-cfg$dn`	Turns on/off the writing of information about DN configuration.
`+/-cfgserv`	Turns on/off the writing of messages from Configuration Server.
`+/-passwd`	Turns on/off the writing of information about passwords.
`+/-udata`	Turns on/off the writing of attached data.
`+/-devlink`	Turns on/off the writing of information about the link used to send CTI messages to the switch (for multilink environments).
`+/-sw`	Reserved by Genesys Engineering.
`+/-req`	Reserved by Genesys Engineering.
`+/-callops`	Reserved by Genesys Engineering.
`+/-conn`	Reserved by Genesys Engineering.
`+/-client`	Turns on/off the writing of additional information about the client’s connection.

Changes Take Effect: Immediately

Specifies—using a space-, comma- or semicolon-separated list—the types of information that are written to the log files.

consult-user-data

Default Value: `separate`
Valid Values:

`separate`	Stores user data for original and consultation calls in separate structures. The data attached to the original call is available for review or changes only to the parties of that call. The data attached to the consultation call is available only to the parties of the consultation call.
`inherited`	Copies user data from an original call to a consultation call when the consultation call is created; thereafter, stores user data separately for the original and the consultation call. Changes to the original call's user data are not available to the parties of the consultation call, and vice versa.
`joint`	Stores user data for an original call and a consultation call in one structure. The user data structure is associated with the original call, but the parties of both the original and consultation calls can see and make changes to the common user data.

Changes Take Effect: For the next consultation call created

Specifies the method for handling user data in a consultation call.

Note: A T-Server client can also specify the `consult-user-data` mode in the `Extensions` attribute `ConsultUserData` key for a conference or transfer request. If it is specified, the method of handling user data is based on the value of the `ConsultUserData` key-value pair of the request and takes precedence over the T-Server `consult-user-data` option. If it is not specified in the client request, the value specified in the `consult-user-data` option applies.

merged-user-data

Default Value: `main-only`
Valid Values:

`main-only`	T-Server attaches user data from the remaining call only.
`merged-only`	T-Server attaches user data from the merging call.
`merged-over-main`	T-Server attaches user data from the remaining and the merging call. In the event of equal keys, T-Server uses data from the merging call.
`main-over-merged`	T-Server attaches data from the remaining and the merging call. In the event of equal keys, T-Server uses data from the remaining call.

Changes Take Effect: Immediately

Specifies the data that is attached to the resulting call after a call transfer, conference, or merge completion.

Note: The option setting does not affect the resulting data for merging calls if the `consult-user-data` option is set to `joint`. (See "consult-user-data" on page 153.)

License Section

The License section contains the configuration options that are used to configure T-Server licenses. They set the upper limit of the seat-related DN licenses (`tserver_sdn`) that T-Server tries to check out from a license file. See "License Checkout" on page 155.

This section must be called `license`.

Notes:

- T-Server also supports the `license-file` option described in the *Genesys 7 Licensing Guide*.
- The License section is not applicable to Network T-Server for DTAG.

If you use two or more T-Servers, and they share licenses, you must configure the following options in the `license` section of the T-Servers.

num-of-licenses

Default Value: `0` or `max` (all available licenses)
Valid Values: `0` or string `max`
Changes Take Effect: Immediately

Specifies how many DN licenses T-Server checks out. T-Server treats a value of `0` (zero) the same as it treats `max`—that is, it checks out all available licenses.

The sum of all `num-of-licenses` values for all concurrently deployed T-Servers must not exceed the number of seat-related DN licenses (`tserver_sdn`) in the corresponding license file. The primary and backup T-Servers share the same licenses, and therefore they need to be counted only once. T-Server checks out the number of licenses indicated by the value for this option, regardless of the number actually in use.

num-sdn-licenses

Default Value: `0` or `max` (All DN licenses are seat-related)
Valid Values: String `max` (equal to the value of `num-of-licenses`), or any integer from `0`–`9999`
Changes Take Effect: Immediately

Specifies how many seat-related licenses T-Server checks out. A value of `0` (zero) means that T-Server does not grant control of seat-related DNs to any client, and it does not look for seat-related DN licenses at all.

The sum of all `num-sdn-licenses` values for all concurrently deployed T-Servers must not exceed the number of seat-related DN licenses (`tserver_sdn`) in the corresponding license file. The primary and backup T-Servers share the same licenses, and therefore they need to be counted only once. T-Server checks out the number of licenses indicated by the value for this option, regardless of the number actually in use.

Note: For Network T-Servers, Genesys recommends setting this option to `0`.

Note: Be sure to configure in the Configuration Database all the DNs that agents use (Extensions and ACD Positions) and that T-Server should control. For further information, see Chapter 2, "DNs and Agent Logins," page 41.

License Checkout

Table 12 shows how to determine the number of seat-related DN licenses that T-Server attempts to check out. See the examples on page 156.

Table 12: License Checkout Rules

Options Settings[a]		License Checkout[b]
num-of-licenses	**num-sdn-licenses**	**Seat-related DN licenses**
max (or 0)	max	all available
max (or 0)	x	x
max (or 0)	0	0
x	max	x
x	y	min (y, x)
x	0	0

a. In this table, the following conventions are used: `x` and `y` - are positive integers; `max` is the maximum number of licenses that T-Server can check out; `min (y, x)` is the lesser of the two values defined by `y` and `x`, respectively.

b. The License Checkout column shows the number of licenses that T-Server attempts to check out. The actual number of licenses will depend on the licenses' availability at the time of checkout, and it is limited to 9999.

Example 1

If...		Then...
Options Settings	**License File Settings**	**License Checkout**
num-of-licences = max	tserver_sdn = 500	500 seat-related DNs
num-sdn-licences = max		

Example 2

If...		Then...
Options Settings	**License File Settings**	**License Checkout**
num-of-licences = 1000	tserver_sdn = 500	500 seat-related DNs
num-sdn-licences = max		

Example 3

If...		Then...
Options Settings	**License File Settings**	**License Checkout**
num-of-licences = 1000	tserver_sdn = 600	400 seat-related DNs
num-sdn-licences = 400		

Example 4

If...		Then...
Options Settings	**License File Settings**	**License Checkout**
num-of-licences = max	tserver_sdn = 5000	1000 seat-related DNs
num-sdn-licences = 1000		

Agent-Reservation Section

The Agent-Reservation section contains the configuration options that are used to customize the T-Server Agent Reservation feature.

This section must be called `agent-reservation`.

Note: The Agent Reservation functionality is currently a software-only feature that is used to coordinate multiple client applications. This feature does not apply to multiple direct or ACD-distributed calls.

request-collection-time

Default Value: `100 msec`
Valid Values: See “Timeout Value Format” on page 170.
Changes Take Effect: Immediately

Specifies the interval that agent reservation requests are collected before a reservation is granted. During this interval, agent reservation requests are delayed, in order to balance successful reservations between client applications (for example, Universal Routing Servers).

reservation-time

Default Value: `10000 msec`
Valid Values: See “Timeout Value Format” on page 170.
Changes Take Effect: Immediately

Specifies the default interval that an AgentDN is reserved to receive a routed call from a remote T-Server. During this interval, the agent cannot be reserved again.

reject-subsequent-request

Default Value: `true`
Valid Values:

`true`	T-Server rejects subsequent requests.
`false`	A subsequent request prolongs the current reservation made by the same client application for the same agent.

Changes Take Effect: Immediately

Specifies whether T-Server rejects subsequent requests from the same client application, for an agent reservation for the same `Agent` object that is currently reserved.

Note: Genesys does not recommend setting this option to `false` in a multi-site environment in which remote locations use the Agent-Reservation feature.

Multi-Site Support Section

The Multi-Site Support section contains the configuration options that are used to support multi-site environments with the Inter Server Call Control (ISCC) feature. The configuration options in this section are grouped with related options that support the same functionality (such as those for Transfer Connect Service or the ISCC/Call Overflow feature).

This section must be called `extrouter`.

For a description of the ways in which T-Server supports multi-site configurations and for an explanation of the configuration possibilities for a multi-site operation, see the "Multi-Site Support" chapter.

Note: In a multi-site environment, you must configure the `timeout`, `cast-type`, and `default-dn` options with the same value for both the primary and backup T-Servers. If you do not do this, the value specified for the backup T-Server overrides the value specified for the primary T-Server.

reconnect-tout

Default Value: `5 sec`
Valid Values: See "Timeout Value Format" on page 170.
Changes Take Effect: At the next reconnection attempt

Specifies the time interval after which a remote T-Server attempts to connect to this T-Server after an unsuccessful attempt or a lost connection. The number of attempts is unlimited. At startup, T-Server immediately attempts the first connection, without this timeout.

use-data-from

Default Value: `active`
Valid Values:

`active`	The `UserData` and `ConnID` attributes are taken from the consultation call.
`original`	The `UserData` and `ConnID` attributes are taken from the original call.
`consult-user-data`	If the value of `consult-user-data` is specified, the following occurs: • Before the transfer or conference is completed, the `UserData` and `ConnID` attributes are taken from the consultation call. • After the transfer or conference is completed, `EventPartyChanged` is generated, and the `UserData` and `ConnID` are taken from the original call.

Changes Take Effect: Immediately

Specifies the call from which the values for the `UserData` and `ConnID` attributes should be taken for a consultation call that is routed or transferred to a remote location.

Note: For compatibility with the previous T-Server releases, you can use the values `consult`, `main`, and `current` for this option. These are aliases for `active`, `original`, and `consult-user-data`, respectively.

report-connid-changes

Default Value: `false`
Valid Values:

`true`	`EventPartyChanged` is generated.
`false`	`EventPartyChanged` is not generated.

Changes Take Effect: Immediately

Specifies whether the destination T-Server generates `EventPartyChanged` for the incoming call when the resulting `ConnID` attribute is different from the `ConnID` attribute of an instance of the same call at the origination location.

match-call-once

Default Value: `true`
Valid Values:

`true`	ISCC does not process (match) an inbound call that has already been processed (matched).
`false`	ISCC processes (matches) a call as many times as it arrives at an ISCC resource or multi-site-transfer target.

Changes Take Effect: Immediately

Specifies how ISCC processes an inbound call that has already been processed.

ISCC Transaction Options

request-tout

Default Value: `20 sec`
Valid Values: See “Timeout Value Format” on page 170.
Changes Take Effect: For the next request for remote service

Specifies the time interval that a T-Server at the origination location waits for a notification of routing service availability from the destination location. Counting starts when the T-Server sends a request for remote service to the destination site.

network-request-timeout

Default Value: `20 sec`
Valid Values: See “Timeout Value Format” on page 170.
Changes Take Effect: For the next network request

For a premise T-Server, this option specifies the time interval that the premise T-Server waits for a response, after relaying a `TNetwork<...>` request to the Network T-Server. For a Network T-Server, this option specifies the time interval that the Network T-Server waits for a response from an SCP (Service Control Point), after initiating the processing of the request by the SCP.

When the allowed time expires, the T-Server cancels further processing of the request and generates `EventError`.

timeout

Default Value: `60 sec`
Valid Values: See "Timeout Value Format" on page 170.
Changes Take Effect: For the next request for remote service

Specifies the time interval that the destination T-Server waits for a call routed from the origination location. Counting starts when this T-Server notifies the requesting T-Server about routing service availability. The timeout must be long enough to account for possible network delays in call arrival.

cast-type

Default Value: `route, route-uui, reroute, direct-callid, direct-uui, direct-network-callid, direct-notoken, direct-digits, direct-ani, dnis-pool, pullback`
Valid Values: `route, route-uui, reroute, direct-callid, direct-uui, direct-network-callid, direct-notoken, direct-digits, direct-ani, dnis-pool, pullback`
Changes Take Effect: For the next request for the remote service

Specifies—using a space-, comma- or semicolon-separated list—the routing types that can be performed for this T-Server.

The valid values provide for a range of mechanisms that the ISCC feature can support with various T-Servers, in order to pass call data along with calls between locations.

Because switches of different types provide calls with different sets of information parameters, some values might not work with your T-Server. See Table 3 on page 63 for information about supported transaction types by a specific T-Server. The "Multi-Site Support" chapter also provides detailed descriptions of all transaction types.

Notes: For compatibility with the previous T-Server releases, you can use the `direct` value for this option. This is an alias for `direct-callid`.

An alias, `route-notoken`, has been added to the `route` value.

direct-digits-key

Default Value: `CDT_Track_Num`
Valid Values: Any valid key name of a key-value pair from the `UserData` attribute
Changes Take Effect: For the next request for the remote service

Specifies the name of a key from the `UserData` attribute that contains a string of digits that are used as matching criteria for remote service requests with the `direct-digits` routing type.

Note: For compatibility with the previous T-Server releases, this configuration option has an alias value of `cdt-udata-key`.

---

default-dn

Default Value: No default value
Valid Values: Any DN
Changes Take Effect: For the next request for the remote service

Specifies the DN to which a call is routed when a Destination DN (`AttributeOtherDN`) is not specified in the client's request for routing. If neither this option nor the client's request contains the destination DN, the client receives `EventError`.

Note: This option is used only for requests with route types `route`, `route-uui`, `direct-callid`, `direct-network-callid`, `direct-uui`, `direct-notoken`, `direct-digits`, and `direct-ani`.

---

register-tout

Default Value: `2 sec`
Valid Values: See "Timeout Value Format" on page 170.
Changes Take Effect: For the next registration

Specifies the time interval after which T-Server attempts to register a dedicated External Routing Point. Counting starts when the attempt to register a Routing Point fails.

register-attempts

Default Value: `5`
Valid Values: Any positive integer
Changes Take Effect: For the next registration

Specifies the number of attempts that T-Server makes to register a dedicated External Routing Point.

route-dn

Default Value: No default value
Valid Values: Any DN
Changes Take Effect: Immediately

Specifies the DN that serves as a Routing Point for the `route` transaction type in the multiple-to-one access mode.

dn-for-unexpected-calls

Default Value: No default value
Valid Values: Any DN
Changes Take Effect: Immediately

Specifies a default DN for unexpected calls arriving on an External Routing Point.

resource-allocation-mode

Default Value: `circular`
Valid Values:

`home`	T-Server takes an alphabetized (or numerically sequential) list of configured DNs and reserves the first available DN from the top of the list for each new request. For example, if the first DN is not available, the second DN is allocated for a new request. If the first DN is freed by the time the next request comes, the first DN is allocated for this next request.
`circular`	T-Server takes the same list of configured DNs, but reserves a subsequent DN for each subsequent request. For example, when the first request comes, T-Server allocates the first DN; when the second request comes, T-Server allocates the second DN; and so on. T-Server does not reuse the first DN until reaching the end of the DN list.

Changes Take Effect: Immediately

Specifies the manner in which T-Server allocates resources (that is, DNs of the `External Routing Point` type and Access Resources with `Resource Type dnis`) for multi-site transaction requests.

resource-load-maximum

Default Value: `0`
Valid Values: Any positive integer
Changes Take Effect: Immediately

Specifies the maximum number of ISCC routing transactions that can be concurrently processed at a single DN of the `External Routing Point` route type. After a number of outstanding transactions at a particular DN of the `External Routing Point` type reaches the specified number, T-Server considers the DN not available. Any subsequent request for this DN is queued until the number of outstanding transactions decreases. A value of `0` (zero) means that no limitation is set to the number of concurrent transactions at a single External Routing Point. In addition, the `0` value enables T-Server to perform load balancing of all incoming requests among all available External Routing Points, in order to minimize the load on each DN.

use-implicit-access-numbers

Default Value: `false`
Valid Values: `true`, `false`

Changes Take Effect: Immediately

Determines whether an External Routing Point in which at least one access number is specified is eligible for use as a resource for calls coming from switches for which an access number is not specified in the External Routing Point. If this option is set to `false`, the External Routing Point is not eligible for use as a resource for calls coming from such switches. If this option is set to `true`, an implicit access number for the External Routing Point, composed of the switch access code and the DN number of the External Routing Point, will be used.

Note: If an External Routing Point does not have an access number specified, this option will not affect its use.

Transfer Connect Service Options

tcs-use

Default Value: `never`
Valid Values:

`never`	The TCS feature is not used.
`always`	The TCS feature is used for every call.
`app-defined`	In order to use the TCS feature for a multi-site call transfer request, a client application must add a key-value pair with a TC-type key and a nonempty string value to the `UserData` attribute of the request.

Changes Take Effect: Immediately

Specifies whether the Transfer Connect Service (TCS) feature is used.

Note: For compatibility with the previous T-Server releases, you can use the value `up-app-depended` for this option. This is an alias for `app-defined`.

tcs-queue

Default Value: No default value
Valid Values: Any valid DN number
Changes Take Effect: For the next request for the remote service

Specifies the TCS DN number to which a call, processed by the TCS feature, is dialed after the originating external router obtains an access number.

ISCC/COF Options

cof-feature

Default Value: `false`
Valid Values: `true`, `false`

Changes Take Effect: Immediately

Enables or disables the Inter Server Call Control/Call Overflow (ISCC/COF) feature.

cof-ci-req-tout

Default Value: `500 msec`
Valid Values: See "Timeout Value Format" on page 170.
Changes Take Effect: For the next COF operation

Specifies the time interval during which T-Server will wait for call data requested with respect to a call originated at another site. After T-Server sends the call data request to remote T-Servers, all events related to this call will be suspended until either the requested call data is received or the specified timeout expires.

cof-rci-tout

Default Value: `10 sec`
Valid Values: See "Timeout Value Format" on page 170.
Changes Take Effect: For the next COF operation

Specifies the time interval that T-Server waits for call data from other T-Servers' transactions. Counting starts when `cof-ci-req-tout` expires.

cof-ci-wait-all

Default Value: `false`
Valid Values:

`true`	T-Server waits for responses from all T-Servers that might have the requested call data before updating the call data with the latest information.
`false`	T-Server updates the call data with the information received from the first positive response.

Changes Take Effect: Immediately

Specifies whether T-Server, after sending a request for matching call data, waits for responses from other T-Servers before updating the call data (such as `CallHistory`, `ConnID`, and `UserData`) for a potentially overflowed call. The waiting period is specified by the `cof-ci-req-tout` and `cof-rci-tout` options.

cof-ci-defer-delete

Default Value: `0`
Valid Values: See "Timeout Value Format" on page 170.
Changes Take Effect: Immediately

Specifies the time interval that T-Server waits before deleting call data that might be overflowed. If set to `0`, deletion deferring is disabled.

cof-ci-defer-create

Default Value: `0`
Valid Values: See "Timeout Value Format" on page 170.
Changes Take Effect: Immediately

Specifies the time interval that T-Server waits for call data from the switch before generating a negative response for a call data request from a remote T-Server. If T-Server detects the matching call before this timeout expires, it sends the requested data.

local-node-id

Default Value: `0`
Valid Values: `0` or any positive integer
Changes Take Effect: Immediately

This option, if enabled, checks all networked calls against the specified `NetworkNodeID` (the identity of the switch to which the call initially arrived). If the `NetworkNodeID` is the same as the value of this option, the request for call information is *not* sent. The default value of `0` disables the functionality of this option. To establish an appropriate `NetworkNodeID`, specify a value other than the default.

Note: This option applies only to T-Server for Nortel Communication Server 2000/2100 (formerly DMS-100).

Event Propagation Option

event-propagation

Default Value: `list`
Valid Values:

`list`	Changes in user data and party events are propagated to remote locations through call distribution topology.
`off`	The feature is disabled. Changes in user data and party events are not propagated to remote locations.

Changes Take Effect: Immediately

Specifies whether the Event Propagation feature is enabled.

Number Translation Option

inbound-translator-<*n*>

Default Value: No default value.
Valid Value: Any valid name
Changes Take Effect: Immediately

Specifies the name of another configuration section as the value for the `inbound-translator` option. For example,

```
inbound-translator-1 = ani-translator
```

where `ani-translator` is the name of the configuration that describes the translation rules for inbound numbers.

Translation Rules Section

The section name is specified by the `inbound-translator-<n>` option. It contains options that define translation rules for inbound numbers.

You can choose any name for this section, provided that it matches the value of the section. Every option in this section corresponds to a rule and must conform to the format described below. You can configure as many rules as necessary to accommodate your business needs.

rule-<*n*>

Default Value: No default value
Valid Value: Any valid string in the following format:
`in-pattern=<input pattern value>;out-pattern=<output pattern value>`
Changes Take Effect: Immediately

Defines a rule to be applied to an inbound number. The two parts of the option value describe the input and output patterns in the rule. When configuring the pattern values, follow the syntax defined in “Using ABNF for Rules” on page 72. See “Configuration Procedure” on page 78 for examples of these rules as well as detailed instructions for creating rules for your installation. For example, a value for this configuration option might look like this:

```
rule-01 = in-pattern=0111#CABBB*ccD;out-pattern=ABD
```

Backup-Synchronization Section

The Backup-Synchronization section contains the configuration options that are used to support a high-availability (`hot standby` redundancy type) configuration.

This section must be called `backup-sync`.

Note: These options apply only to T-Servers that support the `hot standby` redundancy type.

sync-reconnect-tout

Default Value: `20 sec`
Valid Values: See "Timeout Value Format" on page 170.
Changes Take Effect: Immediately

Specifies the time interval after which the backup T-Server attempts to reconnect to the primary server (for a synchronized link).

protocol

Default Value: `default`
Valid Values:

`default`	The feature is not active.
`addp`	Activates the Advanced Disconnect Detection Protocol.

Changes Take Effect: When the next connection is established

Specifies the name of the method used to detect connection failures. If you specify the `addp` value, you must also specify a value for the `addp-timeout`, `addp-remote-timeout`, and `addp-trace` options.

addp-timeout

Default Value: `0`
Valid Values: Any integer from `0–3600`
Changes Take Effect: Immediately

Specifies the time interval that this T-Server waits for a response from another T-Server after sending a polling signal. The default value of `0` (zero) disables the functionality of this option. To establish an appropriate timeout, specify a value other than the default. This option applies only if the `protocol` option is set to `addp`.

addp-remote-timeout

Default Value: `0`
Valid Values: Any integer from `0–3600`
Changes Take Effect: Immediately

Specifies the time interval that the redundant T-Server waits for a response from this T-Server after sending a polling signal. The default value of `0` (zero) disables the functionality of this option. To establish an appropriate timeout, specify a value other than the default. This option applies only if the `protocol` option is set to `addp`.

addp-trace

Default Value: `off`
Valid Values:

`off`, `false`, `no`	No trace (default).
`local`, `on`, `true`, `yes`	Trace on this T-Server side only.
`remote`	Trace on the redundant T-Server side only.
`full`, `both`	Full trace (on both sides).

Changes Take Effect: Immediately

Specifies whether the option is active, and to what level the trace is performed. This option applies only if the `protocol` option is set to `addp`.

network-provided-address

Default Value: `false`
Valid Values:

`false`	T-Server reports the backup host information as configured in the Configuration Layer.
`true`	T-Server reports the backup host information as supplied by the network.

Changes Take Effect: Immediately

Specifies how T-Server reports to its clients the host information about its backup T-Server.

Call-Cleanup Section

The Call-Cleanup section contains the configuration options that are used to control detection and cleanup of stuck calls in T-Server. For more information on stuck call handling, refer to the “Stuck Call Management” chapter in the *Framework 7.5 Management Layer User’s Guide*.

This section must be called `call-cleanup`.

notify-idle-tout

Default Value: `0`
Valid Values: See “Timeout Value Format” on page 170.
Changes Take Effect: Immediately

Specifies the time interval that T-Server waits for a call to be updated from its last update. After this time elapses, if no new events about the call are received, T-Server reports this call as a stuck call. The default value of `0` disables the stuck calls notification.

cleanup-idle-tout

Default Value: `0`
Valid Values: See “Timeout Value Format” on page 170.

Changes Take Effect: Immediately

Specifies the time interval that T-Server waits for a call to be updated from its last update. After this time elapses, if no new events about the call are received, T-Server clears this call as a stuck call, either by querying the switch (if a CTI link provides such capabilities) or by deleting the call information from memory unconditionally. The default value of `0` disables the stuck calls cleanup.

periodic-check-tout

Default Value: `10 min`
Valid Values: See “Timeout Value Format” on page 170.
Changes Take Effect: Immediately

Specifies the time interval for periodic checks for stuck calls. These checks affect both notification and cleanup functionality, and are made by checking the T-Server’s own call information with call information available in the switch. For performance reasons, T-Server does not verify whether the `notify-idle-tout` or `cleanup-idle-tout` option has expired before performing this checking.

Note: Setting this option to a value of less than a few seconds can affect T-Server performance.

Example 1

```
notify-idle-tout = 0
cleanup-idle-tout = 0
periodic-check-tout = 10
```

With these settings, T-Server will not perform any checks for stuck calls.

Example 2

```
notify-idle-tout = 5 min
cleanup-idle-tout = 0
periodic-check-tout = 10 min
```

With these settings, T-Server performs checks every 10 minutes and sends notifications about all calls that have been idle for at least 5 minutes.

Example 3

```
notify-idle-tout = 5 min
cleanup-idle-tout = 20 min
periodic-check-tout = 10 min
```

With these settings, T-Server performs checks every 10 minutes, sends notifications about all calls that have been idle for at least 5 minutes, and attempts to clean up all calls that have been idle for more than 20 minutes.

Security Section

The Security section contains the configuration options that are used to configure secure data exchange between T-Servers and other Genesys components. Refer to the *Genesys 7.5 Transport Layer Security Deployment Guide* for complete information on the security configuration.

Timeout Value Format

This section of the document describes the values to use for those T-Server common options that set various timeouts. The current format allows you to use fractional values and various time units for timeout settings.

For timeout-related options, you can specify any value that represents a time interval, provided that it is specified in one of the following formats:

`[[[`*`hours`*`:]`*`minutes`*`:]`*`seconds`*`][`*`milliseconds`*`]`

or

`[`*`hours`*` hr][`*`minutes`*` min][`*`seconds`*` sec][`*`milliseconds`*` msec]`

Where a time unit name in italic (such as *hours*) is to be replaced by an integer value for this time unit.

Integer values with no measuring units are still supported, for compatibility with previous releases of T-Server. When you do not specify any measuring units, the units of the default value apply. For example, if the default value equals `60 sec`, specifying the value of `30` sets the option to 30 seconds.

Example 1

The following settings result in a value of 1 second, 250 milliseconds:

```
sync-reconnect-tout = 1.25
sync-reconnect-tout = 1 sec 250 msec
```

Example 2

The following settings result in a value of 1 minute, 30 seconds:

```
timeout = 1:30
timeout = 1 min 30 sec
```

Changes from Release 7.2 to 7.5

Table 13 lists the configuration options that:

- Are new or changed in the 7.5 release of T-Server
- Have been added or changed since the most recent 7.2 release of this document

If a configuration option has been replaced with another that enables the same functionality, the new option name and its location in this chapter are noted.

Table 13: Option Changes from Release 7.2 to 7.5

Option Name	Option Values	Type of Change	Details
Security Section (New in 7.5)			
certificate	Specifies the platform-dependent certificate parameters related to the TLS configuration	New	See the *Genesys 7.5 Transport Layer Security Deployment Guide* for complete information on the security configuration
certificate-key			
trusted-ca			

Chapter

10 Configuration Options in Network SIP Server

This chapter describes configuration options unique to the Network SIP Server and includes these sections:

T-Server Section

This section describes configuration options used to support unique Network SIP Server features. Configure these options in the `TServer` section on the `Options` tab for the T-Server Application object in the Configuration Layer.

sip-port

Default Value: `5060`
Valid Value: Any valid TCP/IP port
Changes Take Effect: After T-Server is restarted

Specifies the TCP/IP and UDP port where T-Server listens for requests from SIP client (switch).

t1-timeout

Default Value: `500`
Valid Value: Any integer > = `500`
Changes Take Effect: Immediately

Determines the round-trip estimate (in msec). This option value is the same as the T1 timeout described in the Network Working Group, Request for Comments (RFC): 3261 Category: Standards Track, Rosenberg, et. al., June 2002.

t2-timeout

Default Value: `4000`
Valid Value: Any integer > = `4000`
Changes Take Effect: Immediately

Specifies the maximum retransmit interval (in msec) for `non-INVITE` request and `INVITE` request responses. The option value is the same as T2 timeout described in the RFC3261.

router-timeout

Default Value: `30000`
Valid Value: Any integer > `200`
Changes Take Effect: Immediately

Specifies the maximum time interval (in msec) during which T-Server expects response from the Genesys Universal Routing Server.

ack-timeout

Default Value: `500`
Valid Value: Any integer > `200`
Changes Take Effect: Immediately

Currently this option is not used. Reserved for Genesys Engineering.

noanswer-timeout

Default Value: `32000`
Valid Value: Any integer > `200`
Changes Take Effect: Immediately

Currently this option is not used. Reserved for Genesys Engineering.

udp-packets-to-read

Default Value: `1`
Valid Value: Any integer > `200`
Changes Take Effect: Immediately

Specifies the size of the buffer (in bytes) where UDP packets to read are kept.

udp-packets-to-write

Default Value: `1`
Valid Value: Any integer > `200`
Changes Take Effect: Immediately

Specifies the size of the buffer (in bytes) where UDP packets to write are kept.

sip-routing

Default Value: `redirect`
Valid Value: `redirect`
Changes Take Effect: Immediately

Specifies which SIP method is used to route a call.

Note: Reserved for future use. Do not modify this option.

udp-recvbuf

Default Value: `40960`
Valid Value: `8000-1000000`
Changes Take Effect: Immediately

Specifies the size of the buffer (in bytes) where received UDP packets are kept.

Note: Increase this value to reduce the percentage of lost UDP packets and retransmissions of the switch requests because of lost packets.

udp-sendbuf

Default Value: `40960`
Valid Value: `8000-1000000`
Changes Take Effect: Immediately

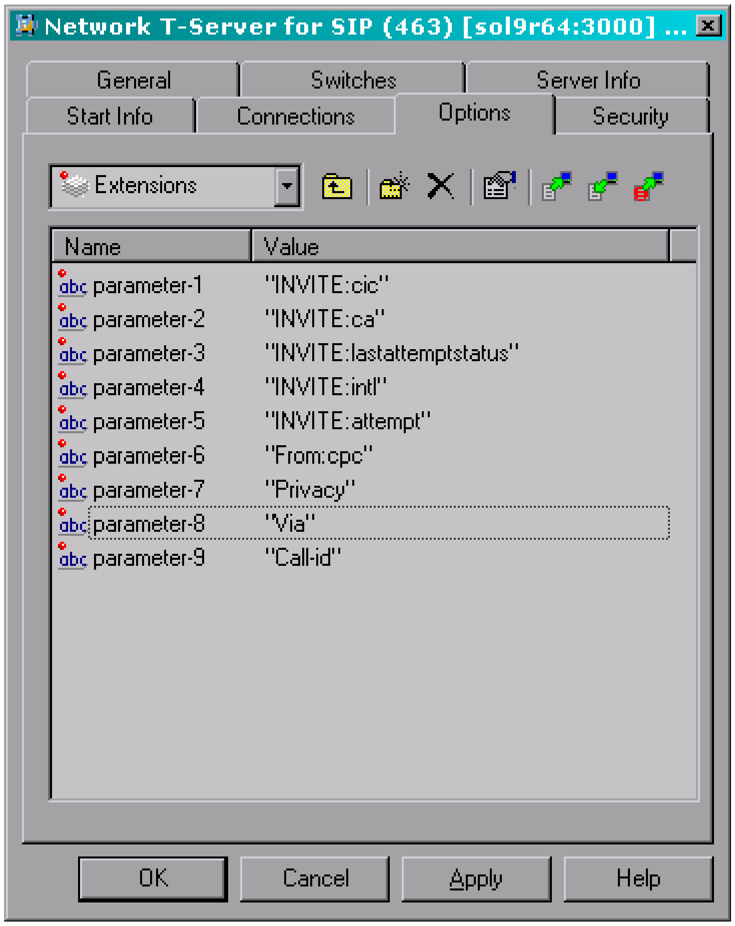
Specifies the size of the buffer (in bytes) where sent UDP packets are kept.

Extensions Section

This section describes configuration options used to make specific SIP headers or header parameters available for URS and other applications. Configure these options in the `Extensions` section on the `Options` tab for the T-Server Application object in the Configuration Layer.

The `Extensions` section lets you configure which additional parameters are mapped from the SIP `INVITE` message to the `AttributeExtensions` of `EventRouteRequest`. This flexible mechanism enables easy retrieval of any additional headers or header parameters from the SIP message and passes them to `EventRouteRequest`. See the “Mapping SIP Parameters to T-Server Event Parameters” on page 119 for more information.

You must name all options created in this section using the form `parameter-`*`n`* where *`n`* is an arbitrary decimal number unique for this section. Figure 11 on page 176 shows an example of the `Extensions` parameters.



Name	Value
parameter-1	"INVITE:cic"
parameter-2	"INVITE:ca"
parameter-3	"INVITE:lastattemptstatus"
parameter-4	"INVITE:intl"
parameter-5	"INVITE:attempt"
parameter-6	"From:cpc"
parameter-7	"Privacy"
parameter-8	"Via"
parameter-9	"Call-id"

Figure 11: Example of the Extensions Parameters

parameter-*n*

Default Value: None
Valid Value: Any character string
Changes Take Effect: Immediately

The value of the `parameter-n` option must start with the name of the SIP header followed by a colon and the name of the `Extension` parameter. If the parameter is taken from the SIP `Request URI`, use the word *INVITE* as the name of the header. If the colon and name of the parameter are absent, put the whole value of the SIP header inside `Attribute Extensions`.

If a parameter configured by the option within the `Extensions` section persists inside the SIP request, it is added as a Key-Value pair inside `AttributeExtensions`. The Key in this pair is the same as the value of the corresponding option inside the `Extensions` section of the configuration.

Changes from 7.2 to 7.5

There are no new or changed options since the previous release of Network SIP Server.

Index