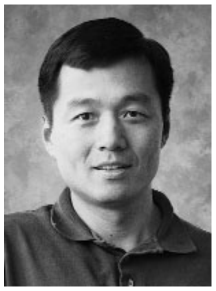
A system administrator's practical guide to SAP System Landscape Directory (SLD)

by George Yu



George Yu
Solution Technology Architect,
SAP Labs

Dr. George Yu is a solution technology architect in SCM/RFID/PLM Solution Management at SAP Labs. He has 20 years of experience in software application research, development, and consulting. George joined SAP in 1998, and is a leading technical expert in the areas of APO system architecture, liveCache database performance and tuning, R/3 Enterprise, Web Application Server, J2EE Engine, CRM, XI, Business Process Management, and Auto-ID Infrastructure. He has been widely involved in customer consultations, conference presentations, and classroom instruction within the SAP community in the Americas, Asia, and Europe. You can reach him at george.yu@sap.com.

With the addition of myriad new solutions such as SAP Supply Chain Management (SCM), Customer Relationship Management (CRM), Sales Automation, and many others, the size of SAP system landscapes has mushroomed — it is not uncommon to see 20+ systems in the data center of a midsized company, or 100+ systems in a multinational enterprise. System administrators often use a Microsoft Excel spreadsheet or a small database program to manually keep track of all the systems in these landscapes, but this approach is limited. Let's say that you are running SAP R/3 4.0B and are planning to implement SAP SCM 4.1 — the spreadsheet won't tell you whether these two systems are compatible, and if they are, at what support package level. Tracking the systems in a multinational enterprise, where each division often has its own set of systems, compounds the challenge even further.

The SAP System Landscape Directory (SLD) of SAP NetWeaver answers this need. SLD serves as a central repository of technical information on the systems installed in your landscape, including release information and interdependencies. SLD was initially released as part of SAP Exchange Infrastructure (SAP XI) 1.0 to store the information required by SAP XI to integrate systems across a landscape. As of SAP Web Application Server (SAP Web AS) 6.40, SLD runs on SAP Web AS Java as a service of the SAP J2EE Engine. With this architectural change, additional applications that require system information, such as SAP Solution Manager, can use the same, centralized information repository instead of using separate repositories, saving time and simplifying maintenance.

While the standard documentation covers the details of SLD installation and configuration steps in the context of the solutions it supports, this article intends to give system administrators, Basis consultants, and project managers a practical overview of how SLD works across the system landscape, enabling you to draw an effective plan for installing and configuring it, and to quickly identify and resolve any potential problems, reducing the overall project implementation time and cost. Along the way I present some best practices, gleaned from my own experiences, for key decision points in the installation and configuration process to help set you up for success with your own SLD configuration.

Note!

This article applies to SAP Web AS 6.40 and SAP NetWeaver Application Server 2004s,[1] and assumes familiarity with the SAP J2EE Engine and SAP XI.[2] Any differences between the 6.40 and 2004s versions of SLD are highlighted.

Let's start by examining the architecture that enables SLD to work with applications across your system landscape, regardless of whether they are ABAP, Java, or even third-party applications.

SLD architecture

SLD is a server application, commonly referred to as the SLD server. It operates based on a client/server architecture. The SLD server collects system information from its data suppliers, such as SAP R/3, SAP NetWeaver Business Intelligence (SAP NetWeaver BI), and SAP Web AS Java systems in the landscape, and provides this information to the SLD clients requesting it, such as SAP XI and SAP Solution Manager systems, regardless of whether the system is Java-based or ABAP-based. To accomplish this, SLD includes both Java and ABAP APIs to access SLD content over the HTTP or HTTPS protocol. Since SLD is a Java application, the ABAP API is implemented on top of the Java API using a Java Connector (JCo) RFC Provider service, which I'll discuss in detail later in the article. Method invocations and data exchanges are wrapped in WBEM-based XML messages.[3]

Figure 1 shows how SLD works. SAP maintains a Product and Production Management System (PPMS) that contains information on all SAP software products and components, including their release versions, maintenance schedules, and support packages. This information is periodically synchronized with a master Component Repository (master CR) in XML format. Customers download the contents of the master CR from the Software Distribution Center at the SAP Service Marketplace (http://service.sap.com/swdc) to keep SAP component information in SLD up to date (see SAP Note 669669). Customers can also add third-party components to SLD manually, so that SLD has a complete picture of all the systems in your landscape, regardless or whether they are SAP or non-SAP applications.

SLD content is organized according to the Common Information Model (CIM) open standard, which is a system information schema of class definitions developed by the Distributed Management Task Force (www.dmtf.org/standards/cim/). The CIM standard describes a common model for information on various areas in a managed enterprise environment, including system, database, device, network, user/security, and application areas. Information is stored in SLD as instances of CIM-based classes — for example, the product mySAP CRM is stored in SLD as an instance of the CIM class SAP_Product. The use of a common data structure enables SLD to store third-party system information as well as SAP system information, and enables the information to be shared

[1] As of SAP NetWeaver 2004s, SAP Web Application Server is known as SAP NetWeaver Application Server. For simplicity, I use the term SAP Web AS to refer collectively to both throughout the article.

[2] For a detailed introduction to SAP XI, see the *SAP Professional Journal* articles "A Beginner's Guide to Implementing SAP Exchange Infrastructure (SAP XI) — Paving the Way to Seamless Integration" (March/April 2005) and "A Beginner's Guide to Implementing SAP Exchange Infrastructure (SAP XI) — Designing and Configuring an SAP XI Integration" (May/June 2005).

[3] Web-Based Enterprise Management (WBEM) is a set of standards developed for managing distributed enterprise systems, including both hardware and software.

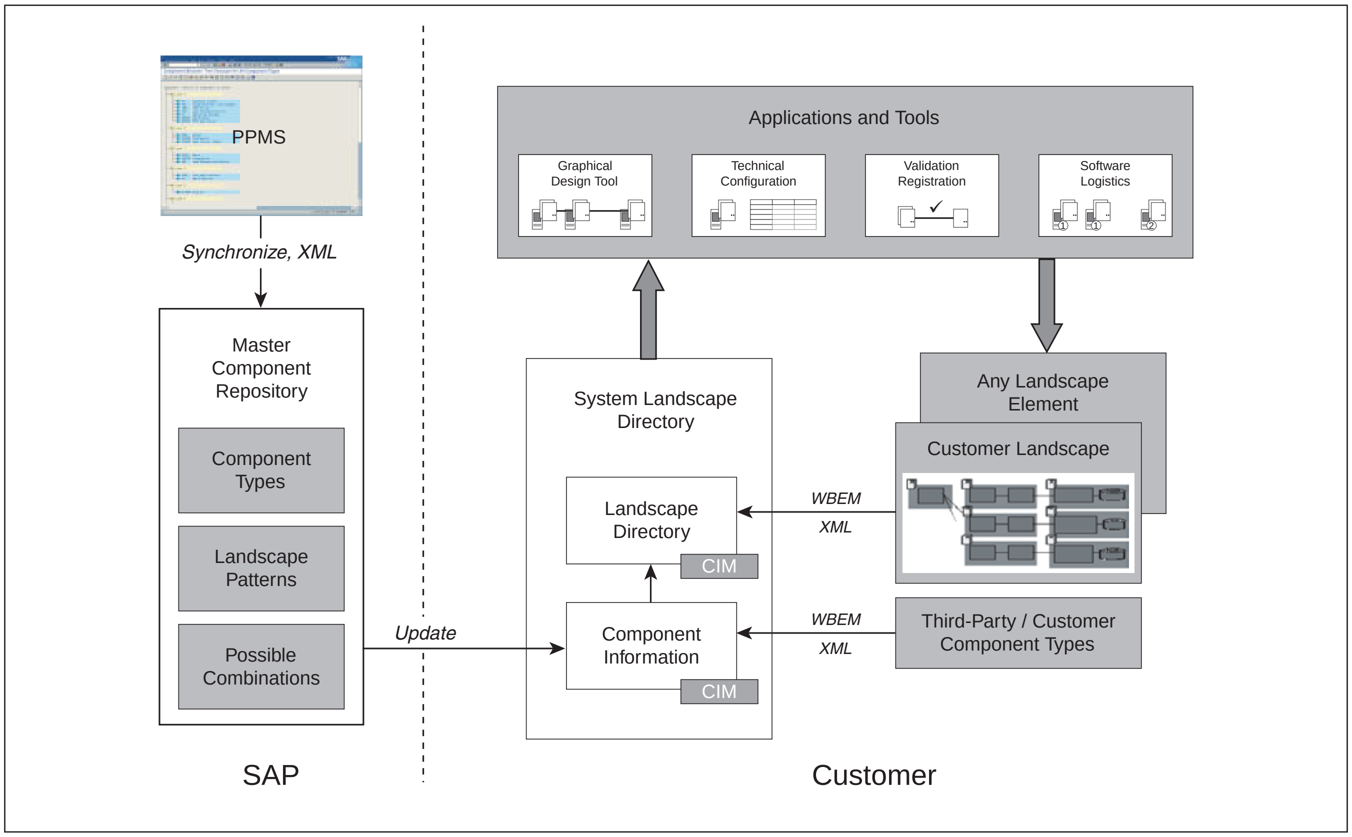


Figure 1 The System Landscape Directory (SLD) concept

across the various applications that use it, such as SAP Solution Manager (for installing, upgrading, and monitoring your system landscape),[4] SAP XI (for enterprise-wide application integration), SAP Supplier Relationship Management (SRM, for integrating internal business systems with external business partners), and SAP Auto-ID Infrastructure (AII, for Radio Frequency Identification).

Depending on your needs, you can install the SAP J2EE Engine (and thus the SLD server, since it is a Java application installed as part of the SAP J2EE Engine) either standalone (i.e., SAP Web AS with the Java stack only) or as part of a full SAP Web AS installation that includes both the ABAP and Java stacks. While the ABAP stack can supply and retrieve data to and from SLD, technically the SLD server does not require the ABAP stack to function, so you can reduce system requirements by installing only the SAP J2EE Engine if you want. If you use a landscape-wide maintenance or integration tool such as SAP Solution Manager or SAP XI, however, both the Java and ABAP stacks are necessary. In this case, SLD is installed as part of SAP Web AS, and both SAP Solution Manager and SAP XI can use the SLD server within their own SAP Web AS.

This installation flexibility makes it possible for a single, independent SLD server to serve the entire system landscape — ABAP and Java-based systems alike. Instead of having to maintain separate information repositories for different solutions, the solutions can leverage the same SLD repository, simplifying maintenance and lowering costs.

[4] For more on SAP Solution Manager, see the articles "SAP Solution Manager — 'Command Central' for Implementing Your mySAP Solutions … and Much More!" (*SAP Professional Journal*, September/October 2003) and "Ensure effective system management by configuring the right infrastructure for SAP Solution Manager using central system landscape maintenance" (*SAP Professional Journal*, September/October 2005).

Now that you understand how SLD data is modeled and stored, let's take a closer look at the different types of information that constitute SLD content. It is important to understand the different types of information and the relationships among them within the SLD server, as the operation of solutions like SAP XI depends on up-to-date, accurate SLD content.

Understanding SLD content

There are three types of SLD content:

- **Component information:** This is information about all available SAP solutions and their possible combinations and dependencies. SAP provides this information for SAP software components in your landscape via periodic updates, similar in concept to support packages. Information on third-party software components in your landscape is maintained manually.
- **Landscape description:** This is a model of the elements installed in your system landscape and the connections between them. Software components in the landscape description are linked to their counterparts in the component information to keep the landscape description information up to date. Landscape descriptions are customer-specific and must be maintained manually.
- **Name reservation:** This is information that defines the unique names used for the development objects in your system landscape. It guarantees that a development object name is unique in the world to avoid conflicts when systems are consolidated, for example. This information is also customer-specific and must be maintained manually.

Let's take a closer look at each of these types and how they serve the different purposes of SLD.

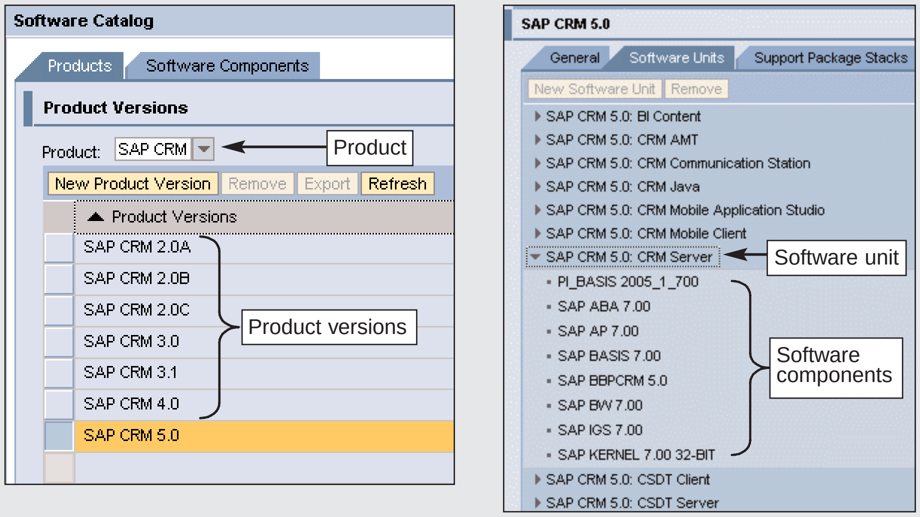
Component information

Component information encompasses the details of all of the software components — i.e., the technical, installable software units that constitute a product — installed in your system landscape. Component information also includes the combination of options and dependencies of a product's software components — i.e., which release of a software component can work with certain other components. For example, since SAP AII 2.1 was developed on SAP Web AS 6.40, prior to the release of SAP ERP Central Component (ECC) 6.0, SAP AII 2.1 can be installed only on SAP Web AS 6.40; it will not work with SAP ECC 6.0.

Figure 2 shows the main elements of component information as they appear in the 2004s version of SLD:

- A **product** is a collection of installable, upgradeable components that is delivered to a customer. Based on the components it contains, a product has different releases, which are known as **product versions**. For example, as you can see in Figure 2, SAP CRM is a product and has versions 2.0A, 2.0B, 2.0C, 3.0, 3.1, 4.0, and 5.0.
- A **software unit** is a grouping of related software components that together with other software units constitute a product. A software unit is a logical link between the product version and the software components that compose that version of the product. In Figure 2, for example, the software unit CRM Server is one of several software units that make up the CRM 5.0 product.
- A **software component** contains the actual technical software that enables a product to operate. In Figure 2, for example, SAP Basis 7.00 (under SAP CRM 5.0: CRM Server) is one of many software components that make up the CRM Server software unit, which in turn is one of many software units that make up the CRM 5.0 product. Software components can be reused among various products — for example, the same SAP Basis 7.00 software component used in the CRM 5.0 product also is used in the SAP ECC 6.0 and SAP AII 4.0 products — which simplifies upgrades and maintenance.

The software component information within SLD corresponds to the component types in the master CR shown in Figure 1. Each update from the component types in the master CR to the component information in SLD includes:



SAP CRM 2.0A
SAP CRM 2.0B
SAP CRM 2.0C
SAP CRM 3.0
SAP CRM 3.1
SAP CRM 4.0
SAP CRM 5.0

SAP CRM 5.0: BI Content
SAP CRM 5.0: CRM AMT
SAP CRM 5.0: CRM Communication Station
SAP CRM 5.0: CRM Java
SAP CRM 5.0: CRM Mobile Application Studio
SAP CRM 5.0: CRM Mobile Client
SAP CRM 5.0: CRM Server
- PI_BASIS 2005_1_700
- SAP ABA 7.00
- SAP AP 7.00
- SAP BASIS 7.00
- SAP BBPCRM 5.0
- SAP BW 7.00
- SAP IGS 7.00
- SAP KERNEL 7.00 32-BIT

SAP CRM 5.0: CSDT Client
SAP CRM 5.0: CSDT Server

Figure 2 Component information displayed in the SLD of SAP NetWeaver 2004s

- The attributes of the SAP products installed in your landscape (e.g., product versions)
- The attributes of the software components installed in your landscape (e.g., the SAP kernel type)
- The releases and support packages associated with the components installed in your landscape (e.g., the latest support package level for a certain software component)

Note!

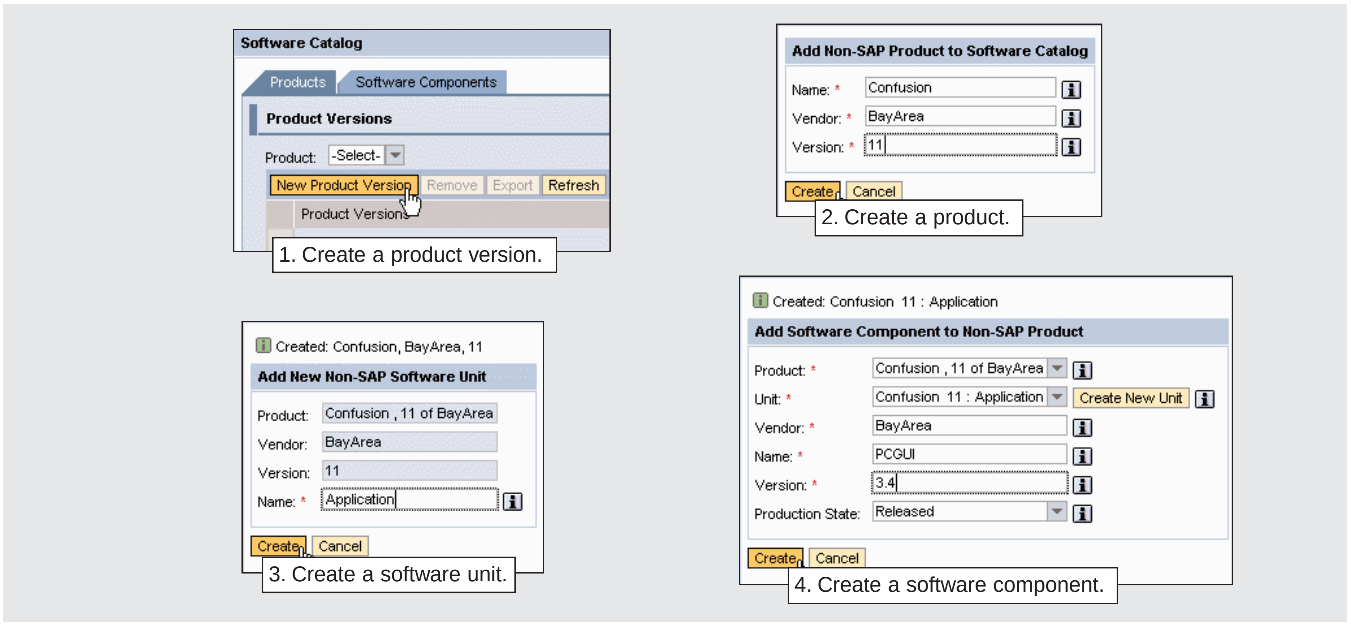
Component information is presented slightly differently in the 6.40 version of SLD. While the components are similarly grouped by product, product version, software unit, and software component, the 6.40 version of SLD includes information such as expiration date (i.e., Warranty Until) for software components, which is no longer displayed in the 2004s version of SLD.

Adding third-party component information to SLD

You can add component information for a third-party product by manually creating a product, product version, software component, and software unit in SLD. Let's say that you are implementing version 11 of a product called Confusion from a company called BayArea, and you want to register this information in SLD. **Figure 3** summarizes the elements of this example product. As you can see, there are two software units: Application, composed of the software

Product and version	Software unit	Software component
Confusion 11	Application	PCGUI 3.4
		MobileGUI 1.0
	Basis	Database 100.0
		ODBC 5.6
		JVM 2.1

Figure 3 Elements of the example third-party product



Software Catalog

Products | Software Components

Product Versions

Product: -Select-

New Product Version | Remove | Export | Refresh

Product Versions

1. Create a product version.

Add Non-SAP Product to Software Catalog

Name: *	Confusion
Vendor: *	BayArea
Version: *	11

Create | Cancel

2. Create a product.

Created: Confusion, BayArea, 11

Add New Non-SAP Software Unit

Product:	Confusion , 11 of BayArea
Vendor:	BayArea
Version:	11
Name: *	Application

Create | Cancel

3. Create a software unit.

Created: Confusion 11 : Application

Add Software Component to Non-SAP Product

Product: *	Confusion , 11 of BayArea
Unit: *	Confusion 11 : Application
Vendor: *	BayArea
Name: *	PCGUI
Version: *	3.4
Production State:	Released

Create New Unit

Create | Cancel

4. Create a software component.

Figure 4 Adding component information for a third-party product in SLD

components PCGUI 3.4 and MobileGUI 1.0, and Basis, composed of the software components Database 100.0, ODBC 5.6, and JVM 2.1.

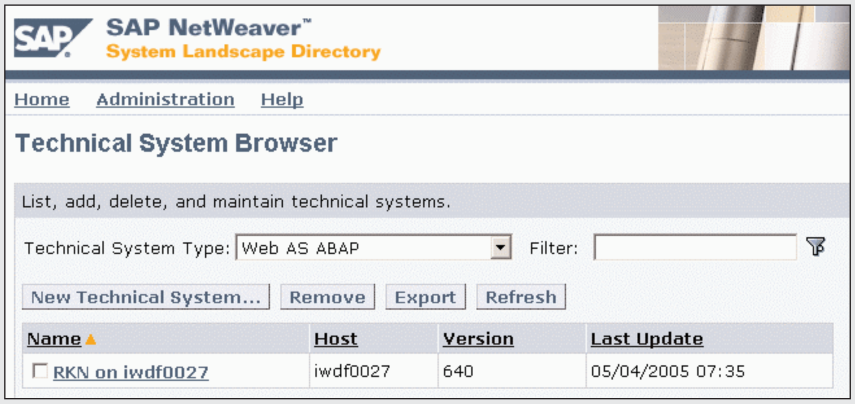
Figure 4 summarizes the steps involved in adding component information for the example third-party product in SLD.

Note!

If you don't immediately have all the information available for a third-party product, you can create a new product in SLD without a software unit and software component and add these elements at a later time, which is useful for planning purposes.

Note!

- If component information is added for a third-party software component, content upgrades from the master CR won't overwrite this information; it is up to administrators to keep third-party information up to date.
- User-defined third-party component information can be exported to other SLD servers via a manual export and import mechanism (more on this later), to keep information consistent and to avoid reentering the information in multiple SLD servers.



Technical System Browser

List, add, delete, and maintain technical systems.

Technical System Type: Web AS ABAP   Filter:

Name	Host	Version	Last Update
RKN on iwdf0027	iwdf0027	640	05/04/2005 07:35

Figure 5 Application system RKN registered as a technical system in SLD

Landscape description

The landscape description provides a model of the installed system landscape, and is manually entered in SLD. There are three types of landscape-related information stored in SLD: technical systems, landscapes, and business systems.

Technical systems

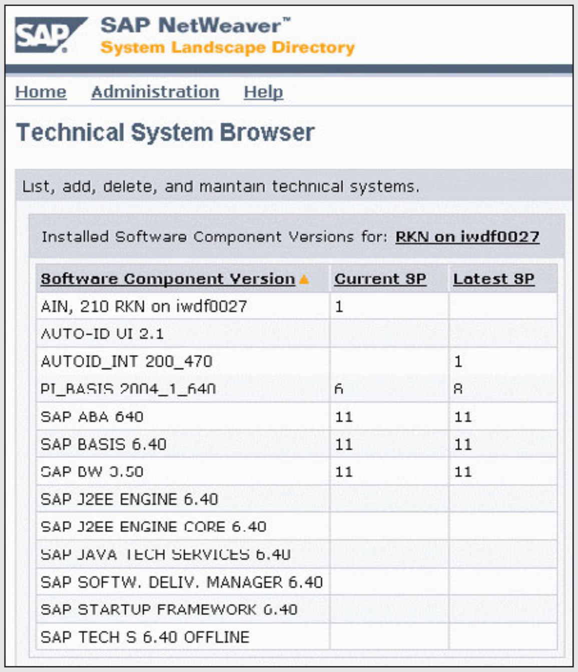
A technical system is the definition of a physical system that provides content to the SLD server. The information provided in technical system definitions includes:

- The physical systems installed in your landscape (identified by the system ID) and the system topology (network addresses, links, etc.)
- The structure of the components installed in your landscape (e.g., database, message server, application server)
- The component information for the software installed in your landscape (e.g., CRM software, ABAP 6.40 software)

The landscape description classes are associated with the component information classes as part of the CIM schema, so when you define a new technical system for your landscape description, you can immediately associate the system with a particular type of software, such as SAP ECC 5.0 or SAP CRM 4.0. Only the software components stored in the component information are available for the technical system definition, so updating SLD component information is a prerequisite for defining the technical systems for your landscape description. You can also register a third-party system as a technical system after defining its software component information.

Let's take a quick look at an example technical system definition. As you can see in **Figure 5**, an SAP application system RKN has been registered in SLD as an SAP Web AS ABAP system. Its host name (iwdf0027), SAP Web AS release number (6.40), and product information (SAP Auto-ID Infrastructure 2.1, located in the lower portion of the screen, which is not visible in Figure 5) are provided automatically by a registration program (more on this registration process later in the article). Clicking on the system name takes you to the display shown in **Figure 6** on page 10, which lists the software components associated with system RKN, along with the latest support packages available (the Latest SP column) and whether they have been installed (the Current SP column), which is derived from SLD component information. As you can see in the example, software components AUTOID_INT 200_470 and PI_BASIS 2004_1_640 do not have the latest support package installed; SAP BASIS 6.40 and SAP BW 3.50 do. A support package has also been installed for component AIN,[5] but the

[5] AIN stands for Auto-ID Node, an old name for Auto-ID Infrastructure.



Technical System Browser

List, add, delete, and maintain technical systems.

Installed Software Component Versions for: RKN on iwdf0027

Software Component Version	Current SP	Latest SP
AIN, 210 RKN on iwdf0027	1	
AUTO-ID UI 2.1		
AUTOID_INT 200_470		1
PI_BASIS 2004_1_640	6	8
SAP ABA 640	11	11
SAP BASIS 6.40	11	11
SAP BW 3.50	11	11
SAP J2EE ENGINE 6.40		
SAP J2EE ENGINE CORE 6.40		
SAP JAVA TECH SERVICES 6.40		
SAP SOFTW. DELIV. MANAGER 6.40		
SAP STARTUP FRAMEWORK 6.40		
SAP TECH S 6.40 OFFLINE		

Figure 6 The software components of the registered application system RKN

Note!

If you don't import the master CR content regularly, the component information in SLD could quickly become obsolete, which can prevent you from accurately defining technical systems in the landscape description. For example, if you don't import the 2.4 version of the master CR content, the component information will not include the SAP AII 4.0 product, and you cannot accurately define an SAP AII 4.0 system.

blank entry under Latest SP indicates that the latest support package is unknown, which means that the SLD content is outdated.

The landscape description information is the most dynamic SLD content. When a database is updated or a new support package is installed on a system that is registered as a technical system in SLD, this delta information is periodically updated, usually every 12 hours, in the SLD server by a batch job running on the system. In comparison, component information is updated in the SLD server only when a user manually imports the latest master CR content from SAP.

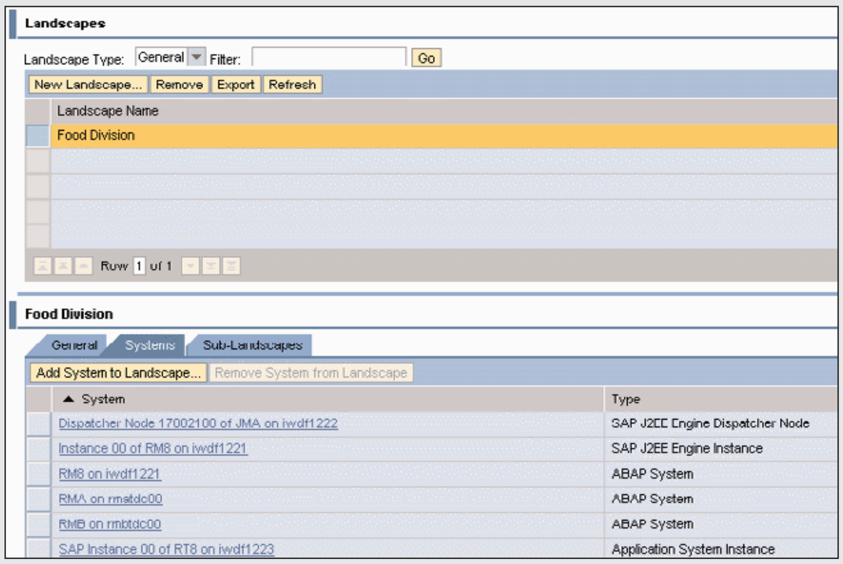
Landscapes

A large enterprise can have a very complicated system landscape involving multiple business units and regions. To help maintain systems themselves, in addition to maintaining software components, their release versions, and their dependencies, the 2004s version of SLD introduces the "landscape" concept as part of the landscape description. A landscape represents a logical environment with complex structures containing multiple components such as systems, services, and installed products.

SLD can define the following types of landscapes:

- **Administration:** A system landscape for administration purposes, such as monitoring
- **General:** A typical system landscape with all types of application systems
- **NWDI System:** A system landscape related to the SAP NetWeaver Development Infrastructure (NWDI)
- **Scenario:** A system landscape created for a specific business scenario in the Software Lifecycle Manager (SLM) of SAP NetWeaver
- **Transport:** A system landscape created for transport purposes (i.e., distributing development objects)
- **Web Service:** A system landscape containing Web service providers and clients

The General landscape type is the closest to a conventional system landscape — for example, an SAP R/3 system, a CRM system, and an SCM system in a customer environment. **Figure 7** shows an example General landscape called Food Division. It contains one SAP J2EE Engine dispatcher node



Landscape Name
Food Division

System	Type
Dispatcher Node 17002100 of JMA on iwdf1222	SAP J2EE Engine Dispatcher Node
Instance 00 of RM8 on iwdf1221	SAP J2EE Engine Instance
RM8 on iwdf1221	ABAP System
RMA on rmatdc00	ABAP System
RMB on rmbtdc00	ABAP System
SAP Instance 00 of RT8 on iwdf1223	Application System Instance

Figure 7 Example General landscape

(17802100 of JMA on iwdf1222), one SAP J2EE Engine instance (00 of RM8 on iwdf1221), three ABAP systems (RM8, RMA, and RMB on different servers), and one SAP application instance (00 of RT8 on iwdf1223).

Business systems

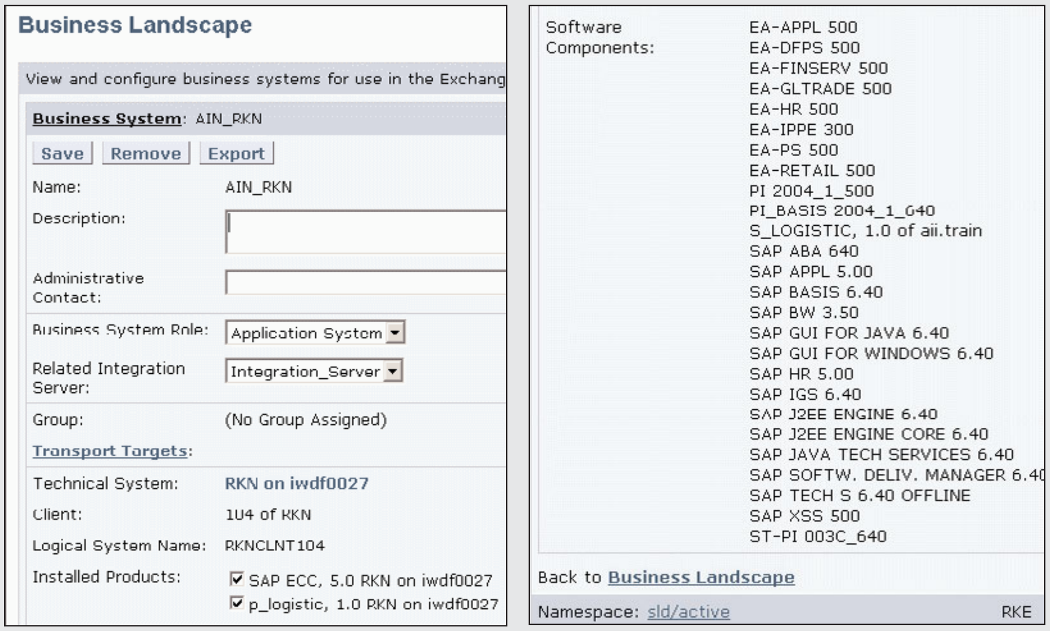
While a technical system defines a physical system, a business system defines a logical unit of a technical system based on its business purposes, similar to a client of an SAP R/3 system. A business system must be associated with a technical system. Similar to the relationship between clients and an SAP R/3 system, the relationship between business systems and a technical system is *n* to 1 — multiple business systems can be associated with a single technical system, but a business system can be associated with only one technical system. This part of the landscape description information is used only by SAP XI.

Figure 8 on page 12 shows the details of an example business system AIN_RKN. As you can see in the screenshot on the left, the business system represents client 104 of technical system RKN and its logical system name is RKNCLNT104. The screenshot on the right lists the software components installed on the system, including ABAP 6.40, BW 3.5, J2EE Engine 6.40, and others. When you create the business system, the physical system information is provided automatically from technical system RKN, and you select the installed software components provided by the SLD component information. This is why updating the SLD component information on a regular basis is important — if you need to register an SAP AII 4.0 business system for your SAP XI integration, for example, and SAP AII 4.0 is not included in the SLD component information, you cannot register the business system as an SAP AII 4.0 system and the integration scenario will not run.

Name reservation

The last type of content stored in SLD is name reservation information, which stores the unique namespace assigned to the SLD server. Starting with SAP R/3 4.0B, SAP began assigning development namespaces to customers and partners to guarantee the uniqueness of the naming conventions used for their ABAP development objects, and to avoid the risk of naming conflicts.[6] With the introduction of the SAP

[6] Customers and partners can request development namespaces from the SAP Service Marketplace at http://service.sap.com/namespaces.



Business Landscape

View and configure business systems for use in the Exchang

Business System: AIN_RKN

Name:	AIN_RKN
Description:	
Administrative Contact:	
Business System Role:	Application System
Related Integration Server:	Integration_Server
Group:	(No Group Assigned)
Transport Targets:	
Technical System:	RKN on iwdf0027
Client:	104 of RKN
Logical System Name:	RKNCLNT104
Installed Products:	☑ SAP ECC, 5.0 RKN on iwdf0027 ☑ p_logistic, 1.0 RKN on iwdf0027

```
Software        EA-APPL 500
Components:     EA-DFPS 500
                EA-FINSERV 500
                EA-GLTRADE 500
                EA-HR 500
                EA-IPPE 300
                EA-PS 500
                EA-RETAIL 500
                PI 2004_1_500
                PI_BASIS 2004_1_640
                S_LOGISTIC, 1.0 of aii.train
                SAP ABA 640
                SAP APPL 5.00
                SAP BASIS 6.40
                SAP BW 3.50
                SAP GUI FOR JAVA 6.40
                SAP GUI FOR WINDOWS 6.40
                SAP HR 5.00
                SAP IGS 6.40
                SAP J2EE ENGINE 6.40
                SAP J2EE ENGINE CORE 6.40
                SAP JAVA TECH SERVICES 6.40
                SAP SOFTW. DELIV. MANAGER 6.4
                SAP TECH S 6.40 OFFLINE
                SAP XSS 500
                ST-PI 003C_640
```

Back to Business Landscape

Namespace: sld/active RKE

Figure 8 Example business system AIN_RKN and its associated software components

NetWeaver platform and SAP Business One, the namespace concept extends to non-ABAP development objects as well, including the names of Java applications, Java packages, development components, installation units, software components, database tables, indexes, and fields.

The assigned namespace — i.e., the SLD server name — serves as a prefix for all the development objects within the system landscape to guarantee the uniqueness of all objects in your SLD server from those defined in other SLD servers. You can also form a namespace hierarchy by assigning sub-namespaces to sub-groups of development objects. Without reserving your SLD server name with SAP, there is no guarantee that the prefix has no duplicate elsewhere in the world, which can result in confusion if you later need to merge development objects from two systems.

This article does not go into detail about name reservation, since it is specific to using SAP NWDI, but it is important to understand the concept to anticipate possible name conflicts when multiple developers are involved.

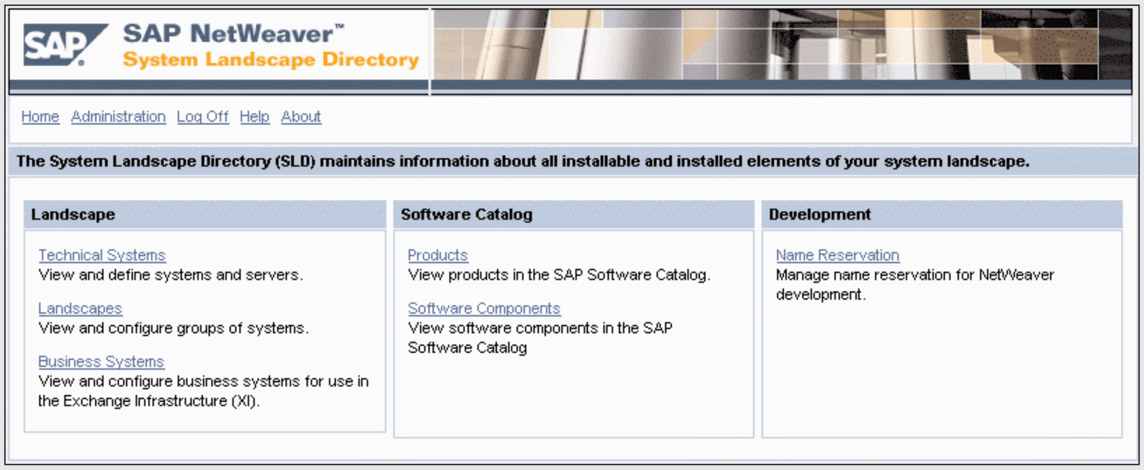
Before turning to SLD configuration, let's take a quick look at the key elements of the SLD user interface to familiarize you with how it is organized, so you can navigate quickly and efficiently to the information you need during configuration and maintenance.

The SLD user interface

The SLD server is accessed via an HTTP connection using a Web browser. The URL takes the format http://<host>:<port>/sld — for example, http://iwdf0027.wdf.sap.corp:50000/sld — where <host> is the server name running the SAP J2EE Engine and <port> is the HTTP port of the SAP J2EE Engine. If you access the SLD server via a Web browser that is local to the server, you can enter localhost as the server name.

The SLD home page, shown in **Figure 9**, is organized into three categories:

- **Landscape** for maintaining the landscape description


SAP NetWeaver™
System Landscape Directory

Home Administration Log Off Help About

The System Landscape Directory (SLD) maintains information about all installable and installed elements of your system landscape.

Landscape

Technical Systems
View and define systems and servers.

Landscapes
View and configure groups of systems.

Business Systems
View and configure business systems for use in the Exchange Infrastructure (XI).

Software Catalog

Products
View products in the SAP Software Catalog.

Software Components
View software components in the SAP Software Catalog

Development

Name Reservation
Manage name reservation for NetWeaver development.

Figure 9 The SLD home page in SAP NetWeaver Application Server 2004s

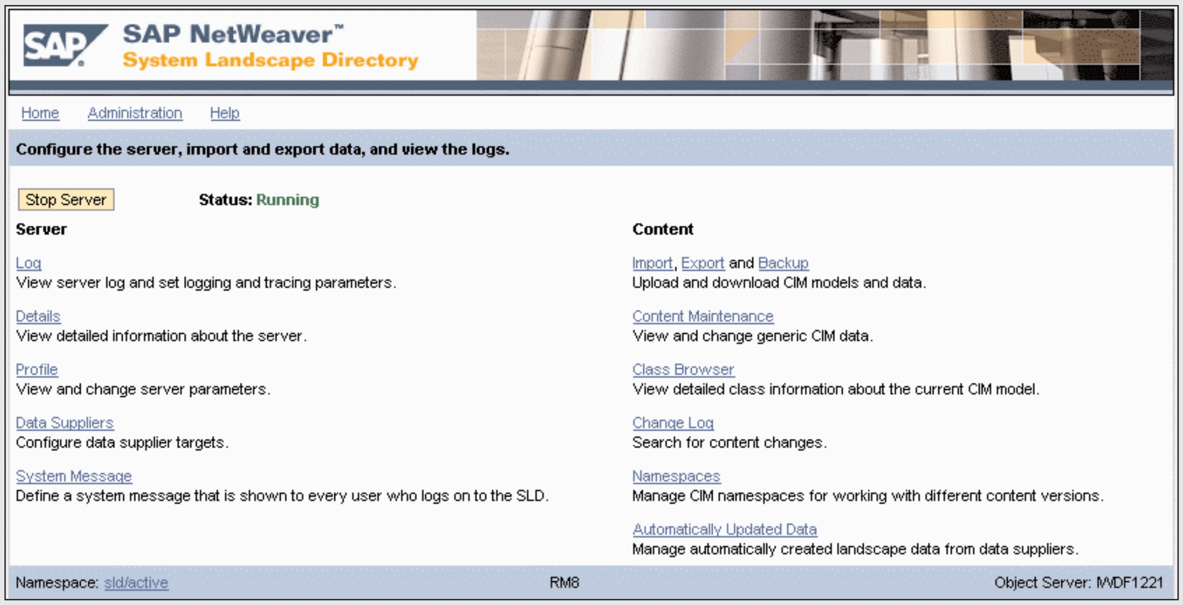
Note!

The HTTP port must be the HTTP port for the SAP J2EE Engine in which the SLD server resides. This can be confusing because SAP Web AS has two stacks — ABAP and Java — each of which has its own HTTP service and port. The HTTP port for the ABAP stack is usually four digits (e.g., 8080) and can be identified via transaction SMICM, where you can create and modify the HTTP and HTTPS ports, or disable/enable them as needed. The port for the SAP J2EE Engine is five digits (e.g., 52000) and follows the format 5xxyy, where xx corresponds to the system number assigned to the SAP Web AS and yy represents the appropriate SAP J2EE Engine port. The SAP J2EE Engine is installed with a set of default ports, including 00 for HTTP, 01 for HTTPS, and 04 for the Visual Admin Tool (known as the SAP NetWeaver Administrator in SAP NetWeaver 2004s). These default ports can be changed — for security reasons, for example — using the SAP J2EE Engine Config Tool.

- **Software Catalog** for maintaining the component information
- **Development** for managing name reservations for development objects

To access the Administration area, shown in **Figure 10** on the next page, simply click on the Administration hyperlink at the top of the SLD home page. The tasks for an administrator are roughly divided into two groups, which are reflected in the organization of the Administration page: Server for server configuration and Content for repository maintenance. You can stop and start the SLD server, which is required for making changes to certain SLD server settings, by clicking on the toggle button above the Server category. The status of the server is displayed to the right of the button. You also can find the assigned namespace of the SLD server (sld/active in the case of Figure 10) at the lower left of the Administration page.

Now that you have a solid foundational understanding of the SLD architecture, the SLD content, and the organization of the user interface, let's start exploring the key areas involved in configuring SLD for your system landscape.



Configure the server, import and export data, and view the logs.

Status: Running

Server

Log
View server log and set logging and tracing parameters.

Details
View detailed information about the server.

Profile
View and change server parameters.

Data Suppliers
Configure data supplier targets.

System Message
Define a system message that is shown to every user who logs on to the SLD.

Content

Import, Export and Backup
Upload and download CIM models and data.

Content Maintenance
View and change generic CIM data.

Class Browser
View detailed class information about the current CIM model.

Change Log
Search for content changes.

Namespaces
Manage CIM namespaces for working with different content versions.

Automatically Updated Data
Manage automatically created landscape data from data suppliers.

Namespace: sld/active RM8 Object Server: MDF1221

Figure 10 The SLD Administration page in SAP NetWeaver Application Server 2004s

Note!

The 6.40 and 2004s versions of the SLD home page both use the three-category format with some minor differences in the category names and the organization of the information. Notably, the 2004s version of SLD added the new Landscapes element (the 6.40 version of SLD did not include this concept), added direct links to Products and Software Components on the home page (previously these were accessed through a Software Catalog link), and moved the CIM Content Maintenance and Class Browser categories to the Administration page (originally these were included on the home page). The 2004s version of SLD also includes a Log Off function and an About link, which lists useful information about your SLD server, including the SLD server version, the CIM model version, the CR content version, and the Java Virtual Machine (JVM) version.

Best practices for configuring SLD

Since SLD installation and configuration guides provide detailed guidance on how to perform the technical steps involved in configuring SLD (see the sidebar on the next page for a complete listing of these steps), instead of repeating this readily available information, in the following sections I provide you with insights into the key decision points that face you during the configuration, including:

- Using a single SLD server vs. distributed SLD servers
- Assigning SLD user roles and access authorizations
- Importing the CIM model and the contents of the master Component Repository
- Specifying the SLD server settings
- Configuring the SLD data supplier bridge
- Configuring the SLD data supplier

The SLD configuration steps

The following is a complete listing of the steps necessary to configure SLD:

1. If you are using NWDI, reserve your SLD server namespace at the SAP Service Marketplace (http://service.sap.com/namespaces).
2. Determine whether to use a single SLD server or distributed SLD servers. If you opt to use a distributed SLD server configuration, you also must decide on the SLD content synchronization mechanism, using either exporting/importing or data forwarding.
3. Install the SAP J2EE Engine either standalone or together with the ABAP stack as part of an SAP Web AS.
4. Assign SLD user roles and access authorizations.
5. Download the latest CIM model and the latest contents of the master Content Repository from the Software Distribution Center at the SAP Service Marketplace (http://service.sap.com/swdc), and import them.
6. Specify the SLD server settings, including the SLD server namespace obtained in Step 1.
7. Configure the SLD data supplier bridge.
8. Register the ABAP-based or Java-based systems as technical systems in the SLD landscape description, and verify that the technical system information is updated automatically as intended.
9. Register the business systems manually if you plan to use SAP XI for integration purposes.
10. Set up a JCo RFC Provider for accessing SLD data from ABAP-based systems. SLD has a built-in API for Java-based systems that requires no manual configuration.
11. Run the SLD check program SLDCHECK on all ABAP-based SLD client systems that use SLD data to verify the communication between the SLD client and the SLD server.

While the SAP NetWeaver 2004 installation program SAPInst runs some of these steps automatically, it is important that you verify the settings so that you can make any necessary corrections or additions.

For the technical details of performing these steps, refer to the SLD configuration and installation guides, which you can find in the Media Library for SLD at the SAP Service Marketplace (http://service.sap.com/sld).

- Accessing SLD data

I also present some best practices for addressing these decision points, and hard-won lessons that I learned first-hand while setting up an integrated SAP RFID solution, which involves SAP R/3, SAP Event Management, SAP XI, and SAP AII, to help set you up for success with your own configuration.

We’ll start with the first key decision that you will face during your SLD configuration: whether to use a single SLD server or distributed SLD servers.

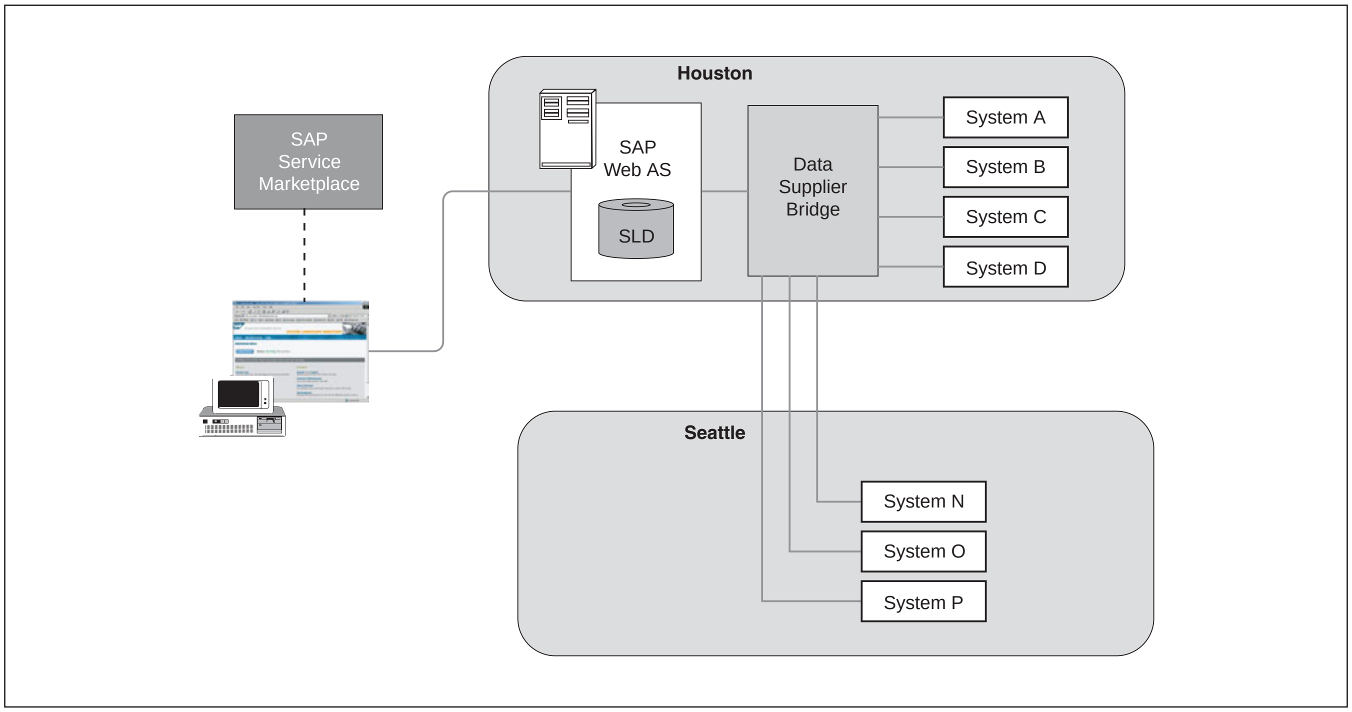


Figure 11 A single SLD server configuration

Using a single SLD server vs. distributed SLD servers

In a complicated enterprise-wide system landscape, the number of required SLD servers is largely dependent on customer needs, but there are a few considerations that are common to any customer:

- Cost
- Security
- Data integrity
- Performance
- Availability

In the next sections we'll look at these considerations in the context of using a single SLD server vs. distributed SLD servers, and the pros and cons of each.

Using a single SLD server

A single server that serves all systems in an enterprise, including systems on different networks in different geographical locations, is the simplest configuration, is the easiest to maintain, and provides a centralized repository for an entire landscape.

Figure 11 shows a typical system landscape with a single SLD server configuration. As you can see, there are two locations, one in Houston and one in Seattle. The SLD server is located in Houston, where it receives periodic component information updates from the SAP Service Marketplace, and it communicates with the systems in the landscape via a data supplier bridge (more on configuring data supplier bridges in a later section). Let's apply the five evaluation criteria — cost, security, data integrity, performance, and availability — to this example configuration:

- **Cost:** Since there is only one SLD server, hardware and maintenance costs are at a minimum.
- **Security:** There is no obvious security risk from the outside, such as Internet intruders, with this configuration. If test systems and production systems are mixed in this landscape, however, there is the risk that test system users can access production systems.

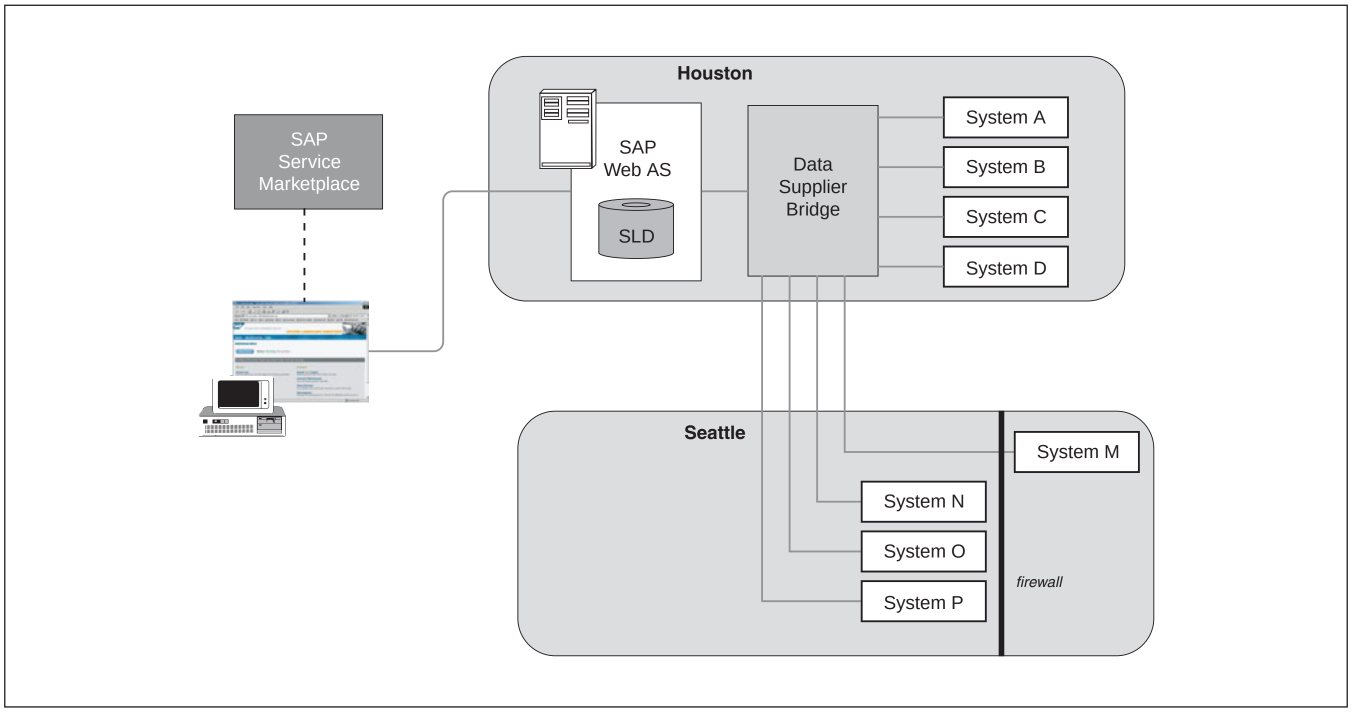


Figure 12 A single SLD server configuration with a system outside the firewall

- **Data integrity:** There is no need to consolidate and synchronize data from additional SLD servers, so there is minimum risk of data integrity problems.
- **Performance:** If there are very few clients of the SLD server, all located in Houston, performance should not be a problem. But performance could be a problem if there is an SAP XI system or a Web Dynpro system in Seattle, because these systems rely heavily on the SLD server, and in turn place a greater demand on its performance than other SLD clients.
- **Availability:** Since there is only one SLD server, downtime due to hardware failure or applying support packages, for example, can affect the entire landscape, including production systems.

Let's now modify the example landscape slightly to reflect an added element found in many typical landscapes — an online store. As you can see in **Figure 12**, the system supporting the online store, System M, is located in Seattle, outside of the firewall. In this case, the single SLD server configuration presents an increased security risk, since System M has access to the SLD server from outside of the firewall. In this scenario, a single SLD server should not serve both the internal and external systems.

In summary, a single SLD server approach is ideal for a midsized environment with a small number of systems, though you will need to weigh lower costs against potential security and availability risks. This approach is not ideal in cases where you want to isolate the production environment, where you want to segregate different sets of application systems for geographic or business purposes, or where data volume and performance demands are significant. In these situations, a distributed SLD server configuration is best, which we will look at next.

Using distributed SLD servers

For an enterprise with a complicated landscape, distributed SLD servers may be necessary for performance and security reasons. In such cases, SAP recommends keeping a single SLD server at the top of the hierarchy that has an overall view of the entire system landscape; other SLD servers in the hierarchy

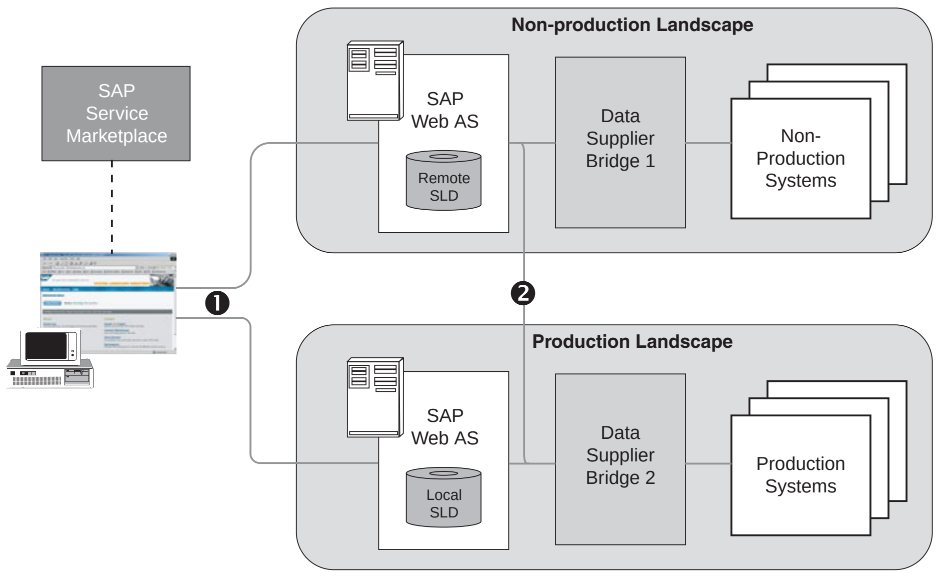


Figure 13 Two SLDs for two separate system landscapes

then are configured to have no content synchronization or limited content synchronization with the SLD server at the top of the hierarchy via content forwarding or content copying (see the sidebar on the next page for more on these synchronization options).

Here, I focus on a distributed SLD server configuration that incorporates content forwarding. In this case, a remote SLD server collects information from local SLD servers to keep data consistent across the landscape and to maintain an overall view of the enterprise. While multiple SLD servers generally mean higher costs for operations and hardware acquisitions, the more SLD servers you have, the better the systems in different geographic locations will perform, and the better protection you will have for production systems behind firewalls.

Let's take the SLD topology in **Figure 13** as an example. Here, the systems are divided into two landscapes: production and non-production. With this configuration, the production systems are protected from exposure to the non-production systems, increasing security. Each system landscape has a single SLD server that simultaneously receives periodic component information updates from the SAP Service Marketplace (❶) and each SLD communicates with the systems in the landscape via a data supplier bridge, which I'll discuss in detail later in the article. The SLD server in the non-production landscape is the remote SLD server, so whenever a

Note!

For performance and security reasons, it is not recommended to have an SLD server in the production environment serve non-production SLD clients or provide information to a remote SLD server in a non-production environment. A production SLD server should be completely isolated from non-production environments, with no data entering or exiting. It is always best to avoid any unnecessary data access to production systems to avoid compromising the integrity of your production environment.

Synchronizing content among distributed SLD servers

There are two ways of synchronizing SLD content across distributed SLD servers: content forwarding and content copying.

- **Content forwarding:** With content forwarding which is shown in Figure 13, a local server forwards its content to a remote server. Forwarding happens automatically via a data supplier bridge whenever a local server receives new content from its data suppliers. With this topology, the content in the remote SLD server is a complete set, including the content of all its associated local SLD servers. The local SLD server contains only its own data, unless it also serves as a remote server for another local server that feeds it data. For example, let's say that there are three SLD servers in a large enterprise: Server A is a remote SLD server at the corporate level that contains information about the entire company. Server B is a local SLD server that contains information about the entire pharmaceutical division. Server C is another local server that contains information about the production environment of the pharmaceutical division. When Server C receives data from a data supplier, it forwards its content to Server B. When Server B receives data from Server C, it forwards its content to Server A.
- **Content copying:** With content copying, you manually export content from one SLD server and import it into another. The server that exports its content is called the master server and the server that imports content is called the subsidiary or "slave" server. This topology is used when you don't want direct data exchange between the SLD servers, but you want them to have identical content. A good example is importing master Component Repository content from the SAP Service Marketplace to your own SLD server. As a customer, you want the latest content from SAP, but SAP does not allow you to receive the content with a direct connection. Similarly, if you entered a lot of third-party software information in one SLD server and don't want to repeat the manual entry work, you can export the content from the master SLD server and import it into subsidiary SLD servers. However, to guarantee that data is consistent and maintained by only one SLD server, a subsidiary SLD server can import content from only one master system. Of course, one master server can serve multiple subsidiary servers.

I prefer the content forwarding approach because it automates the data exchange among the SLD servers via the data supplier bridge. This approach does have its limitations. Only data from SLD data suppliers is forwarded; manually entered data is not forwarded. Configuring a data supplier bridge is covered in detail later in the article.

With content copying, on the other hand, you can exchange manually entered data between two SLD servers. In this case, you need to perform the exporting and importing and keep track of content updates manually. As with any manual process, however, this method makes it easy for data to become inconsistent, so try to limit its scope to certain data, such as manually entered third-party software component information. You can find information on the SLD content exporting and importing steps in the "System Landscape Directory, SAP NetWeaver 2004s SPS09" document available from the Media Library at http://service.sap.com/sld, so I won't go into the details of the process here. However, keep in mind that with this approach you need a well-thought-out plan from the very beginning to maintain consistent SLD content among your SLD servers. Once you have completed your first full export, be sure to export and import subsequent delta changes on a regular basis.

landscape description update is sent to the local SLD server in the production system, that same information is synchronized with the remote SLD server, as illustrated by ❷. With this synchronization, the non-production SLD server does not need to access the production SLD server, guaranteeing that the remote SLD server always has up-to-date system information for the entire enterprise without jeopardizing the security of the production landscape.

Let's see how this example configuration, using content forwarding, stacks up against our evaluation criteria:

- **Cost:** Two SLD servers are definitely more expensive than one SLD server. But if the local production SLD server hardware fails, the remote SLD server can provide a backup of the content using the SLD export and import mechanism.
- **Security:** The production landscape is fully insulated from the non-production landscape.
- **Data integrity:** By replicating all the data sent to the local SLD server in the production landscape to the remote SLD server in the non-production landscape, there should be no data integrity problems.
- **Performance:** With a limited number of systems in each landscape, the workload for each SLD server should not be as high as using one SLD server. Although the remote SLD server receives data from both landscapes, it should not have a big impact due to the infrequency of incoming data.
- **Availability:** With multiple SLD servers, any potential failures or system downtimes are localized. There should be no enterprise-wide effects.

In summary, a distributed SLD server approach is ideal for most system landscapes, since most landscapes tend to grow and become increasingly complex over time. Even if your current landscape is simple and a single SLD server approach might be sufficient for now, consider implementing a distributed approach in anticipation of a more complicated landscape in the future. SAP Note 936318 details the steps of splitting content from a single SLD server to multiple SLD servers.

Note!

A more comprehensive study of SLD server topology can be found in the "Planning Guide - System Landscape Directory" document available from the Media Library at http://service.sap.com/sld. SAP Notes 954820 and 764393 also feature good discussions on this subject.

Note!

Keep in mind that there is no special hardware sizing requirement for an SLD server — the hardware resources required for an SAP Web AS is sufficient to support an SLD server running on it. The SLD server is only a reference system; it is not a transaction-oriented system like SAP R/3, so it does not perform resource-intensive operations. Data input/output activity also is limited (for example, an SLD data supplier usually sends system data to an SLD server only twice a day, and only changed data is sent), as is SLD server access (for example, SAP XI retrieves common landscape information from the SLD server during its design, configuration, and monitoring phases, but normally not during its runtime).

Recommendations for considering a single SLD server vs. distributed SLD servers

Based on the discussions I have had with customers and consultants, the decision between using one or many SLD servers is similar to the decision between using one or many SAP XI systems in your landscape — it depends on the cost, performance, data integrity, and internal politics. Keep the following in mind when determining which approach makes the most sense for your landscape:

- Cost (system and its maintenance), security, and data integrity are the three major factors in deciding on your SLD server configuration.
- For a small landscape, a single SLD server is sufficient. It is the simplest and least costly approach.
- For a complex landscape involving multiple systems, multiple SLD servers can overcome the security, performance, and availability shortcomings of a single SLD server.
- There is more work involved in multiple SLD servers, including software, hardware, and administration, which you will need to weigh against the security, performance, and availability benefits you gain.

Once you've made the fundamental decision on whether to use a single SLD server or multiple SLD servers, the next critical step in your SLD configuration is assigning SLD user roles and authorizing access. We'll look at this next.

Assigning SLD user roles and access authorizations

As an independent Java application, SLD has its own user authorizations. SLD defines nine SLD security roles for allowing access to an SLD server, two of which (LcrSupport and DataSupplierLD) are new with the 2004s version of SLD:

- **LcrUser:** This role is for read access to SLD data. It is the minimum security role required for accessing an SLD server; otherwise, access is denied during the logon process. This user role does not allow viewing of the SLD Administration area.
- **LcrClassWriter:** This role is for creating, modifying, and deleting CIM classes. It also includes the authorizations of the LcrUser role.
- **LcrSupport:** This role is for read-only access to all SLD data and UIs, including the SLD Administration area, and is used for SAP support.
- **LcrInstanceWriterCR:** This role is for creating, modifying, and deleting CIM instances of the Component Information subset. It also includes the authorizations of the LcrUser role.
- **LcrInstanceWriterNR:** This role is for creating, modifying, and deleting CIM instances of the Name Reservation subset. It also includes the authorizations of the LcrUser role.
- **LcrInstanceWriterLD:** This role is for creating, modifying, and deleting CIM instances of the Landscape Description subset. It also includes the authorizations of the LcrUser role.
- **DataSupplierLD:** This role is for creating, modifying, and deleting CIM instances of the Landscape Description subset as a data supplier without access to the SLD UI.
- **LcrInstanceWriterAll:** This role is for creating, modifying, and deleting all types of CIM instances. It also includes the authorizations of the LcrUser, LcrInstanceWriterCR, LcrInstanceWriterLD, and LcrInstanceWriterNR roles.
- **LcrAdministrator:** This role provides the highest authorization access to the SLD server. It is for administrative tasks for both systems and applications. It also includes the authorizations of all other roles. Without this security role, the Administration hyperlink does not appear on the SLD home page.

Note!

A user with SAP J2EE Engine Administrator rights automatically has LcrAdministrator rights, since the SLD server is a component of the SAP J2EE Engine.

ABAP security role	SLD user role	Recommended use
SAP_SLD_GUEST	LcrUser	Includes read permission for all SLD content
SAP_SLD_DEVELOPER	LcrInstanceWriterNR	In addition to LcrUser rights, includes write permission for development object names
SAP_SLD_CONFIGURATOR	LcrInstanceWriterLD	In addition to LcrUser rights, includes write permission for system landscape elements
SAP_SLD_ORGANIZER	LcrInstanceWriterCR LcrInstanceWriterAll	In addition to LcrUser rights, includes write permission for component information
SAP_SLD_ADMINISTRATOR	LcrAdministrator	In addition to LcrUser rights, includes administration permission
SAP_SLD_DATA_SUPPLIER*	DataSupplierLD	Includes data supplier permission without UI access
SAP_SLD_SUPPORT*	LcrSupport	Includes read permission for all SLD content, including administration data
* New as of SAP NetWeaver Application Server 2004s.		

Figure 14 SLD security role mapping

If you have only the SAP J2EE Engine installed, you simply can assign one of these authorizations to your users using the SAP J2EE Engine Visual Admin Tool (known as the SAP NetWeaver Administrator in SAP NetWeaver 2004s). In most cases, however, the ABAP stack is installed along with the SAP J2EE Engine, so that applications such as SAP XI and SAP Solution Manager can access the SLD server, which means that many users will already have an existing login ID for the ABAP stack. How do you assign user authorizations for both the Java and ABAP stacks without maintaining two separate sets of logon IDs for users? SAP Web AS 6.40 and higher includes seven ABAP user roles (listed in **Figure 14**) that are mapped to the SLD Java user roles, so that you simply can add one of these roles to an existing ABAP user role to enable SLD access.

Note!

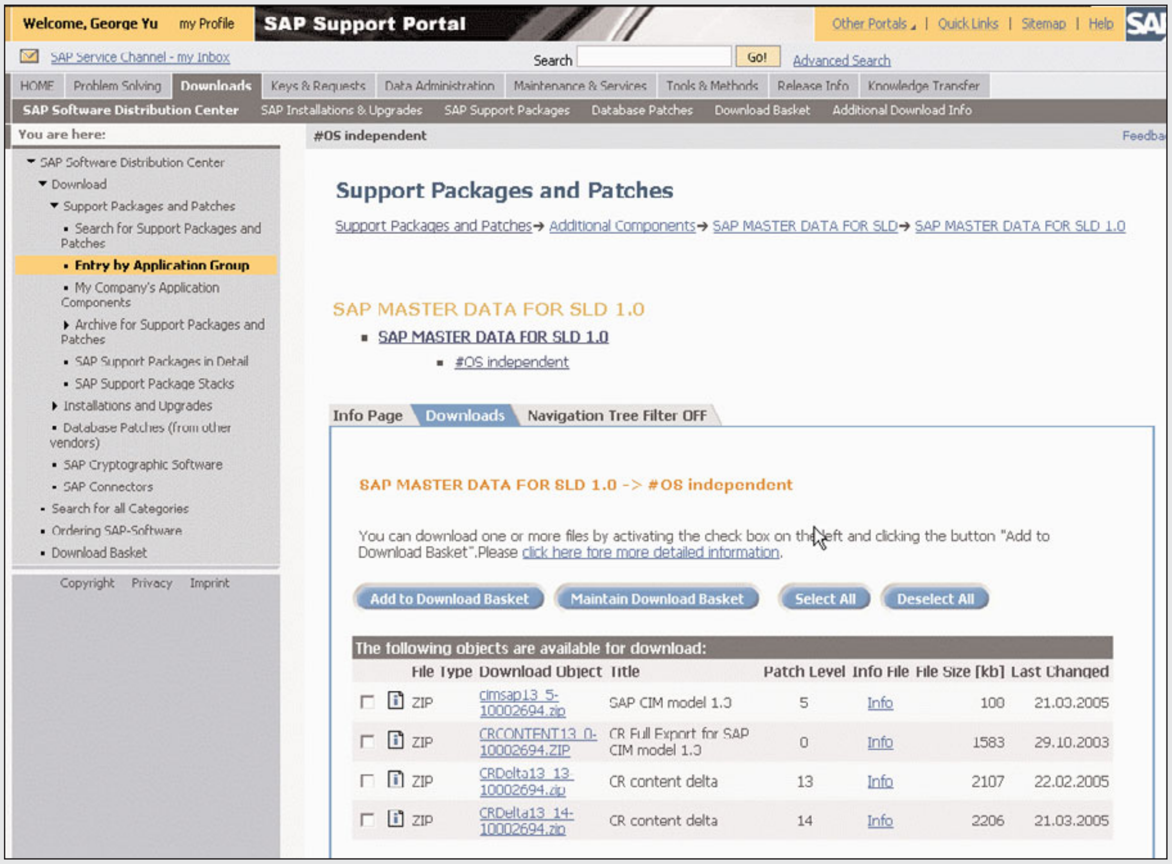
The LcrClassWriter SLD security role does not have a corresponding ABAP role defined, since a user with permission to create or adapt classes will likely be an administrator with full access authorizations.

During installation, the SAP J2EE Engine tries to map the SLD user roles to their corresponding ABAP roles. If the mapping does not exist — for example, if you initially installed an ABAP-only SAP Web AS and are subsequently adding the SAP J2EE Engine — you need to create the mapping between them using the SAP J2EE Engine Visual Admin tool. The details are described in the "Post-Installation Guide - SLD of SAP NetWeaver 2004s" document available from the Media Library at http://service.sap.com/sld.

Recommendations for assigning SLD user roles and access authorizations

Assigning SLD authorizations is not much different from assigning authorizations in the SAP R/3 world, a familiar and well-known task for SAP administrators that I will not belabor here. Do keep in mind the following, however:

- In most cases, the SAP J2EE Engine administrator is also the SLD administrator, so you can verify if the SAP_J2EE_Admin user profile is granted to the user.
- If you don't want the SLD administrator to have SAP J2EE Engine administrator rights, you need to add the SAP_SLD_Administrator role to the user.


Welcome, George Yu
my Profile
SAP Support Portal
Other Portals | Quick Links | Sitemap | Help
SAP Service Channel - my Inbox
Search
Go!
Advanced Search
HOME
Problem Solving
Downloads
Keys & Requests
Data Administration
Maintenance & Services
Tools & Methods
Release Info
Knowledge Transfer
SAP Software Distribution Center
SAP Installations & Upgrades
SAP Support Packages
Database Patches
Download Basket
Additional Download Info
You are here:
#OS independent
SAP Software Distribution Center
Download
Support Packages and Patches
Search for Support Packages and Patches
Entry by Application Group
My Company's Application Components
Archive for Support Packages and Patches
SAP Support Packages in Detail
SAP Support Package Stacks
Installations and Upgrades
Database Patches (from other vendors)
SAP Cryptographic Software
SAP Connectors
Search for all Categories
Ordering SAP-Software
Download Basket
Copyright Privacy Imprint
Info Page
Downloads
Navigation Tree Filter OFF
Add to Download Basket
Maintain Download Basket
Select All
Deselect All

Support Packages and Patches

Support Packages and Patches → Additional Components → SAP MASTER DATA FOR SLD → SAP MASTER DATA FOR SLD 1.0

SAP MASTER DATA FOR SLD 1.0

- SAP MASTER DATA FOR SLD 1.0
  - #OS independent

SAP MASTER DATA FOR SLD 1.0 -> #OS independent

You can download one or more files by activating the check box on the left and clicking the button "Add to Download Basket".Please click here fore more detailed information.

The following objects are available for download:

	File Type	Download Object	Title	Patch Level	Info File	File Size [kb]	Last Changed
☐	ZIP	cimsap13_5-10002694.zip	SAP CIM model 1.3	5	Info	100	21.03.2005
☐	ZIP	CRCONTENT13_0-10002694.ZIP	CR Full Export for SAP CIM model 1.3	0	Info	1583	29.10.2003
☐	ZIP	CRDelta13_13-10002694.zip	CR content delta	13	Info	2107	22.02.2005
☐	ZIP	CRDelta13_14-10002694.zip	CR content delta	14	Info	2206	21.03.2005

Figure 15 Downloading SLD component data from the SAP Service Marketplace

- For the majority of other SLD users, I provide user authorizations only as needed.

Now that you've assigned the appropriate authorizations to the appropriate users, you next need to prepare the SLD contents by importing the CIM model and the contents of the master CR from the Software Distribution Center at the SAP Service Marketplace (http://service.sap.com/swdc).

Importing the CIM model and the contents of the master CR

The master Component Repository (CR) contains information about all of the software components available from SAP. When a new product becomes available, a new support package is released to the master CR, or when a maintenance schedule has changed, this information is released to the master CR. The CR content is released from SAP in full content form and as subsequent delta content (i.e., the content changes since the last full content release).

In addition to updating SLD with content from the master CR, the CIM model itself is modified frequently by SAP based on the changes made by the Distributed Management Task Force. Since the data based on the old model does not work with the new model, you must periodically update the CIM model used in SLD.

Figure 15 displays the location from which the master CR files can be downloaded to your local SLD server. From the Software Distribution Center at the SAP Service Marketplace (http://service.sap.com/swdc), a registered SAP customer can download all licensed software and its relevant support packages, such as SAP Web AS 6.40, SAP SCM 4.1, SAP CRM 4.0, and so on. You can download the master CR content files by navigating to Download → Support Packages and Patches → Entry by Application Group in the left-hand navigation pane, and then in the right pane selecting Additional Components → SAP MASTER DATA FOR SLD → SAP MASTER DATA FOR SLD 1.0 → OS Independent. For the 2004s version of SLD, select SAP MASTER DATA FOR SLD 2.0.

There are four types of master CR content-related files available at the SAP Software Distribution Center:

- A file containing the **CIM model definition**, such as cimsap13_5-10002694.zip, where 13 represents the CIM model release number, 5 represents the patch level, and 10002694 represents the model content compilation number
- A file containing the **initial CR content**, such as CR_Content.zip, which is usually delivered on the SAP J2EE Engine installation CD
- A file containing the **full CR content** based on the latest CIM model, such as CRCONTENT13_0-10002694.zip, where 13 represents the CIM model release number, 0 represents the patch level, and 10002694 represents the full content compilation number (if the model number is the same as that on the SAP J2EE Engine installation CD, this file is identical to CR_Content.zip)
- A file containing the **delta CR content** based on the latest full master CR content file, such as CRDelta13_14-10002694.zip, where 13 represents the CIM model release number, 14 represents the patch level, and 10002694 represents the full content compilation number

The file containing the CIM model, the file containing the full master CR content, and the file containing the delta CR content must be downloaded to your SLD server to ensure the information in SLD is up to date. (The file containing the initial CR content is only necessary if it is based on the same latest CIM model.) With the ongoing changes to the CIM model and the master CR content, these three file types are released periodically by SAP, similar to support packages for other software, so be sure to check them frequently.

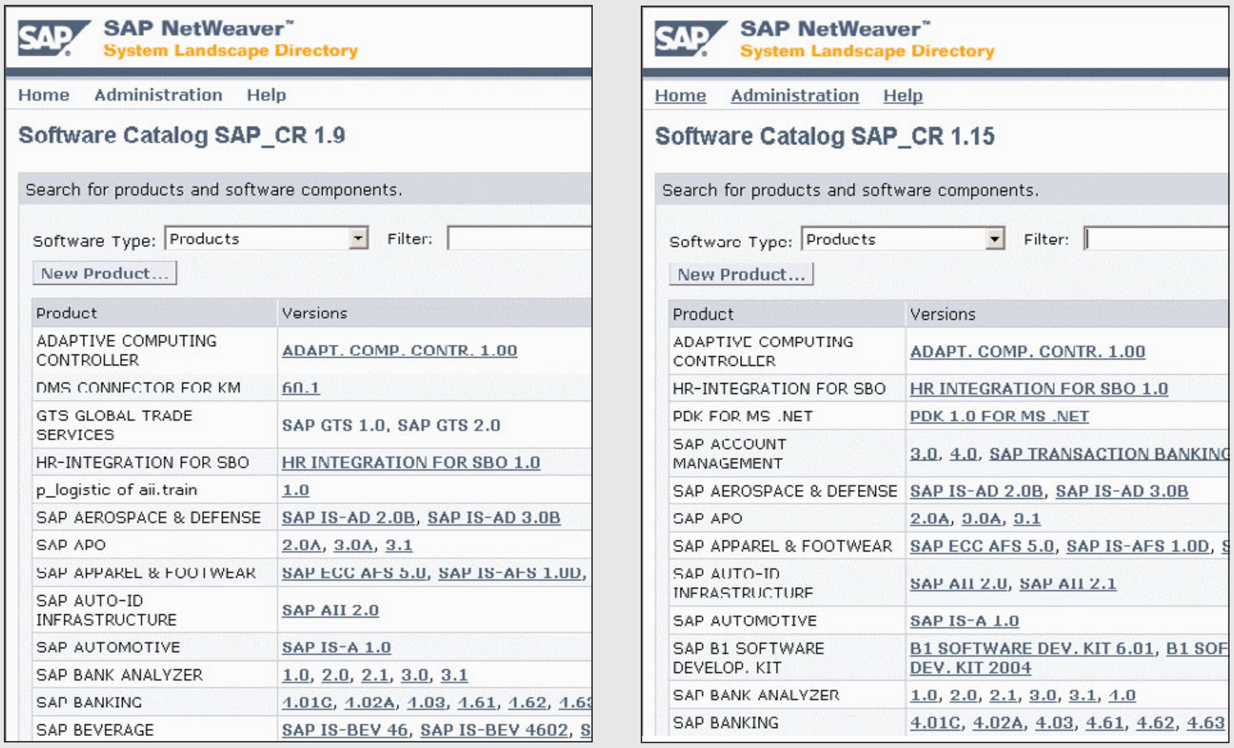
The SLD component data can be downloaded either to your local PC or to a server. If you download the data to your local PC, it will be moved to the SLD server during the import process. Right before the import starts, the data should be located at the \\<server>\usr\sap\<SAPSID>\SYS\global\sld\model\ directory.

After downloading the CR files to your SLD server, click on the Administration hyperlink on the SLD home page and then click on the Import link under Content to import both the CIM model file and the content files. Prior to importing a content file, the SLD importing program checks the CIM model information. If the content file is based on a CIM model newer than the currently installed CIM model, a warning message is displayed and the import is not executed until the CIM model is updated. In this case, you need to download the CIM model and import it first.

To check the current CR content, click on the Software Catalog link on the SLD home page, which takes you to the list of software components. **Figure 16** shows two software catalogs, one for SAP_CR 1.9 and another for SAP_CR 1.15. The numbers represent the release version of the CR content — in this case, 1.15 is newer than 1.9. Several differences are visible:

- Some products listed in SAP_CR 1.9 have been moved due to name changes, such as GTS Global Trade Service, which became SAP Global Trade Service (GTS).
- Some new products have been added, such as the PDK for .NET.
- New releases have become available for certain products, such as SAP Auto-ID Infrastructure 2.1.

This example clearly shows the importance of keeping the CR content current so that you have the latest information to work with. In some cases, a lack of current information will even prevent a solution from functioning. To set up an SAP XI integration scenario, you need to define a business system, which uses the component information stored in SLD. If you



Software Catalog SAP_CR 1.9

Search for products and software components.

Software Type: Products Filter:

Product	Versions
ADAPTIVE COMPUTING CONTROLLER	ADAPT. COMP. CONTR. 1.00
DMS CONNECTOR FOR KM	60.1
GTS GLOBAL TRADE SERVICES	SAP GTS 1.0, SAP GTS 2.0
HR-INTEGRATION FOR SBO	HR INTEGRATION FOR SBO 1.0
p_logistic of aii.train	1.0
SAP AEROSPACE & DEFENSE	SAP IS-AD 2.0B, SAP IS-AD 3.0B
SAP APO	2.0A, 3.0A, 3.1
SAP APPAREL & FOOTWEAR	SAP ECC AFS 5.0, SAP IS-AFS 1.0D,
SAP AUTO-ID INFRASTRUCTURE	SAP AII 2.0
SAP AUTOMOTIVE	SAP IS-A 1.0
SAP BANK ANALYZER	1.0, 2.0, 2.1, 3.0, 3.1
SAP BANKING	1.01C, 1.02A, 1.03, 1.61, 1.62, 1.63
SAP BEVERAGE	SAP IS-BEV 46, SAP IS-BEV 4602, S

Software Catalog SAP_CR 1.15

Search for products and software components.

Software Type: Products Filter:

Product	Versions
ADAPTIVE COMPUTING CONTROLLER	ADAPT. COMP. CONTR. 1.00
HR-INTEGRATION FOR SBO	HR INTEGRATION FOR SBO 1.0
PDK FOR MS .NET	PDK 1.0 FOR MS .NET
SAP ACCOUNT MANAGEMENT	3.0, 4.0, SAP TRANSACTION BANKING
SAP AEROSPACE & DEFENSE	SAP IS-AD 2.0B, SAP IS-AD 3.0B
SAP APO	2.0A, 3.0A, 3.1
SAP APPAREL & FOOTWEAR	SAP ECC AFS 5.0, SAP IS-AFS 1.0D, S
SAP AUTO-ID INFRASTRUCTURE	SAP AII 2.0, SAP AII 2.1
SAP AUTOMOTIVE	SAP IS-A 1.0
SAP B1 SOFTWARE DEVELOP. KIT	B1 SOFTWARE DEV. KIT 6.01, B1 SOF DEV. KIT 2004
SAP BANK ANALYZER	1.0, 2.0, 2.1, 3.0, 3.1, 4.0
SAP BANKING	4.01C, 4.02A, 4.03, 4.61, 4.62, 4.63

Figure 16 CR 1.9 content and CR 1.15 content

don't have the right component information in SLD, you cannot define the business system, and the integration scenario won't run.

Recommendations for importing the CIM model and the contents of the master CR

Having the most up-to-date CR content and CIM model is critical. Toward this end, keep the following in mind:

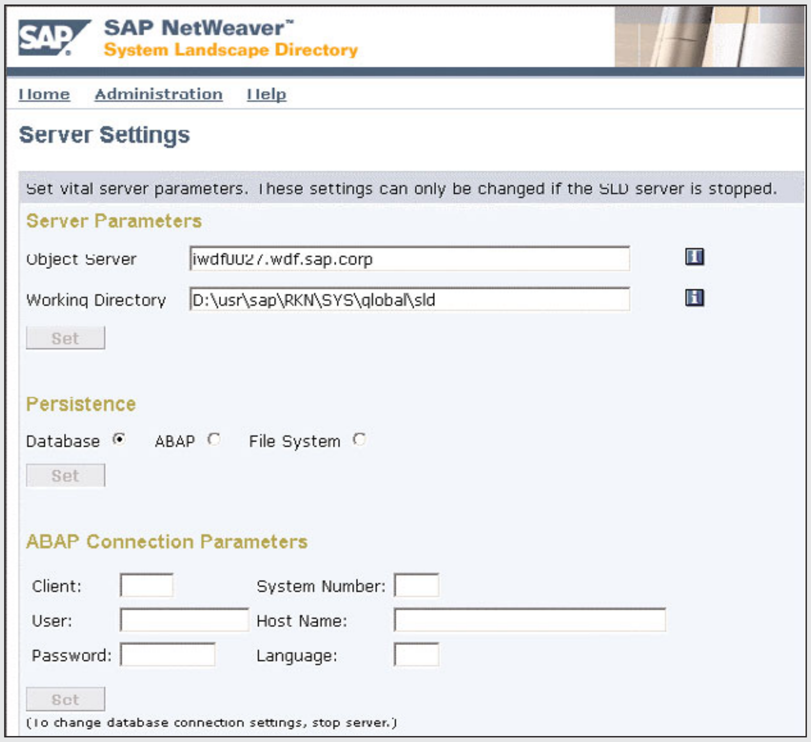
- Get the latest CIM model and the full master CR content based on it during the initial SLD installation. It takes 1+ hours to do the import, so get it out of the way as soon as possible.

- Be sure to update both the CIM model and software component information periodically to include the latest information. This could be done along with applying any support packages, since you will already have the system down.

With the CIM model and CR content updated, the server must be prepared to store content, receive system information, and respond to requests for information. We'll examine this over the next few sections. First, we'll look at how to set up the SLD server so that it can store content.

Specifying the SLD server settings

The SLD server is installed by default during the SAP J2EE Engine installation; there is no manual installation of the SLD server itself. While a dialog appears during SAP Web AS installation asking if there is an SLD server to be used, this is intended for configuring the system as a data supplier to be connected to an



Server Settings

Set vital server parameters. These settings can only be changed if the SLD server is stopped.

Server Parameters

Object Server	iwdf0027.wdf.sap.corp
Working Directory	D:\usr\sap\RKN\SYS\global\sld

Set

Persistence

Database [✓] ABAP File System

Set

ABAP Connection Parameters

Client:		System Number:	
User:		Host Name:	
Password:		Language:	

Set

(To change database connection settings, stop server.)

Figure 17 Defining the SLD server settings in the 6.40 version of SLD

Note!

As of the 2004s version of SLD, the SLD server settings have been incorporated into the server profile parameters. To access the server profile parameters, click on the Profiles hyperlink on the SLD Administration page and select Server Settings or All in the Section dropdown list. You can edit the server profile in a Web browser, or you can download it as a local file and then upload it after making the changes. If you are testing some parameters, it is a good idea to download your current settings to a file so that you can restore them easily. There is no version control mechanism to restore previous parameter settings.

existing SLD server (more on configuring data suppliers in a later section). The SLD software is installed regardless of your answer to the prompt.

To set up an SLD server, you first need to define the SLD server settings (**Figure 17**). To access the SLD server settings screen in the 6.40 version of SLD, click on the Administration hyperlink on the SLD home page and choose Server → Server Settings.

For the simplicity of this discussion, we will look at the 6.40 version of the SLD interface shown in Figure 17. As you can see, there are three components of the server settings:

- **Server Parameters** define the server name and its working directory:
  - The Object Server parameter

(iwdf0027.wdf.sap.corp in the example) together with the CIM namespace (sld/active in the example, located in the lower portion of the screen, which is not visible in Figure 17) identifies the physical location of SLD. The parameter value can be an ABAP namespace reserved in the SAP Service Marketplace or, if you just want to install a test or development system, it can be the host name of the SAP J2EE Engine. If you do not use a reserved namespace for the Object Server parameter, however, you cannot use this SLD server name for name reservation in the SAP NetWeaver Java Development Infrastructure to distinguish your development objects, which can result in confusion if you later decide to merge development objects from two systems.

- The Working Directory parameter is used to store application configuration files — for example, the CIM model and software component import/export files. The working directory also can be used to store SLD data (discussed next). If there are multiple SAP J2EE Engine instances serving the same SLD server (for load balancing or high-availability purposes, for example) the path must point to the same directory. The SLD servers on these SAP J2EE Engine instances should share the same working directory. The working directory is created automatically during SLD installation on the file system. By default, it is /usr/sap/<SAPSID>/SYS/global/sld for a Unix platform and \\<server>\usr\sap\<SAPSID>\SYS\global\sld for a Microsoft Windows platform.

- **Persistence** specifies where the SLD data is stored. The SLD server supports three options for saving data: database, ABAP, and file system. SAP recommends database persistence, so that all SLD data is stored in a database schema as part of the SAP J2EE Engine. The ABAP persistence option is for saving SLD data as part of the ABAP database, and the file system persistence option is for (local) test purposes only, in which case the data is stored in the working directory. Neither ABAP persistence nor file system persistence is recommended in a customer environment to eliminate any dependencies and because data is more secure in a database than in a plain file at the OS level.

- **ABAP Connection Parameters** provide the system and user logon information for the ABAP database if ABAP persistence is selected for storing the SLD data. If ABAP persistence is not selected, leave this section blank.

Note!

Be sure to click on the Set button in each setting area after making your changes, otherwise the changes will not be saved.

To make server setting changes, you first need to stop the SLD server by clicking on the toggle button in the upper left of the SLD Administration page. You then need to restart the SLD server for the changes to take effect.

Note!

Although starting and stopping the SLD server has no direct impact on the SAP J2EE Engine, you won’t be able to access the SLD server if the SAP J2EE Engine is not running. There are two reasons for this:

- The SLD server is a service of the SAP J2EE Engine.
- The communication to the SLD server depends on the HTTP service of the SAP J2EE Engine.

Therefore, a running SAP J2EE Engine is mandatory for working with an SLD server.

Recommendations for specifying the SLD server settings

Keep the following in mind when you are setting up your SLD server:

- Don't make changes to server parameters unless you are absolutely sure. Otherwise, you could accidentally end up using a different SAP J2EE Engine for your SLD server than the one you intended, for example.
- Always use database persistence for SLD data so that the data is constantly backed up and you can recover from any disasters. Plain files at the OS level are not necessarily backed up at many data centers.
- Remember to stop the SLD server to make changes, and restart it afterward.
- The 2004s version of SLD includes additional profile parameters, such as time-out and tracing capabilities, that you can use to fine-tune SLD server performance. In my experience, the default settings are appropriate in most cases.

With the SLD server settings in place, you need to configure the SLD data supplier bridge to receive data from those systems in the landscape. We'll look at this next.

Configuring the SLD data supplier bridge

The systems that provide information to an SLD server are called data suppliers. While the SLD server is a Java-based application, it is able to communicate with ABAP-based systems as well as Java-based systems. The data supplier bridge is the key that makes this possible.

The SLD data supplier bridge has two primary tasks:

- Transform the system data from data suppliers into CIM-compliant XML format before sending it to the SLD server.
- Act as a central receiving point for all SLD data from suppliers and forward the data to both local and remote SLD servers.

Figure 18 illustrates the working mechanism of an SLD data supplier bridge. As you can see, there are two SLD servers: Local SLD Server and Remote SLD Server. Each SLD server contains a data supplier bridge with two interfaces, one for ABAP-based programs (RFC Server) and another for Java-based programs (HTTP Servlet). The data sent to Local SLD Server can be forwarded to Remote SLD Server from the local data supplier bridge. Remote SLD Server can receive information from other data suppliers in addition to what is forwarded from Local SLD Server.

ABAP-based SAP systems communicate with each other using Remote Function Calls (RFCs). To make it possible to communicate with any ABAP-based systems, the SLD data supplier bridge uses a gateway service between an ABAP-based data supplier and the Java-based SLD server.

Note!

When you install a standalone SAP J2EE Engine on a server, the RFC gateway service is not installed by default, so you will need to install this gateway service manually if there are ABAP-based data suppliers. This is similar to installing a gateway service for each dialog instance in the SAP R/3 world.

If you prefer not to install a dedicated gateway service — for example, if you want to do a quick test of a connection to an SLD server — you can "borrow" an RFC gateway from an existing system that has a gateway service, as long as both the data supplier and the SLD server go through this gateway service to communicate with each other. In other words, you will use the same gateway service to link a data supplier to an SLD server. Using a similar technique, you can install a gateway service within a

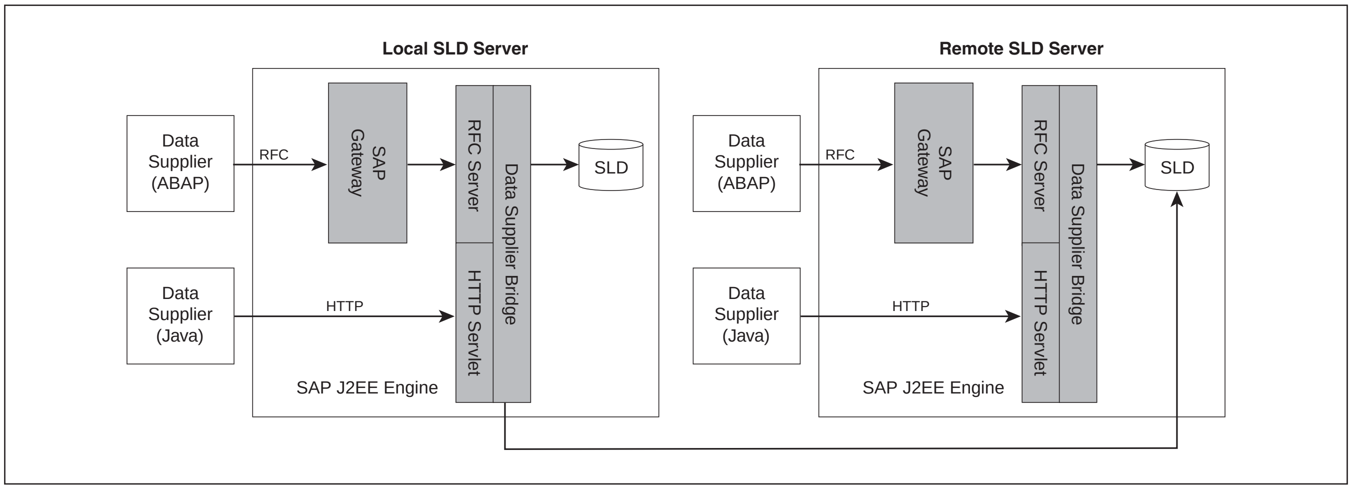


Figure 18 SLD data supplier architecture

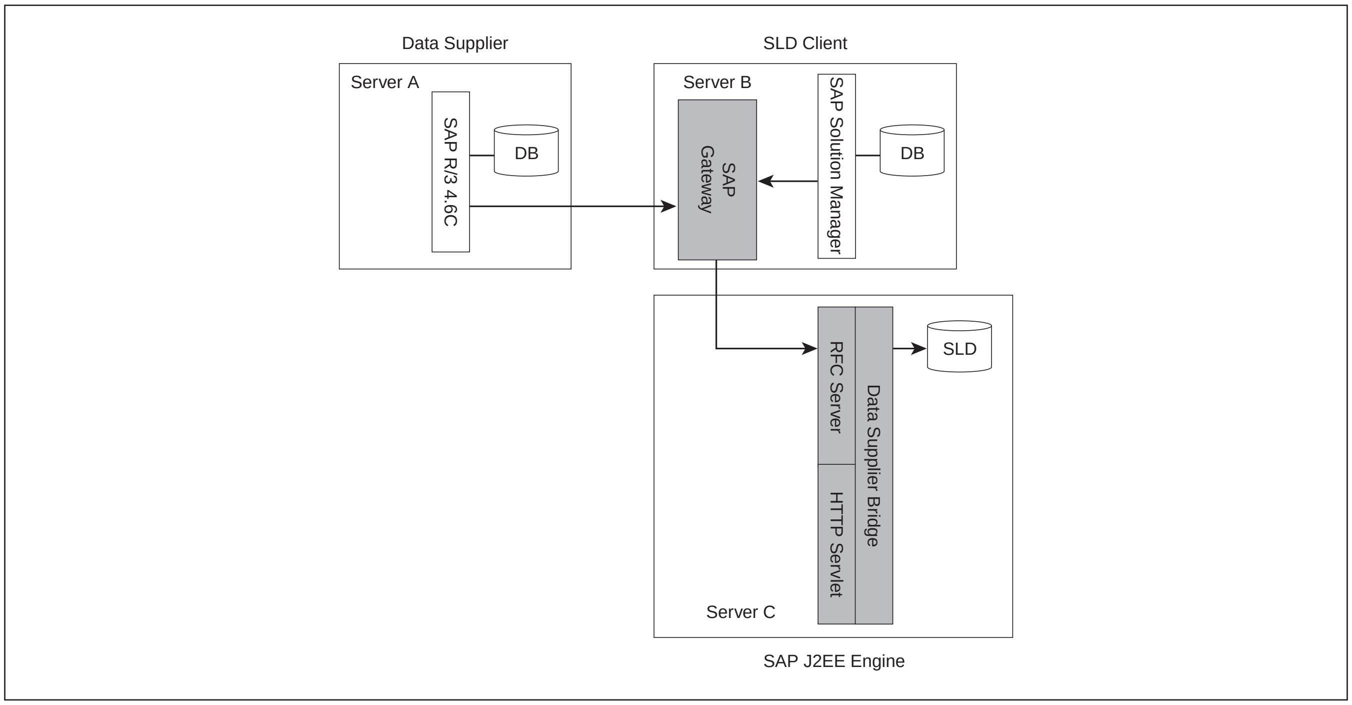


Figure 19 Using an available gateway service for an SLD server

demilitarized zone to insulate and increase the security of the SLD server itself.

Let's look at an example. **Figure 19** shows three systems in a landscape: an SAP R/3 4.6C system as a data supplier on Server A, an SLD server on a standalone SAP J2EE Engine on Server C, and an SAP Web AS 6.40 system on Server B that is an SLD client system and contains both the Java and ABAP stacks to support SAP Solution Manager. Let's assume there is no gateway service on the server where the SLD server resides — it has only an SAP J2EE Engine and its database. We can send the SLD data from the SAP R/3 4.6C system to the SAP Web AS gateway service on Server B, and have the SLD data supplier bridge on Server C point to the same SAP Web AS gateway

Note!

One gateway service cannot be shared by multiple SLD servers. Each SLD server requires a dedicated gateway service to avoid any confusion about which SLD server it serves.

service. With this arrangement, the communication between the data supplier SAP R/3 4.6C system on Server A and the SLD server on Server C is possible.

If the data supplier is a Java system, it can send the SLD data directly to the SLD data supplier bridge via HTTP, since it is a communication between two Java programs. There is no RFC gateway service involved. Although it is possible to use an RFC connection to send data to an SLD server from a Java program, it is not recommended since it makes a simple connection overly complicated and has a negative effect on performance.

If a local SLD data supplier bridge forwards data to a remote SLD server, as shown in Figure 18, it will forward only the following types of information:

- ABAP system data provided by the ABAP SLD data supplier (transaction RZ70, SLD Administration)
- SAP J2EE Engine system data provided by the Java SLD data supplier
- Server host data provided by the SAP OS collector program SAPOSCOL
- Other system data provided by non-ABAP/ non-Java SLD data suppliers via the program SLDREG,[7] available as of SAP NetWeaver 2004s

In this case, changes made manually to the content of an SLD server — for example, changes to the landscape definition and component information — via the SLD user interface, or via SAP XI or SAP Solution Manager, are not forwarded to another SLD server. These types of changes are kept in the first SLD server, or you can manually export them to a subsidiary server. Only data generated by data collection programs can be forwarded to a remote SLD server.

In an upcoming release of SAP NetWeaver, there are plans to provide automatic synchronization of all SLD data, including manually entered data, so that only the CIM model must be manually updated. This will dramatically reduce the operation effort for large customers that have complex SLD landscapes.

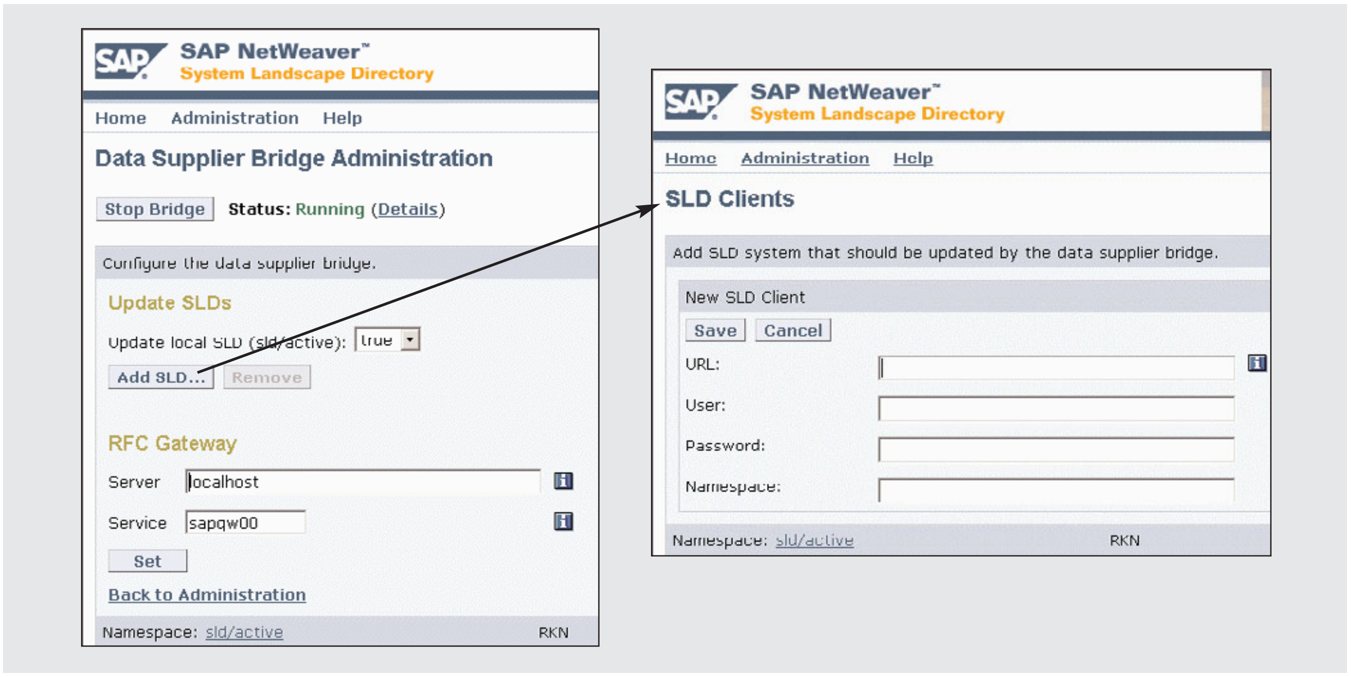
SLD data supplier bridge administration

To enable automatic content forwarding for synchronizing multiple SLD servers, as mentioned earlier in the discussion on distributed SLD servers, you need to configure the data supplier bridge as described here. To access the data supplier bridge administration screen (**Figure 20**), click on the Administration link on the SLD home page, and then click on Server → Data Supplier Bridge. There are two sections: Update SLDs and RFC Gateway.

In the Update SLDs section, you need to specify where the SLD data supplier bridge should send the SLD data. If the SLD server receiving the data resides on the same server as the SLD data supplier bridge, select true for the Update local SLD field. Select false to forward the SLD data to a destination on a remote SLD server host. In the example in Figure 20, this local server acts as both an SLD bridge and an SLD data store.

If you want this SLD server to keep a local copy of the SLD content, but also forward it to a remote SLD server, select true for the Update local SLD field and click on the Add SLD button, as shown in Figure 20. In this example, the local SLD server acts as a Java SLD data supplier, so the communication between the SLD data supplier bridge and the remote SLD server follows the HTTP protocol. For this reason, you enter a URL (such as http://iwdf0025:50000) instead of a server name to identify the server. Make sure the user, password, and namespace information is for the

[7] Program SLDREG builds a connection between the SLD server and a C-based data supplier application. Details on this program can be found in the “System Landscape Directory, SAP NetWeaver 2004s SPS09” document available from the Media Library at http://service.sap.com/sld.



Data Supplier Bridge Administration

Status: Running (Details)

Configure the data supplier bridge.

Update SLDs

Update local SLD (sld/active): true

RFC Gateway

Server: localhost

Service: sapqw00

Namespace: sld/active RKN

SLD Clients

Add SLD system that should be updated by the data supplier bridge.

New SLD Client

URL:

User:

Password:

Namespace:

Namespace: sld/active RKN

Figure 20 SLD data supplier bridge administration

remote SLD server, not the current SLD server containing the data supplier bridge.

You can use this mechanism to forward data from both a Java client and an ABAP client, because at this point the communication is between two Java applications, a local SLD bridge and a remote SLD server.

The RFC Gateway section in Figure 20 is for enabling the SLD data supplier bridge to receive data from an ABAP-based data supplier. In this case, enter the gateway service information for the SLD server. Otherwise, you can leave this section blank; a Java-based system will not require a gateway service, since the communication will be between two Java programs. Your settings for the data supplier bridge will take effect when you restart the SLD server.

Note!

In the 2004s version of SLD, namespace information can no longer be entered on the SLD Clients screen. The remote server takes the default sld/active namespace.

Note!

In the 2004s version of SLD, the gateway service information has been moved to the SLD server profile parameters.

Recommendations for configuring the SLD data supplier bridge

Keep the following in mind when configuring a data supplier bridge for your SLD server:

- Make sure you have only one gateway service serving one SLD server.

- Using a temporary gateway service for an SLD server is possible, but it is better to install a dedicated one for your SLD server so that you do not become confused by which SLD server or ABAP system it is for.

- The data supplier bridge is the entry point for all SLD data. Make sure it points to the desired SLD server. If you do not receive data from the SLD server's data suppliers, check whether you selected true for updating the local SLD namespace, and whether you have specified a gateway service to receive ABAP system input.

- When using multiple SLD servers, consider using a data supplier bridge to forward system information from one SLD server to another. The reason is that each data supplier can only send its data to one SLD server.

With the SLD data supplier bridge prepared to receive information and direct it to the SLD server (or servers), it's time to configure the data suppliers that send the information. We'll look at the key considerations involved in this configuration next.

Configuring the SLD data supplier

SLD data collection is based on a "push" mechanism. The SLD data supplier periodically "pushes" (or sends) data to the SLD server (the data supplier bridge, to be precise) via a batch program scheduled on the data supplier. This usually happens during startup of the data supplier system, and at a set time interval afterward. The default interval is every 720 minutes (12 hours). You can change this default time interval to update the SLD server according to your system information changes in a shorter or a longer period of time. If a data supplier stops sending the latest information to the SLD server (if the data supplier bridge is disabled or the batch job is canceled on the data supplier, for example), the SLD client (for example, the SAP XI system) can retrieve only what exists on the SLD server; there is no way to "pull" (or request) an update of the data from the SLD data supplier.

To guarantee a continuous update of SLD data, you need to configure the data supplier properly. There are three types of data suppliers: ABAP-based, Java-based, and non-ABAP/non-Java-based. In this article, I examine only ABAP-based and Java-based data suppliers. For details on configuring non-ABAP/non-Java-based data suppliers, which are beyond the scope of this article, see the SLD documentation.

Using an ABAP-based data supplier

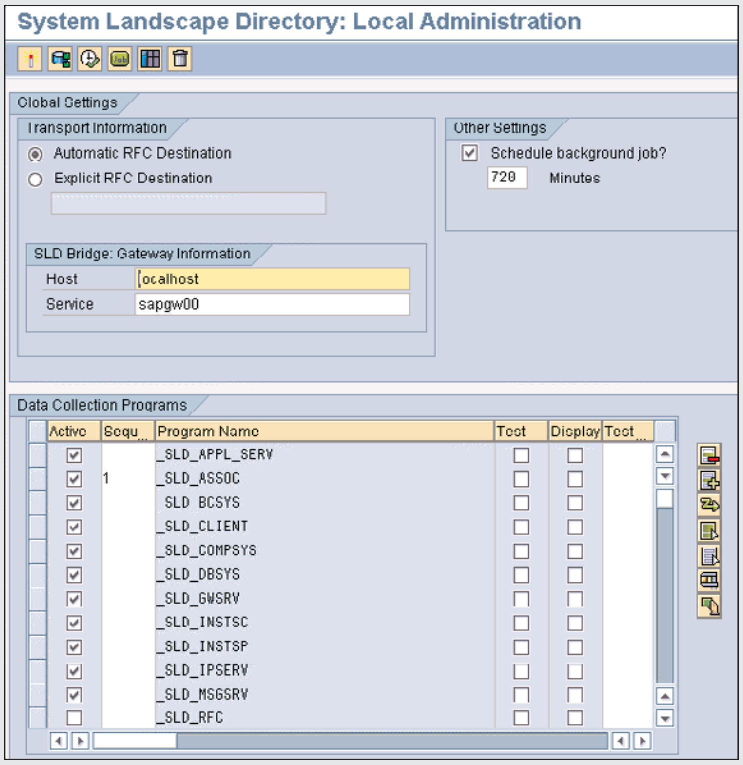
In addition to SAP ECC 5.0 and 6.0 systems, SAP R/3 4.0B to 4.7 systems can act as SLD data suppliers to an SLD server as long as adequate support packages are applied. SAP Note 584654 details the support package requirement for each SAP R/3 release. The SLD data is sent via an RFC connection to the RFC gateway service of an SLD server.

The collected ABAP system information includes:

- System name
- System license number
- Central instance (host name, instance number)
- Message server (host name, port)
- Dialog instances (host name, port)
- Clients (client number, ALE name)
- Installed software components

An ABAP-based data supplier is configured in transaction RZ70 (SLD Administration). **Figure 21** shows the RZ70 screen in an SAP NetWeaver 2004s ABAP system. Information needed for the configuration is divided into four sections:

- The **Transport Information** section specifies the details of the SLD data delivery vehicle (i.e., how the system information is delivered to a designated SLD server). There are two choices for the RFC call to the gateway service of the SLD data supplier bridge: Automatic RFC Destination and



System Landscape Directory: Local Administration

Global Settings

Transport Information

(●) Automatic RFC Destination

( ) Explicit RFC Destination

SLD Bridge: Gateway Information

Host	localhost
Service	sapgw00

Other Settings

[✓] Schedule background job?

720 Minutes

Data Collection Programs

Active	Sequ...	Program Name	Test	Display	Text ...
✓		_SLD_APPL_SERV	☐	☐	
✓	1	_SLD_ASSOC	☐	☐	
✓		SLD BCSYS	☐	☐	
✓		_SLD_CLIENT	☐	☐	
✓		_SLD_COMPSYS	☐	☐	
✓		_SLD_DBSYS	☐	☐	
✓		_SLD_GWSRV	☐	☐	
✓		_SLD_INSTSC	☐	☐	
✓		_SLD_INSTSP	☐	☐	
✓		_SLD_IPSERV	☐	☐	
✓		_SLD_MSGSRV	☐	☐	
☐		_SLD_RFC	☐	☐	

Figure 21 ABAP-based SLD data supplier configuration in transaction RZ70

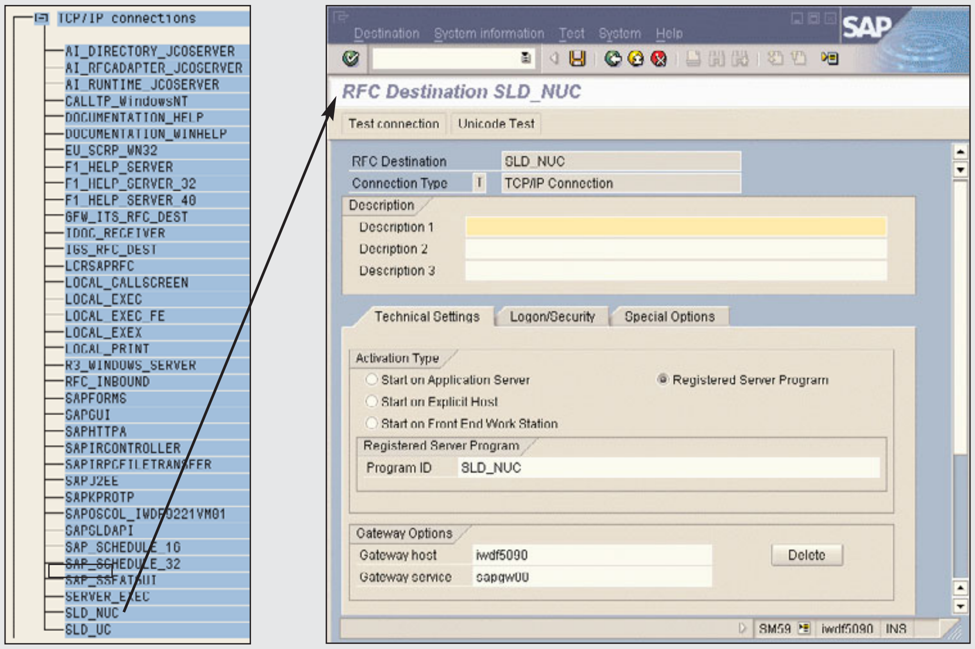
Explicit RFC Destination. The default choice is Automatic RFC Destination, which creates the destination SLD_NUC for a non-Unicode data supplier system or SLD_UC for a Unicode data supplier system. The system chooses the proper RFC destination based on the characteristics of your data supplier system. I recommend using Automatic RFC destination for the sake of convenience. Otherwise, you have to manually create a new RFC destination.

Figure 22 on page 34 shows the details of the automatically generated RFC destination SLD_NUC in transaction SM59 (Maintain RFC Destinations). As you can see, the RFC refers to a registered program SLD_NUC (SLD_UC in the case of the SLD_UC RFC destination). The purpose of these registered programs is discussed in the upcoming section on accessing SLD data from an ABAP-based system.

Note!

It is good practice to verify the SLD server gateway service information during the SLD server configuration even though it is automatically generated. If you provided incorrect information during installation, make changes to the gateway service accordingly here. Otherwise, the information will be sent to the wrong SLD server, or you won't see the expected information in your SLD server.

If you select Explicit RFC Destination, you first need to define an RFC connection and then enter its name in the provided space in Figure 21. When



TCP/IP connections
- AI_DIRECTORY_JCOSERVER
- AI_RFCADAPTER_JCOSERVER
- AI_RUNTIME_JCOSERVER
- CALLTP_WindowsNT
- DOCUMENTATION_HELP
- DOCUMENTATION_WINHELP
- EU_SCRP_WN32
- F1_HELP_SERVER
- F1_HELP_SERVER_32
- F1_HELP_SERVER_40
- GFW_ITS_RFC_DEST
- IDOC_RECEIVER
- IGS_RFC_DEST
- LCRSAPRFC
- LOCAL_CALLSCREEN
- LOCAL_EXEC
- LOCAL_EXEC_FE
- LOCAL_EXEX
- LOCAL_PRINT
- R3_WINDOWS_SERVER
- RFC_INBOUND
- SAPFORMS
- SAPGUI
- SAPHTTPA
- SAPIRCONTROLLER
- SAPIRPCFILETRANSFER
- SAPJ2EE
- SAPKPROTP
- SAPOSCOL_IWDF0221VM01
- SAPSLDAPI
- SAP_SCHEDULE_16
- SAP_SCHEDULE_32
- SAP_SSFATGUI
- SERVER_EXEC
- SLD_NUC
- SLD_UC

RFC Destination SLD_NUC

RFC Destination	SLD_NUC
Connection Type	T TCP/IP Connection

Description

Description 1	
Decription 2	
Description 3	

Activation Type
- ○ Start on Application Server
- ○ Start on Explicit Host
- ○ Start on Front End Work Station
- ● Registered Server Program

Registered Server Program

Program ID	SLD_NUC

Gateway Options

Gateway host	iwdf5090
Gateway service	sapgw00

SM59 iwdf5090 INS

Figure 22 RFC destination details in transaction SM59

defining an RFC connection for a Unicode system, make sure you select the Unicode option on the Special Options tab in Figure 22. The RFC connection is based on the nature of the data supplier system, not the SLD server. If it is not a Unicode system, leave this option out.

- The **SLD Bridge: Gateway Information** section is for specifying the RFC gateway service of the SLD data supplier bridge. For an ABAP-based data supplier, the RFC gateway service is mandatory. Although it appears that the same gateway service information is provided in transaction RZ70 (Figure 21) and transaction SM59 (Figure 22), the settings are not redundant. Once I had difficulty sending data from an ABAP-based system to an SLD server. The cause was missing gateway service information in transaction RZ70. After entering the correct gateway service information, the SLD server received system updates immediately.

Note!

The gateway service information in transaction RZ70 applies to all clients associated with an ABAP-based data supplier system. This means that information from all clients of the data supplier system go to the same SLD server. Currently, there is no simple solution to let each client point to a different SLD server from the same system.

- The **Other Settings** section specifies a time interval to collect and send SLD data. The default setting of 720 minutes (12 hours) is sufficient in most cases, considering your system information should not change that frequently. When all the settings are complete in transaction RZ70, a batch job is scheduled according to this time interval.

Data collection program	System element for which data is provided	Recommended use
_SLD_APPL_SERV	Application servers/instances	When ABAP system-related information is needed, such as SID, host name, and instance profile
_SLD_BCSYS	SAP system	When ABAP Basis-related information is needed, such as TMS, ABAP version, and system number
_SLD_CLIENT	Clients	When client-specific information is needed for each available client, such as client number, currency type, location, and last changed by
_SLD_COMPSYS	Hosts	When computer host information is needed, such as operating system and version, RAM, and IP address
_SLD_DBSYS	Database	When the underlying database information is needed, such as the brand, release number, and database name
_SLD_GWSRV	Gateway service	When the gateway service information is needed, such as message port and service name
_SLD_INSTSC	Installed software components	When the installed software component information is needed, such as SAP R/3 Basis and its release number
_SLD_INSTSP	Installed support packages	When the support packages of all installed software components are needed, such as SPAM version and Basis SP level
_SLD_IPSERV	Network services	When the dispatcher and message server information is needed, such as their name and port number
_SLD_MSGSERV	SAP message server	When the message server information is needed, such as its name and port number
_SLD_RFC	RFC destinations	When the entire list of RFC destinations is needed, such as their name, target host, and registered program (generally not recommended)

Figure 23 SLD data collection ABAP program list

- The **Data Collection Program** section allows you to select from a group of available reporting programs that collect information ranging from application server information to installed SAP software components. A complete list is available in **Figure 23**. Among them, the selection of program _SLD_RFC is not recommended. This program collects all registered RFC connections, which can take a very long time if you have hundreds of RFC destinations. In addition, RFC destination information is more or less internal to the data supplier, and not useful to other systems.

You manage the data collection programs you want to use with the icons on the right side of this section in Figure 21. From top to bottom, they are:

Delete Row: Deletes a selected row containing a data collection program.

Insert Row: Inserts an empty row to add a new data collection program.

Transfer: Transfers a saved set of data collection programs from the database to transaction RZ70.

Select All: Selects all the data collection programs on the screen.

Deselect All: Deselects all the data collection programs on the screen.

Test: Tests the selected data collection program.

Proposal: Proposes to use default data collection programs.

A set of selected programs can be stored in the database so that you can work on them when you return to transaction RZ70. If the program you are interested in is not listed because you didn't select it last time, you can click on the Transfer or Proposal icon to transfer all available programs to the list and then make your choice.

After all information has been entered in Figure 21, you need to activate the current configuration settings and start the data collection. Let's take a look at the icons listed at the top from left to right (note that two of the icons are available only in the 2004s version of SLD, and therefore do not appear in the 6.40 version of SLD):

- **Activate Current Configuration (F8):** Activates all current settings and saves them in the database so they can be recalled when you next run transaction RZ70.
- **Reload Configuration from Database (F6):** Reloads the last saved configuration data.
- **Start Data Collector and Job Scheduling (F5):** Starts all currently selected data collection programs, and assigns them to a batch job to run at a specified time interval.
- **Schedule Job (F7):** Schedules a job to run the currently selected data collection programs. This icon is new with the 2004s version of SLD.
- **Start Data Collector without Job Scheduling (F9):** Starts all currently selected data collection programs without scheduling a job. This is useful for testing your selected programs (that is, testing the communication between the data supplier and the SLD server) before running a batch job. This icon is new with the 2004s version of SLD.
- **Deletes SLD Batch jobs (Shift+F2):** Deletes the currently scheduled data collection job.

Note!

Make sure the Schedule Background Job? option in the Other Settings section is selected when you click on the Start Data Collector and Job Scheduling icon, so that you can specify the time interval information. Otherwise, no batch job is scheduled and SLD data won't be sent to the SLD server periodically.

Based on my own experience, I usually follow these steps when I work with transaction RZ70:

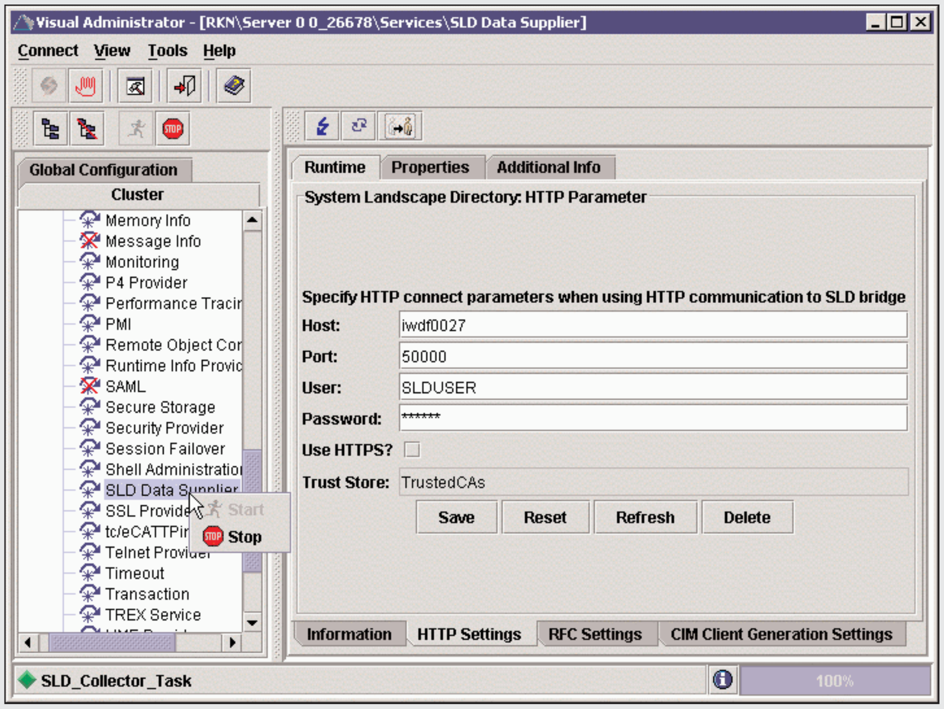
1. Select Automatic RFC Destination.
2. Select the Schedule Background Job? option and specify a time interval no less than 60 minutes; 720 minutes should suffice in most cases.
3. Specify the gateway service.
4. Select all the data collection programs except for the _SLD_RFC program.
5. Run a test to see if the collected data is sent to the SLD server.
6. Schedule a batch job to run the data collection programs.
7. Activate the current configuration.
8. Verify the data sent to the SLD server by the batch job after the data has spent several hours on the SLD server to make sure the batch job is running properly.

Using a Java-based data supplier

The Java-based data supplier settings are easier than the ABAP-based settings. The HTTP servlet of the SLD data supplier bridge can communicate with a Java system via the HTTP protocol. Although a Java-based data supplier can deliver its SLD data via RFC connection, it is not recommended for performance reasons, as discussed previously.

As a prerequisite to configuring an SAP J2EE Engine as an SLD data supplier, the property SynchPermissionsWithDatabase of the SAP J2EE Engine must be set to true. This parameter is located in the SAP J2EE Engine Visual Admin Tool via the menu path Cluster → Server → Service → Security Provider → Properties.

The Java-based SLD data supplier runs as a service on an SAP J2EE Engine. You can access this service from the SAP J2EE Engine Visual Admin Tool by following the path Cluster → Server → Services → SLD Data Supplier (**Figure 24**). From here, you can enter the SLD server access information in the tab



System Landscape Directory: HTTP Parameter

Specify HTTP connect parameters when using HTTP communication to SLD bridge

Host:	iwdf0027
Port:	50000
User:	SLDUSER
Password:	******
Use HTTPS?	☐
Trust Store:	TrustedCAs

Figure 24 Java-based SLD data supplier in the SAP J2EE Engine Visual Admin Tool

Runtime → HTTP Settings, and the time interval for sending the SLD data in the Properties tab. The changes take effect after the SLD data supplier service has been restarted. It takes about two minutes to send the first SLD data to the SLD server after restarting the service.

Checking the data update status from the data supplier

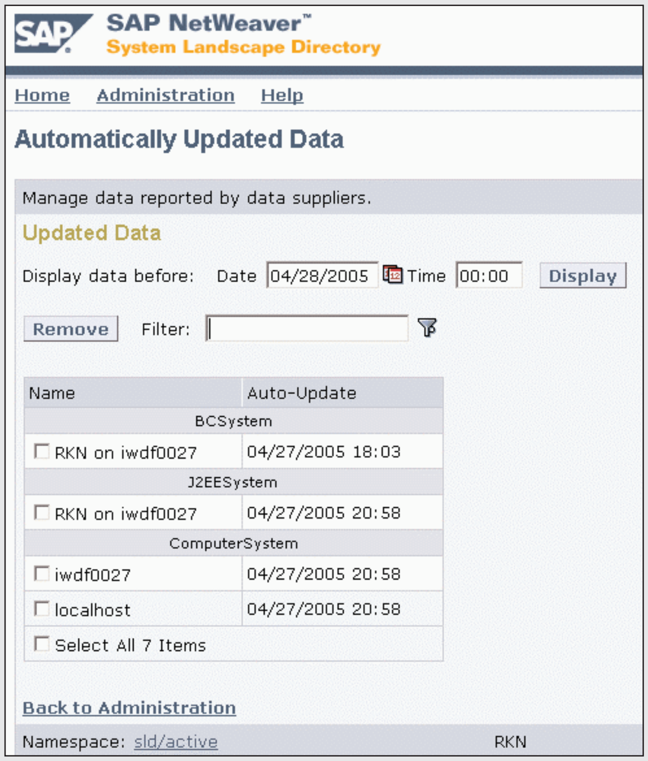
You can view the data update status from the data supplier by clicking on the Automatically Updated Data hyperlink on the SLD Administration page. **Figure 25** on the next page shows the Automatically Updated Data display for the 6.40 version of SLD. The data is divided into two columns, Name and Auto-Update, and three sections:

- BCSystem for the ABAP stack
- J2EESystem for the Java stack
- ComputerSystem for OS information

On an SAP Web AS, the ABAP stack and Java stack are viewed as two separate data suppliers. Each follows its own data reporting schedule, as the data update time stamps are different. For example, system RKN reports ABAP information at 18:03, but Java information at 20:58. The information reported in ComputerSystem can be from either the ABAP stack or the Java stack. The latest update is displayed.

Note!

The Automatically Updated Data display is slightly different in the 2004s version of SLD. It has three columns instead of two, and is no longer divided into three sections. Instead, the newly added Supplier column indicates if the system belongs to one of two categories: BCSystem or J2EESystem. There is no longer a ComputerSystem category.



Automatically Updated Data

Manage data reported by data suppliers.

Updated Data

Display data before: Date 04/28/2005 Time 00:00 Display

Remove Filter:

Name	Auto-Update
BCSystem	
☐ RKN on iwdf0027	04/27/2005 18:03
J2EESystem	
☐ RKN on iwdf0027	04/27/2005 20:58
ComputerSystem	
☐ iwdf0027	04/27/2005 20:58
☐ localhost	04/27/2005 20:58
☐ Select All 7 Items	

Back to Administration

Namespace: sld/active RKN

Figure 25 Checking reported data from the data suppliers

Recommendations for configuring the SLD data supplier

Keep the following in mind when configuring data suppliers for your SLD server:

- The ABAP stack and Java stack report their data separately to the SLD server. For a complete picture, you need to configure both the ABAP stack and Java stack as data suppliers.
- Gateway service information is necessary when reporting ABAP system data. Make sure the ABAP system and the SLD bridge are connected through the same gateway service. There is only one gateway service for one SLD server.
- Frequent data collection is not necessary. The default 720 minutes (12 hours) should suffice in most cases.
- The SLD data can be forwarded to multiple remote SLD servers. This increases the flexibility of having a distributed SLD server environment.

With the SLD content ready, it's time to make the data available to the solutions that need it.

Accessing SLD data

Both ABAP-based and Java-based systems need to access SLD content. For example, when a Java-based Web Dynpro application is accessed, the JCo connection defined in SLD is used, and when the SAP XI Integration Engine is configured, the business system information in SLD is retrieved.

Accessing SLD data from a Java-based system is simple. Since it is a Java client calling a Java server, the Java API built into SLD requires no manual configuration. In essence, the Java client program sends an HTTP request to an SLD server, and the server sends the result back accordingly.

The ABAP API, however, is a bit more complicated, so here we will take a detailed look at the

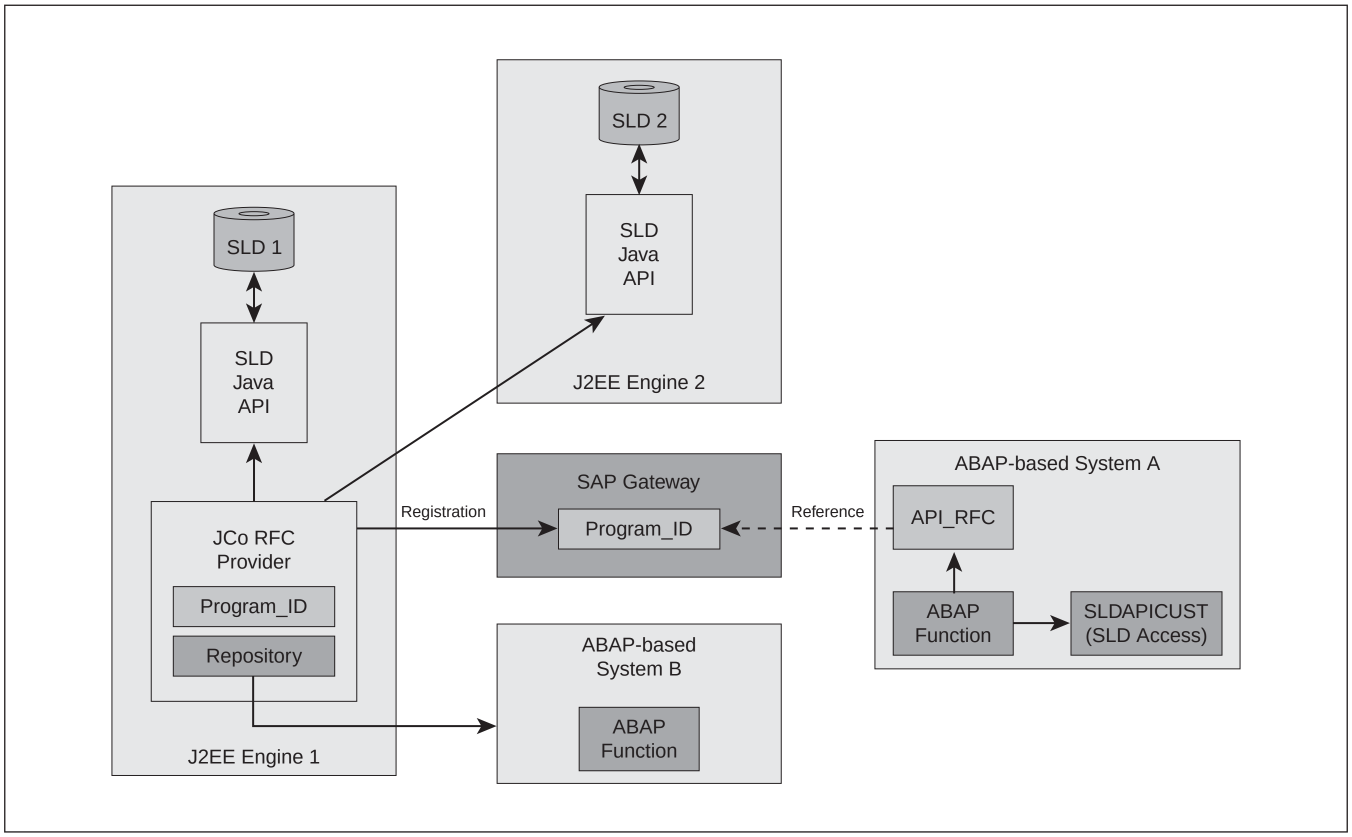
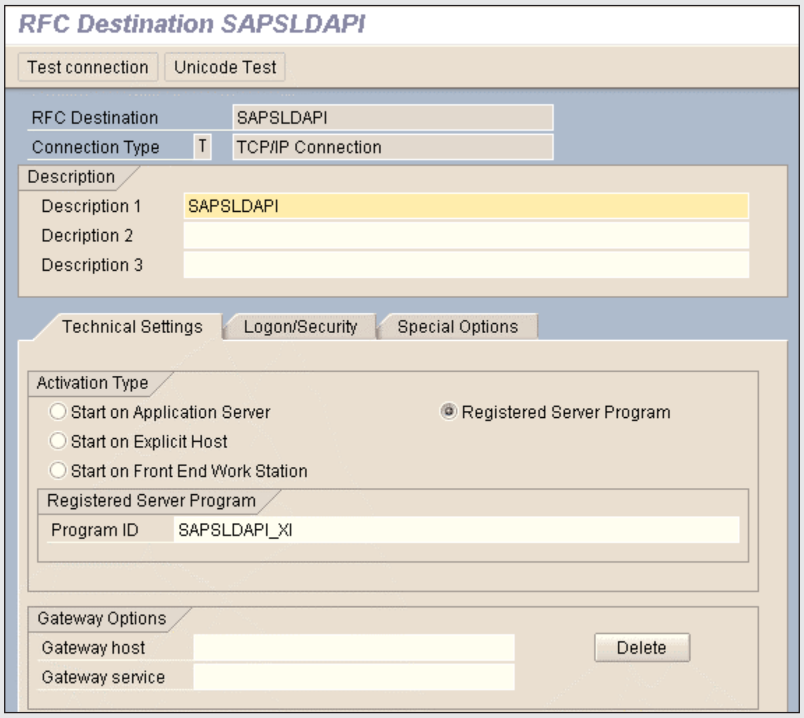


Figure 26 ABAP API for the SLD server

mechanism that enables an ABAP-based client to access the Java-based SLD server. Since SAP XI and SLD are so closely connected (see the download available at www.SAPpro.com for a brief overview of how they interact), I use SAP XI to illustrate how the mechanism works. While SAP XI is used in the example, this same mechanism is used to access SLD data from all ABAP-based SLD clients, which you also will find in SAP Solution Manager, SAP SRM, and SAP AII implementations, so you can apply what you learn here to these implementations as well.

Figure 26 illustrates how a function in an ABAP-based system calls a Java-based SLD server. The SAP J2EE Engine provides a Java Connector (JCo) RFC Provider service for processing ABAP to Java requests. This JCo RFC Provider creates an RFC gateway that serves as a communication tunnel between the ABAP stack and the Java stack and registers itself at the gateway with the name Program_ID. The JCo RFC Provider acts as an RFC server, forwarding the ABAP request to the SLD server via a Java API. The JCo RFC Provider is created on the SAP J2EE Engine with two components — an RFC destination with a Program_ID name and a Repository. The JCo RFC Provider registers its Program_ID name with the RFC gateway service so that the RFC destination on the ABAP-based system (API_RFC on System A in Figure 26) can reference it using a registered program that has the same Program_ID name.

Since the JCo RFC Provider has no knowledge of the ABAP function that calls the SLD server, it needs to access the ABAP system to retrieve the function information. For this reason, the Repository of the JCo RFC Provider stores the access information for the ABAP system. In most instances, the Repository points to the system making the RFC call. In some cases, however, the Repository points to a second system containing that same ABAP function. For example, let's say that a support package has been


RFC Destination SAPSLDAPI

RFC Destination		SAPSLDAPI
Connection Type	T	TCP/IP Connection

Description

Description 1	SAPSLDAPI
Decription 2	
Description 3	

Activation Type: ( ) Start on Application Server, ( ) Start on Explicit Host, ( ) Start on Front End Work Station, [✓ Registered Server Program]

Registered Server Program

Program ID	SAPSLDAPI_XI

Gateway Options

Gateway host	
Gateway service	

Figure 27 Server program associated with the RFC connection

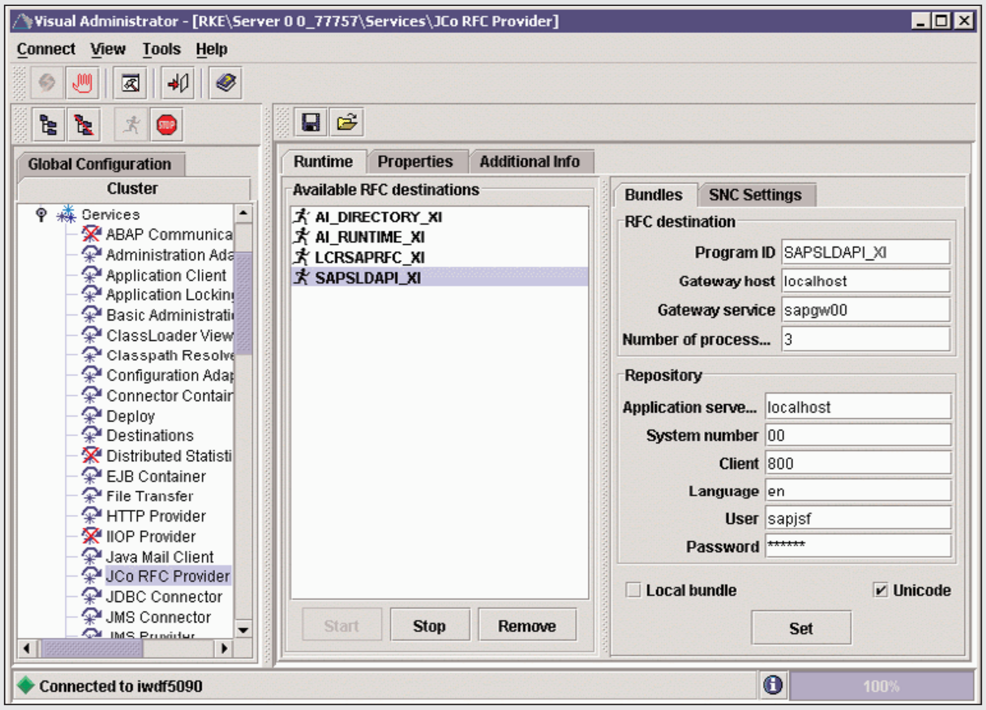
applied to the ABAP function definition in System B, but not in System A — when System A makes the call, the Repository can point to System B to retrieve the updated function definition. In Figure 26, this means that System A makes the call via API_RFC, but the Repository reads the ABAP function information from System B.

For flexibility, SLD server access information is stored in transaction SAPAPICUST on the ABAP-based system. In an environment with multiple SAP J2EE Engines, the SLD server can be located on an SAP J2EE Engine other than the one acting as the RFC server. For example, there are two SAP J2EE Engines in Figure 26 — the JCo RFC Provider is located on J2EE Engine 1, which is acting as the RFC server, but the server to which the ABAP-based System A wants to connect, SLD Server 2, is located on SAP J2EE Engine 2. Transaction SAPAPICUST can store access information for multiple servers (SLD Server 1 and SLD Server 2, in this case), enabling the JCo RFC Provider to call SLD Server 2 on behalf of System A.

Note!

Accessing a Java-based program from an ABAP program is common in SAP solutions. For example, SAP AII is an ABAP program, while printers and scanners are controlled by Java-based applications. SAP AII accesses these devices via a pair of RFC destinations — one for the RFC gateway and another for holding Java application access information, similar to storing access information in transaction SAPAPICUST.

With this background information, let's take a look at how to build this RFC-based mechanism to enable an ABAP-based system to access SLD data using SAP XI as an example.



Available RFC destinations

- AI_DIRECTORY_XI
- AI_RUNTIME_XI
- LCRSAPRFC_XI
- SAPSLDAPI_XI

RFC destination

Program ID	SAPSLDAPI_XI
Gateway host	localhost
Gateway service	sapgw00
Number of process...	3

Repository

Application serve...	localhost
System number	00
Client	800
Language	en
User	sapjsf
Password	******

[ ] Local bundle [✓ Unicode]

Figure 28 RFC destination registered with the JCo RFC Provider service

Building the mechanism for accessing SLD data

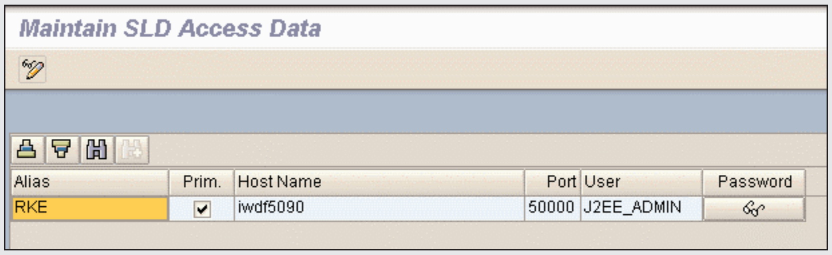
In an SAP XI system, the RFC destination created to link to SLD is called SAPSLDAPI. This RFC name cannot be changed. SAPSLDAPI is created as a TCP/IP connection during SAP XI system installation and is associated with the registered program SAPSLDAPI_XI (**Figure 27**), which corresponds to the Program_ID SAPSLDAPI_XI registered in the JCo RFC Provider of the SAP J2EE Engine (**Figure 28**). While the Program_ID can have any name you want, I prefer to include the RFC name as part of the Program_ID name for easy identification purposes. For example, SAPSLDAPI_XI indicates that the program is defined for RFC SAPSLDAPI, and the JCo RFC Provider is on the SAP XI system. To register SAPSLDAPI_XI with the JCo RFC Provider service, in the SAP J2EE Engine Visual Admin Tool, select Cluster → Server → Services → JCo RFC Provider → Runtime. The Repository section stores access information to the ABAP system. In the example, it is the ABAP system on the same server (localhost) with system number 00, client 800, and user sapjsf.

The SLD server access information is defined in transaction SLDAPICUST (**Figure 29** on page 42). It defines the HTTP access information: hostname, port, user, and password. The Primary option is important. If there are multiple entries listed, only the checked entry is active.

Recommendations for accessing SLD data from ABAP-based systems

Keep the following in mind when configuring the mechanism for accessing SLD data from ABAP-based systems:

- The mechanism for accessing SLD data from an ABAP program is a potential failure point for your configuration. A good understanding of this mechanism can help you quickly troubleshoot problems.



Alias	Prim.	Host Name	Port	User	Password
RKE	✓	iwdf5090	50000	J2EE_ADMIN	

Figure 29 Access information in transaction SLDAPICUST

Running the SLD check program

A program called SLDCHECK is available for performing a quick check of the communication between an ABAP client program and the SLD server. Let's say you are running this program on an SAP XI system. It runs through a series of checks, including:

- Testing the RFC connection to the SLD Java client
- Launching the SLD home page in a separate browser window
- Retrieving data from the SLD server (e.g., a list of business systems, and a particular system's business system registration in the SLD)
- Checking access to the Exchange Profile

At the end of the SLD configuration, a successful run of the program SLDCHECK is a good indication that a client program is able to access SLD data as intended. If there are any errors, fix them right away. It is much more difficult to identify SLD data access as the cause of system access problems later on.

Keep in mind that there are two places for storing user information, one for the SLD Java API (transaction SLDAPICUST) and one for the ABAP Repository (JCo RFC Provider Repository). Either can block a successful access of SLD data.

- To test SLD data access, run transaction SLDCHECK as your first step. It can expose problems in many cases. See the sidebar above for more on the SLDCHECK program.

Conclusion

With more and more SAP applications relying on SLD data to function, a good understanding of SLD is critical to a smooth-running system landscape. This article has provided an in-depth discussion on the design concept and mechanisms behind many SLD functionalities, so that you are well equipped to apply sound principles to your own SLD configuration. The resource listing available at www.SAPpro.com is a good starting point for familiarizing yourself further with this topic, but nothing equals hands-on experience, and there is no time like the present. With this guide in hand, you can get started exploring the world of SLD on your own with confidence.