CONNX Planning Guide

Table Of Contents

Introduction

The CONNX Enterprise Planning guide is designed for CONNX product administrators. It can help them to develop their CONNX installation and maintenance strategy.

Development of a strategy is a very broad exercise. This document is intended to be relatively comprehensive; it provides guidance that is not included in the CONNX Installation Guide or the CONNX User Guide.

Managing data in the modern enterprise requires an effective combination of processes, people and technology. Interoperability, the ability of diverse systems or products to work together with minimal customer effort, is a critical property of any component in the enterprise. The problem would be simple if everyone used the same limited tool set under strict control but this rarely happens.

Legacy, mainframe, and non-relational databases continue to provide mission-critical functionality to organizations competing in the twenty-first century's fast-paced, international marketplace. In today's dynamic global business climate, leading-edge organizations require access to all their business information - no matter where that information resides. And if an enterprise relies on previous-era components, this adds to data management complexity.

CONNX provides a standardized interface to business data regardless of changes in the underlying database or structures. CONNX offers .NET or JDBC functionality to customer sites using both relational and non-relational data stores as if data access was being performed on a single relational database. CONNX enables you to present all your data to familiar standards-based tools on any platform. This flexibility gives you the freedom to choose the business intelligence and development solutions that are best for your business needs. CONNX can handle large amounts of data in many formats. It is easily adaptable to all environments thereby enabling organizations to scale from project to enterprise; from two-tiered to n-tiered; and from one to many data sources.

CONNX Architecture

<u>**View from 10,000 Feet**</u>

CONNX has positioned itself in the marketplace as "Simplified Data Access" for over 20 years But what does "Simplified Data Access" really mean?

- Real-time, read/write, SQL access to non-relational and relational data sources, using open data access standards (ODBC, OLE DB, JDBC and .NET).
- Connecting to multiple databases, using virtually any ODBC/OLE DB standards-based compliant tool/application.
- Integrating multiple disparate data sources no matter where they reside, giving a single SQL view of the data as if it exists in a single relational database.
- No disruption to existing systems or extensive programming.

This high-level "view" of CONNX helps to explain how it's done.

Here is a selection of relational and legacy ISAM/hierarchical data sources that CONNX supports:

RELATIONAL	DB2, SqlServer, Oracle, MySQL, Sybase, Informix

HIERACHICAL w/ Record Locking	Oracle DBMS

ISAM with Transaction support	ADABAS, RDB
ISAM with Record Locking	VSAM, CISAM
ISAM with File Locking	DataFlex
ISAM with Journaling	RMS

It is difficult to provide end users a consistent data access interface because different data sources have different external capabilities (such as record locking). A significant development effort would be required to extract a single value from both legacy systems data and direct modern relational API. CONNX provides end users a consistent interface to access their data no matter what the source, shielding the user from the complexity, development and quality assurance costs while providing consistent access methods and functionality across all the supported DBMS systems.
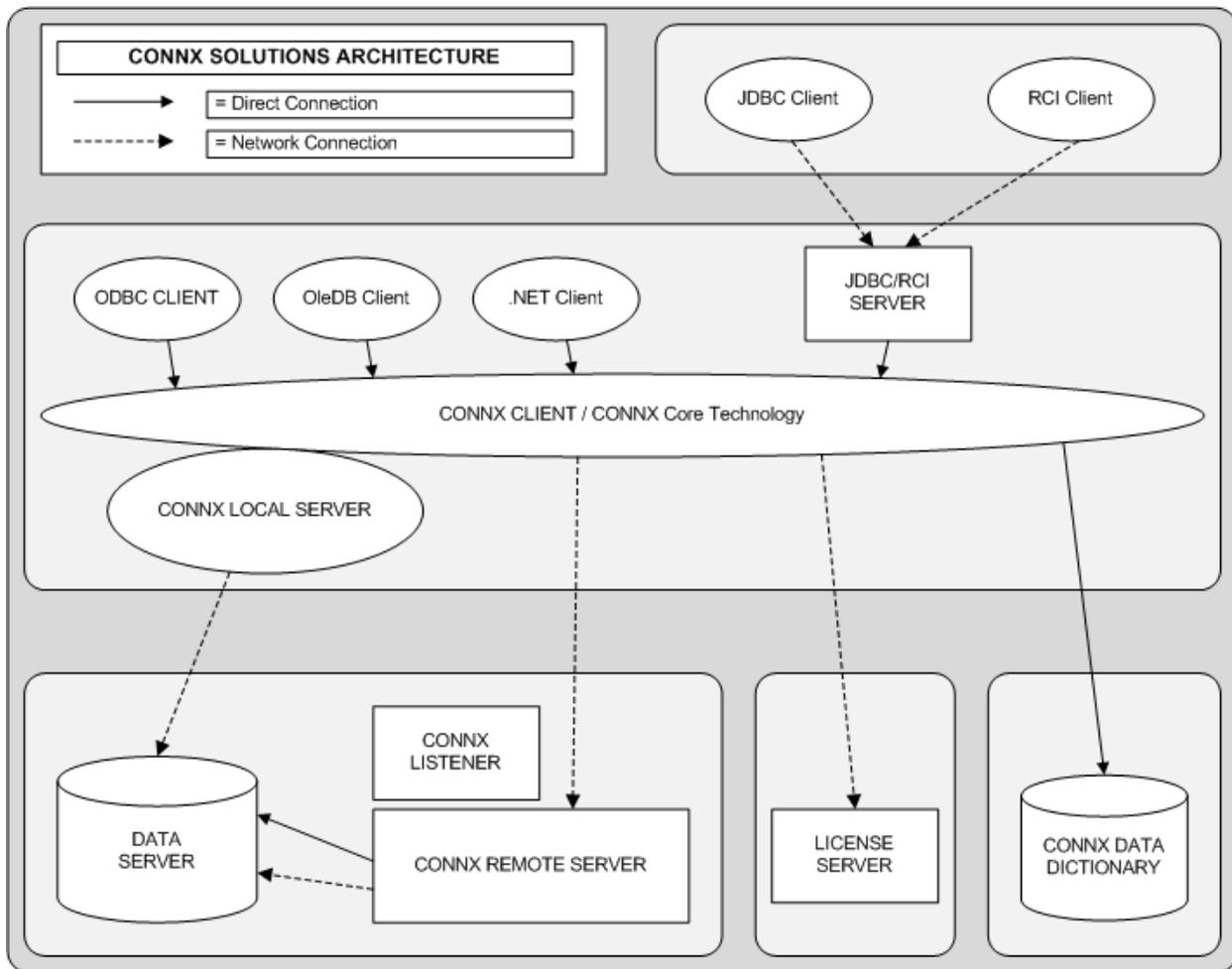
Figure 1. Organizational chart of CONNX Client and Server Components

As Figure 1 shows, CONNX uses a client/server based networked architecture. The CONNX client can be installed on every desktop in the organization or only on a select number of machines that serve data through a web interface or SOA application.

CONNX supports a variety of enterprise organizational models.

If all components reside on the same machine, it may be possible to use either a special in-memory pipe for faster communication, or the operating system loop-back device for ultra-fast communications.

CONNX supports the following access methods:
- ODBC
- OLEDB
- .NET
- JDBC
- RCI/ Embedded (currently only available to ADABAS users)

All the access methods will route through the client and then to a server which has the most direct contact with any back-end database.

A CONNX server may either be a process separate from the CONNX Client or it may be a DLL/library extension to the CONNX Client. Usually non-relational ISAM databases are implemented in a separate CONNX Remote server process and relational databases are implemented as DLL /library extensions to the CONNX Client. Any supported database with an API that services a network protocol will be implemented as a DLL/library extension to the CONNX Client. The workload is about the same whether using a DLL/library extension or a remote server .

The CONNX client and server perform the following distinct functions:

CONNX CLIENT	Data Conversion
	SQL to native translation
	Metadata lookups (requires CDD access)
	Query Plan creation
	Data Joining
CONNX SERVER	Physical Data Access
	Indexed data filtering
	Non-indexed data filtering
	Distributed Data Joining

The DLL/library extension workload is determined by the capabilities of the back-end database:

Relational data source	Filtering and physical data access is off-loaded to the relational database rather than being done by the CONNX client
CONNX Remote server process (usually a legacy ISAM database)	CONNX server is responsible for all of the work listed above in the CONNX server column.

The CONNX Remote server contains a CONNX Listener component, a separate process that spawns the correct type of independent server processes to service user requests. The CONNX Listener requires a minimal amount of system resources.

The JDBC/RCI server can be considered a CONNX client with a listener attached, with the additional functionality and protocol support needed by JDBC and RCI applications. The JDBC/RCI server supports multiple threads of execution and directly links to the end user's CONNX application if they share the same machine.

The CONNX License server service is a networked application that combines the elements of both a listener and a server. The CONNX License server requires a minimal amount of system resources.

The CONNX Client and CONNX License server service are available on Windows, Linux, Linux Z-Frame, AIX, Solaris, HPUX-RISC, and HPUX-Itanium. Because the CONNX CDD manager and license configuration utilities are only available on Windows, CONNX requires at least one Windows machine to create and manage CDDs and to install and configure CONNX licenses. Windows and UNIX have separate CONNX client configurations and configuration utilities.

A Typical Architecture Example

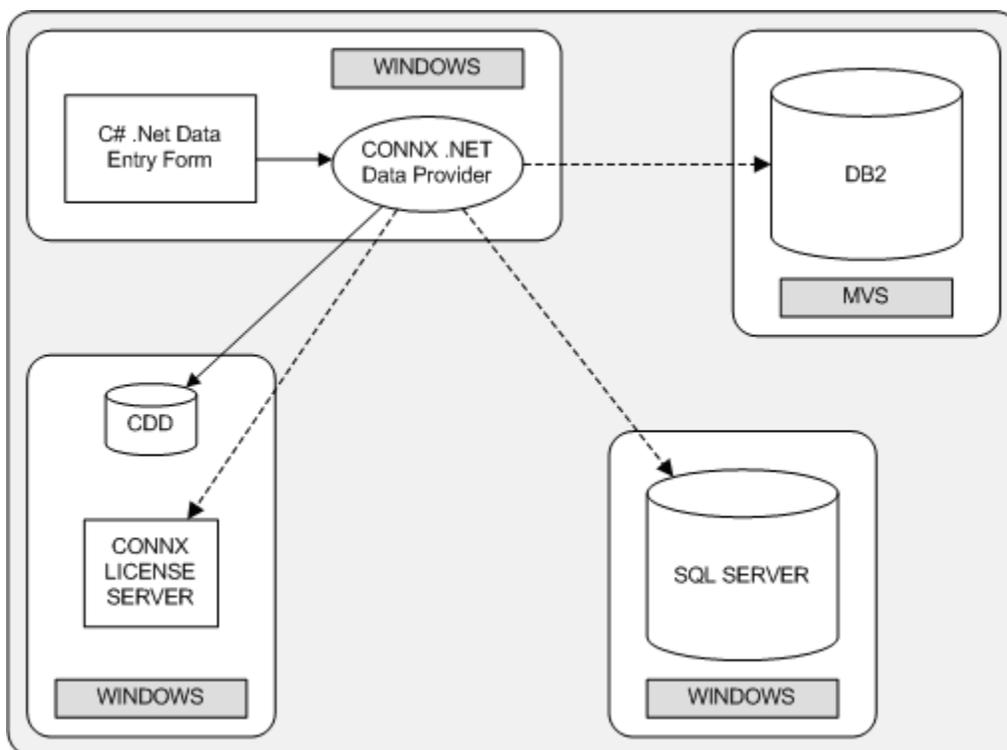Figure 2 outlines a typical (moderately complex) customer site.

Figure 2. C# DataEntry form on Windows using the CONNX .NET Data Provider connecting to SQLServer on Windows and DB2 on MVS.

This typical scenario has four machines:

- The user application (C# DataEntry form) resides on Windows. The CONNX Client is installed on the user application machine.

- The DB2 database resides on a MVS machine

- The SQL Server database resides on a Windows machine

- The CONNX CDD and License server service reside on a Windows machine. This machine has a shared folder (with read/write access permissions) for CDD access.

In this scenario, the customer selected a machine to run the CONNX License server with a guaranteed uptime level. Due to the CONNX networked architecture, there are no CONNX specific components required on either database machine.

This diagram does not show the development configuration and any test and production environments. In those environments, the CDD may not reside on the same machine as the license server; if it did, each deployment phase might have its own directory.

This example illustrates some of the typical architectural and management challenges in a real-world scenario. This gives us a view of the architecture that we can refer to in the remaining sections as we

begin to unravel the complexity of Enterprise Planning. A comprehensive examination of typical deployment scenarios are outlined in the section entitled **CONNX Scenarios**.

CONNX Planning

General Planning Concepts

A successful configuration plan contains reasonable platform, network and software component forecasts for the production environment.

The CONNX Planning section provides the criteria for reviewing project scope, making initial resource and cost estimates and defining the hardware and software components needed to support a CONNX product install. This section also defines criteria for preparing the development and test environments and deploying the CONNX product family to production.

Component Configuration

When planning the CONNX install service configuration, here are some considerations:

1. The CONNX Client must be installed on every machine that accesses data. This may mean many CONNX Client installs for a desktop application or only a few installs for organizations using distributed or web service framework. Each configuration has its particular installation, administration and maintenance constraints.

2. The CDD repository location is very important. The CDDs must be available to every client machine; you may need to open a share on a file server and set appropriate access permissions. You may need multiple CDD copies (the development, test and deploy cycles may each require a separate copy); this can increase administration complexity.

3. A CONNX Remote server must be installed on remote hosts when a local server DLL cannot access data. You may have to open up a range of firewall ports and coordinate across departments to gain access to a particular machine.

4. The CONNX License server must be installed on a single server that has access to the LAN. Choose a dedicated server to ensure uninterrupted access to CONNX features.

The following discussion will be broken down into the constituent aspects of a typical CONNX installation. Please refer to Figure 1 (Organizational chart of CONNX Client and Server Components) throughout the discussion to help facilitate a clearer understanding.

Estimating Throughput Requirements

For a graphic representation of the CONNX architecture, see Organizational chart of CONNX Client and Server Components.

In order to estimate CONNX throughput, you must know how much bandwidth is available and how much bandwidth is being used.

CONNX is designed to reduce the amount of data traveling across the network. Most database servers have excellent data-bus bandwidth to move data quickly from the disk into resident memory and CONNX takes advantage of this speed. CONNX is able to tune queries which can result in greater speed.

After you determine the system requirements for your federated server environment, to plan the throughput requirements requires a plan for each of the components (including the CONNX client machines, each of the database and remote servers, and the CDD). This can be challenging because you must consider:

- the possible configurations' complexity
- the autonomous nature of the different remote data sources
- how great the throughput workload will be. This is even more difficult if you do not have CONNX currently installed.

No matter how large the available bandwidth, data can only flow as fast as its slowest component.

To estimate bandwidth usage, calculate the number of bytes per query and the processing power per query.

- Bytes per query indicates how much data is flowing across the network and how much temporary storage is required to hold a result set while it is being processed or consumed on the client or the server. Data can be stored either in resident memory (which is orders of a magnitude faster than disk access) or in temporary disk files. The greater the amount of data that must be stored in temporary disk files, the greater the risk of disk utilization. And as the amount of data that flows over the network increases, the amount of time required to return the result set to the client increases as well.

  Bytes per query can be calculated with some accuracy depending on the CONNX application environment. The following are responsible for most of the bandwidth usage:

  - Number of columns,
  - summed bytes per column,
  - number and cardinality of tables involved, and (ultimately)
  - number of rows returned, contribute to bandwidth usage.

  In an ad-hoc query environment, hard estimates are unlikely as opposed to estimates in a tightly controlled production application environment where defined throughput statistics are possible.

- Processing power per query indicates how computationally expensive a particular result set is to produce. To measure processing power per query, use the query plan; reading the query plan and tuning queries for performance is beyond the scope of this document.

Important: Query tuning is crucial to reducing processing power use. CONNX will perform exactly as your queries ask it to perform; if queries are not tuned for performance, processing power will suffer.

General Guidelines for Estimating Throughput Requirements

1. Timing information is critical to any estimation of throughput. If you cannot gain access to low level code to isolate the bottleneck, make an adjustment to the query parameters and get a benchmark estimate by contrast. When you add timing information to an application, be careful as it can interfere with query execution.

2. Monitoring a variety of targeted queries with some degree of control over byte width and query plan will provide a much clearer view than merely selecting all of the columns and all of the records from a single table.

3. A query that can be held entirely in resident memory will be faster than one that has to use temporary disk storage. Unless your environment has the appropriate amount of production hardware, hundreds of concurrent users will dramatically increase disk utilization. Reduce the number of columns and number of result set rows wherever possible in a production environment.

4. Select the appropriate reporting intervals for effective throughput estimation. Effective estimation means filtering out the extreme utilizations and focusing on the median or second highest utilizations, depending on the degree of risk tolerated in sizing estimates. Estimation accuracy depends on gathering accurate metrics over a representative (large enough sample) interval, and being able to project a future workload that includes the number of concurrent users, the frequency of existing queries, and the profile and frequency of new queries.

5. Primary decisions about which records match the criteria of a SQL query request are made on the server. The order of tables processed in a join is decided on the client as are other activities like data conversion and sorting. These CONNX core design elements can influence the choices made in allocating hardware or in placement of components.

Estimating Component Requirements

For a graphic representation of the CONNX architecture, see [Organizational chart of CONNX Client and Server Components](#).

Install the CONNX client component on any machine that will retrieve data.

For standalone desktop applications install the CONNX client on any machine where the desktop application is installed.

For distributed architecture applications (multiple machines retrieve data to be viewed on a user desktop) such as Web applications, there are two common scenarios:

- Each Web server running server side scripts that access data would require its own copy of the CONNX Client.
- The Web server is connected to an SOA service that runs on a separate set of machines; each of the separate set of machines would require its own copy of the CONNX Client.

Each machine running a standalone application requires its own copy of the CONNX Client.

When there is a distributed or web application, choose the location of the CONNX Client based on performance requirements. Even though the CONNX core offloads much of the work to the database or remote server, the entire returned data set may have to reside on a client machine. If there are hundreds of users connecting to a web server simultaneously accessing data from that same machine through CONNX then hundreds of data sets may be called into memory on the same machine. This may also include temporary tables required for multi-table joins and sorting.

Estimating Server Requirements

For a graphic representation of the CONNX architecture, see Organizational chart of CONNX Client and Server Components.

Because the CONNX Listener automatically spawns CONNX Remote servers to handle incoming requests, you may not need multiple listeners per machine.

The CONNX Remote server is normally located on the database machine because CONNX Remote servers are tightly coupled with the database. or in the case of some ISAM storage methods, the file system. ADABAS is the only exception; if you have installed Entire-Network, the database and the CONNX Remote server may physically reside on different machines.

Estimating License Requirements

For a graphic representation of the CONNX architecture, see <u>Organizational chart of CONNX Client and Server Components</u>.

License usage for the CONNX product family is based on database access. <u>Each</u> database defined in a CDD represents a separate connection (no matter what database type) and requires <u>one license seat per back-end type</u> (even if the database definition is actually a clone of an existing database) when a connection has been made to the CDD. Connection count may be limited to a subset of defined databases within the CDD by using the **application** field on the CONNX Integrated Logon screen or during the CONNX Data Source setup. More information on the use of the **application** feature can be found in the CONNX User Guide.

For web server or distributed access you can purchase an NTIER license for each server; this allows unlimited connections to that server. NTIER licensing is appropriate for web based enterprise applications or applications distributed in a large-scale SOA project.

You may wish to consider purchasing redundant or fail-over licenses so that you have a primary and a back-up license server; this can help ensure uninterrupted access to CONNX features. It can also make sense to purchase separate licenses for your development and testing systems as well as your production system.

Planning Component Installation

For a graphic representation of the CONNX architecture, see <u>Organizational chart of CONNX Client and Server Components</u>.

As you install the CONNX component installation across the enterprise, you may have to coordinate with other departments.

Server or mainframe administrators may have to install the CONNX Remote server component; network administrators may coordinate the CONNX Client rollout. Depending on your organization's environment, you may need to plan for a phased CONNX component rollout - first to a test environment and then from test to production.

The CONNX API is used to communicate between the CONNX Client and the CONNX Remote server; it determines the compatibility between client and server components. Compatibility breaks occur when an incompatible version is installed.

- The CONNX Client is backwards compatible
- The CONNX Remote server is not forward compatible.
- For JDBC users, the client jar is not forward compatible but the JDBC/RCI server is backwards compatible.

For the best performance and feature quality, the CONNX Client and the CONNX Remote server should be on the same version level, although this is not a system requirement.

Planning the SQL Catalog (CONNX CDD)

In CONNX, the SQL Catalog is called the CONNX CDD.

The CONNX CDD is the cornerstone of the CONNX product family. It contains the metadata that describe all of the organization's enterprise data.

Your installation can have multiple CDDs. You might create one CDD for the development environment databases, a second CDD that is an exact clone of the development CDD, and a third CDD containing an entirely different set of databases for your production environment. An organization usually has a collection of ad-hoc CDDs it uses for off-line development and testing along with a master CDD it uses in production.

The CONNX Client determines the current view of the CDD. The current CONNX Client connection contains the current view of the CDD. If the CDD is modified while this connection is running, the CONNX Client will not recognize the changes until a new connection is spawned.

Keep the production and development CDDs that you consider project resources in a controlled access environment under a revision control system such as a source repository.

The CDD is the CONNX sole point of entry to data. Only the CONNX CDD Manager can create the initial database connection required for data access. Determining where to install the CDD Manager is as important as deciding where to keep the CDDs.

When you enable connection pooling (instead of reusing an existing connection), CONNX creates a fresh connection for any altered CDD so only the current CDD version is the metadata repository connection.

CONNX CDDs may also be linked together in a parent/child relationship.

Scalability Concerns

Scalability is an integral part of the CONNX design and architecture. Not only can CONNX continue to function effectively and efficiently when the environment is resized, but CONNX can take full advantage of the added resources.

The heart of CONNX is a distributed SQL engine. The CONNX SQL engine takes advantage of both the client and the server resources, and is designed to reduce the amount of data traveling across the network.

With its distributed architecture, CONNX can take advantage of available client CPU cycles, freeing server resources to handle other tasks. CONNX's collaborative design unites the best client and server features and increases overall query performance.

CONNX Administration

General Administration Concepts

This section discusses CONNX in the classic development/test/deploy/administrate cycle.

Promoting an application from a development environment to an enterprise production environment requires:

- planning
- preparation
- coordination
- testing

In this multi-step process, the application:

- is deployed into one or more intermediate environments where it is integrated with other applications
- undergoes rigorous testing
- runs in configurations that are scaled up or in which application resources are increasingly distributed.

CONNX provides world class technical support to help you with your enterprise-wide project plans. Contact your sales representative for more information.

Development and Testing

To ensure a smoother transition to your production environment, first integrate CONNX into your development and testing environments.

Having CONNX integrated into your separate development and testing environments will help your enterprise. In most cases, you can identify problems and needed feature requests without impacting your production environment. If you believe there is a problem in the product or in the application and you need technical support's help, it is easier to isolate the problem to a simple reproducible example in a non-production environment.

CONNX uses standard APIs which minimizes the CONNX learning curve. CONNX assumes that you are familiar with ODBC, JDBC, OleDB and .NET. You will find the ODBC, JDBC, OleDB and .NET documentation for your environment extremely helpful if you run into problems.

The development and testing environments are usually isolated from the production environment, and often from each other. Each isolated environment requires its own CDDs and its own set of CONNX licenses.

CONNX uses the CDD as a metadata repository to make different data sources appear as one. Treat the CDD like code and add it to a source repository and back it up on a regular basis. If CDD changes break an application you may need to roll back to a previous CDD version.

Deployment

When you've finished testing, you can deploy the final application to the production server. This may require coordinating and scheduling time with several departments.

Whenever you deploy to a new machine on a separate or different environment, be careful of the operating system differences that can skew results. Here are some examples of potential problems:

- If the testing UNIX and the production UNIX systems have different locale settings, the application may not perform at all, or it may produce different results. Because of environment differences, the same system library search path settings on one UNIX machine will produce correct results but on another machine will produce the wrong results.

- Different Windows systems can have different directory permissions.

- Different mainframe environments may require technical resources or administrative scheduling to install the server in the correct data set or library.

- VMS test area installations could require the individual install option; whoever install this must understand DCL .

CDD Management

General CDD Management Concepts

The CDD is a snapshot of the metadata at import time. Over time, the CONNX tables structures will change. This section describes several strategies you can use to manage table structure changes in a controlled fashion.

If your organization has discrete development, test, and production environments, maintain three separate data dictionaries for each environment. This isolates each environment from each other; you can test any changes in one environment without affecting the others. When it comes time to propagate changes from development CDD to test CDD, or from a test CDD to a production CDD, there are several techniques.

- If there are no changes to the metadata (exactly the same between the environments), the CDD file can be copied from one name to another.

  Note: This usually occurs when the database or server name changes, but nothing else has changed

  Use the CDD manager to change the name of the database and/or server by selected the red database cylinder and making the appropriate changes. The exact metadata tested in one environment will be used in the next environment.

  This option will require some downtime, as any connections to the CDD will need to be closed before a new copy can replace the old. Shut down the JDBC Server and any client connections to the data dictionary. This is typically done during a nighttime batch window or a planned weekend outage.

- If the metadata between the two environments are different enough that a CDD file copy will not work, use the data dictionary manager to re-import the changed tables.

  This will require some downtime, as any connections to the CDD (such as the JDBC Server or any client connections to the data dictionary) will need to be closed before a new copy can replace the old version. This is typically done during a nighttime batch window or a planned weekend outage.

- For some data adaptors, such as ADABAS, the metadata can be changed by using DDL SQL statements, such as **Create Table Description** or **Create Table Cluster Description**. Using these DDL statements, old metadata definitions can be removed, and new definitions created.   Since the DDL statements can be issued using a variety of interfaces to CONNX (JDBC, ODBC, .NET, OLEDB), there is no downtime required for this type of metadata update. The CDDs can be updated while online and in use, simply by using the required SQL statements.

Controlling Access

Some care should be exercised with the modification of a production CDD. To prevent unwanted modification of the CDD contents via the CONNX CDD manager, use the security measures outlined in the CONNX User Guide and the CONNX Installation Guide .

Controlling physical access to a CDD may be a concern. A CDD must be made available to any CONNX client that will be accessing data. The CDD is actually a database, and read/write access is required to the directory that contains the CDD, and the CDD file itself.

The following is a list of major security considerations:

1. Access to the Data Dictionary Manager
   The Data Dictionary Manager can only be installed when someone is in possession of a CONNX License file. Without access to a CONNX license file, the installation routine will not make the CONNX Data dictionary manager available. Minimize the number of individuals who have access to the CONNX data dictionary manager by physically securing the license files provided.

2. Securing created Data Dictionary files with file server security mechanism.
   Typical CDD files reside on a central network share so all users can access the same CDD. Secure this file share so that only authorized users have access to the CDDs.

3. Securing Data Dictionary Files via internal password
   Data Dictionary Files are encrypted for security reasons. You can also password protected Data Dictionary Files. If you set a password on the data dictionary, do not lose it as you will not be able to modify the CDD without it.

The tables, views and other SQL entities within the CDD can be secured a variety of ways. Please refer to the CONNX User Guide for more details.

Updating Indexes

In many cases the structure of the columns in the table remains intact, but indexes are changed, added or removed.

Index metadata can be refreshed on any table by using the **Refresh index** button in the data dictionary manager, without performing a complete re-import of the object metadata.

Index information for the complete data dictionary can be refreshed by using the **Update Statistics** option in the data dictionary manager. This is equivalent to selecting **Refresh index** on every table in the CDD.

The **Refresh Index** and **Update Statistics** functions are only available via the CDD Manager.

CONNX Performance

General Performance Concepts

CONNX has been developed with performance in mind and strives to maintain the highest level performance in the marketplace.

Getting the highest performance is most important in applications that require fast user response. For any application that relies on multiple machines or resources, system performance is only as good as the slowest component.

Because operating system and networking environments vary, we are unable to provide a performance calculator or hard statistics that would work for all our customers. Developing and monitoring performance for a particular installation is best performed on-site in your specific environment. Customer can modify their current environment so that CONNX can achieve peak performance.

This chapter gives general performance improvement guidelines. Use the remaining sections for high level planning. We will detail some specific concepts and methods that can be employed to increase or maintain CONNX performance.

Query Tuning

SQL query tuning is an essential performance component.

Because query tuning specifics depend on the underlying database structure, this document cannot provide a comprehensive examination of query tuning. Your database documentation should contain specific query tuning information; there are also many third party books on SQL query tuning available for purchase.

Much of the specific database information on SQL query tuning is also applicable to CONNX. For example, the natural relationships between sort order and index selection can help you decide when and how to use an ORDER BY clause. And understanding the relationship between table cardinality, index selection, join order, and usage of various operators is essential to understand SQL query performance.

CONNX provides access to the query plan via the {**statistics**} function. Please refer to the CONNX User Guide for more information.

CPU

Even though multi-processors and memory are relatively inexpensive, CPU speed can still be a problem.

Raw computing power may be a problem depending on the application and expected result set size. Some CONNX operations (like sorting and aggregate operators) may be CPU intensive. Although CONNX core technology uses optimized algorithms, some operators must access each record no matter how large the data set.

Many organizations have multiple intensive applications running concurrently on a single machine. If you reduce the number of intensive applications concurrently running on a single machine (by adding more hardware to the network) processing should speed up.

On average, the CONNX remote servers use less CPU then the client. You can minimize some of the computing cost if you offload processing to the backend database. This is an internal CONNX design feature.

The CONNX Remote servers tend to be installed on legacy systems. There are many specific legacy CONNX hardware configurations such as RMS, DBMS and RDB running on VAX hardware running OpenVMS. Remember that the fastest VAX is much slower than today's average PC.

In some cases it may be possible to cluster machines or to use a Network File System implementation such that the data on a slow machine can be accessed from a much faster machine.

In a later section we will examine local versus remote ISAM access.

Memory

Random Access Memory (RAM), or memory, can cause performance problems if application memory requirements exceed the RAM capacity. This will cause the results to be swapped to disk.

Because the CONNX Client may be required to hold several record sets in resident memory to complete join or sorting operations, it has the largest memory requirement. To minimize memory usage when the result set is large enough, use disk based algorithms that sort in-memory pointers to the data.

The CONNX Listener uses very little memory and should never dominate resident resources.

CONNX Servers will batch record sets and may hold a number of resources they need to maintain the source database in a cursor state.

Input and Output

Disk I/O

Disk access has traditionally always been the bottle neck for database related systems. This includes retrieval and updating of physical data, storing temporary result sets and logging. For more information, see Estimating Throughput Requirements.

Network I/O

Usually local LAN access with Ethernet is not a bottleneck.

The system loopback device is an internal CONNX feature. If you use the system loopback device, it can speed up component communications. If you plan to use the loopback interface, install the components locally.

Depending on your application's scale, you may get the best results if you create a distributed setup where the database resides on a different machine than the CONNX data access components.

Due to system resource thrashing, overall performance may decrease if too many applications are competing for system resources.

For more information, see Estimating Throughput Requirements.

Connection Pooling

To speed up establishing a connection, enable connection pooling. Connection pooling is much faster because many of the internal objects needed to facilitate a connection have been allocated and initialized. The JDBC/RCI server has connection pooling enabled by default.

The CONNX Remote server is spawned for the life of a connection. Normally, when the connection ends, the remote server drops away. For pooled connections, the remote server does not drop away but none of the original SQL processing context remains.

For more information on connection pooling, refer to the CONNX User Guide.

CONNX Scenarios

General Scenario Information

In the following scenarios we will outline some typical customer environments.

These examples show how you can design and administer CONNX using a variety of client, sever and database platforms. These are not meant to be an exhaustive examination of possible architectural configurations.

We are giving you templates for what is possible. There are few limits to the number of ways in which CONNX is employed in the enterprise. You can infer more complex (or less complex) scenarios from these relatively simple examples.

Most examples connect to two data sources, one local server and one remote server to establish a sense for each type of connectivity.

Scenarios use this diagram format

DIAGRAM LEGEND

→ = Direct Connection

------► = Network Connection
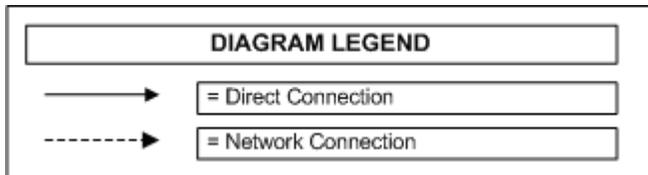
Scenario 1: Local and remote CONNX servers

General Scenario Information contains useful information about the scenarios.

This scenario shows a distributed VB .NET desktop application using OleDB to connect to both Oracle on Windows and to VSAM on Z/OS (CICS.)
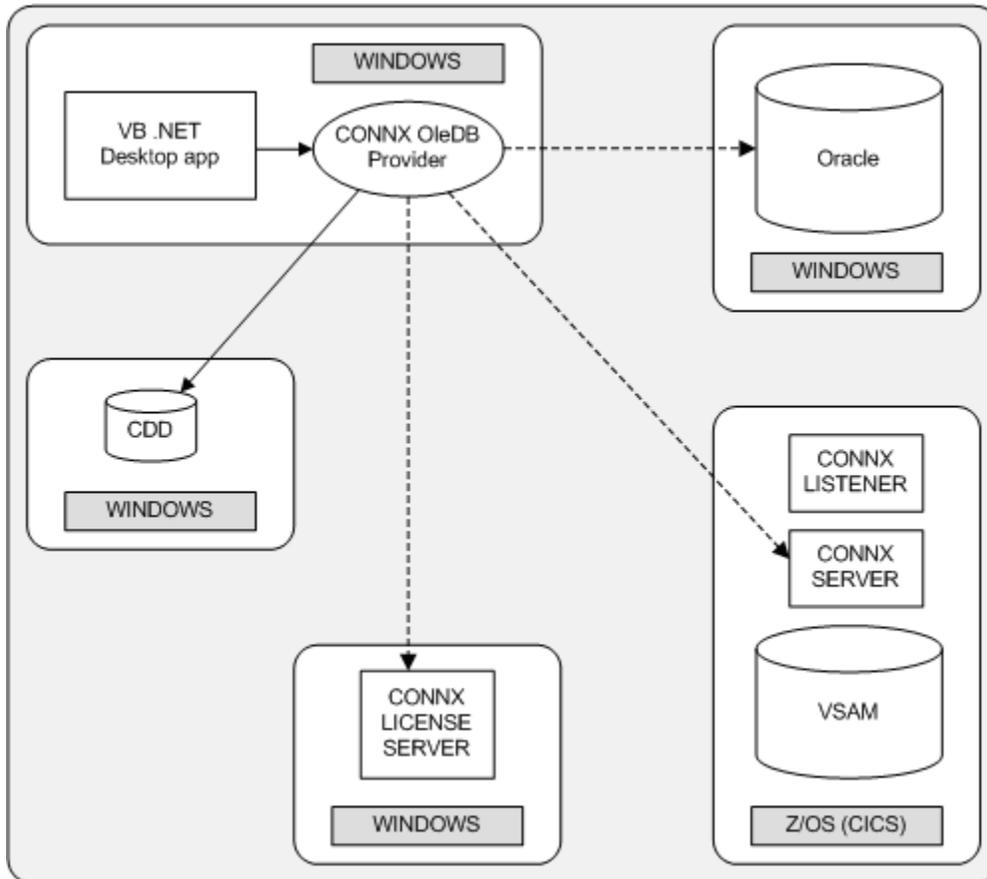
The desktop application uses both local and remote CONNX servers. The CDD is located on a different file server than either the CONNX Client or the CONNX License server.

If your CDD is being shared by multiple .NET applications, pay special attention to security. If the CDD configuration and the file server directory have the wrong security settings, your applications won't have the correct CDD access.

The license server can be installed on its own machine. Doing so could lower the application downtime risk.

The VB .NET distributed desktop application uses OleDB to connect to both Oracle on Windows and VSAM on Z/OS (CICS).

The CDD is on a file server that is accessible to the CONNX Client and the CONNX License server is on a separate machine.

Your organization's CDD may be shared by multiple NET application instances. Providing the necessary security provisions for the CDD configuration and for the directory share on the file server will ensure correct access to the CDD. The license server need not be installed on a separate machine although you may choose to do so to guarantee machine uptime.

Scenario 2: .NET via SQL Server

General Scenario Information contains useful information about the scenarios.

This scenario shows a NET application using the SQL Server Provider connecting to RMS on VMS via a SQLServer linked server and SQLServer on Windows. This .NET application indirectly uses CONNX via SQL Server linked server technology.
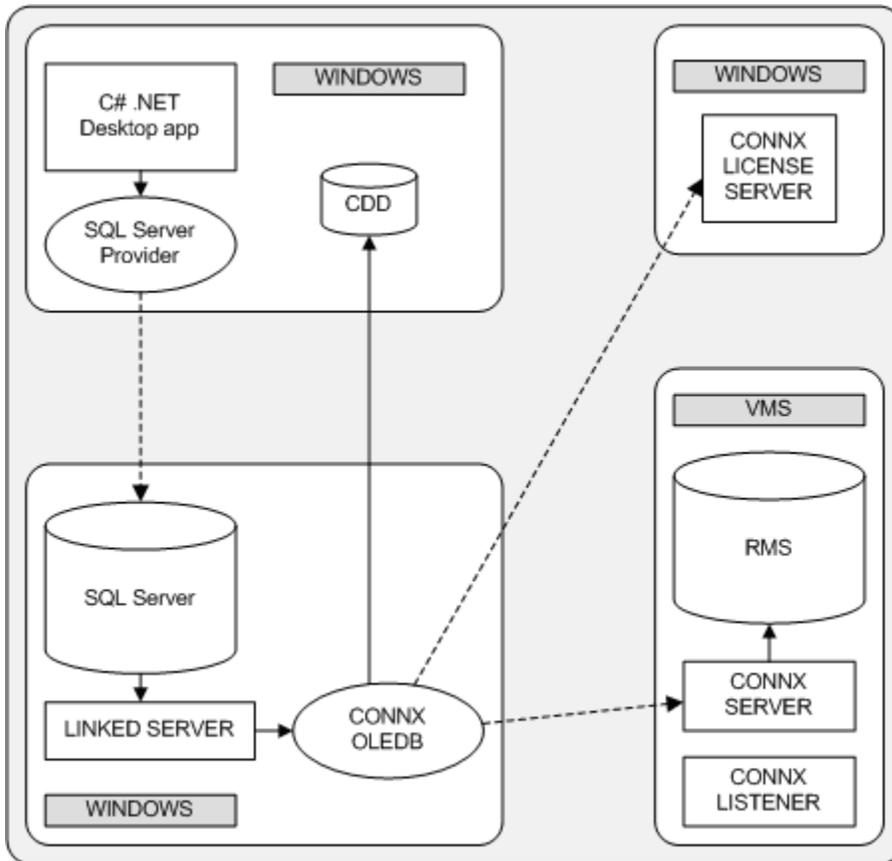
With CONNX, a legacy data source is accessed as if it were just another relational table. The user can create stored procedures and other routines as if the disparate data sets all belonged to the same database. CONNX provides the seamless connectivity and access to the legacy data source.

In this scenario, the CDD is on the user's machine. This is typical in a development environment where CDD changes are stored on the user's local machine and propagated to the server when development is complete. In the production environment, move the CDD closer to the CONNX Client or to a dedicated file server.

Scenario 3: Unix Client

General Scenario Information contains useful information about the scenarios.

This scenario shows a C++ ODBC Application using UNIXODBC on Linux connecting to Oracle on Solaris and RMS on VMS.
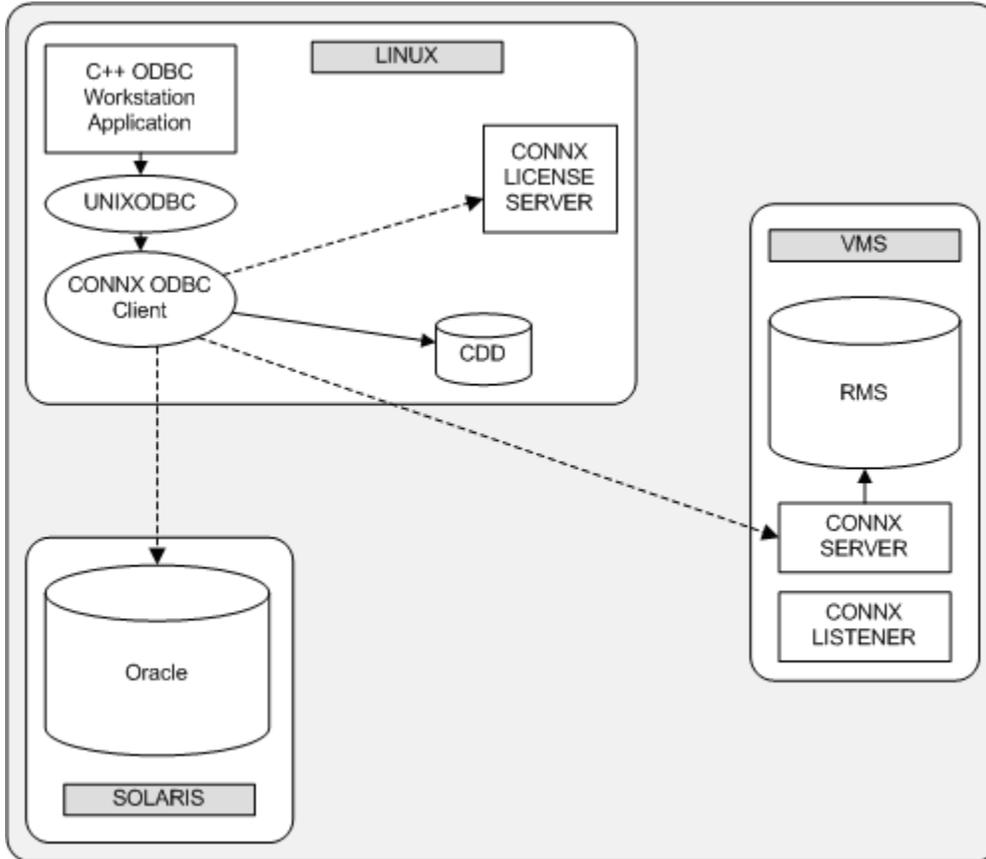
There are no Windows machines in our configuration diagram. When you need to configure the license server or the CDD, there must be at least one Windows machine available. It can be removed when configuration is complete.

UNIXODBC is the driver manager in our scenario. If the driver configuration manager allows it, any available ODBC compliant driver manager should work.

In the example, the license server runs on the user application machine. This is a very common configuration. Only one CONNX License server will be installed on the LAN.

Scenario 4: Oracle and Java

General Scenario Information contains useful information about the scenarios.

This scenario shows a JAVA standalone application on Windows using the Oracle JDBC Client connecting to Oracle on Solaris and accessing VSAM under VSE (CICS) via Oracle Heterogeneous Services.
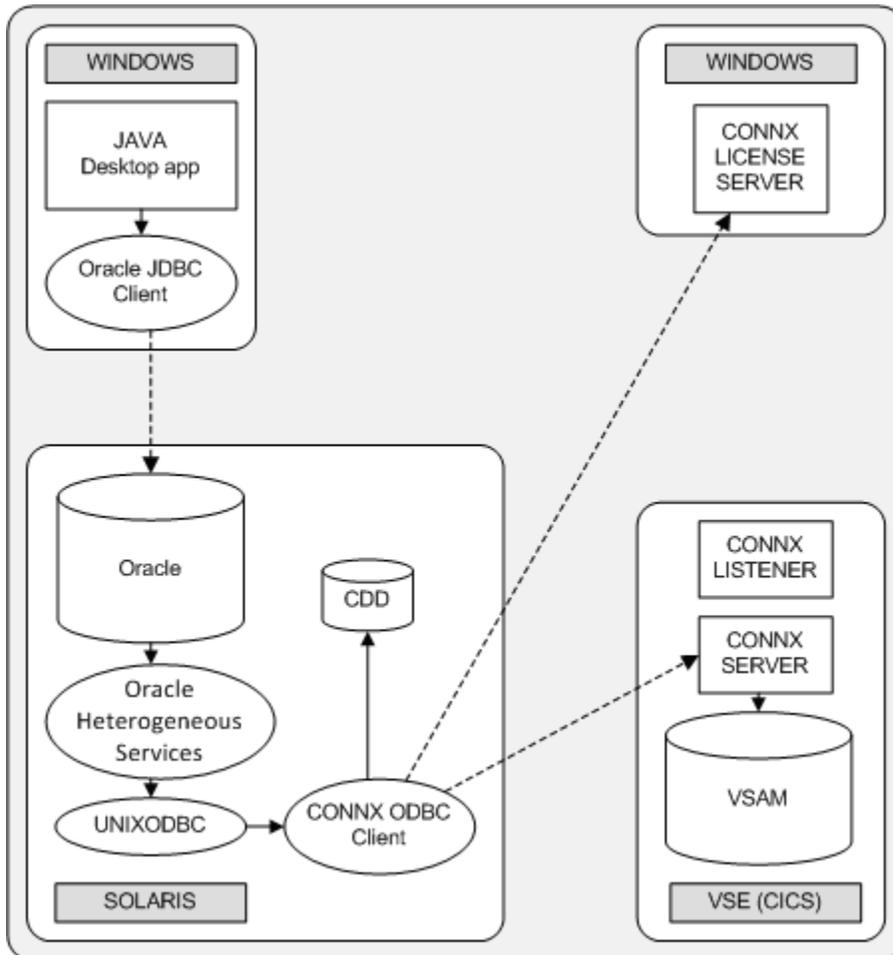
This Oracle Heterogeneous Services example in this scenario is analogous to the SQL Server linked servers example outlined in a previous scenario.

The Java desktop application uses the Oracle JDBC driver to connect to an Oracle database and access both Oracle and VSAM data.

In this scenario, a legacy data source is accessed as if it were just another relational table. The user can create stored procedures and other routines as if the disparate data sets all belonged to the same database. CONNX provides the seamless connectivity and access to the legacy data source.

Store the CDD where the CONNX Core Client technology can access it; in our scenario, the CONNX client and CDD both reside on the Solaris machine.

Scenario 5: ASP .NET

General Scenario Information contains useful information about the scenarios.

This scenario shows an IIS ASP .NET application (webforms) connecting to Oracle on Linux and Microfocus on HPUX.
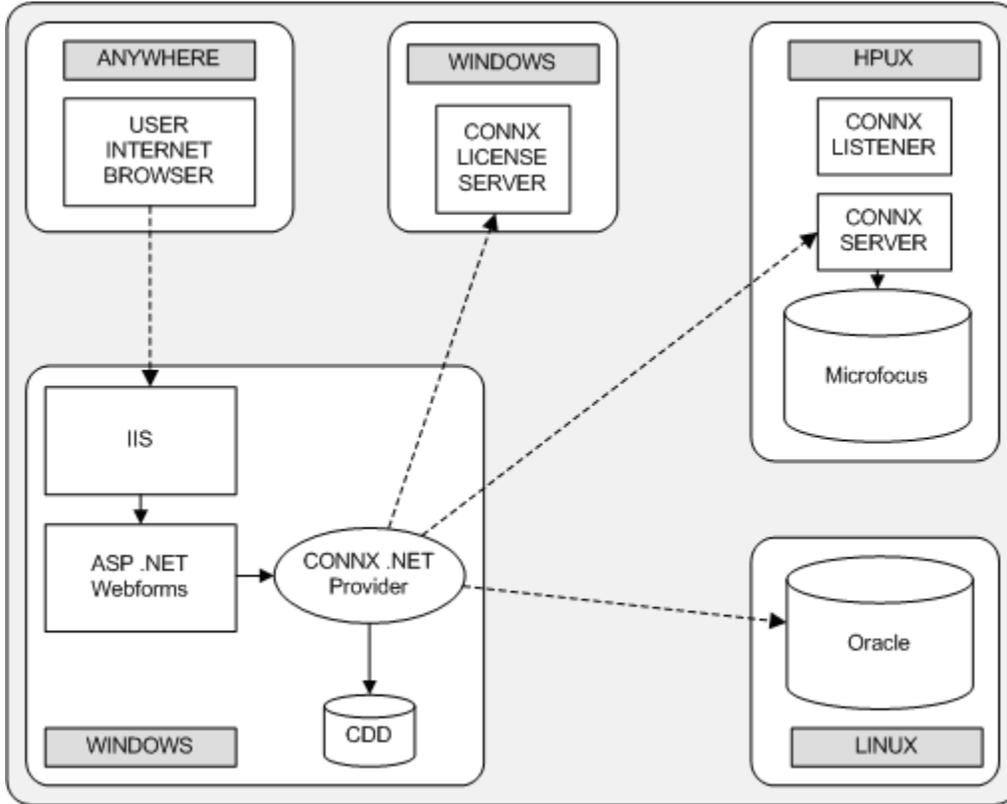
In this scenario, a legacy data source is accessed as if it were just another relational table. The user can create stored procedures and other routines as if the disparate data sets all belonged to the same database. CONNX provides the seamless connectivity and access to the legacy data source.

Scenario 6: NTIER

General Scenario Information contains useful information about the scenarios.

This scenario shows a C++ ODBC Web Service Application using UNIXODBC with Apache on Linux connecting to DB2 on AS400 and VSAM on MVS.
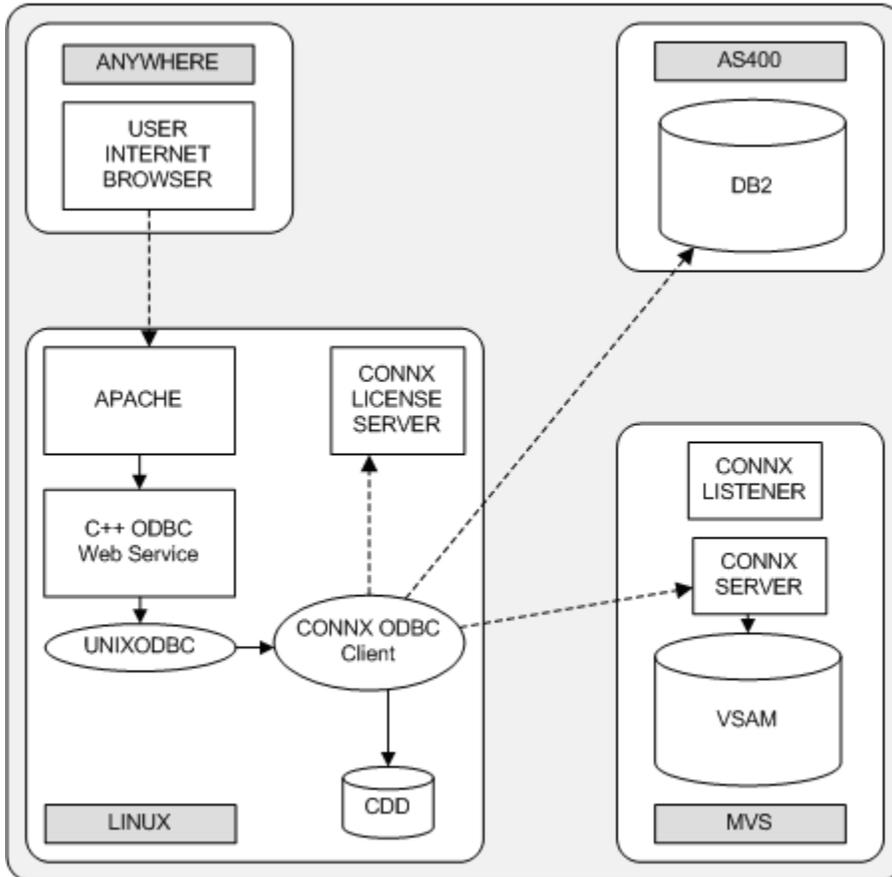
This architectural layout describes a web or NTIER application. The C++ ODBC Web Service uses NTIER because NTIER had been enabled in this machine's sqlregistry (CONNX's configuration repository for UNIX systems).

NTIER gives the end user unlimited connections from a designated client machine. Each NTIER license is single machine specific.

The NTIER license is ideal for web server use because it allows unlimited connections, although only one session can be active at a time. If you want to have multiple active sessions, consider purchasing multiple NTIER licenses.

Scenario 7: JBOSS and Hibernate on Linux

General Scenario Information contains useful information about the scenarios.

This scenario shows a JBOSS application using Hibernate on Linux and the JDBC/RCI server on Linux connecting to ADABAS on Z-Linux and DB2 on MVS.
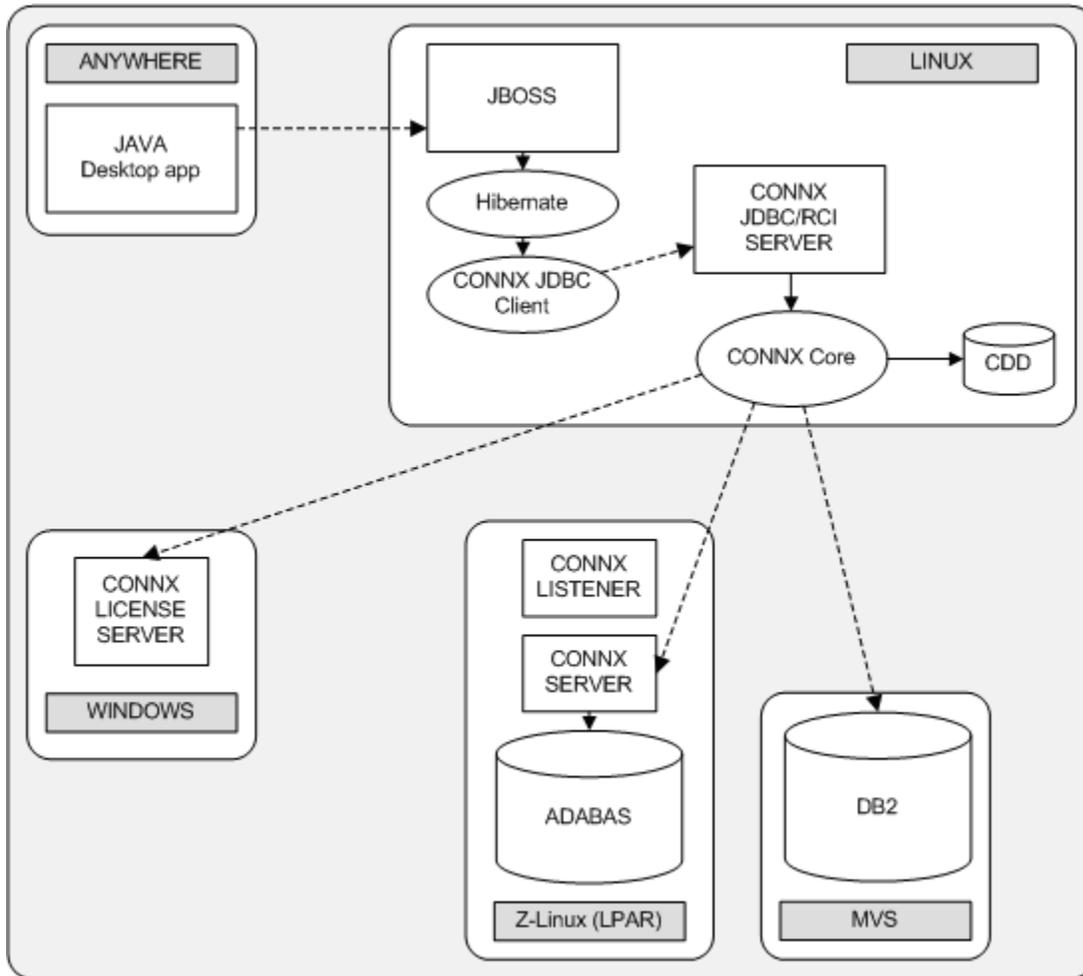
CONNX supports Java connectivity to any CONNX Client platform.

CONNX for JDBC is a Type 3 networked driver. Java access to CONNX is supported from any platform that supports Java version 1.3 and above, including VMS.

Our scenario uses Hibernate within JBOSS, a popular open-source Java Service Container. The CONNX JDBC Client jar and the JDBC/RCI Server are both located on the same machine. This allows for very fast communication between client and the JDBC/RCI middleware via the system network loopback interface.

Scenario 8: JDBC Servlet and Websphere

General Scenario Information contains useful information about the scenarios.

This scenario shows both a JDBC servlet application using Websphere on HPUX and the JDBC/RCI server on Windows connecting to SQLServer on Windows and CISAM on AIX.
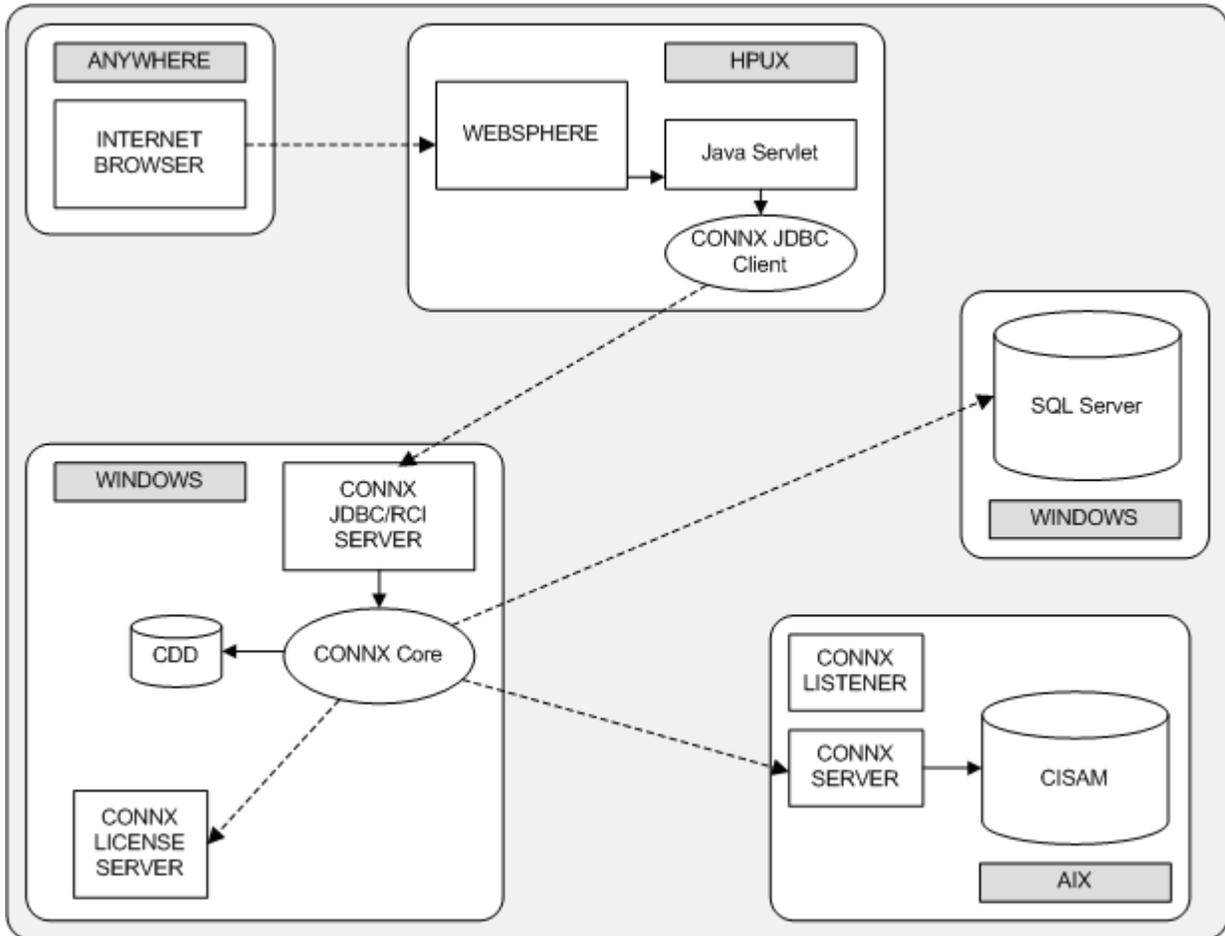
IBM WebSphere is used as a Java Web Container to host a Java Servlet. The CONNX JDBC jar can communicate over the network to a CONNX JDBC/RCI server running on the same computer as a CONNX Client.

Scenario 9: JDBC Web Services and Tomcat

General Scenario Information contains useful information about the scenarios.

This scenario shows both a JDBC web service application using Tomcat on Solaris and the JDBC/RCI server on Solaris connecting to DB2 on AIX and IMS on MVS.
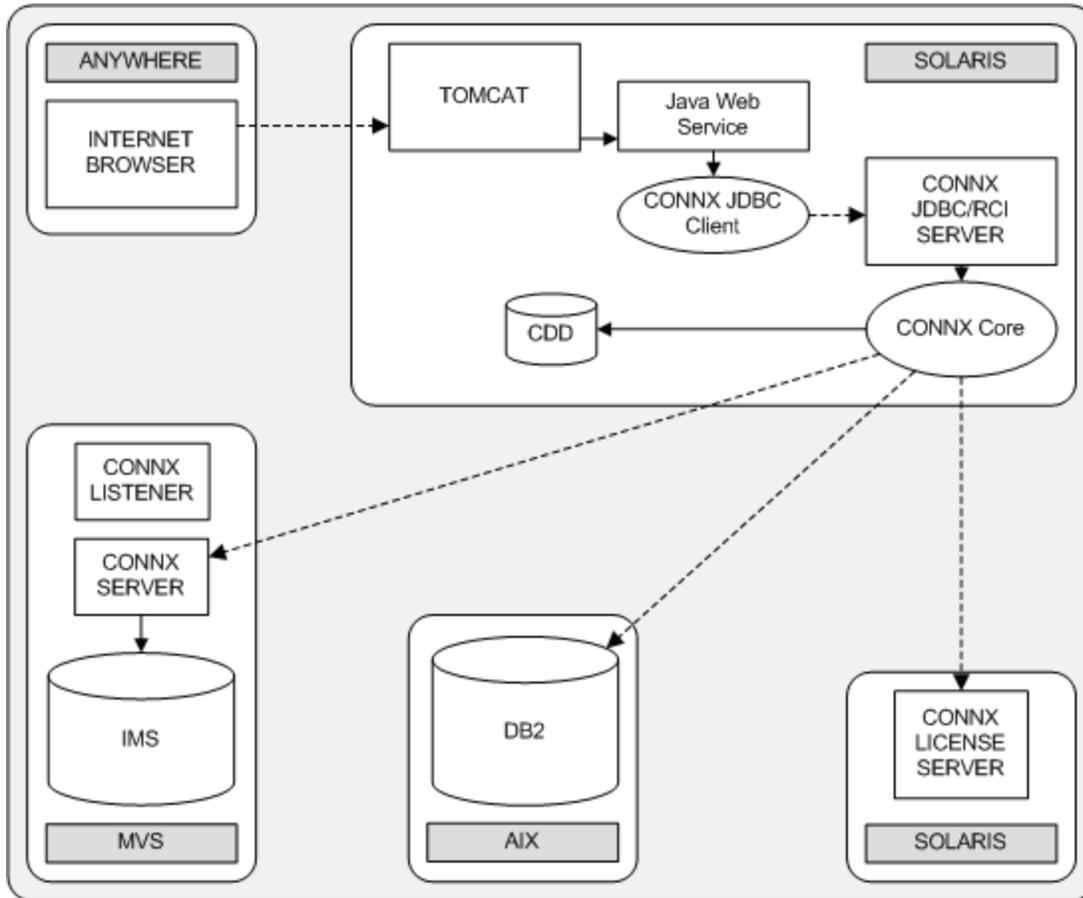
In this example, Tomcat, a popular open-source implementation from Apache, is used as a Web Container.

Scenario 10: RCI Embedded COBOL

General Scenario Information contains useful information about the scenarios.

This scenario shows both a RCI Embedded COBOL application on Z/OS and the JDBC/RCI server on Z-Linux in another LPAR connecting to ADABAS on Z/OS and ADABAS on AIX.
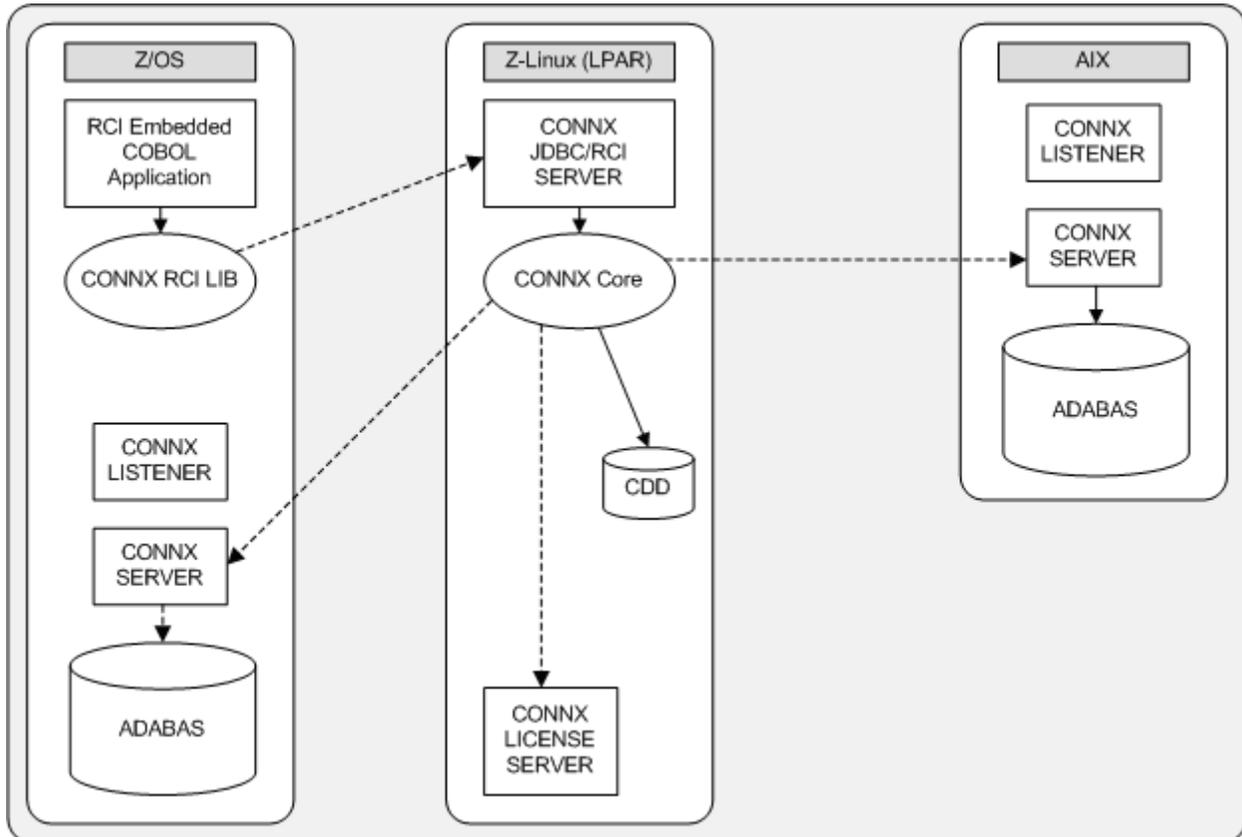
The CONNX RCI for Embedded SQL is only available to CONNX for ADABAS customers.

In this example we are running Z/OS in one VM LPAR and Z-Linux in another VM LPAR so they are considered to be physically located on one machine. We are connecting to ADABAS data that resides on both Z/OS and AIX.

We are also connecting to ADABAS data via a COBOL application that has embedded SQL statements that have been precompiled for the RCI. This technology allows for fast database development utilizing 3GL languages.

Scenario 11: RCI Embedded C

General Scenario Information contains useful information about the scenarios.

This scenario shows both a RCI Embedded C application on Solaris and the JDBC/RCI server on Windows connecting to SQL Server on Windows and ADABAS on Solaris.
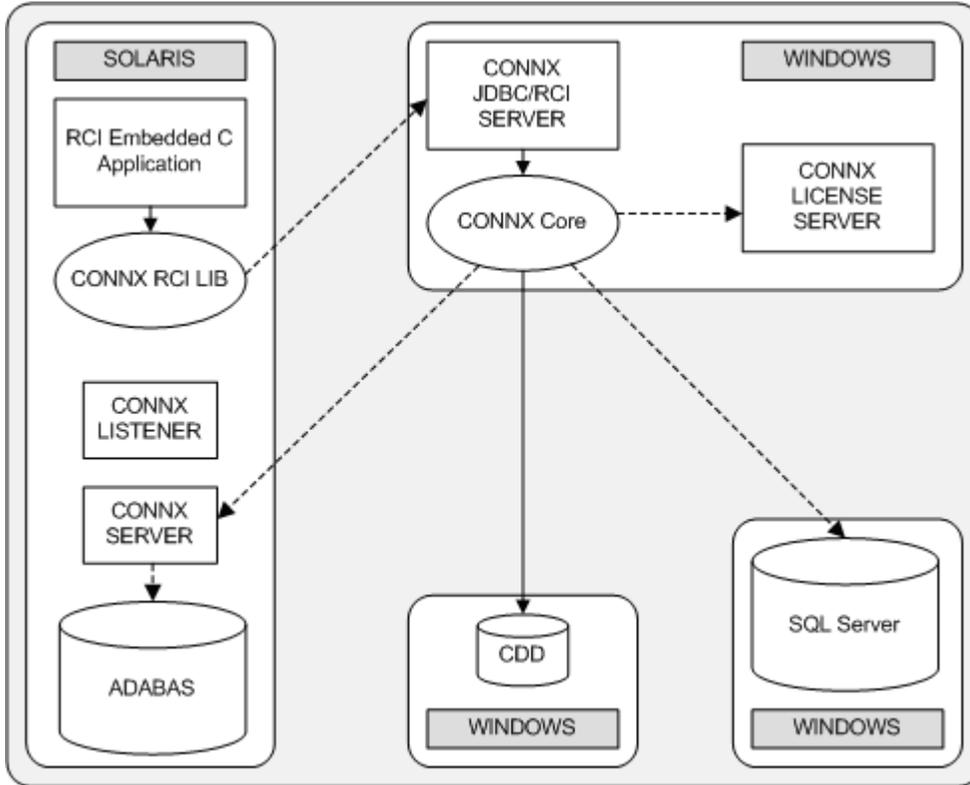
The CONNX RCI for embedded SQL is only available to CONNX for ADABAS customers.

In this example we have a C application that has embedded SQL statements that have been precompiled for the RCI. Our C application can access ADABAS and SQL Server data joined together as if the data was from the same homogenous database.

Appendix A - Technical Support

Technical Support

CONNX products can be installed only as licensed by CONNX Solutions. Software cannot be transferred from one computer to another or to another portion of your network before registration procedures are completed and without the written consent of CONNX Solutions.

This guide assumes that you are familiar with the operating procedures of your computer and with the Microsoft® Windows® environment. For information about your file system and the Microsoft® Windows® environment, refer to the Windows documentation.

If you encounter problems with the installation or operation of CONNX, contact an authorized CONNX reseller or CONNX Technical Support as follows:

Telephone: (888) 930-2727 (For users with a Maintenance Support Contract.)

Telephone: (425) 519-6600 (For international inquiries.)

Fax for all users: (425) 519-6601

E-mail: support@connx.com

If you purchased CONNX through a value-added reseller (VAR), please contact them for support.

CONNX technical support personnel are available Monday through Friday, between 6:00 a.m. and 5:00 p.m., Pacific Time. If you have questions regarding your maintenance support, see your CONNX contract for details.

It is important that you register your copy of CONNX either by completing the registration card supplied with the CONNX CD-ROM or by visiting the CONNX Web site at www.connx.com

Proper registration ensures receipt of Service Pack notices, tips and hints for CONNX operation, related CONNX information, and uninterrupted technical support.

Adabas SQL Gateway users:

Product support and technical assistance for the Adabas SQL Gateway (CONNX for Adabas) are available through your local Software AG Regional Support Center, Software AG's ServLine24, or your Software AG Account Manager.

Access to ServLine24 can be found at the following addresses:

https://empower.softwareag.com/default.asp

http://www.softwareag.com

Adabas SQL Gateway support can also be reached via e-mail at support@softwareag.com.

Trial copies and pre-release versions are covered by separate contracts. Please contact your Software AG Account Manager for more information.

Appendix B - Trademarks and Copyrights

Trademarks and Copyright

Legal Notices

This notice is valid for all products of the Adabas®, Bolero®, EntireX®, Natural® and Tamino® (the "Product") product line and associated Software AG products.

IMPORTANT: PLEASE READ BEFORE INSTALLING THE SOFTWARE

You are not allowed to install or use the Product without a corresponding license agreement.

If you have entered into a license agreement with one of Software AG's subsidiaries or distributors (the "License Agreement") the installation and use of the Product is subject to your acceptance of additional terms which are provided for you in the section Additional Terms.

If you have not entered into a License Agreement you must execute the license agreement with Software AG which is provided for you in the section Software AG License Agreement for Trial Versions of Software AG Software ONLY.

Copyright Notice

Adabas®, Bolero®, EntireX®, Natural® and Tamino® are Products of Software AG, a German corporation with its principal place of business at Uhlandstrasse 12, 64297 Darmstadt.

Software AG is the sole owner of all industrial property rights and copyright to the Products and accompanying user documentation or has the respective distribution rights. References made in or on the Products to the copyright or to other industrial property rights must not be altered, deleted or obliterated in any manner.

© Copyright 1998-2016 CONNX Solutions, Inc. All Rights Reserved.

© Copyright 2017-2019 Software AG, Uhlandstrasse 12, 64297 Darmstadt, Germany 2016. All rights reserved.

Trademark Notice

The name Software AG and/or all Software AG Products are either trademarks or registered trademarks of Software AG. Other product and company names mentioned herein may be the trademarks of their respective owners.

CONNX, FTL, InfoNaut, InfoNaut Professional, and CONNX Data Synchronization are trademarks or registered trademarks of Software AG, in the USA and other countries.

Detailed information on trademarks and patents owned by Software AG and/or its subsidiaries is located at http://softwareag.com/licenses.

This software may include portions of third-party products. For third-party copyright notices, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software AG Products / Copyright and Trademark Notices of Software AG Products". These documents are part of the product documentation, located at http://softwareag.com/licenses and/or in the root installation directory of the licensed product(s).

Use, reproduction, transfer, publication or disclosure is prohibited except as specifically provided for in your License Agreement with Software AG.

The following are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries:Microsoft, Microsoft Access, Microsoft Excel, Microsoft Internet Information Server,

Microsoft .NET, Microsoft SQL Server, Microsoft Visual Basic, Microsoft Visual C++, Microsoft Visual FoxPro, Microsoft Visual Studio, Microsoft Windows.

The following are either registered trademarks or trademarks of their respective companies or organizations in the United States and/or other countries: Adabas - Software AG; Extra!, Attachmate - Attachmate Corporation; Borland C++ - Inprise Corporation; Crystal Reports - Seagate Software; DataFlex - Data Access Corporation; DB2, AIX, AS/400, C-ISAM, OS/2, OS/390, OS/400, IBM Communications Server, IBM Personal Communications, IBM eNetwork Personal Communications, IBM eNetwork Communications Server - International Business Machines Corporation; Dharma ODBC Integrator - Dharma Systems, Inc.; FOCUS -- IBI, Inc.; HP-UX -- Hewlett-Packard Company; Impromptu - Cognos Incorporated; Itanium - Intel Corporation; JetForm - JetForm Corporation; Linux - Linus Torvalds; NetWare for SAA, Novell - Novell, Inc.; Micro Focus -- Micro Focus International Limited; Oracle, Oracle Rdb, Oracle CODASYL DBMS, Oracle Developer, Form Builder, and Report Builder - Oracle Corporation; Paradox - Corel Corporation; Pathworks, Network File Transfer, RMS, DECnet, VMS, OpenVMS, VAXServer, AlphaServer - Compaq Computer Corporation; PowerBuilder - Sybase, Inc.; POISE (People Oriented Information Systems for Education) -- Campus America/Jenzabar; POWER*flex* - POWER*flex* Corporation; RUMBA, Wall Data - NetManage, Inc.; SCO -- Caldera Systems, Inc.; SCT, Plus2000 - Systems and Computer Technology Corporation; SIS - University of California Berkeley; Solaris, Sun - Sun Microsystems, Inc.; Unix - The Open Group.

Portions of the **PostgreSQL Data Base Management System** are copyrighted © 1996-2002, PostgreSQL Global Development Group
Portions Copyright © 1994-7 Regents of the University of California
Permission to use, copy, modify, and distribute this software and its documentation for any purpose, without fee, and without a written agreement is hereby granted, provided that the above copyright notice and this paragraph and the following two paragraphs appear in all copies.
IN NO EVENT SHALL THE UNIVERSITY OF CALIFORNIA BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, INCLUDING LOST PROFITS, ARISING OUT OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE UNIVERSITY OF CALIFORNIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

THE UNIVERSITY OF CALIFORNIA SPECIFICALLY DISCLAIMS ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE PROVIDED HEREUNDER IS ON AN "AS IS" BASIS, AND THE UNIVERSITY OF CALIFORNIA HAS NO OBLIGATIONS TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

SAS Institute Inc.
SAS/C OnlineDocTM, Release 7.00, Cary, NC: SAS Institute Inc., 2001.
SAS/C OnlineDocTM, Release 7.00, April 2001
SAS/C OnlineDocTM, Release 7.50, April 2004
Copyright © 2001 by SAS Institute Inc., Cary, NC, USA.
All rights reserved. Produced in the United States of America.
U.S. Government Restricted Rights Notice
Use, duplication, or disclosure of this software and related documentation by the U.S. government is subject to the Agreement with SAS Institute and the restrictions set forth in FAR 52.227-19, Commercial Computer Software - Restricted Rights (June 1987).
SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.
April 2001
SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

dbCrypto
 Copyright (c) 2002, Dr Brian Gladman, Worcester, UK.
 All rights reserved.

SIS

ICU Library

FOR ANY CLAIM, OR ANY SPECIAL INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION

WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

Hiredis

Copyright (c) 2009-2011, Salvatore Sanfilippo <antirez at gmail dot com>

Copyright (c) 2010-2011, Pieter Noordhuis <pcnoordhuis at gmail dot com>

All rights reserved.

Redistribution and use in source and binary forms, with or without

modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice,

  this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice,

  this list of conditions and the following disclaimer in the documentation

  and/or other materials provided with the distribution.

* Neither the name of Redis nor the names of its contributors may be used

  to endorse or promote products derived from this software without specific

  prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND

ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED

WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE

DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR

ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES

(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;

LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON

ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT

(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS

SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

OCILib

Licensed under Apache 2.0

SAFMQ

  Copyright 2004-2007 Matthew J. Battey, Omaha, NE

  Licensed under the Apache License, Version 2.0 (the "License");

  you may not use this file except in compliance with the License.

  You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

FreeTDS

Copyright (C) 1998-1999 Brian Bruns

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details. You should have received a copy of the GNU Library General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

ACE Library

Copyright and Licensing Information for ACE(TM), TAO(TM), CIAO(TM), DAnCE(TM), and CoSMIC(TM)

ACE(TM), TAO(TM), CIAO(TM), DAnCE>(TM), and CoSMIC(TM) (henceforth referred to as "DOC software") are copyrighted by Douglas C. Schmidt and his research group at Washington University, University of California, Irvine, and Vanderbilt University, Copyright (c) 1993-2009, all rights reserved. Since DOC software is open-source, freely available software, you are free to use, modify, copy, and distribute--perpetually and irrevocably--the DOC software source code and object code produced from the source, as well as copy and distribute modified versions of this software. You must, however, include this copyright statement along with any code built using DOC software that you release. No copyright statement needs to be provided if you just ship binary executables of your software products.

You can use DOC software in commercial and/or binary software releases and are under no obligation to redistribute any of your source code that is built using DOC software. Note, however, that you may not misappropriate the DOC software code, such as copyrighting it yourself or claiming authorship of the DOC software code, in a way that will prevent DOC software from being distributed freely using an open-source development model. You needn't inform anyone that you're using DOC software in your software, though we encourage you to let us know so we can promote your project in the DOC software success stories.

The ACE, TAO, CIAO, DAnCE, and CoSMIC web sites are maintained by the DOC Group at the Institute for Software Integrated Systems (ISIS) and the Center for Distributed Object Computing of Washington University, St. Louis for the development of open-source software as part of the open-source software community. Submissions are provided by the submitter ``as is'' with no warranties whatsoever, including any warranty of merchantability, noninfringement of third party intellectual property, or fitness for any particular purpose. In no event shall the submitter be liable for any direct, indirect, special, exemplary, punitive, or consequential damages, including without limitation, lost profits, even if advised of the possibility of such damages. Likewise, DOC software is provided as is with no warranties of any kind, including the warranties of design, merchantability, and fitness for a particular purpose, noninfringement, or arising from a course of dealing, usage or trade practice. Washington University, UC Irvine, Vanderbilt University, their employees, and students shall have no liability with respect to the infringement of copyrights, trade secrets or any patents by DOC software or any part thereof. Moreover, in no event will Washington University, UC Irvine, or Vanderbilt University, their employees, or students be liable for any lost revenue or profits or other special, indirect and consequential damages.

DOC software is provided with no support and without any obligation on the part of Washington University, UC Irvine, Vanderbilt University, their employees, or students to assist in its use, correction, modification, or enhancement. A number of companies around the world provide commercial support for DOC software, however. DOC software is Y2K-compliant, as long as the underlying OS platform is Y2K-compliant. Likewise, DOC software is compliant with the new US daylight savings rule passed by Congress as "The Energy Policy Act of 2005," which established new daylight savings times (DST) rules for the United States that expand DST as of March 2007. Since DOC software obtains time/date and calendaring information from operating systems users will not be affected by the new DST rules as long as they upgrade their operating systems accordingly.

The names ACE(TM), TAO(TM), CIAO(TM), DAnCE(TM), CoSMIC(TM), Washington University, UC Irvine, and Vanderbilt University, may not be used to endorse or promote products or services derived from this source without express written permission from Washington University, UC Irvine, or Vanderbilt University. This license grants no permission to call products or services derived from this source ACE(TM), TAO(TM), CIAO(TM), DAnCE(TM), or CoSMIC(TM), nor does it grant permission for the name Washington University, UC Irvine, or Vanderbilt University to appear in their names.

UMac

Copyright (c) 1999-2000 Ted Krovetz (tdk@acm.org)

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee, is hereby granted, provided that the above copyright notice appears in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the names of the University of California and Ted Krovetz not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission.

The Regents of the University of California and Ted Krovetz disclaim all warranties with regard to this software, including all implied warranties of merchantability and fitness. In no event shall the University of California or Ted Krovetz be liable for any special, indirect or consequential damages or any damages whatsoever resulting from loss of use, data or profits, whether in an action of contract, negligence or other tortious action, arising out of or in connection with the use or performance of this software.

Gigabase

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the Software), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHOR OF THIS SOFTWARE BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

System.Net.FtpClient

Copyright (c) 2013 J.P. Trosclair

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

SharpSSH

Copyright (c) 2002-2015 Atsuhiko Yamanaka, JCraft,Inc.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice,   this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL JCRAFT, INC. OR ANY CONTRIBUTORS TO THIS SOFTWARE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT

LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Mentalis

Copyright © 2002-2007, The Mentalis.org Team

All rights reserved.

http://www.mentalis.org/

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

- Neither the name of the Mentalis.org Team, nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER

CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NLog

Copyright (c) 2004-2016 Jaroslaw Kowalski <jaak@jkowalski.net>, Kim Christensen, Julian Verdurmen

All rights reserved.

Redistribution and use in source and binary forms, with or without

modification, are permitted provided that the following conditions

are met:

* Redistributions of source code must retain the above copyright notice,   this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

* Neither the name of Jaroslaw Kowalski nor the names of its   contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR

CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

MariaDB ODBC Driver

The MariaDB client libraries for C, Java and ODBC are distributed under the LGPL license, version 2.1 or later. The LGPL license allows you to distribute these MariaDB client libraries freely with any application.

About Software AG, Inc.

Based in Reston, Va., Software AG, Inc. is a wholly owned subsidiary of Software AG headquartered in Darmstadt, Germany. Founded in 1969, Software AG continues to be Europe's largest and a leading global provider of system software and services enabling enterprise data integration and management. Software AG's products control the central IT processes of thousands of renowned companies worldwide including Lufthansa, Siemens, Citibank, Merck, DaimlerChrysler, Sony, BP and Telefonica. Software AG develops products and solutions that support the XML (Extensible Markup Language) standard. XML simplifies the exchange of documents and data as well as the integration of cutting-edge Web applications into traditional IT architectures. In 2003, the corporation achieved 422 million euros in total revenue. Software AG has representation in 59 countries worldwide, and currently employs a staff of approximately 2,600.

Software AG and/or all Software AG products are either trademarks or registered trademarks of Software AG.

All other product and company names mentioned herein are used for identification purposes only, and may be the trademarks or service marks of their respective owners.

Index