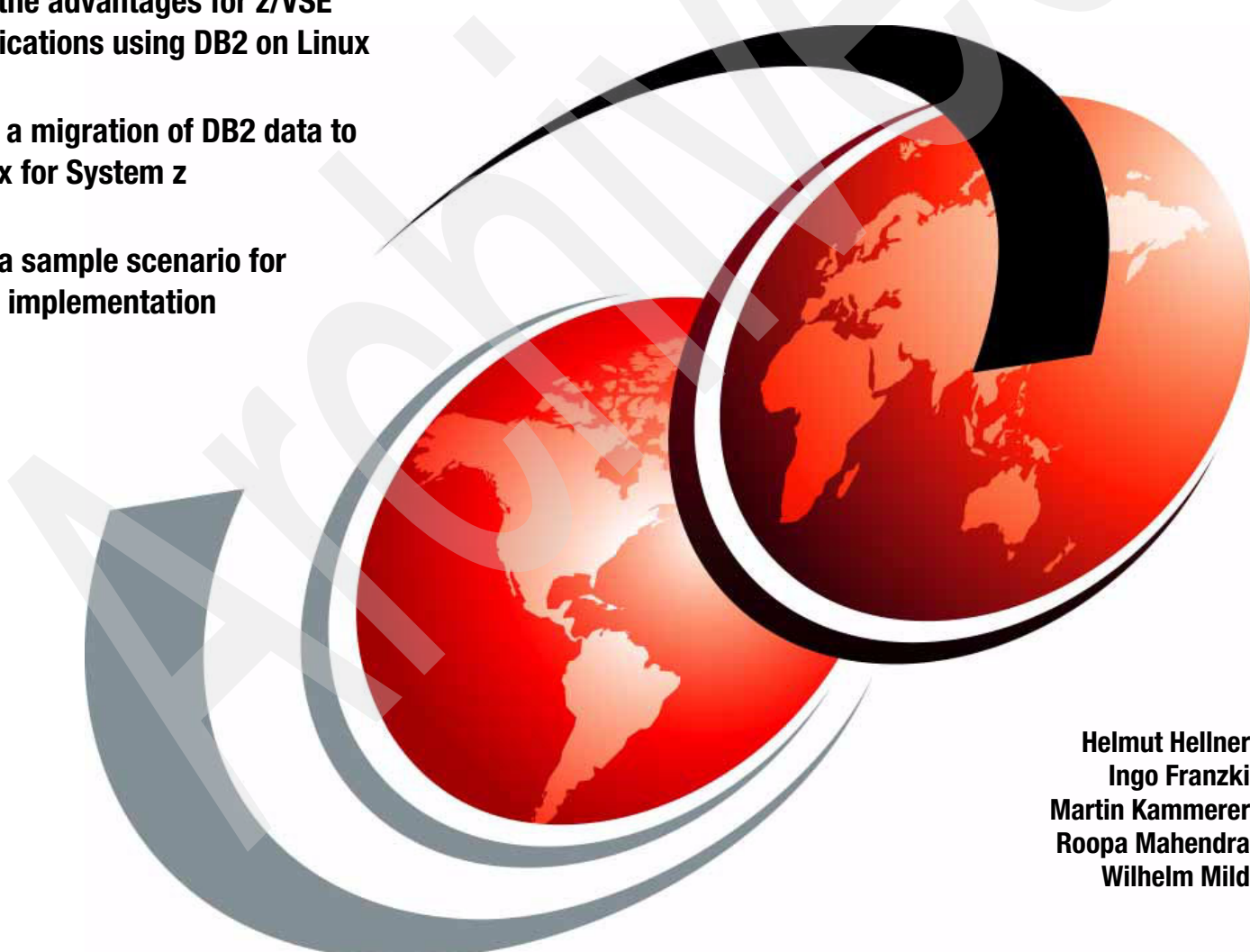


# z/VSE Using DB2 on Linux for System z

See the advantages for z/VSE applications using DB2 on Linux

Plan a migration of DB2 data to Linux for System z

Use a sample scenario for your implementation



Helmut Hellner  
Ingo Franzki  
Martin Kammerer  
Roopa Mahendra  
Wilhelm Mild





International Technical Support Organization

**z/VSE Using DB2 on Linux for System z**

February 2010

Archived

**Note:** Before using this information and the product it supports, read the information in “Notices” on page vii.

Archived

**First Edition (February 2010)**

This edition applies to Version 7, Release 5, Modification 0 of IBM DB2 Server for VSE & VM (product number 5697-F42) and DB2 9.7 for Linux, UNIX, and Windows, Enterprise Server Edition (product number 5765-F41).

© Copyright International Business Machines Corporation 2010. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

# Contents

<b>Notices</b> .....	vii
Trademarks .....	viii
<b>Preface</b> .....	ix
The team who wrote this book .....	ix
Become a published author .....	x
Comments welcome .....	xi
<b>Chapter 1. Overview of a future oriented DB2 environment</b> .....	1
1.1 Business values of using DB2 for Linux on System z .....	2
1.2 z/VSE benefits of using DB2 for Linux on System z .....	3
1.3 Architectural options .....	4
1.3.1 Logical data integration with IBM InfoSphere Federation Server .....	5
1.3.2 Physical data integration on DB2 for Linux on System z .....	6
1.4 Considerations for a rehosting project .....	7
1.4.1 Overview of the steps for a rehosting project .....	7
1.4.2 Skills required in a rehosting project .....	9
1.4.3 Required documentation .....	10
<b>Chapter 2. Planning DB2</b> .....	11
2.1 Hardware options .....	12
2.1.1 Virtualization options with LPAR and z/VM .....	13
2.1.2 IBM System Storage disk considerations .....	14
2.2 DB2 in z/VSE and z/VM .....	17
2.2.1 Considerations for DB2 in a z/VSE environment .....	18
2.2.2 CPU planning .....	18
2.2.3 Memory sizing .....	20
2.2.4 I/O dependency .....	21
2.2.5 Workload behavior .....	21
2.2.6 Outbound database connections .....	21
2.3 Linux considerations .....	22
2.3.1 Kernel .....	22
2.3.2 File system I/O .....	23
2.3.3 I/O processing for disk .....	23
2.3.4 I/O scheduling .....	24
2.3.5 I/O options .....	25
2.3.6 Logical volumes .....	27
2.3.7 Storage pool striping .....	27
2.3.8 DB2 container striping .....	28
2.3.9 Summary of considerations .....	29
2.4 Network planning .....	29
2.4.1 Throughput measurements for different network connection types .....	31
2.5 Database considerations .....	37
2.5.1 Database type .....	37
2.5.2 ASCII and EBCDIC considerations .....	37
<b>Chapter 3. Setting up and customizing the environment</b> .....	39
3.1 General environment considerations .....	40
3.2 Setting up the DB2 enterprise server for Linux on System z .....	42

3.2.1	Installing DB2	42
3.2.2	Configuring DB2 for Linux on System z as a server	51
3.2.3	Configuring DB2 for Linux on System z for access to remote databases	52
3.2.4	Defining the rehosting database	54
3.3	Setting up a federated database environment	54
3.3.1	Setting up a federated database	55
3.4	Setting up the network	62
3.4.1	An overview of HiperSockets	63
3.4.2	Configuring the z/VSE network	63
3.4.3	Configuring the Linux network	64
3.4.4	HiperSockets specifics	72
3.4.5	Network specifics in Linux distributions	73
3.5	Setting up DB2 Server for VSE Client Edition	75
3.5.1	Preparing z/VSE	75
3.5.2	Preparing the CICS system	76
3.5.3	Preparing DB2	78
3.5.4	Installing DB2 Server for VSE Client Edition	86
3.5.5	Verifying the installation	90
3.5.6	Considerations for an implicit connection	95
3.5.7	Connection pooling in an online resource adapter	95
3.6	DB2 Server for VSE	96
3.6.1	Preparing the application server to use TCP/IP	96
3.6.2	Preparing the application requester to use TCP/IP	98
3.6.3	Enabling DRDA for DB2 Server for VSE	98
3.6.4	DB2 block size considerations in z/VSE and z/VM	98
3.7	DB2 Server for VM Client Edition	99
3.7.1	Setting up the communication directory	100
3.8	DB2 Server for VM	100
3.8.1	Preparing the application server to use TCP/IP	100
3.8.2	Preparing the application requester to use TCP/IP	102
3.9	DRDA interfaces in z/VSE	102
3.9.1	DB2 Server for VSE & VM interface considerations	102
3.9.2	PROTOCOL parameter considerations	103
	<b>Chapter 4. DB2 data migration and application dependencies</b>	<b>105</b>
4.1	Data migration	106
4.1.1	Prerequisites for data migration	107
4.1.2	Migrating data from DB2 Server for VSE & VM	107
4.2	Migrating the database access packages	111
4.2.1	Creating access packages	112
4.2.2	Creating a package by using CBND and batch binding	112
4.2.3	Binding online applications using CBND	113
4.2.4	Binding batch applications by using the batch binder	113
4.2.5	Package loading and rebinding	114
4.3	Application considerations	114
4.3.1	Implicit connect setup	115
4.3.2	Connection pooling	115
4.3.3	Fetch and insert blocking	117
4.3.4	Compatibility of DB2 VSE with DB2 for Linux, UNIX, and Windows	117
4.3.5	Transferring database object permissions	119
4.4	Moving to production	120
	<b>Chapter 5. Monitoring and tuning</b>	<b>123</b>

5.1 Best practice activities for monitoring and tuning . . . . .	124
5.2 Performance activities . . . . .	126
5.2.1 Collecting data . . . . .	126
5.2.2 Analyzing the data . . . . .	126
5.2.3 Tuning the systems . . . . .	127
5.2.4 Running the test case . . . . .	127
5.3 Linux system monitoring and tuning . . . . .	127
5.3.1 Monitoring with sysstat utilities . . . . .	128
5.3.2 Disk monitoring with the iostat utility . . . . .	128
5.3.3 Disk statistics monitoring with the dasd statistics utility . . . . .	129
5.3.4 SCSI disk statistics for the FCP . . . . .	129
5.3.5 Linux process monitoring with the top utility . . . . .	130
5.3.6 Monitoring with the oprofile utility . . . . .	130
5.3.7 Linux on System z system check . . . . .	134
5.3.8 Additional tools for Linux and z/VM monitoring . . . . .	134
5.4 DB2 for Linux on System z monitoring and tuning . . . . .	135
5.4.1 DB2 Design Advisor . . . . .	135
5.4.2 DB2 Workload Manager . . . . .	136
5.4.3 SQL and XQuery Explain facility . . . . .	140
5.4.4 Snapshot monitor . . . . .	141
5.4.5 DRDA tracing of activity with DB2 for Linux . . . . .	142
5.4.6 Additional DB2 monitoring tools . . . . .	142
5.5 Network monitoring and tuning . . . . .	143
5.5.1 Network trace in z/VSE . . . . .	144
5.5.2 Network activity in Linux on System z . . . . .	145
5.5.3 HiperSockets considerations . . . . .	148
5.5.4 DB2 BLOCK SIZE considerations . . . . .	148
5.6 z/VSE system monitoring and tuning . . . . .	148
5.6.1 z/VSE system monitoring facilities . . . . .	149
5.6.2 Stand-alone monitor tools and products . . . . .	156
5.6.3 CPU considerations . . . . .	157
5.6.4 Memory analysis . . . . .	158
5.6.5 I/O considerations . . . . .	159
5.6.6 z/VM and LPAR considerations . . . . .	160
5.7 DB2 Server for VSE & VM monitoring and tuning . . . . .	161
5.7.1 Operating system tools for monitoring database workload behavior . . . . .	161
5.7.2 DB2 Server for VSE & VM tools . . . . .	161
5.8 Monitoring z/VSE applications and tuning recommendations . . . . .	163
5.8.1 Monitoring connections . . . . .	163
5.8.2 Authentication considerations . . . . .	164
5.8.3 Batch application monitoring . . . . .	164
5.8.4 CICS application monitoring . . . . .	165
5.8.5 SQL query tuning . . . . .	165
5.8.6 DB2 applications . . . . .	166
5.8.7 Best practices for designing high-performance DB2 applications . . . . .	166
5.8.8 Tuning considerations for migrated VSAM applications . . . . .	167
<b>Appendix A. Configuration examples . . . . .</b>	<b>169</b>
The IPINIT00.L TCP/IP configuration member in z/VSE . . . . .	170
File definitions in CICS . . . . .	173
The ARISIVRR.Z parameter file for the DB2 setup . . . . .	175
<b>Appendix B. Database manipulation . . . . .</b>	<b>179</b>

The db_udcs.c program . . . . .	180
The sql819a.h header file. . . . .	182
The crtnick.sqc C program . . . . .	183
<b>Related publications</b> . . . . .	187
IBM Redbooks . . . . .	187
Other publications . . . . .	187
Online resources . . . . .	188
How to get Redbooks. . . . .	189
Help from IBM . . . . .	189
<b>Index</b> . . . . .	191

Archived



# Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.*

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.


## COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

# Trademarks

IBM, the IBM logo, and [ibm.com](http://www.ibm.com) are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.shtml>

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

AIX®	DS8000®	Redbooks®
CICS®	ECKD™	Redbooks (logo)  ®
Cognos®	FICON®	S/390®
DB2 Connect™	HiperSockets™	System Storage™
DB2 Universal Database™	IBM®	System z®
DB2®	InfoSphere™	Tivoli®
developerWorks®	OMEGAMON®	VM/ESA®
Distributed Relational Database Architecture™	Optim™	WebSphere®
DRDA®	OS/390®	z/OS®
DS6000™	OS/400®	z/VM®
	pureXML®	z/VSE™

The following terms are trademarks of other companies:

Snapshot, and the NetApp logo are trademarks or registered trademarks of NetApp, Inc. in the U.S. and other countries.

Java, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

# Preface

Data is one of the most critical and valuable assets of a business. Critical strategic decisions can be made more quickly and effectively when they are based on complete, accurate, and timely operational data. From this point of view, it is important to have an enterprise data management architecture that supports a flexible global view of the business. A global view helps enable decisions that quickly reflect current conditions and anticipate future requirements. Faster, better informed decisions can help drive substantive changes that enhance value.

Many environments today are heterogeneous with a high quantity and diversity of data. In this IBM® Redbooks® publication, we help enterprise architects and IT managers with these environments make decisions for a centralized database or data warehouse. We recommend a centralized data management environment on Linux® on System z®. We include guidance for IBM z/VSE™ and Linux specialists to reorganize existing IBM DB2® VSE data and build a database environment with continuous operation in Linux on System z.

We begin this book by describing the possibilities and advantages of enterprise data management and different technical ways to realize it. Then we discuss planning, which is important for setting the foundation of the architecture that is implemented. We explain the hardware considerations for capacity and performance planning. For the z/VSE system and Linux on System z, we describe considerations for operation in a logical partition (LPAR) and in a virtualized environment with IBM z/VM®. In addition, we discuss the disk behavior for different workloads, storage dependencies, network connections, and DB2 database considerations.

We also guide you in customizing the DB2 server for z/VSE, z/VM, and DB2 on Linux to allow existing z/VSE and z/VM applications to access the database on Linux on System z. We include the data migration, application considerations, dependencies, compatibility, monitoring, and tuning possibilities in such an environment.

## The team who wrote this book

This book was produced by a team of DB2, Linux, and z/VSE specialists from the IBM Development Laboratory in Germany and IBM India Software Labs in Bangalore, India. It was published by the ITSO in Poughkeepsie, New York.

**Helmut Hellner** is a z/VSE developer in the IBM Development Laboratory in Boeblingen, Germany. After several years of working with IBM OS/390® and VM, he joined the z/VSE Development and Service Team. His focus is on security in z/VSE. He studied computer science at the University of Stuttgart, Germany, and graduated in 1981.

**Ingo Franzki** is part of the z/VSE Development Team at the IBM Development Laboratory in Boeblingen, Germany. He joined IBM in 1997. His main focus is on z/VSE connectors, security, and performance. He is a consultant for solutions that involve distributed environments with z/VSE. Ingo is often a speaker at IBM events, user group meetings, and workshops.

**Martin Kammerer** is the team leader of the Linux on System z performance team at the IBM Development Laboratory in Boeblingen, Germany. He has worked more than 25 years in the area of operating system development and performance test, of which the last 20 years were with IBM. His current responsibility is leading, planning, and reviewing performance evaluations for new hardware support, new Linux functions, new Linux distributions, and major updates; handling customer performance problems; and contributing to Linux on System z performance conference sessions. Martin holds a degree in computer science and is certified as a Project Management Professional.

**Roopa Mahendra** is the Product Architect on the DB2 Server for VSE & VM development team in the India Software Labs in Bangalore, India, and has been with IBM since 2006. After initially working on DB2 on z/OS®, Roopa began working on DB2 Server for VSE & VM. Her main focus is on core engine and optimization area of DB2 Server for VSE & VM.

**Wilhelm Mild** is an IBM Certified and Open Group Master Certified IT Integration Architect in the IBM Development Laboratory in Boeblingen, Germany. After earning a degree in computer science, he dedicated more than a decade to VSAM development and data management and has documented this information in the various Redbooks publications that he has co-authored. Wilhelm currently designs solution architectures for heterogeneous environments with z/VSE, z/VM, and Linux on System z. He also teaches workshops and speaks at many international conferences and customer events.

Thanks to the following people for their contributions to this project:

Siegfried Langer  
Edmund Wilhelm

A special thank you goes to the ITSO center in Poughkeepsie, New York, and Mike Ebbers, Project Leader.

## Become a published author

Join us for a two- to six-week residency program! Help write a book dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You will have the opportunity to team with IBM technical professionals, Business Partners, and Clients.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you will develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

[ibm.com/redbooks/residencies.html](http://ibm.com/redbooks/residencies.html)

## Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks in one of the following ways:

- ▶ Use the online **Contact us** review Redbooks form found at:

[ibm.com/redbooks](http://ibm.com/redbooks)

- ▶ Send your comments in an e-mail to:

[redbooks@us.ibm.com](mailto:redbooks@us.ibm.com)

- ▶ Mail your comments to:

IBM Corporation, International Technical Support Organization  
Dept. HYTD Mail Station P099  
2455 South Road  
Poughkeepsie, NY 12601-5400

Archived

Archived

# Overview of a future oriented DB2 environment

The decision to establish modern data management is often accompanied by the need for a solid, high performance hardware platform. The proven, highly evolved IBM System z platform is designed as a high performance, secure, scalable, highly available, cost-effective server that addresses the need for around-the-clock operation. On the System z platform, different workloads take advantage of this high availability and scalability. In addition to traditional workloads, such as those that use IBM z/VSE, new workloads that use Linux on System z and z/VM virtualization technology benefit from the characteristics of the System z platform.

Open Source Linux also enables the centralization of data by using the IBM DB2 Portfolio. The DB2 Portfolio starts with the DB2 database server and can be enhanced with additional components to integrate distributed data and allow for efficient data analysis.

A similar idea is to rehost a DB2 database from z/VSE or z/VM to create a centralized data management architecture in Linux on System z. This architecture combines operational data from different platforms for better operational processing, better administration, and ad hoc data analysis. A centralized environment can help enhance business values, such as faster time to market for products and services. In addition, centralized data management provides the platform for better business analysis, business intelligence, and faster decisions.

In this chapter, we provide an overview of the topics covered in this book. We give conceptual insight into the objectives with a focus on the following areas:

- ▶ “Business values of using DB2 for Linux on System z” on page 2
- ▶ “z/VSE benefits of using DB2 for Linux on System z” on page 3
- ▶ “Architectural options” on page 4
- ▶ “Considerations for a rehosting project” on page 7

## 1.1 Business values of using DB2 for Linux on System z

The globalization factor increasingly drives today's business, meaning that key business applications must be available at any time, day or night. For example, prime business hours in one region might be night hours in another region.

Internet technologies help to enable a global business and increase the business opportunity. To drive a global business, the solution platform provided to customers must be secure, reliable, and available.

DB2 is optimized to deliver industry-leading performance across multiple workloads, while lowering administration, storage, development, and server costs. Designed for large and mid-sized servers, DB2 Enterprise Server Edition is database software for applications that require limitless flexibility and scalability.

DB2 Enterprise Server Edition is packed with the following features, among others, that reduce the overall costs:

- ▶ IBM pureXML® to help reduce development costs and improve business agility
- ▶ Industry-leading deep compression capabilities that reduce storage needs up to 80%
- ▶ Large scalability to handle high volume transaction processing
- ▶ Advanced workload management and performance tuning capabilities
- ▶ Maximum data availability to minimize the costs associated with planned and unplanned downtime
- ▶ Self-tuning memory management and autonomic features that reduce day-to-day database maintenance expenses

While companies are increasing their focus on business optimization initiatives, the information that is essential for optimizing business performance remains locked in numerous silos spanned across the enterprise. Organizations need access to trusted information in order to respond quickly and precisely to business changes and boardroom priorities, as well as to gain a sustainable competitive advantage.

The IBM System z platform and virtualization techniques provide the ability to manage operating systems, storage devices, and networks in a way that brings huge advantages to the business through consolidation and reduced overall complexity. They help enable operational efficiency and are economically attractive. For z/VSE users, these advantages can be realized with the System z platform running Linux on System z. This environment is standard, running on large scalable hardware with high virtualization capabilities.

Virtualized servers work with a large DB2 database or a pool of DB2 databases for the best support of the business processes. Direct access to the existing applications and service-oriented architecture (SOA) integration of business processes on z/VSE brings the advantage of both platforms to the enterprise.

Figure 1-1 on page 3 shows Linux on System z as a central data and access hub with integration of the z/VSE processes. The portal can be the central access point to all applications. You can integrate existing z/VSE applications by using Host Access Transformation Services (HATS) or SOA. The IBM WebSphere® Application Server is the central Web application server that hosts HATS and portal applications. All applications take advantage of the highly scalable DB2 database and the integration possibilities.



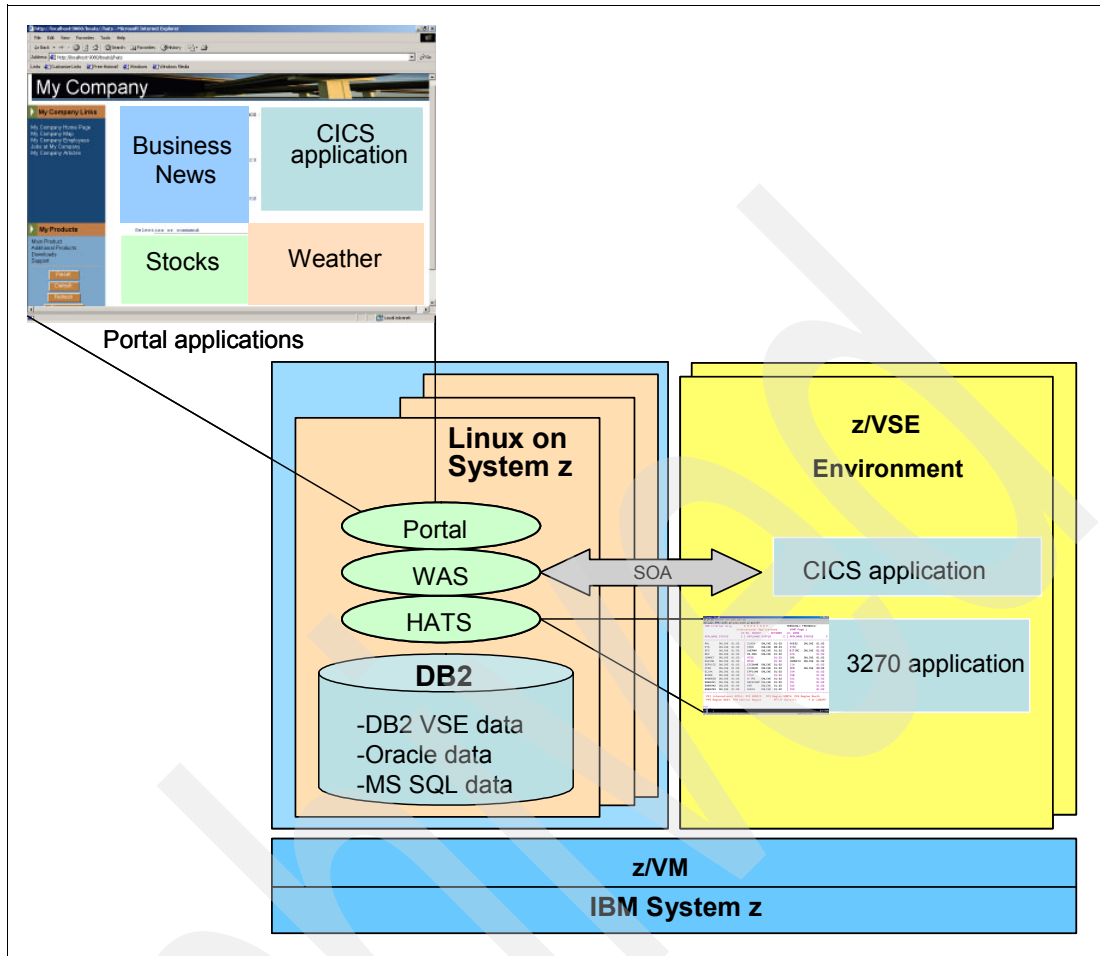


Figure 1-1 Central access through Linux on System z

## 1.2 z/VSE benefits of using DB2 for Linux on System z

The z/VSE operating system is designed for massive parallel workloads. It uses different types of batch and online applications. It also uses different data formats. Data stored in DB2 Server for VSE & VM is one of these data formats.

Because of a high level of development focus for DB2 on open platforms, DB2 for Linux on System z contains enhanced functionality.

**Note:** The product name is DB2 9.7 for Linux, UNIX, and Windows, Enterprise Server Edition.

This functionality might make rehosting of the DB2 database from z/VSE or z/VM to DB2 for Linux on System z an attractive and cost-effective solution. For example, new and existing applications on z/VSE can access the database on Linux and take advantage of the newer technology, scalability, and availability around the clock. Management tasks and service tasks in DB2 on Linux run parallel with your production work. For some customers, this type of solution becomes a critical necessity to provide information management services to their clients.

The following reasons might encourage rehosting:

- ▶ You want to diminish the current functional limitations of DB2 Server for VSE & VM by using DB2 for Linux on System z.
- ▶ You want to mitigate the current capacity constraint of 64 GB per database in DB2 Server for VSE & VM.
- ▶ You want more administration and productivity tools for DB2.
- ▶ You desire to reduce costs by modernizing applications and consolidating data.

By rehosting the database, the advantages are tremendous for the entire environment and the enterprise. You can efficiently integrate existing z/VSE applications with a centralized database on Linux on System z.

## 1.3 Architectural options

DB2 for Linux on System z takes advantage of running on a large scalable server with virtualization capabilities. It can simultaneously drive a large number of clients. With a wide selection of tools that work with DB2 on Linux, it supports the different business requirements from fast access to data, to efficient data analysis and report generations.

DB2 for Linux on System z can be the central data hub of your enterprise. You can integrate or consolidate the distributed data either logically or physically in the same database on Linux on System z, easing access to the enterprise data pool and the analysis of this data.

The approach to *logically integrate* the data in DB2 for Linux on System z documented here enables the integration of a variety of data on different platforms. A central database access point in Linux on System z is a solution that uses the IBM InfoSphere™ Federation Server. The InfoSphere Federation Server provides a single view and access in real time to operational data across systems and architectures in a single request.

The *physical integration* of data into a DB2 database on Linux on System z is the real enterprise database where all data and its dependencies can be reflected and combined. The type of access from distributed platforms to this enterprise database is standard relational access, and the access is transparent.

In addition to data integration, you might also benefit from the comprehensive data warehouse and business intelligence (BI) infrastructure available on Linux on IBM System z. The following major components can be part of this infrastructure:

- ▶ IBM Cognos® BI  
The exploration of information and report generation
- ▶ IBM InfoSphere Information Server  
The data integration software
- ▶ IBM DB2 Alphablox  
The analytical capabilities for decision makers
- ▶ IBM DataQuant  
The easy to use query, format, and editing of data

For installations that currently use DB2 Server for VSE & VM, the following integration options are available:

- ▶ Logical integration with the IBM InfoSphere Federation Server
- ▶ Physical integration by rehosting the database on Linux on System z

Based on your business requirements, your solution for an enterprise database can contain a mixed environment with logical integration and physical integration of other distributed databases. You can implement an enterprise database in steps starting with a logical integration and proceeding one database at a time to a final enterprise database.

### 1.3.1 Logical data integration with IBM InfoSphere Federation Server

IBM InfoSphere Federation Server establishes a federated database on Linux on System z. With IBM InfoSphere Federation Server, you create a mapping that points to the real DB2 Server for VSE & VM.

If you have multiple databases, you can map them at the same time into one federated database on Linux. Different data sources, such as the following types, can be integrated into the IBM federated database:

- ▶ DB2 for Linux, UNIX®, and Windows®
- ▶ DB2 Server for VSE & VM
- ▶ DB2 for z/OS
- ▶ Microsoft® SQL
- ▶ Oracle
- ▶ CA Datacom
- ▶ VSAM files
- ▶ Flat files

Each query request against the federated database can contain columns from the DB2 Server for VSE & VM and other databases simultaneously. The result is a combined set of columns from the different data sources obtained with a single, standard SQL query. You can use this method to query production data in real time or to populate a data warehouse from the various data sources in the enterprise. The federation scenario provides for a logical integration, but it does not physically combine all the data in one central database. The physical residence of the data impacts centralized data management and service. We describe how to centralize your data into one big database in the next section.

Figure 1-2 shows a logical database build using InfoSphere Federation Server.

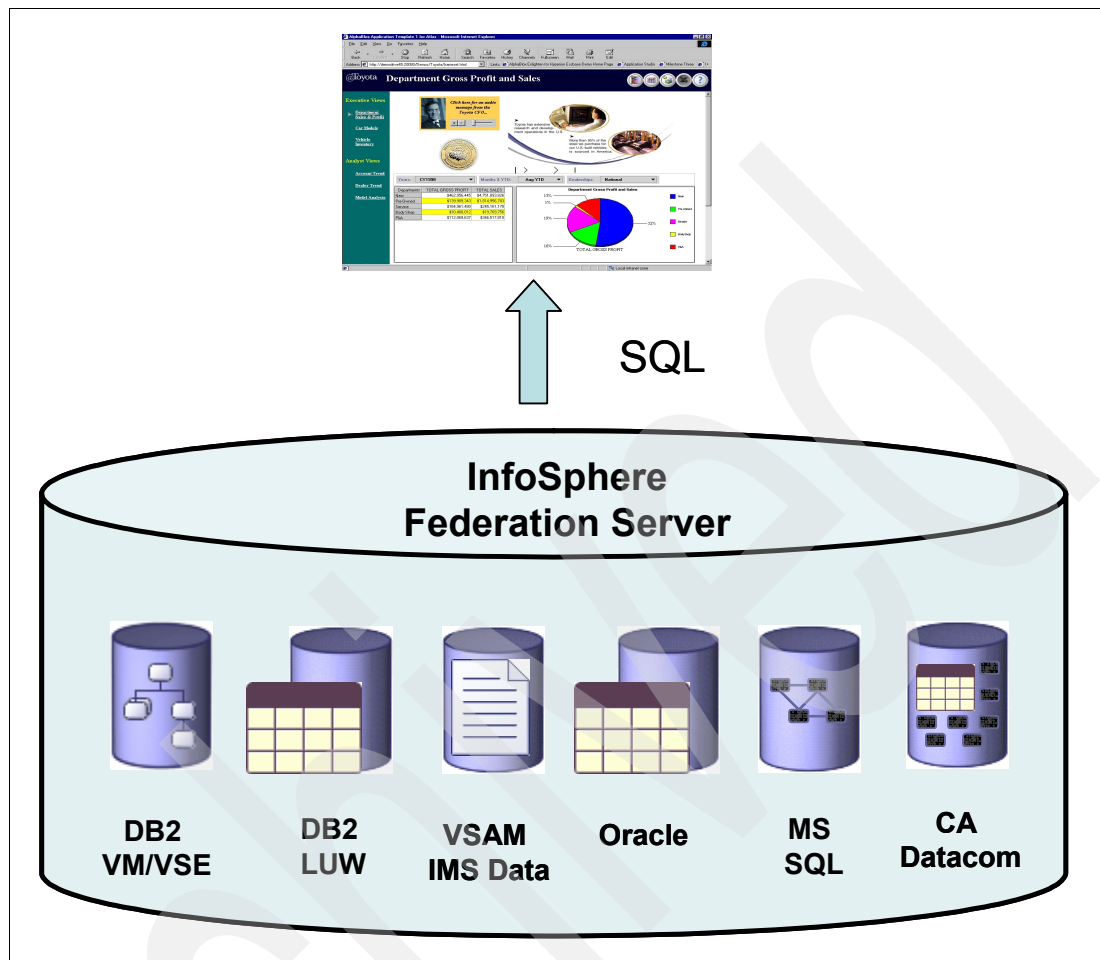


Figure 1-2 Logical database access with InfoSphere Federation Server

### 1.3.2 Physical data integration on DB2 for Linux on System z

The physical integration or consolidation of all data into a single, large database gives the most advanced and comprehensive view to the data of an enterprise. System z and DB2 for Linux on System z can handle such enterprise databases. A single data pool to manage, back up, and analyze can provide many advantages. Consolidation of databases from distributed platforms reduces the complexity of the database and the hardware environment.

In installations with DB2 Server for VSE & VM, with this consolidation, you can use the newest technology, functions, database tools, and standard integration methods. We call this physical integration *rehosting* or *migration* of the database. In this book, we provide the steps to rehost a DB2 database that was initially on z/VSE or z/VM to DB2 for Linux on System z.

Figure 1-3 on page 7 shows an enterprise database on Linux that contains several databases from distributed systems and data from DB2 Server for VSE & VM. With this concept, you can build a data warehouse and analyze your data with IBM COGNOS BI.

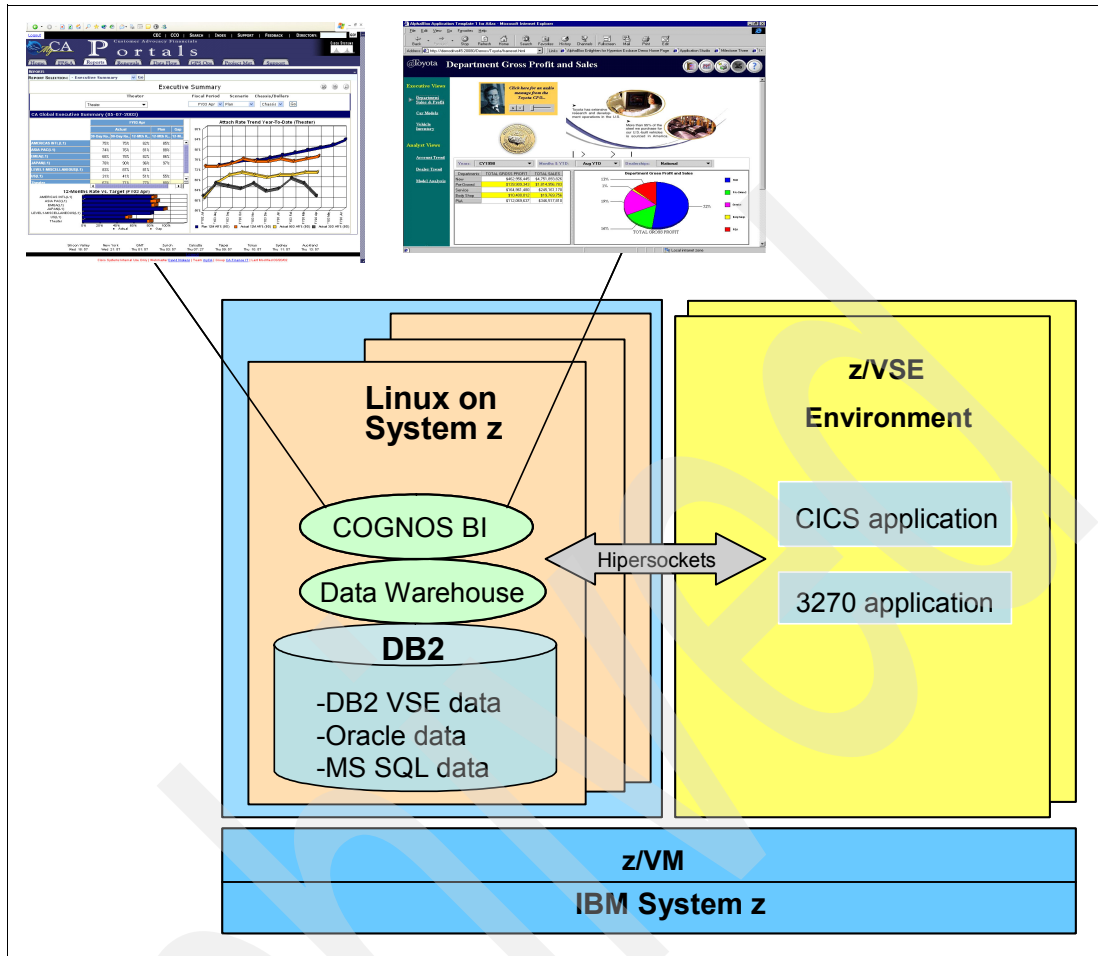


Figure 1-3 Linux on System z as a central hub

## 1.4 Considerations for a rehosting project

For a rehosting project, you must have the required skills and a specified sequence of work items that must be executed as explained in the following sections.

### 1.4.1 Overview of the steps for a rehosting project

To achieve a successful rehosting project from DB2 Server for VSE & VM to DB2 for Linux on System z:

1. Assess what needs to be done.

Determine what needs to be done and how long it will take. You can perform this best by doing a systematic analysis of each of the following tasks that are needed and by estimating how long each task will take.

2. Plan the project.

After you understand the scope of work, plan each step in detail with the aid of a project plan.

3. Acquire the necessary skills, education, and training.

Before the rehosting work begins, the team involved in the project should have or acquire the specific skills that are needed. See 1.4.2, “Skills required in a rehosting project” on page 9.

4. Configure the development environment.

To prepare for the project, establish a development environment. This environment consists of the hardware equipment, operating systems, tools, and DB2 products that are needed to complete the rehosting.

5. Create users and groups, and grant permissions.

After your system administrator and database architecture specialists are trained in DB2 and the associated tools, create the user and group accounts that will be used to access the database. Also assign the correct privileges and authorities to those accounts.

6. Port the database.

Replicating the structure of the source DB2 database in the target DB2 database is the first key conversion task.

7. Port the database application objects.

The contents of a database usually consist of more than just tables of data. In addition, convert and migrate objects that contain programming logic, such as stored procedures, and access packages on DB2 Server for VSE & VM to DB2 for Linux on System z.

8. Install additional database components and products.

Add any functionality, beyond what is provided by DB2 itself, that might be needed to adequately support your applications. Such functionality might include various products and components for information integration, Web integration, replication, federation, high availability, and backup or recovery that you want to consider as part of the porting project.

9. Perform data migration.

After the DB2 database structure is created, populate it at a minimum with test data. Each database system treats data types slightly different and offers various methods of importing and exporting data.

10. Test the applications.

Test your applications carefully. Sometimes application code changes are required because of differences in SQL “dialect” that are implemented by DB2 for Linux, UNIX, and Windows and DB2 Server for VSE & VM.

11. Tune the system’s performance.

After the system is rehosted, plan for time to tune it. In tuning the performance, you might need to change the database and network parameters, the layout of the tables, the referential constraints, or the application code to optimize the queries. You might also need to add indexes.

12. Establish a maintenance strategy.

After the port is complete, consider the maintenance of the application and database. Establish a maintenance strategy early in the development cycle. To ensure the necessary knowledge to keep the application running well, also educate the maintenance staff during the project.

### 13. Perform acceptance testing.

In order to decide when the rehosting is ready to become part of the production cycle, perform a solution integration test. The closer that this test can simulate your production environment, the more you reduce risks and provide a good level of comfort within your organization.

### 14. Document the setup and any changes.

Part of the rehost must include detailed documentation about the application and database setup and any changes that were made. Establish the plan for documentation early on, and monitor it throughout the port to ensure that adequate information is available to the maintenance and support staff.

We used the following steps to realize the physical rehosting of a DB2 Server for VSE & VM database to a DB2 for Linux on System z database and the adaptations to the environment:

1. Plan the disk organization based on the database workload.
2. Install DB2 for Linux on System z.
3. Define a single-byte character DB2 database in Linux for the z/VSE applications.
4. Set up the network connectivity between Linux and z/VSE (HiperSockets™).
5. Set up the DB2 Server for VSE & VM database directory to point to DB2 on Linux.
6. Enable Distributed Relational Database Architecture™ (DRDA®) connectivity between z/VSE and Linux on System z.
7. Set up the federated database in Linux for data migration.
8. Make adaptations in DB2 Linux to allow implicit connections from CICS® and batch applications.
9. Migrate the DB2 data from DB2 Server for VSE & VM to DB2 on Linux.
10. Rebuild access packages in DB2 for Linux.
11. Perform test scenarios with applications.
12. Monitor application and database behaviors and take tuning actions if required.

## 1.4.2 Skills required in a rehosting project

A rehosting project for DB2 Server for VSE & VM to DB2 for Linux on System z requires skills in the following areas:

- ▶ Linux:
  - System setup, performance, and tuning
  - DB2 for Linux on System z expertise: DRDA versus local
  - DB2 for Linux on System z tuning for remote DB2 applications
- ▶ z/VSE:
  - z/VSE System experience
  - DB2 Server for VSE & VM expertise with DRDA communications

Because of these prerequisites, we do not explain all basic terms of these areas. In this book, we assume that you are familiar with these terms when we explain how we executed the steps to rehost DB2 Server for VSE & VM, mention the pitfalls to reduce errors, and provide guidance for similar projects.

### 1.4.3 Required documentation

Before you begin a rehosting project, ensure that you have the following documentation:

- ▶ For disk planning

*IBM System Storage DS8000: Architecture and Implementation*, SG24-6786

- ▶ For network planning

– *IBM System z Connectivity Handbook*, SG24-5444

– *HiperSockets Implementation Guide*, SG24-6816

- ▶ DB2 Server for VSE 750 Manuals

[http://www.ibm.com/support/docview.wss?rs=66&context=SSEPF&dc=DA410&uid=swg27011615&loc=en\\_US&cs=utf-8&lang=en](http://www.ibm.com/support/docview.wss?rs=66&context=SSEPF&dc=DA410&uid=swg27011615&loc=en_US&cs=utf-8&lang=en)

- ▶ IBM DB2 program directories for server and client edition

<http://www.ibm.com/software/data/db2/vse-vm/directory.html>

In addition, consult the IBM DB2 for Linux, UNIX and Windows page at the following address:

<http://www.ibm.com/software/data/db2/9/>



## Planning DB2

In this chapter, we discuss how to plan the configuration of the IBM System z hardware and software components that are required for z/VSE by using DB2 for Linux on System z. We show the options for a mixed DB2 workload on the System z platform.

Specifically we discuss the following topics:

- ▶ “Hardware options” on page 12
- ▶ “DB2 in z/VSE and z/VM” on page 17
- ▶ “Linux considerations” on page 22
- ▶ “Network planning” on page 29
- ▶ “Database considerations” on page 37

## 2.1 Hardware options

The IBM System z platform is a large scalable server that is capable of running various workloads on different operating systems. It has two layers of virtualization:

- ▶ Logical partition (LPAR)

The LPAR is the hardware virtualization layer in which a System z server can be “divided” into multiple virtual servers with a high isolation level. The LPAR hypervisor assigns hardware resources, such as processors, memory, and attachments, to disk storage for all virtual servers. The resource assignments are rather static. Therefore, the flexibility to change resources is limited while the LPAR is active.

On the System z server, the workload is always running in LPARs. The workload can be a native operating system such as z/VSE, z/OS, Linux, and the virtualization hypervisor z/VM.

- ▶ z/VM hypervisor

The z/VM hypervisor is a software virtualization layer in which multiple virtual guest systems can run within an LPAR. With z/VM, you can dynamically assign resources, such as disk, memory, CPU, and network, to guest systems as needed. The guest systems can share these resources based on the SHARE settings in z/VM. You can create new guest systems relatively fast in z/VM when you need them or add resources to existing guests. Based on priorities, you can define which guests are favored in their request for resources.

The IBM System z server supports the following processor types:

- ▶ Central processors (CPs), also known as *standard processors*, can run any operating system, including z/VSE, z/OS and the z/VM hypervisor. If you run z/VM, you can run multiple z/VSE systems as guests under this z/VM.
- ▶ Integrated Facility for Linux (IFL) is the specialty engine that can run only Linux on System z and the z/VM hypervisor. If you run z/VM, you can run multiple Linux guests on the System z server.
- ▶ System z Integrated Information Processors (zIIPs) are specialty engines that are supported by z/OS only to run DB2 workloads.
- ▶ System z Application Assist Processors (zAAPs) are specialty engines that are supported by z/OS only to run Java™ workloads.

**CPU:** We often use the term *CPU* in this document to mean a CP (standard processor), unless otherwise noted.

For this book, only CPs and IFL processors play a role. You choose different types of processors mainly for economical reasons. The software that runs on a CP is charged based on the size of the processor, meaning its speed. A CP can run on different capacity settings to best suit the required capacity. You choose the capacity setting that best suits your performance requirements to keep the price as low as possible. An IFL processor always runs on full capacity, similar to all specialty engines. The software that runs on IFL processors is charged based on the number of processors.

In this book, we use an environment that consists of two LPARs. Both LPARs run the z/VM hypervisor. z/VSE runs in one LPAR as a guest under z/VM, and Linux for System z runs in the other LPAR as a guest under z/VM. The Linux LPAR has IFL processors, while the z/VSE LPAR has one CP. Figure 2-1 illustrates this environment.

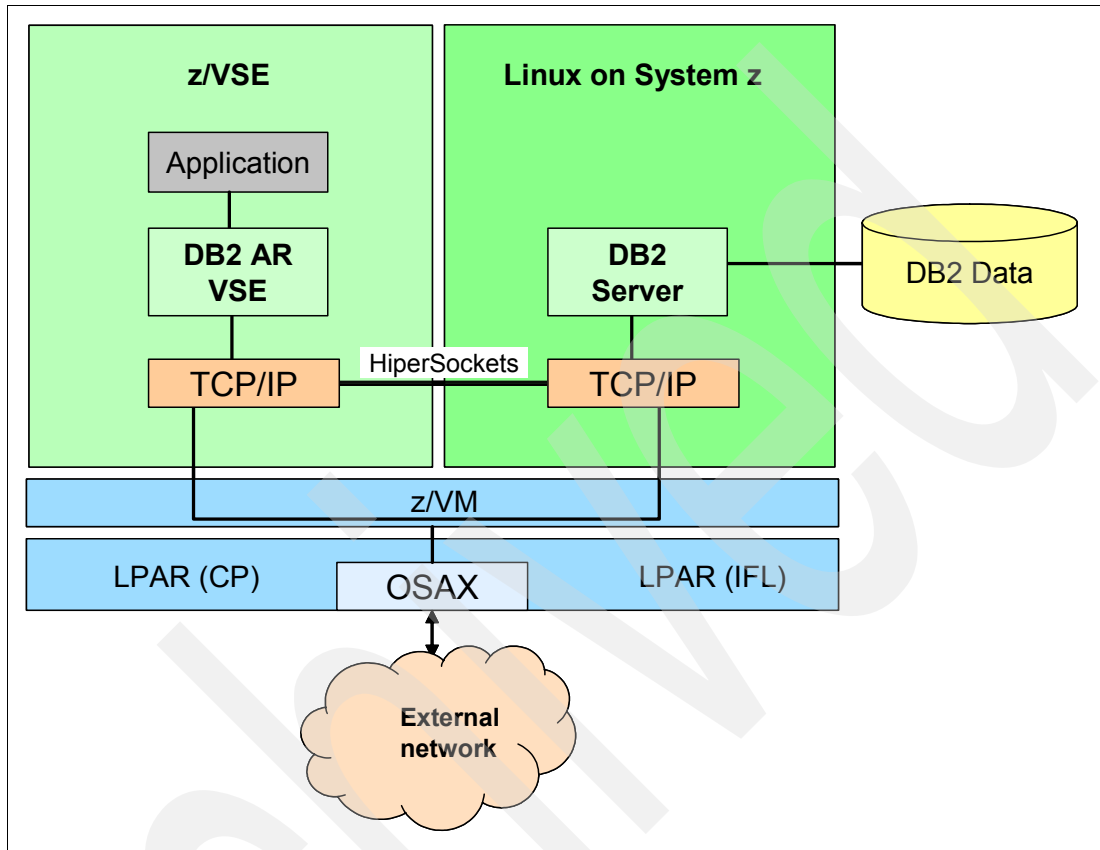


Figure 2-1 z/VSE and Linux on System z in a virtualized environment with z/VM

### 2.1.1 Virtualization options with LPAR and z/VM

z/VSE systems and Linux on System z run native in an LPAR or as a z/VM guest. Both the LPAR hypervisor and z/VM represent a virtualization layer.

In a virtualized environment with z/VM, overcommitting resources is a common practice. For example, overcommitting resources means that your systems detect more real storage than actually exists. In most cases, the systems do not fully use all the memory that they seem to have available. The virtualization layer provides the system with the required resources if needed, for example, through paging out other memory areas. In general, overcommitting resources does not hurt, but it might reduce system performance if it is overdone. You must monitor the virtualization layer z/VM to check if it is doing heavy paging because of memory overcommitment. For more information, see Chapter 5, “Monitoring and tuning” on page 123.

Similar to memory, the CPUs that a guest system detects are virtual CPUs. It is also possible that the guest system has dedicated CPUs, in which case, they are not shared. If the CPUs are shared, the guest system might not get the full power of a CPU. The underlying virtualization layer assigns only a portion of the real CPUs to the guest. The portion is based on the priorities, shares, capping, and CPU consumption of other guest systems that run concurrently.

Therefore, follow these tips:

- ▶ When you define the total number of shared processors in LPARs, make sure that it does not exceed more than four times the number of real available processors.
- ▶ As z/VM guest, use as few virtual processors as possible. The number of guest processors (virtual processors) should be less than or equal to the number of processors of the z/VM LPAR. Defining more virtual processors does not help to complete more work.
- ▶ Run huge database servers natively in an LPAR instead of as a z/VM guest.
- ▶ Allocate enough guest memory to busy Linux database servers on z/VM to avoid paging for this guest.
- ▶ Define at least 2 GB of expanded storage for z/VM.
- ▶ Size a Linux database server as a z/VM guest in such a way that Linux does not swap.
- ▶ Because DB2 data caching occurs in the database buffer pool, it is better to correctly size the buffer pool rather than to add more Linux page cache.
- ▶ In case a Linux guest uses z/VM minidisks for the database files, disable minidisk caching for these disks. It is better to give the space for minidisk caching to the guest and add it to the buffer pool.
- ▶ Because of the desired separation of database data in the storage server for large databases, you can only dedicate full pack minidisks, Extended Count Key Data (ECKD™) direct access storage devices (DASDs), or Small Computer System Interface (SCSI) disks to the Linux guest. Dedicated disks have the advantage that the Linux environment can be easily moved from the guest level to a separate LPAR without z/VM.

## 2.1.2 IBM System Storage disk considerations

Different operating systems can use IBM System Storage servers as main storage servers. For example, you can connect the IBM System Storage DS6000™ and DS8000® to a storage area network (SAN) and directly to a System z server.

In a DS8000, you can configure two different types of disk devices:

- ▶ The traditional mainframe disk, called *DASD*, uses the ECKD format. The System z host connects to the DASD in a storage server by using FICON® Express cards with the Fibre Connection (FICON) protocol.
- ▶ The SCSI disk, with a logical unit number (LUN), can be configured. The connection from a host to the LUN in the storage server is the same FICON Express card that runs the Fibre Channel Protocol (FCP), in this case.

### Connectivity considerations

In the System z environment, provide multiple paths to a disk subsystem. Typically, you must define four paths. In the hardware configuration definition (HCD) or I/O configuration data set (IOCDs), you define the channels that can access each logical control unit (LCU) in the DS8000. The same IOCDs describes which port on a FICON Express card is for use with FICON and which port is for use with FCP connections.

### ***Parallel Access Volume feature of the DS8000 series***

Parallel Access Volume (PAV) is an optional licensed feature on the DS8000 series. PAV requires the purchase of the FICON attachment feature in the turbo models. With PAV, you can access multiple volumes simultaneously through alias names. For such access, create ECKD base volumes and PAV aliases for them.

### **HyperPAV benefits**

HyperPAV has been designed to provide an even more efficient PAV function. HyperPAV helps to implement larger volumes and to scale I/O rates without the need for additional PAV alias definitions. You can exploit the FICON architecture to reduce overhead, improve addressing efficiencies, and provide storage capacity and performance improvements.

HyperPAV offers the following benefits:

- ▶ More dynamic assignment of PAV aliases to improve efficiency
- ▶ More flexibility to changing workloads
- ▶ Simplified management of aliases

### **DS8000 cache**

Disk cache, or the DS8000 term *processor memory*, is especially important to System z server-based I/O. The cache in the DS8000 contributes to high I/O rates and helps to minimize I/O response time.

### **Data placement**

After you determine the disk space and number of disks required by your different hosts and applications, you must make a decision about data placement. Several options are possible for creating logical volumes.

You can select an extent pool that is owned by one internal DS8000 server. For example, you can have just one extent pool per server, or you can have several extent pools per server. The ranks of extent pools can come from the arrays on different device adapter pairs.

To configure your storage:

1. Create arrays. Configure the installed disk drives as either RAID5, RAID6, or RAID10 arrays.
2. Create ranks. Assign each array to either a fixed block (FB) rank or a count key data (CKD) rank.
3. Create extent pools. Define extent pools, associate each with either Server 0 or Server 1, and assign at least one rank to each extent pool.
  - ▶ If you want to take advantage of storage pool striping, assign multiple ranks to an extent pool. Note that with the DS8000 GUI, you can start directly with the creation of extent pools. (The arrays and ranks are automatically and implicitly defined.)
4. Create volume groups where FB volumes will be assigned, and select the host attachments for the volume groups.
5. Create System z LCUs. Define their type and other attributes, such as subsystem identifiers (SSIDs).
6. Create striped System z volumes. Create System z CKD base volumes and PAV aliases for them.

For optimal performance, spread your data across as many hardware resources as possible. Although RAID5, RAID6, or RAID10 already spreads the data across the drives of an array, it is not always enough. *Storage pool striping* and *logical volume striping* are two approaches that you can use to spread your data across even more disk drives.

**Storage pool striping:** Striping is a technique for spreading data across several disk drives in such a way that the I/O capacity of the disk drives can be used in parallel to access data on the logical volume. The easiest way to stripe data is to use the DS8000 internal storage pool striping. You use extent pools with more than one rank, and you use storage pool striping when allocating a new volume. This striping method is independent of the operating system.

**Striping at the operating system level:** Linux on System z has the option to stripe data across several (logical) volumes by using the Linux Logical Volume Manager (LVM). LVM striping is a technique for spreading the data in a logical volume across several disk drives in such a way that the I/O capacity of the disk drives can be used in parallel to access data on the logical volume. The primary objective of striping is to achieve high performance reading and writing of large sequential files, but with benefits for random access.

For a comparison of storage pool striping and Linux LVM striping, see 2.3.6, “Logical volumes” on page 27. Other examples for applications that stripe data across the volumes include the SAN Volume Controller (SVC) and IBM System Storage N series Gateways.

**Double striping:** Double striping at the storage subsystem level and at the operating system level does not enhance performance any further.

As is common for data placement and to optimize the DS8000 resources utilization, follow these tips:

- ▶ Equally spread the LUNs and volumes across the DS8000 servers. Spreading the volumes equally on rank group 0 and 1 balances the load across the DS8000 units.
- ▶ Use as many disks as possible. Avoid idle disks even if all storage capacity will not be initially used.
- ▶ Distribute capacity and workload across disk adapter pairs.
- ▶ Use multirank extent pools.
- ▶ Stripe your logical volume across several ranks.
- ▶ Consider placing specific database objects, such as logs, on different ranks.
- ▶ For an application, use volumes from both even and odd numbered extent pools. Even numbered pools are managed by server 0, and odd numbered pools are managed by server 1.
- ▶ For large, performance-sensitive applications, consider two dedicated extent pools: one managed by server 0 and the other managed by server 1.
- ▶ Consider different extent pools for different arrays. If you use storage pool striping, this approach ensures that your ranks are equally filled.

**Important:** Balance your ranks and extent pools between the two DS8000 internal servers. Half of the ranks should be managed by each internal server.

For more details, see *IBM System Storage DS8000: Architecture and Implementation*, SG24-6786.

## 2.2 DB2 in z/VSE and z/VM

We consider an original environment that uses z/VSE applications with DB2 data. The original environment can have DB2 data in the database in either z/VM or z/VSE.

If you want to access a remote database from z/VSE or z/VM, you need the DB2 client code, called *DB2 application requester*. You can use it as a client only function with the DB2 Server for VSE Client Edition, the DB2 Server for VM Client Edition, or DB2 Server for VSE & VM where the DB2 application requester is included.

### Database in z/VM

If the data is stored in a database in z/VM, DB2 is installed on z/VM and on z/VSE. The z/VSE applications access the data with the guest sharing protocol. In this book, we explain how to rehost this data in DB2 for Linux on System z and leave the applications running on z/VSE. For this scenario, you configure z/VSE access to the DB2 database in Linux directly by using the DB2 Server for VSE Client Edition.

If you have z/VM applications that will access DB2 on Linux, enable and configure the DB2 application requester function or use DB2 Server for VM Client Edition. z/VM needs the application requester, even if the DB2 data no longer exists on z/VM. The application requester is required when the DB2 precompiler compiles applications with SQL statements and access to a remote DB2 database is needed.

**Important:** You can use the DB2 Server for VM Client Edition to enable z/VM applications to access the data on DB2 on Linux on System z. However, z/VM cannot act as a router. Therefore, you cannot use DB2 Server for VM Client Edition from z/VSE applications to access data on DB2 on Linux on System z.

### Database in z/VSE

In case DB2 Server for VSE & VM is installed on z/VSE only and the data is stored in the DB2 database in z/VSE, the applications will use in the new environment the direct way, through TCP/IP to access the data on Linux on System z. Measurements show that the majority of applications perform well when accessing DB2 on Linux on System z instead of the local DB2 database.

**Note:** The z/VSE TCP/IP information in this book is based on TCP/IP for VSE/ESA distributed by IBM, program number 5686-A04.

z/VSE needs the application requester, even if the DB2 data no longer exists on z/VSE. It is required when the DB2 precompiler compiles applications with SQL statements and the interfaces to access a remote DB2 database are needed.

The DB2 Server Client Editions can coexist with other DB2 server installations on the same system. The planning and setup for z/VSE applications to access DB2 on Linux on System z *must be done* in z/VSE and Linux on System z. In this book, we explain how to do this setup and the communication setup between z/VSE and Linux with and without the use of z/VM.

### DB2 Server Client Editions for z/VSE and z/VM

DB2 Server for VSE Client Edition and DB2 Server for VM Client Edition provide the flexibility to install and use only the client component to access remote DB2 databases. If you install DB2 Server for VSE Client Edition V7R5 or DB2 Server for VM Client Edition V7R5, the Distributed Relational Database Architecture (DRDA) capabilities are enabled per default.

For the DB2 Client Editions, consider the following points:

- ▶ You can install the DB2 Server Client Edition on z/VSE or on z/VM.
- ▶ You can use the z/VSE applications to access DB2 for Linux on System z with the DB2 Server for VSE Client Edition on z/VSE.
- ▶ You can use the z/VM applications to access DB2 for Linux on System z with the DB2 Server for VM Client Edition on z/VM.
- ▶ You *cannot* use z/VSE applications to access DB2 for Linux on System z with the DB2 Server for VM Client Edition on z/VM only.

## 2.2.1 Considerations for DB2 in a z/VSE environment

When installing DB2 Server for VSE Client Edition on a z/VSE system, you must have a supported environment with z/VSE. DB2 Server for VSE Client Edition requires VSE/VSAM, which is included in z/VSE operating system package. This operating system also includes the following licensed program products that DB2 for VSE requires:

- ▶ VSE/POWER for Query/Report Writing support
- ▶ VSE/ICCF for an interactive development environment
- ▶ CICS Transaction Server for VSE/ESA for online support

If you use double-byte character set (DBCS) data, consider the following tips:

- ▶ In COBOL/VSE programs for z/VSE, you can use SQL identifiers, SQL host variables, and SQL labels with DBCS characters in SQL statements. The COBOL Kanji preprocessor is not required.
- ▶ PL/I programs that use DBCS do not require additional preprocessing. The PL/I compiler supports DBCS.
- ▶ FORTRAN and Assembler programs do not support DBCS variables and constants, which does not prevent you from using DBCS in dynamically defined SQL statements.

In general, static and dynamic SQL query transactions in CICS have the following requirements:

- ▶ Approximately 6 MB to 10 MB of storage for CICS and the ISQL interface
- ▶ A minimum of a 6 MB partition for batch support when using DB2 Server for VSE Client Edition
- ▶ For the DRDA application requester, approximately an additional 1 MB for the resident online resource adapter

You need z/VSE 4.1 or later to use the fast and standard EZASMI assembler interface IBM TCP/IP interface with the enabled DB2 client feature for online applications.

## 2.2.2 CPU planning

z/VSE dispatches all active tasks based on their status and priorities for the available CPU processing power.

Although z/VSE supports up to 10 active processors, do not use more than three CPUs per z/VSE system. Only run with two or three CPUs if your workload is able to exploit the additional CPUs. Turning on more CPUs than the workload can exploit causes additional overhead, with no benefit. If you want to exploit additional CPUs, be sure that the number of partitions you are running is at least one more than the number of CPUs.



One z/VSE application running in a partition, for example F8, can only exploit the power of one CPU. If more than one CPU is active, one partition does not benefit from additional CPUs. This consideration is especially important for applications running under CICS. One CICS region runs in a partition and can, therefore, only get the power of one CPU. All applications running within that CICS region only get a portion of the power of one CPU. Because of these considerations, z/VSE runs best on a fast single CPU system.

To determine whether the workload can exploit more than one CPU, enter the QUERY TD command as follows:

```
SYSDEF TD,RESETCNT          reset the counters
// run your workload
QUERY TD                    display the counters
```

Example 2-1 shows the output from entering the QUERY TD command.

*Example 2-1 Output of the QUERY TD command*

```
query td
AR 0015 CPU STATUS SPIN_TIME NP_TIME TOTAL_TIME NP/TOT
AR 0015 00 ACTIVE 0 428 688 0.622
AR 0015
-----
AR 0015 TOTAL 0 428 688 0.622
AR 0015
AR 0015 NP/TOT: 0.622 SPIN/(SPIN+TOT): 0.000
AR 0015 OVERALL UTILIZATION: 2% NP UTILIZATION: 1%
AR 0015
AR 0015 ELAPSED TIME SINCE LAST RESET: 27641
AR 0015 1I40I READY
```

In addition to the total used CPU time (TOTAL\_TIME) in milliseconds and the CPU utilization in percent, the value for the non-parallel part of the CPU time (NP\_TIME) is important. The amount of non-parallel CPU time indicates the number of CPUs that the workload can exploit. The value for NP/TOT shows the ratio of non-parallel part. It is also called the *nonparallel share* (NPS). In Example 2-1, an NPS of 0.622 means that 62.2% of the workload is non-parallel CPU time.

You can use the following formula to calculate how many CPUs that workload can use:

$$\text{Max. CPUs exploited} = 0.8 / \text{NPS}$$

In Example 2-1, the workload can exploit  $0.8 / 0.622 = 1.3$  CPUs. Therefore, a second CPU does not bring much benefit in this case.

**Tip:** The observation noted here is workload dependent. During heavy workload times, enter the QUERY TD command (as shown) multiple times. You might find that, at times, your workload can exploit two CPUs, for example, but during other times, it can exploit only one CPU.

### Partition size and priority planning

The partition priority settings ensure that important applications have enough CPU power if they are running under a CPU constraint. However, in most cases, priorities only make a difference if they are running under a CPU constraint. As long as enough unused CPU power is available, every partition, even the one with the lowest priority, gets its part of the CPU power.

The general rule of thumb is to set a server partition (a partition that acts as a server for other partitions) to a higher priority than the partitions that are using the services from the server

partition. With this setting, the TCP/IP partition is set to a higher priority than the applications that are using TCP/IP services, for example, DB2 applications or CICS.

You can display and adjust partition priorities by entering the PRTY command. Example 2-2 shows that partition F1 (running VSE/POWER) has the highest priority and dynamic class Z has the lowest priority. You can also build a partition balancing group by placing an equal sign (=) between one or multiple partitions.

*Example 2-2 Displaying the partition priorities*

---

```
PRTY
AR 0015 PRTY Z,Y,S,R,P,C,BG,FA,F9,F8,F6,F5,F4,F2,F7,FB,F3,F1
AR 0015
AR 0015 1I40I  READY
```

---

Example 2-3 shows that F9, F8, and F6 have equal priorities and build a partition balancing group.

*Example 2-3 Defining a partition balancing group*

---

```
PRTY Z,Y,S,R,P,C,BG,FA,F9=F8=F6,F5,F4,F2,F7,FB,F3,F1
AR 0015 PRTY Z,Y,S,R,P,C,BG,FA,F9=F8=F6,F5,F4,F2,F7,FB,F3,F1
AR 0015
AR 0015 SHARE F9= 100, F8= 100, F6= 100
AR 0015 1I40I  READY
```

---

You can also set the relative shares of each partition in the group, as shown in Example 2-4.

*Example 2-4 Defining a relative share*

---

```
PRTY SHAR,F9=300,F8=200,F6=50
AR 0015 PRTY Z,Y,S,R,P,C,BG,FA,F9=F8=F6,F5,F4,F2,F7,FB,F3,F1
AR 0015
AR 0015 SHARE F9= 300, F8= 200, F6= 50
AR 0015 1I40I  READY
```

---

## 2.2.3 Memory sizing

A workload change using DB2 Server for VSE Client Edition means a change in the requirement of memory. The DB2 client is running in the context of the calling application, such as the CICS partition. The application needs additional storage for DB2 client. For batch, consider that the partition itself will run the DB2 client code. You must plan for this fact because most areas in z/VSE, such as static and dynamic partitions, have a predefined size. The DB2 client code might affect z/VSE for doing paging.

Paging is done if the application running in a partition needs more memory than the amount of real memory that is allocated. Plan on running z/VSE without paging to avoid additional overhead that comes with paging.

With the actual z/VSE features, you can allocate up to 2 GB of memory for one partition and up to 32 GB for a single z/VSE system, which gives you the possibility to exploit the latest capacity features in z/VSE systems for dynamic workload growth.

For more details about memory analysis and tuning, see 5.6.4, “Memory analysis” on page 158.

## 2.2.4 I/O dependency

I/O is not a major factor for a DB2 workload that uses the DB2 client to communicate with a DB2 server on Linux over the network. However, if applications also access other resources, such as files, clusters, and tapes, I/O might still play a role in the overall performance picture. Therefore, it is important to consider all the activities that run at a time. See 5.6.5, “I/O considerations” on page 159, which explains how to find and minimize such constraints.

## 2.2.5 Workload behavior

The application’s behavior is different when it uses a remote database that is running on Linux on System z than when using a local database that is running on z/VSE or on z/VM by using guest sharing. Such local databases use a different protocol to communicate than remote databases.

The access to a local database uses the Cross Partition Communication (XPCC) protocol. The access to a remote database uses the DRDA protocol to communicate with the remote database server over a TCP/IP connection.

The DRDA protocol is much more complex than local communication using XPCC. TCP/IP is the communication layer when using a remote database. The use of DRDA and TCP/IP enlarges the code path length. The DB2 application requester code, including the DRDA protocol handling, is running in the context of the calling application. For a batch application accessing DB2 on Linux, the DB2 application requester code is running in the batch partition. For online CICS applications, the DB2 application requester code is running in the CICS partition. Therefore, you might see increased CPU requirements in the application’s partition and in the TCP/IP partition, compared to a local database.

However, the CPU requirements of the database server itself are moved to the Linux on System z system, freeing up some CPU resources on z/VSE or z/VM. This might be especially notable when a lot of complex SQL queries are executed. Linux on System z typically runs on an IFL specialty engine that always runs at the full speed of the processor. Therefore, it usually has much more capacity than the processor on which z/VSE is running.

## 2.2.6 Outbound database connections

Based on the number of concurrent z/VSE applications that access the database, plan for parallel connections for the remote database. Instead of establishing individual connections for each application, use a *connection pool*. By using a connection pool, the applications share logical connections with each other.

You establish two different types of connections to a remote database:

- ▶ Physical connections

You most likely implement a physical connection by establishing a TCP/IP connection to the remote database server. In most cases, establishing a physical connection is more expensive than establishing a logical connection. Establishing a physical connection usually involves TCP/IP name lookup, connection establishment, and handshaking. Using connection pooling in DB2 Server for VSE & VM V7R5 and later helps to reduce the cost of a connection establishment for connections.

The number of connections defined in the connection pool is workload dependent. You might have physical connections that are currently unused, for example, that are not assigned to any logical connection.

- ▶ Logical connections

A logical connection is established by using the DB2 CONNECT command. It performs a handshake over a physical connection. The handshake process includes the authentication process to verify user ID and password.

See Chapter 5, “Monitoring and tuning” on page 123, to learn about the different tools to monitor and tune both types of connections. For information about the setup and use of the connection pool, see Chapter 3, “Setting up and customizing the environment” on page 39.

## 2.3 Linux considerations

For a database environment on Linux, keep in mind the dependencies of the Linux subsystems. Already in the planning phase, consider the behavior of the Linux kernel, I/O, and disk storage access for a database environment.

### 2.3.1 Kernel

A swap disk for Linux is mandatory. Otherwise, Linux memory management does not use the entire available memory, and Linux starts killing processes when short on memory. Ensure that the swap device and the Linux memory together are large enough to host the maximum number of concurrent processes and files in memory. For more information about process and memory sizes, see Chapter 5, “Monitoring and tuning” on page 123.

To ensure that you have enough hosting space, make the following changes to the kernel parameters in the `/etc/sysctl.conf` file. Enable the `sysctl` service by entering the following command:

```
chkconfig boot.sysctl on
```

The `sysctl` command reads the `sysctl.conf` file during boot time.

Insert a line for each kernel parameter according to `kernel.parameter = value`.

Shared memory has the following kernel parameters:

<b>kernel.shmall</b>	Defines the available memory for shared memory in 4 KB pages.
<b>kernel.shmmax</b>	Defines the maximum size of one shared memory segment in bytes.
<b>kernel.shmmni</b>	Defines the maximum number of shared memory segments.

The DB2 buffer pools use shared memory. Therefore, adapt these parameters to your specific database configuration.

To set the shared memory kernel parameters, use the following strategy:

- ▶ Define the values for `shmall` and `shmmax` large enough, that is to the full memory size, so that they are not a limitation.
- ▶ Ensure that no ongoing swap activity is occurring.
- ▶ Leave at least 5% of memory free. You can control the memory by using the `free` command.

Use the following kernel parameter for the semaphore limits:

**kernel.sem** This parameter consists of four values. The first value defines the number of semaphores that can be in one array. The second value indicates the maximum number of semaphores across the system. The third value is the maximum number of semaphore operations that are possible within one semop call. Finally, the fourth value limits the number of allocatable semaphore arrays.

Use the following kernel parameters for message queue limits:

**kernel.msgmni** Defines the maximum number of queues system wide.

**kernel.msgmax** Defines the maximum size of a message in bytes.

**kernel.msgmnb** Defines the default size of a queue in bytes.

For the semaphore and message queue limits, see also the recommendations from the database product.

## 2.3.2 File system I/O

In all current Linux distributions, format the disks with the file systems as ext3. Choose an appropriate journaling mode to avoid unnecessary processor cycles. The modes are *journal*, *ordered*, and *writeback*. In most cases, *ordered* or *writeback* is sufficient. You can switch off *atime* in the file system. The *atime* mode requests the access time to the database files and might not be of interest. The database itself monitors data access on a more meaningful level.

Place temporary files on a RAM disk instead of a disk device. Temporary files are often small and do not consume a large amount of space. A RAM disk ensures the maximum speed for all I/O operations on this device. Therefore, do not place temporary files on journaling file systems because these files are not required after a system boot.

## 2.3.3 I/O processing for disk

With Linux, you can use FICON-attached ECKD disks and FCP-attached SCSI disks. If you use FICON-attached ECKD DASDs, consider the following points:

- ▶ The traditional mainframe I/O works with a one-to-one (1:1) mapping between the host subchannel and DASD.
- ▶ The system serializes all I/Os to the same DASD at the corresponding subchannel in the host's channel subsystem. Only one I/O can be outstanding at a time. To circumvent this serialization, IBM developed Parallel Access Volumes for storage servers. You must establish a set of DASD numbers, one of which is the real DASD number. All other DASD numbers are alias numbers that can be mapped inside the storage server to the one real DASD. With PAV, Linux can use several subchannels simultaneously.
- ▶ Linux queues all I/Os, which cannot be issued immediately because they are serialized at the subchannels.
- ▶ For disk formatting, the default for the disk block size is 4 KB. Do *not* change this value. If you specify smaller sizes, the disk fragmentation increases to unacceptable values.
- ▶ FICON path groups provide high availability for DASD devices. You can define them in IOCDs. The channel subsystem chooses an operational path, which is transparent for Linux.
- ▶ Linux does load balancing by using FICON path groups and additionally by using PAV devices. The host's channel subsystem handles the FICON path groups. No action is required for Linux. PAV devices are available in Linux as separate devices, and Linux has to balance between them.

The Linux multipath daemon manages the traditional PAV devices. You must define multipathing with a multibus mode. The system administrator must provide special definitions to put this mechanism in place. The I/O requests use all paths in this group in a round-robin policy, switching from one path to the next in intervals as defined in the `rr_min_io` in `/etc/multipath.conf` file. Because all queuing is in Linux, use one `rr_min_io`. The Linux DASD driver handles HyperPAV devices that are defined in the storage server. The driver uses the real device number without any administrator intervention.

If you use FCP-attached SCSI disks, consider the following points:

- ▶ You can immediately issue several I/Os against a LUN. The queue depth value of 32 limits the amount of requests that can be pending to the storage server at the same time. Queuing can occur in the FICON Express card (FCP mode), the storage server, or both. The system queues additional I/O request in Linux.
- ▶ The disk blocks are 512 bytes, and you cannot change this value.
- ▶ You can achieve high availability by using Linux multipathing with the multipath mode failover. If you choose this mode, all I/O requests use the primary path to the LUN. If this path is not operational, Linux takes an alternate path from the failover group. The system administrator must make additional definitions in the `/etc/multipath.conf` file to put this mechanism in place. However, the failover mode keeps this additional utilization at a minimum level.
- ▶ Linux multipathing with the multibus multipath mode does the load balancing. The I/O requests use all paths in this group in a round-robin policy, switching from one path to the next in intervals as defined in the `rr_min_io` in `/etc/multipath.conf` file. Switching after each I/O request might not be appropriate because the queue depth and the I/O request size influence the efficiency. For `rr_min_io`, do not choose values smaller than 100.

**Linux multipath daemon:** The Linux multipath daemon increases processor utilization.

### 2.3.4 I/O scheduling

Read or write operations result in I/O requests for one of the two disk drivers, either the SCSI driver or the DASD driver, depending on whether the target disk is an FCP or SCSI LUN or a FICON or ECKD DASD. While these I/O requests are in process through the kernel disk I/O layers, Linux performs following generic operations on them:

1. Linux merges I/O requests, targeting adjacent disk blocks into one bigger I/O request, to reduce the total amount of requests and to reduce the overall response time from the storage server. The maximum size of one I/O request for ECKD DASD and for SCSI LUNs is 512 KB.
2. Linux places the requests in an optimized order to achieve the specified scheduling goals.

In Linux, the following I/O schedulers are available:

- ▶ Deadline scheduler

This scheduler is the default setting in Novell SUSE and Red Hat distributions for Linux on System z. The strategy behind this scheduling policy avoids read request starvation. Usually, writes to disk are not time critical because the application is not waiting for completion of the operation and continues processing. In the case of a read, the application might wait until the data is transferred from the storage server to the host and is then blocked until the operation is completed. Therefore, in general, this scheduler prefers reads against writes.

- ▶ Completely fair queuing scheduler (*cfq* scheduler)
 

With this scheduling policy, all users of a particular disk can execute about the same number of I/O requests over a given time. It does not distinguish between reads and writes.
- ▶ Noop scheduler
 

Linux performs only request merging without sorting the requests.
- ▶ Anticipatory scheduler (*as* scheduler)
 

This scheduler uses physical disks and not storage subsystems. It works well only on desktop computers with built-in disk drives.

The I/O scheduler can be set in the `/etc/zipl.conf` file as shown in Example 2-5.

*Example 2-5 I/O scheduler settings*

---

```
[ip12GB8CPUdead1]
target = /boot/zipl
image = /boot/image
ramdisk = /boot/initrd
parameters = "maxcpus=<n> dasd=<dev-nr> root=/dev/dasda1 elevator=deadline"
```

---

Possible values for `elevator` are `deadline`, `cfq`, `noop`, and `as`. As a good practice, specify `elevator=deadline`.

### 2.3.5 I/O options

Apart from the standard I/O processing, you can set two special I/O options, direct I/O and asynchronous I/O.

#### Direct I/O

On a block device, the direct I/O option works similar to the raw devices from the older Linux kernels of release 2.4. The I/O bypasses the Linux page cache and avoids the copy operation to or from the page cache.

Direct I/O offers the following advantages:

- ▶ Direct I/O avoids copy operations, which saves processor cycles.
- ▶ Direct I/O saves memory because the Linux page cache is not used.

Direct I/O has the following disadvantages:

- ▶ Direct I/O might not merge I/O requests as extensively as when the page cache is used.
- ▶ Direct I/O operations to a file require that all requests to this file from all applications must use this option. Otherwise inconsistencies and data corruption can occur if a program relies on the page cache contents of a file and another program modifies the same file directly on the disk.

#### Asynchronous I/O

In the “normal” I/O mode, the issuer of a read request waits until the data is available and the process is blocked meanwhile. If you use asynchronous I/O, the application can issue an I/O request to the kernel and continue normal operation. Later it checks the state of the I/O request. The process does not wait until the kernel has executed the request.

Asynchronous I/O offers the advantage that, instead of waiting, the process performs other work in the same application.

Asynchronous I/O has the following disadvantages:

- ▶ Asynchronous I/O does not work immediately as installed. You have to program your application to handle the asynchronous posting. However, all database products have implemented this feature. Therefore, this drawback is negligible.
- ▶ The processor seems to be busy more often because the wait time for I/O no longer exists. This might be of interest if the Linux system has already been working at a high processor utilization rate without asynchronous I/O. In this case, the benefit of asynchronous I/O might not be great.

**Note:** With DB2, direct I/O is possible only with FCP or SCSI LUNs.

### Linux attachment considerations

When you use the FICON Express4 card type in the System z for links to the storage server, the throughput with SCSI LUNs might be limited if the disk I/O is not using the direct I/O option. Use direct I/O whenever using SCSI disks, because it often improves the throughput.

When you use ECKD DASDs, the absolute throughput numbers are lower compared to those of SCSI LUNs. Figure 2-2 shows throughput comparisons of an I/O workload with 32 simultaneous threads, each thread operating on its own file and disk. The comparison was done by using a FICON Express4 card.

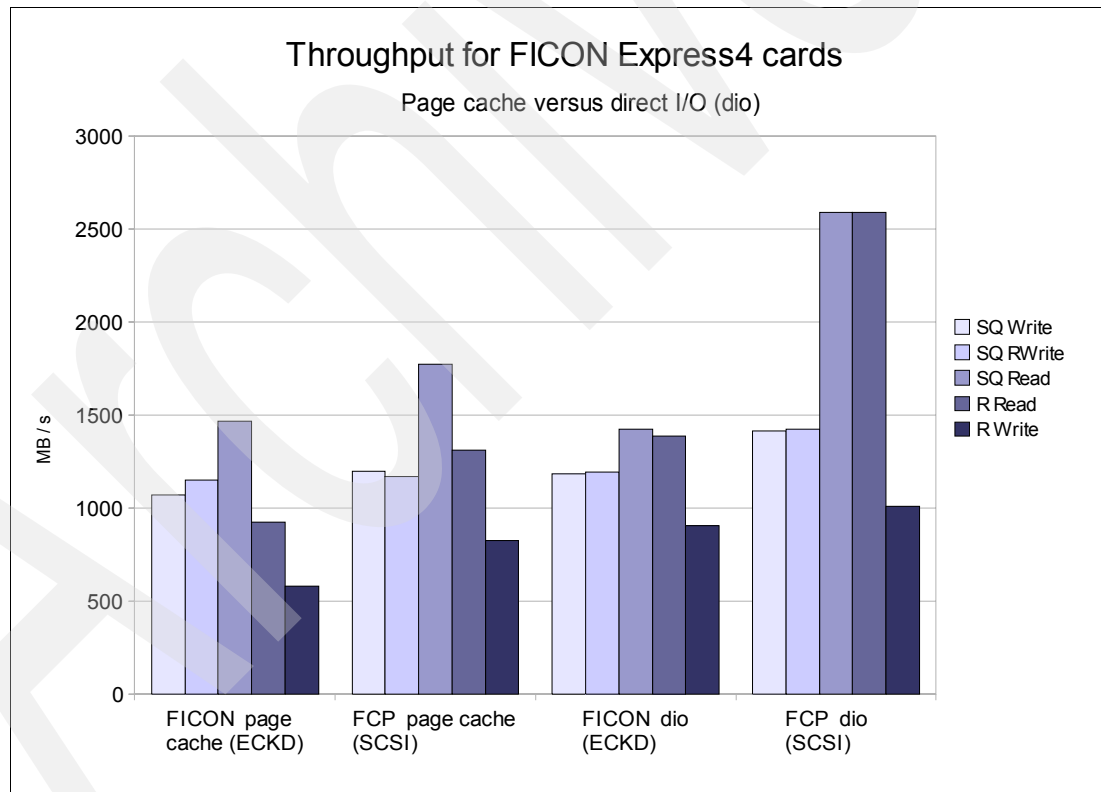


Figure 2-2 Throughput comparison of an I/O workload with 32 simultaneous threads



## 2.3.6 Logical volumes

A *logical volume* is a single disk device within Linux that is represented on several disks in the storage server. Logical volumes are created for the following reasons:

- ▶ You can easily increase the size of a logical volume. Especially if you are using ECKD DASD devices, the size of one DASD is limited to the equivalent of a 3390 model 54 (55 GB). You can only create a larger volume with a logical volume by adding several DASDs to it. For SCSI LUNs, a larger volume is not the main motivation because you can configure a much bigger LUN within the storage server.
- ▶ The volume is extendable. If the volume is full at a later point, you can simply extend and append it without copying the data on the existing volume to a new and bigger volume.
- ▶ You can use a bigger portion of the storage server's infrastructure. With disks in different ranks, you can keep all server sides busy in the storage server. I/O to different ranks benefits from a higher internal I/O bandwidth in the storage server because several device adapters can be used simultaneously.
- ▶ You can create a load balance for this volume and increase the throughput rate. You create a striped logical volume as described in 2.3.7, "Storage pool striping" on page 27. Instead of working on a single disk in the storage server, the I/O requests switch from one disk to the next in intervals of the stripe size. Linux simultaneously issues I/O requests to different stripes.

From this list, linear logical volumes benefit from the first three reasons. For striped logical volumes, all four points apply.

Do not use logical volumes for `/root` or `/usr`. If such a volume becomes corrupted, the system is unusable because you can no longer use neither the base infrastructure nor the commands. Logical volumes require more processor cycles than single physical disks because an additional software layer in Linux must determine the actual target device on the storage server. This problem increases when more physical disks are used for the logical volume.

The goal is to get a balanced load on all paths and physical disks. Therefore, use as many host channels as possible. For ECKD DASDs, Linux switches the paths automatically.

For FCP attachments, the Linux multipath daemon manages the FCP LUNs with a usage plan. If you take a disk from another rank, it uses another path. For each new disk in a logical volume, switch the ranks between server sides and device adapters. Select the disks from as many ranks as possible.

## 2.3.7 Storage pool striping

With License Machine Code 5.30xx.xx, the DS8000 storage servers offer storage pool striped volumes. This feature stripes the extents of a DS8000 storage pool striped volume across multiple RAID arrays.

You can define ECKD DASD and SCSI disks (LUNs) as storage pool striped volumes. Similar to the principle of a Linux logical volume, a storage pool striped volume improves throughput for some workloads. The stripe size is fixed at 1 GB granularity. Random workloads generally benefit more than sequential ones. A storage pool striped volume resides on one storage server and cannot span storage over multiple system servers.

You can combine storage pool striped volumes with striped Linux logical volumes or DB2 container striping.

The biggest advantage of storage pool striping is that the disk placement in selected ranks is no longer necessary. This makes system administration much easier.

Figure 2-3 shows a throughput comparison of a storage pool striped ECKD DASD, using HyperPAV compared to a Linux logical volume (LV) with striping of eight ECKD DASD devices.

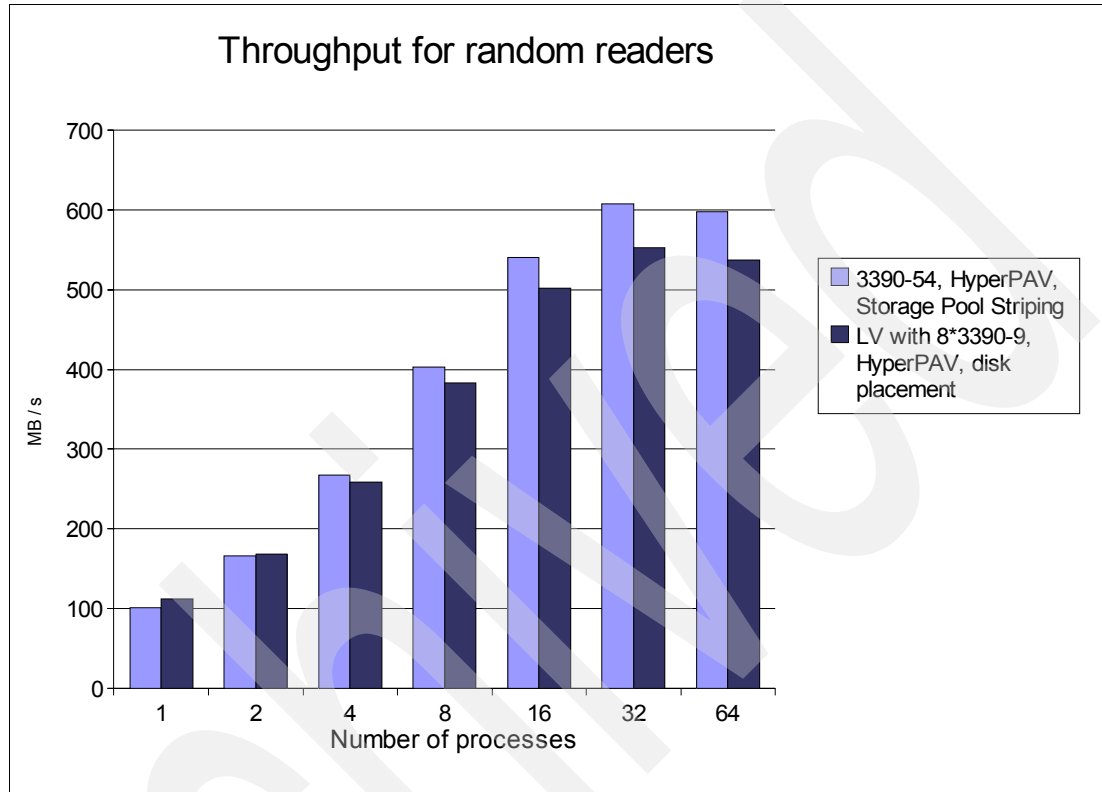


Figure 2-3 Throughput comparison of a storage pool striped ECKD DASD

### Storage server caching mode

If you use ECKD DASDs, you can modify the caching mode of the storage server in Linux. For random database workloads, the caching mode “record” returns the best result.

In Linux with kernel 2.6, you can change the caching mode with the **tunedasd** tool as follows:

```
# tunedasd -g /dev/dasda normal (0 cyl)
# tunedasd -c record /dev/dasda Setting cache mode for device </dev/dasda>...
Done.
# tunedasd -g /dev/dasda record (0 cyl)
```

The modifications are lost if you reboot the Linux system. Therefore, write a script with a set of **tunedasd** commands.

## 2.3.8 DB2 container striping

In DB2, space is allocated in multiples of the page size and are grouped in extents. *Extents* are contiguous regions of data within a container. The size of an extent is defined when the table space is created. You cannot change the extent size for a table space after it is defined. A table space can be defined over multiple containers.

The extent size for a table space also denotes the number of pages that are written to a container before skipping to the next container. The database manager cycles repeatedly through the containers of a table space as data is stored. This ensures proper striping of data pages across several containers. Striping is advantageous if each container resides on a different physical disk because the disks can be accessed in parallel. DB2 Universal Database™ for UNIX and Microsoft® Windows always allocates new pages in a table space in units of one extent.

### 2.3.9 Summary of considerations

To achieve the maximum throughput on Linux, use the following tips:

- ▶ For FICON or ECKD DASD:
  - Use storage pool striped disks.
  - Use HyperPAV (SUSE Linux Enterprise Server 11).
  - Use large volumes.
- ▶ For FCP or SCSI LUNs:
  - Use a Linux logical volume with striping.
  - Use multipathing with the failover mode.

## 2.4 Network planning

Your network topology is important in a rehosting solution. The virtualization layer that you use and the involved systems influence the behavior of your network.

In an environment similar to the example in Figure 2-4 on page 30, the z/VSE and Linux on System z servers communicate in the System z internally. This communication is an advantage because no wires nor physical routers are necessary between the servers. Therefore, the network communication consists of the connections between virtualized servers in the System z and connections to the external network.

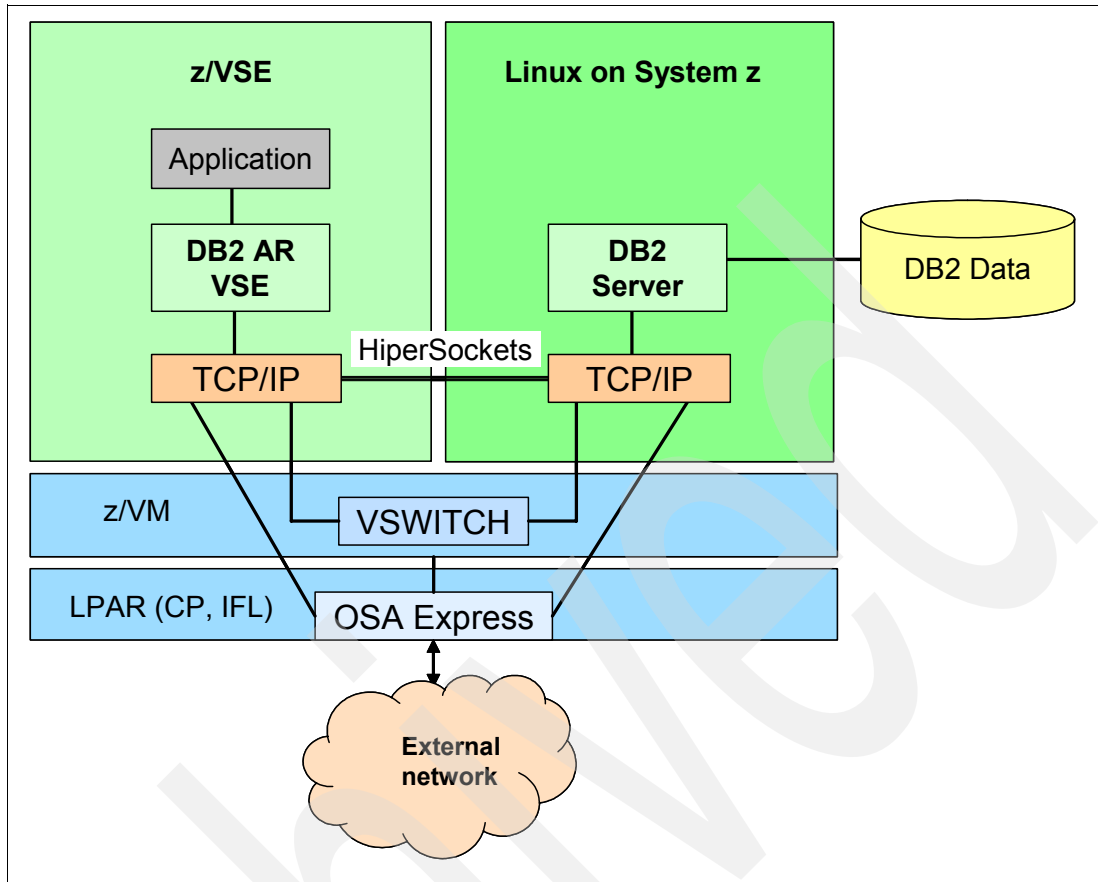


Figure 2-4 Network alternatives

Depending on an implementation in LPARs or as z/VM guests, three options are possible for network communication:

- ▶ HiperSockets
- ▶ Virtual switch in z/VM (VSWITCH)
- ▶ Shared OSA Express card

For host internal connections, consider the following tips:

- ▶ For z/VSE and Linux in an LPAR:
  - HiperSockets with LPAR-to-LPAR communication
  - Shared OSA (workload dependent) for z/VSE installations with smaller CPU capacity

The difference between HiperSockets and shared OSA is that HiperSockets works synchronously with the CPU clock speed, where shared OSA works asynchronously with its own clock speed.

- ▶ For z/VSE and Linux as a z/VM guest:
  - The VSWITCH technology in z/VM for guest-to-guest communication

This option is available only with z/VM V5R4 and later. For highly used network devices, the z/VM VSWITCH can be configured with link aggregation. Within Linux, OSA port scaling is possible with channel bonding.

**Note:** In z/VM V5R4, IFL and CP processors can be in the same LPAR.

- HiperSockets communication between z/VSE and Linux on System z
- Shared OSA (workload dependent) for z/VSE installations with smaller CPU capacity

For host external connections, consider the following tips:

- ▶ For z/VSE or Linux in an LPAR:
  - OSA Express network cards, for example, OSA Express3 and 10 Gb Ethernet cards
- ▶ For z/VSE or Linux on System z as z/VM guest:
  - The VSWITCH technology in z/VM with 10 Gb Ethernet cards and OSA Express attached
  - Directly attached OSA to the z/VSE or Linux guest, for maximum throughput and minimal processor utilization

## 2.4.1 Throughput measurements for different network connection types

In this section, we compare different network connection types. In making these comparisons, we measured network throughput and processor utilization per transaction. To build these workloads, we used AWM, an IBM benchmark that simulates network workload. Each test used different Maximum Transmission Units (MTUs) and 1, 10, and 50 simultaneous connections.

We modeled two types of transactional workload:

- ▶ Request/response (RR)
  - A connection to the server is opened once for a 5 minute time frame.
- ▶ Connect/request/response (CRR)
  - A connection is opened and closed for every request and response.

To simulate the online transactions, we used RR 200/1000, meaning that we sent 200 bytes from the client to the server and received 1000 bytes in response. For the simulation of Web site accesses, we used CRR 64/8 KB, meaning that we sent 64 bytes from the client to the server and received 8 KB in response. We simulated database queries with RR 200/32 KB, meaning that we sent 200 bytes from the client to the server and received 32 KB in response.

We used the following connection types:

- ▶ Linux in LPAR with OSA Express2, 10 gigabit, and MTU sizes of 1492 and 8992 bytes
- ▶ Linux in LPAR with HiperSockets and an MTU size of 32 KB
- ▶ Linux as a z/VM guest with VSWITCH, OSA Express2, 10 gigabit, and MTU sizes of 1492 and 8992 bytes
- ▶ Linux as z/VM guest with VSWITCH, guest-to-guest, and MTU sizes of 1492 and 8992 bytes

The measured scenarios used the following naming conventions:

- |                |   |
|----------------|---|
| <b>OSA_1</b>   | Linux in an LPAR with OSA Express2, 10 gigabit, an MTU size of 1492 bytes, and one connection |
| <b>OSA_j_1</b> | Linux in LPAR with OSA Express2, 10 gigabit, an MTU size 8992 bytes, and one connection       |
| <b>HS_1</b>    | Linux in LPAR with HiperSockets, an MTU size of 32 KB, and one connection                     |

- VS\_OSA\_1** Linux as z/VM guest with VSWITCH, OSA Express2, 10 gigabit, an MTU size of 1492 bytes, and one connection
- VS\_OSA\_j\_1** Linux as z/VM guest with VSWITCH, OSA Express2, 10 gigabit, an MTU size of 8992 bytes, and one connection
- VS\_v\_1** Linux as z/VM guest with VSWITCH, guest-to-guest, an MTU size of 1492 bytes, and one connection
- VS\_v\_j\_1** Linux as z/VM guest, VSWITCH, guest-to-guest, an MTU size of 8992 bytes, and one connection

The scenarios with 10 and 50 simultaneous connections are marked at the end with “\_10” and “\_50”.

Figure 2-5 shows the throughput in transactions per second for the Web site access scenario. The large MTU size shows advantages over the standard MTU size. Virtual guest-to-guest connections are faster than connections that use OSA cards. HiperSockets between LPARs is the fastest connection type.

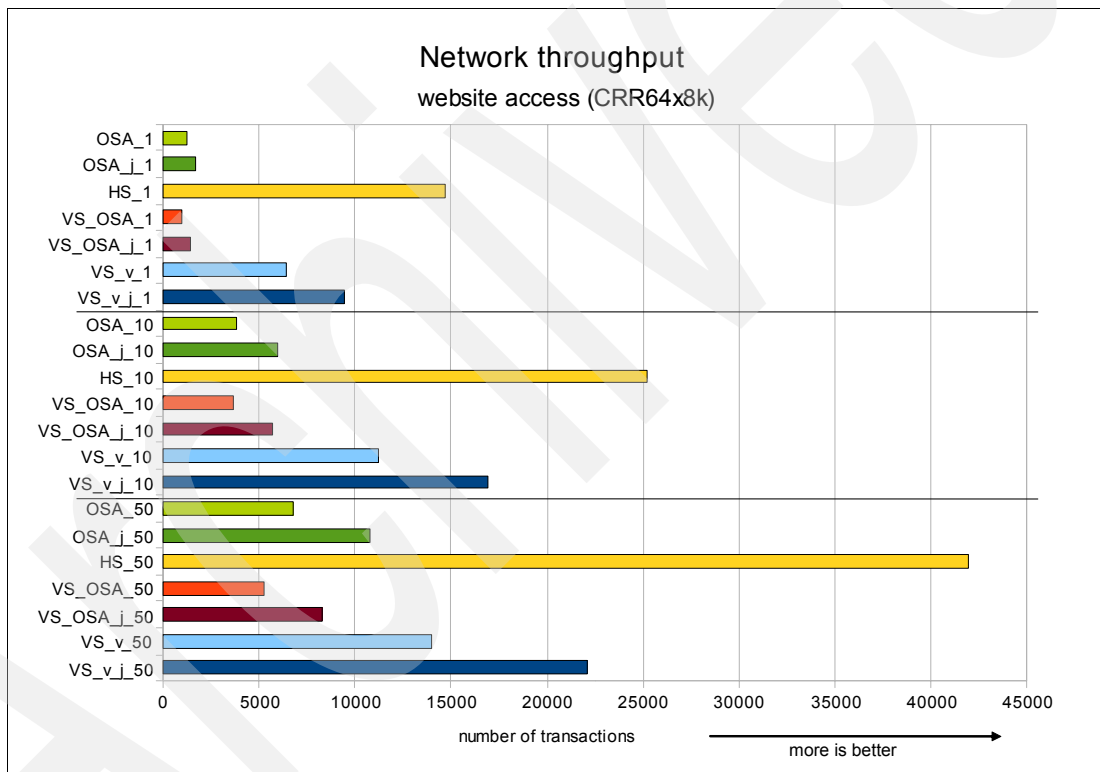


Figure 2-5 Network throughput with Web site access

Figure 2-6 shows the consumed processor time per transaction for the Web site access scenario. The large MTU size shows advantages over the standard MTU size. The OSA connections with Linux in LPAR are the lowest consumers.

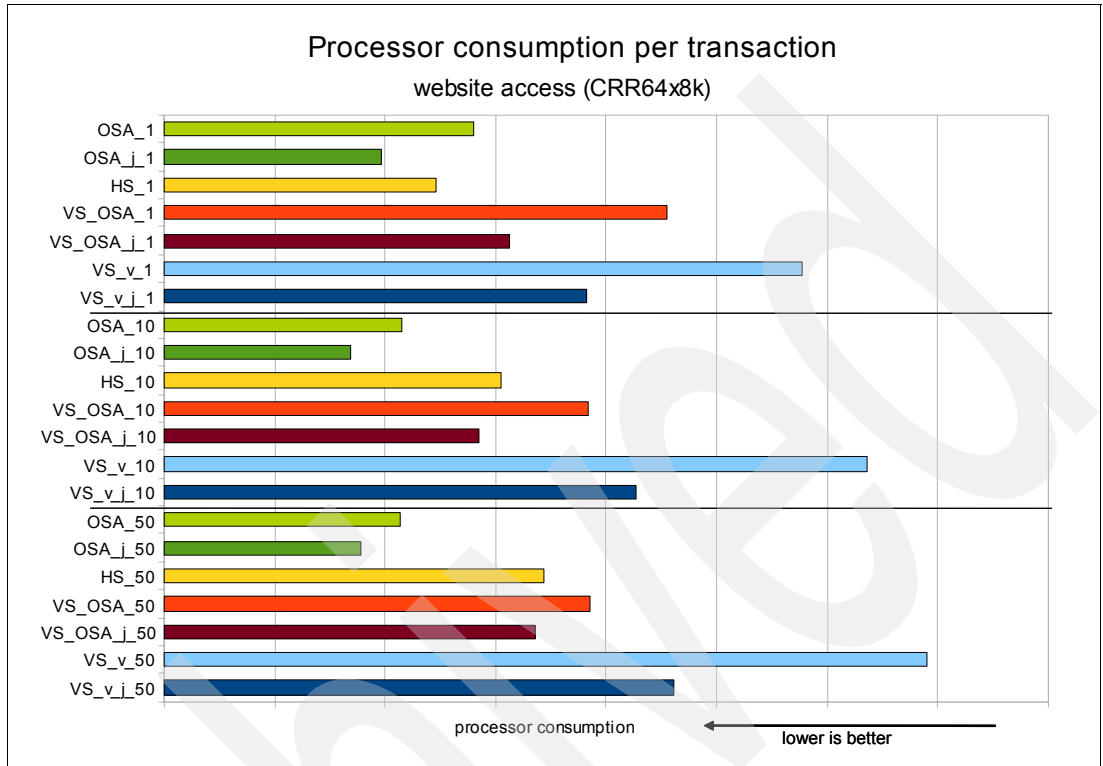


Figure 2-6 Processor time consumption per transaction for Web site access

Figure 2-7 shows the throughput in transactions per second for the online transaction scenario. The MTU size has no influence on the throughput. Virtual guest-to-guest connections are faster than connections that use OSA cards. HiperSockets between LPARs is the fastest connection type.

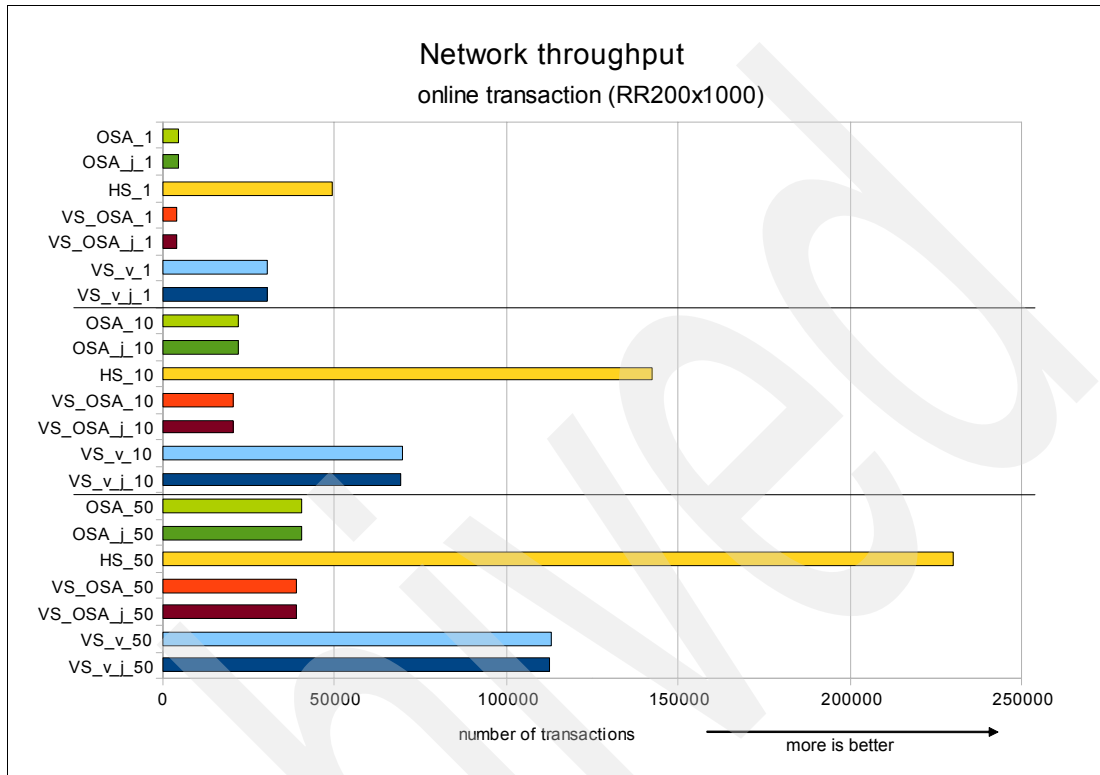


Figure 2-7 Network throughput for the online transaction



Figure 2-8 shows the consumed processor time per transaction for the online transaction scenario. More connections reduce the processor consumption per transaction in the OSA scenarios.

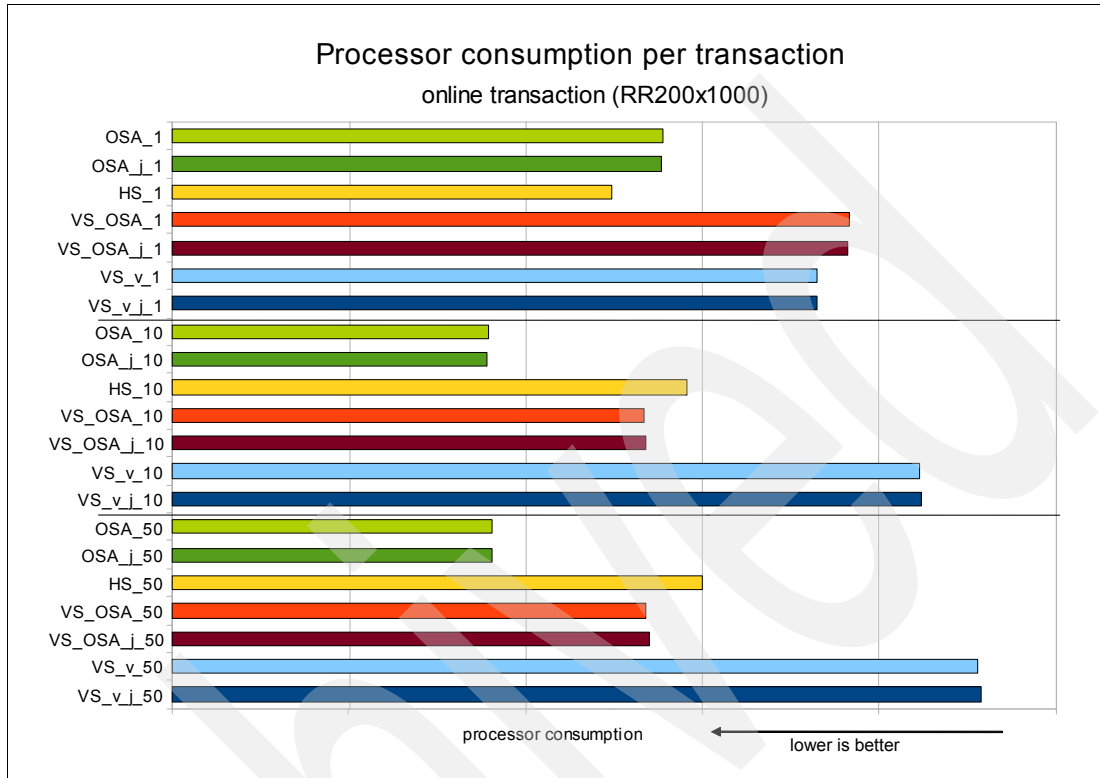


Figure 2-8 Processor time consumption per transaction for the online transaction

Figure 2-9 shows the throughput in transactions per second for the database query scenario. The large MTU size shows advantages over the standard MTU size. Virtual guest-to-guest connections are faster than connections that use OSA cards. HiperSockets between LPARs is the fastest connection type.

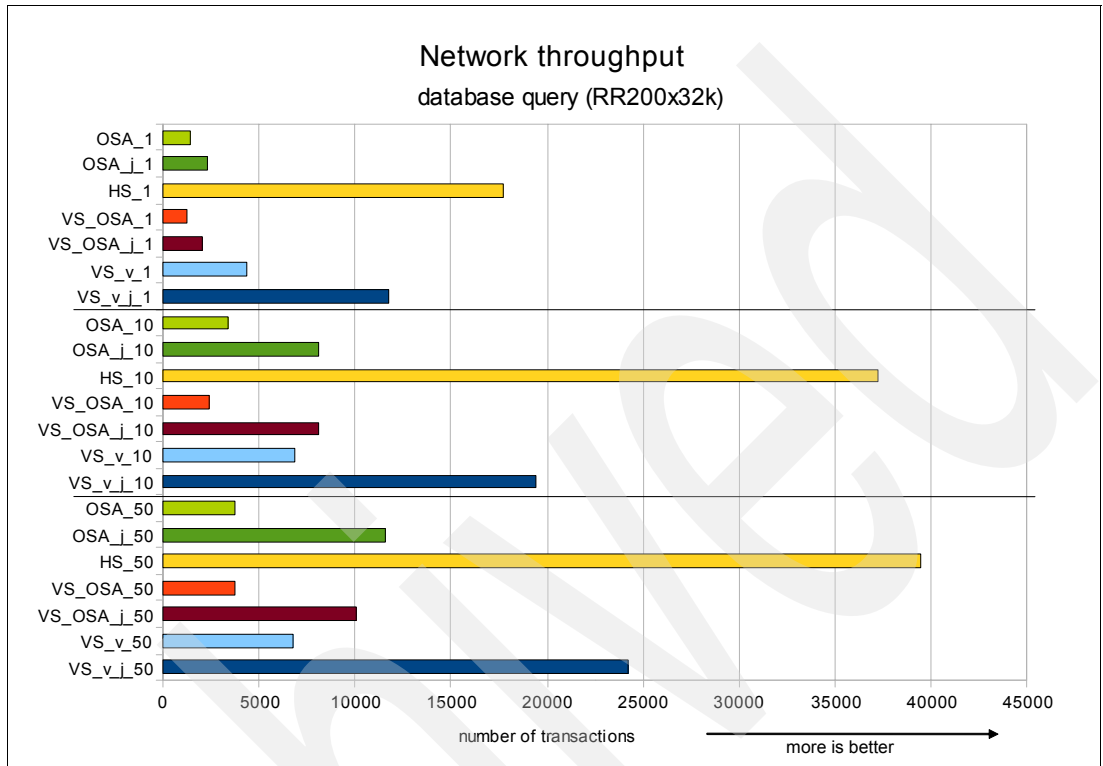


Figure 2-9 Network throughput for the database query

Figure 2-10 shows the consumed processor time per transaction for the database query scenario. The large MTU size shows big advantages over the standard MTU size. The OSA and the HiperSockets connections with Linux in LPAR are the lowest consumers.

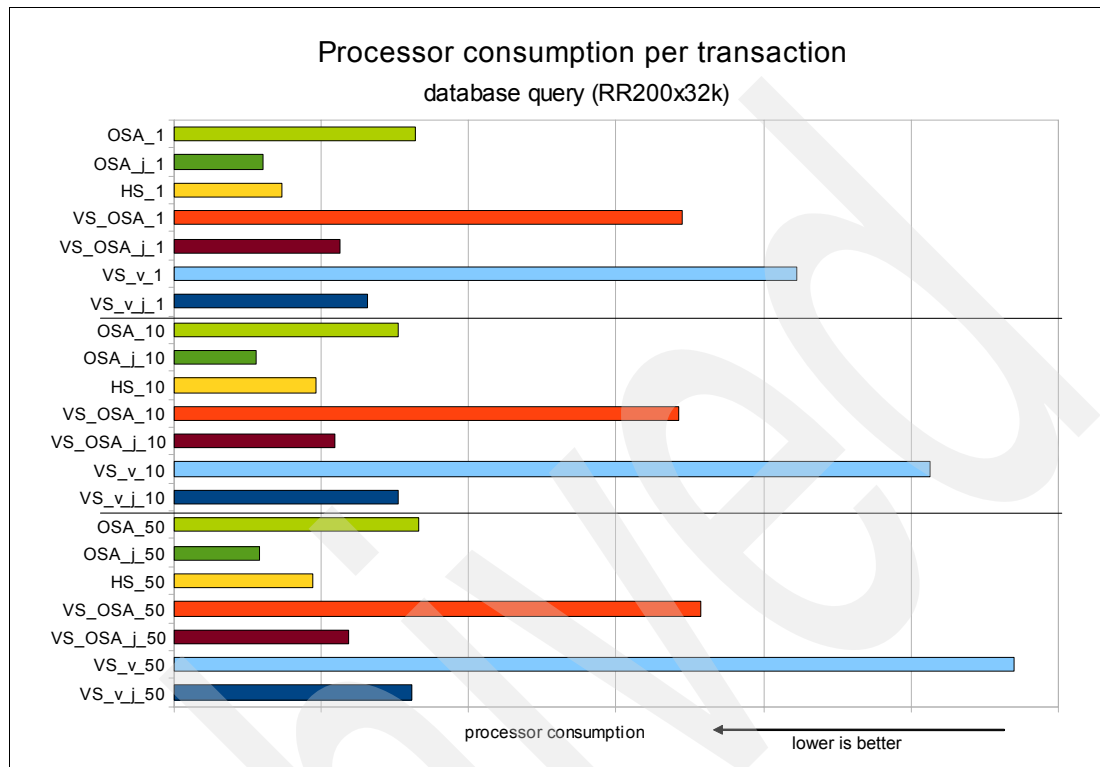


Figure 2-10 Processor time consumption per transaction for the database query

## 2.5 Database considerations

In a distributed database environment, consider the specifics in the different database management systems (DBMS) and the impact for your network. The DB2 DBMS in Linux is aligned with the distributed ASCII type of systems. Plan for a database that works with z/VSE applications and with an Extended Binary Coded Decimal Interchange Code (EBCDIC) environment.

### 2.5.1 Database type

As a good practice, build a single-byte character database in Linux on System z for z/VSE applications. By default, Linux uses a Unicode database that can cause incompatibilities with z/VSE applications. This incompatibility occurs because of the fixed length of columns expected in z/VSE and the variable length of columns when using Unicode-type databases.

### 2.5.2 ASCII and EBCDIC considerations

The z/VSE environment uses EBCDIC collation. By default, a DB2 on Linux database uses ASCII collation. Alphanumeric columns that contain numbers and letters sorts differently in Linux than under z/VSE, which affects programs that rely on EBCDIC ordering. This difference in ordering inhibits automatic matching of z/VSE and Linux test results.

The order in which character data is sorted in a database depends on the structure of the data and the collating sequence defined for the database. In general terms, a collating sequence is a defined ordering for character data that determines whether a particular character sorts higher, lower, or the same as another character. In the ASCII character set, digits come before letters in the sorting sequence, and all of the uppercase characters come before all of the lowercase characters. For the EBCDIC character set, the lowercase characters precede the uppercase characters, and digits come after letters.

DB2 for Linux, UNIX, and Windows, by default, uses the ASCII collating sequence, where DB2 Server for VSE & VM uses the EBCDIC collating sequence to store data. These methods result in alphanumeric columns that contain numbers and letters sorted differently in Linux than under z/VSE and affects programs that rely on EBCDIC ordering.

You can solve this behavior by using one of the following options:

- ▶ Option 1: Use an EBCDIC collated DB2 database in the Linux environment.

This option indicates appropriate use of an EBCDIC collated database in DB2 for Linux, UNIX, and Windows. The alternate code page does not impact application or database performance.

This option has the advantage that all z/VSE and z/VM applications see the same sort order as before. The impact might be that new applications on Linux also see an EBCDIC sort order instead of the ASCII sort order.

- ▶ Option 2: Use the present DB2 ASCII collated database.

You must determine and modify the programs that are affected by the collation difference. You can use differences in test results to identify programs that require modification to operate properly with ASCII ordering.

This option has the advantage that all applications work with the new ASCII collated DB2 for Linux database. The impact is the identification of applications and their adoptions.

- ▶ Option 3: Use the federated database approach.

Using an ASCII database in Linux, create a federated database with an EBCDIC collating sequence on top for the z/VSE and z/VM applications, as explained in the following steps:

- a. Create an EBCDIC collated DB2 database.
- b. For each table, create a nickname in the EBCDIC database that points to that table in the ASCII collated DB2 database where the data resides.
- c. From z/VSE programs, connect to the EBCDIC database on Linux.

This option has the advantage that existing applications on z/VM and z/VSE connect to the federated EBCDIC database and use the EBCDIC collating order. Also the newer applications on Linux can access the ASCII database and see the ASCII collating order. The impact is a small overhead for the z/VM and z/VSE applications because of increased path length.

In this implementation, we use option 3.

## Setting up and customizing the environment

In this chapter, we explain how to connect the systems and the databases to Linux on System z. The setup and customization are based on an environment with an existing DB2 installation that has a DB2 server on z/VSE or a DB2 server on z/VM.

We explain how to set up the following aspects of the environment:

- ▶ The DB2 database on Linux for System z as an application server for connections from z/VSE and z/VM applications
- ▶ A federated database with an Extended Binary Coded Decimal Interchange Code (EBCDIC) sort sequence for z/VSE and z/VM applications
- ▶ The network for DB2 Server for VSE & VM application requester and Linux on System z to use TCP/IP connectivity and Distributed Relational Database Architecture (DRDA)
- ▶ DB2 Server for VSE Client Edition and DB2 Server for VM Client Edition to access data on DB2 for Linux on System z

Specifically, we include the following sections:

- ▶ “General environment considerations” on page 40
- ▶ “Setting up the DB2 enterprise server for Linux on System z” on page 42
- ▶ “Setting up a federated database environment” on page 54
- ▶ “Setting up the network” on page 62
- ▶ “Setting up DB2 Server for VSE Client Edition” on page 75
- ▶ “DB2 Server for VSE” on page 96
- ▶ “DB2 Server for VM Client Edition” on page 99
- ▶ “DB2 Server for VM” on page 100
- ▶ “DRDA interfaces in z/VSE” on page 102

## 3.1 General environment considerations

In this book, we describe an environment in which DB2 Server for VSE Client Edition accesses DB2 for Linux on System z. The implemented network topology uses HiperSockets. For the DRDA communications between DB2 Server for VSE & VM and DB2 for Linux on System z, we use TCP/IP.

The DRDA environment provides the architecture through which you access DB2 relational data, across distributed operating systems. You can run the DB2 application requester and the DB2 application server on different systems. In this implementation, we installed and configured DB2 Server for VSE Client Edition on the same z/VSE system where the DB2 server exists.

Our environment contains the following operating systems and relational database products:

- ▶ Systems:
  - z/VM V5R4M0
  - z/VSE V4R2M1
  - Linux SUSE Linux Enterprise Server 10
- ▶ Databases:
  - DB2 Server for VSE & VM V7R5
  - DB2 Server for VSE Client Edition V7R5
  - DB2 Server for VM Client Edition V7R5
  - DB2 9.7 for Linux, UNIX, and Windows, Enterprise Server Edition

The implementation that we describe in this book is based on the following system definitions for z/VSE, z/VM, and Linux on System z. Table 3-1 shows the system definitions for z/VSE, which is connected by using HiperSockets.

Table 3-1 z/VSE system definitions

Name	Value
z/VSE system name	ZVSEDB2
IP addresses of the z/VSE system	HS: 172.16.0.6 external: 9.152.86.244
Database name	SQLDS
DB2 VSE listener port	446
User ID with DBA authority	DB2ADMIN
Password	myvsepw

Table 3-2 shows the z/VM system definitions.

Table 3-2 z/VM system definitions

Name	Value
z/VM system name	MYVMSYS
IP address of the z/VM system	9.152.86.245
Database name	VMDB2
DB2 VM listener port	8030

Name	Value
User ID with DBA authority	DB2ADMIN
Password	myvmpw

Table 3-3 shows the system definitions for Linux on System z.

Table 3-3 Linux system definitions

Name	Value
Linux system name	lnxdb2
IP addresses of the Linux system	HS: 172.16.0.5 external: 9.152.26.187
Database name	BIGLNxDB
SVCENAME in etc/services file	db2c_db2inst1
SVCENAME port	50001/tcp
User ID with SYSADM authority	db2inst1
Password	mylinpw

Figure 3-1 shows the implementation, its network parameters, and its components.

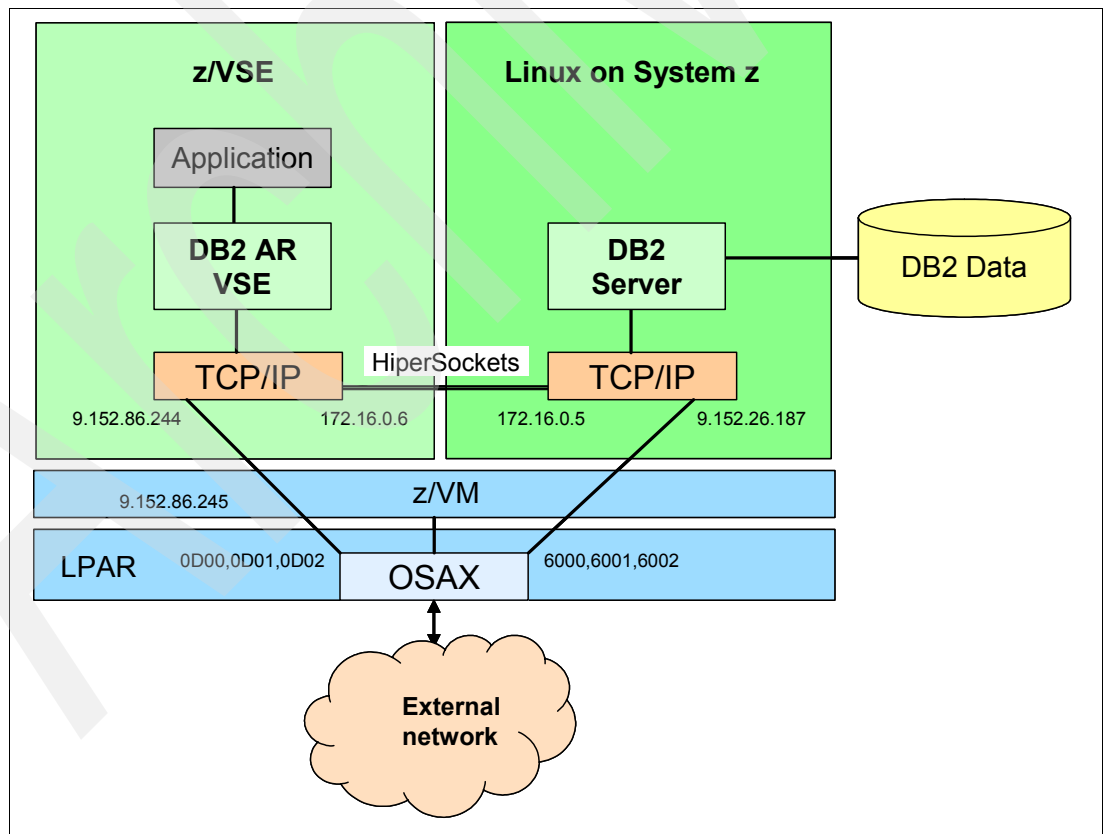


Figure 3-1 Implementation with network parameters and components

## 3.2 Setting up the DB2 enterprise server for Linux on System z

In this section, we explain how to configure DB2 for Linux on System z to access remote databases on z/VSE and to accept requests from remote DB2 clients such as DB2 Server for VSE Client Edition or DB2 Server for VM Client Edition.

### 3.2.1 Installing DB2

To install DB2 for Linux on System z, consider that Linux is a multi-user system. Installing DB2 from root user ID provides more flexibility.

Use the DB2 Setup wizard to define your installation preferences and to install your DB2 database product on your system. To start the DB2 Setup wizard:

1. If the DB2 database product is on a DVD, change to the directory where the DB2 database product DVD is mounted by entering the following command:

```
cd /dvdrom
```

Here, /dvdrom represents the mount point of the DB2 database product DVD.

2. If you downloaded the DB2 database product image, unpack the product file:

- a. Extract the product file:

```
gzip -d product.tar.gz
```

Replace *product* with the name of the product that you downloaded, for example DB2\_ESE\_97\_Linux\_zSeries.tar.gz.

- b. Untar the product file on your Linux operating systems:

```
tar -xvf product.tar
```

Replace *product* with the name of the product that you downloaded, for example DB2\_ESE\_97\_Linux\_zSeries.tar.

- c. Change directory to the unpacked product package:

```
cd ./product
```

In this installation, the product name directory is *ese* as shown in the following example:

```
cd ./ese
```

**National Language Package:** If you downloaded a National Language Package, untar it into the same directory. Untarring the package in this manner creates the subdirectories, for example *./n1pack*, in the same directory. Also, it helps the installer to automatically find the installation images without prompting.

3. Enter the `./db2setup` command as follows from the directory where the database product image resides to start the DB2 Setup wizard:

```
./ese./db2setup
```



4. In the IBM DB2 Setup Launchpad window (Figure 3-2), either view the installation prerequisites and release notes, or proceed directly to the installation. From this window, you can also review the installation prerequisites and release notes for late-breaking information if you have Internet access.

Select **Install a Product** and click **Install New** to start the installation.

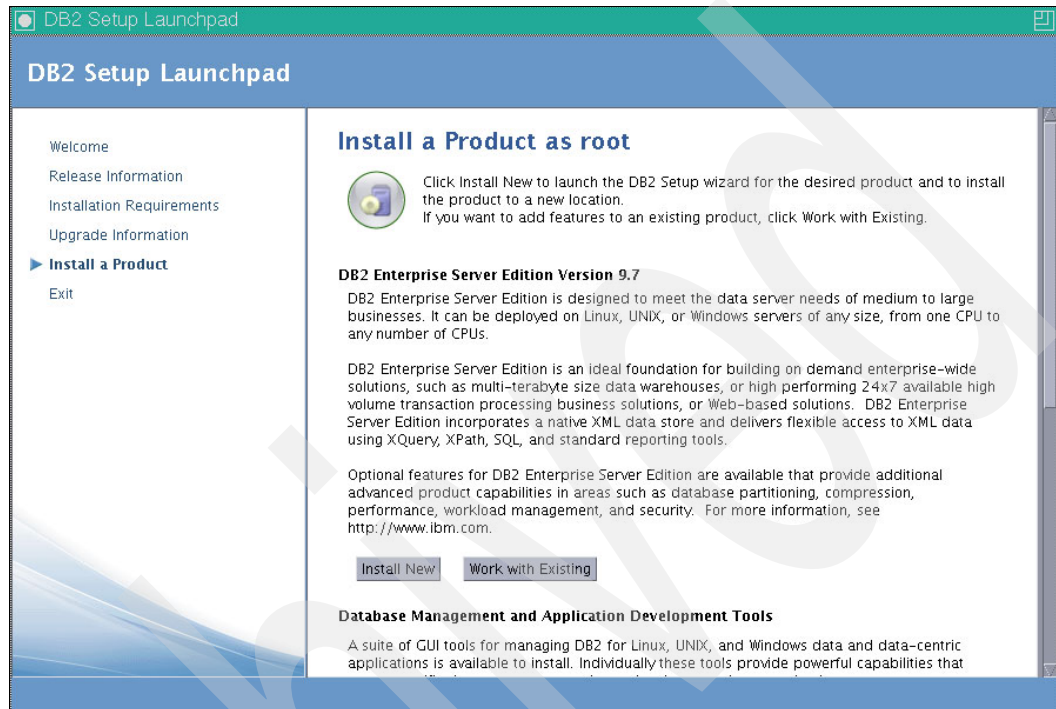


Figure 3-2 DB2 Setup Launchpad

5. Proceed through the DB2 Setup wizard installation panels and make your selections.

**Help:** Installation help is available to guide you through the steps. To invoke the installation help, click **Help** or press F1. You can click **Cancel** at any time to end the installation.

6. In the Select the installation type panel (Figure 3-3), select **Typical** for a typical installation. Click **View Features** to see the selected features.

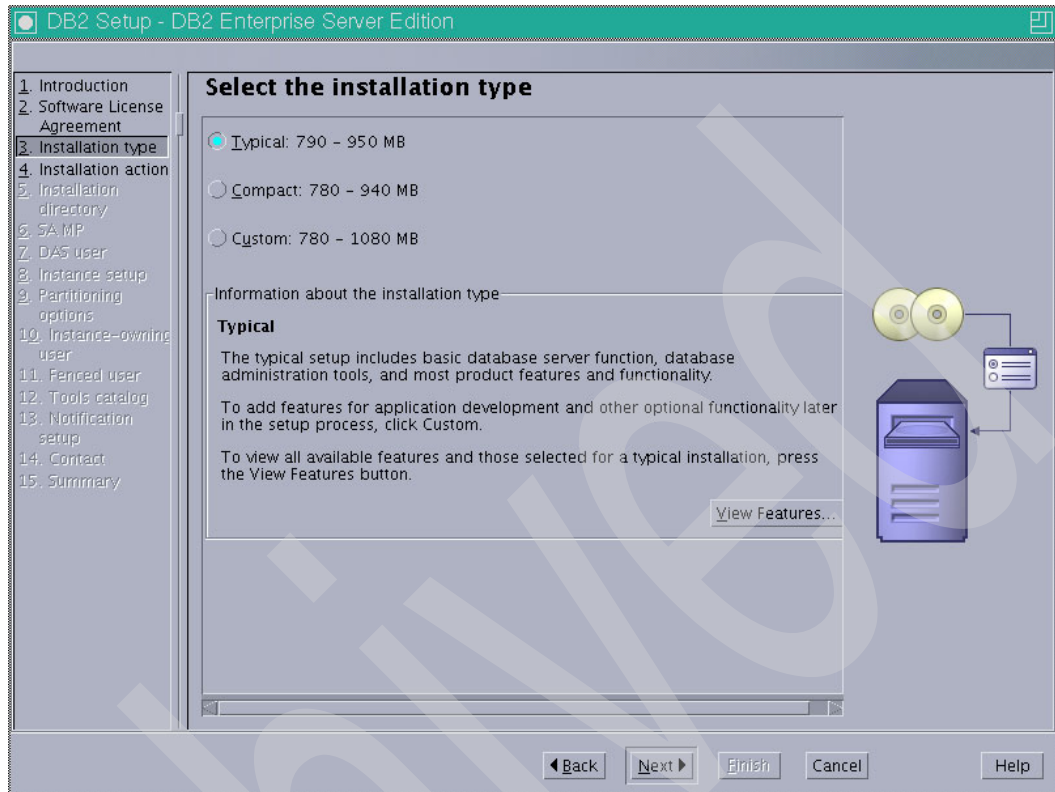


Figure 3-3 Installation window for DB2 on Linux

Figure 3-4 shows the selected features that will be installed. Click **Close** to return to the DB2 installation window (Figure 3-3 on page 44).

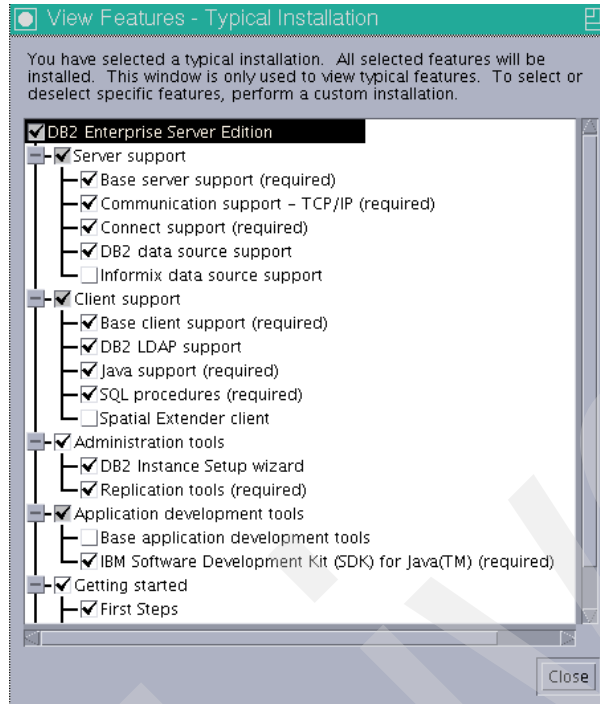


Figure 3-4 Default features selected for a typical installation

From the DB2 installation window, click **Next**.

- In the Set user information for the DB2 Administration Server panel (Figure 3-5), enter the user ID and group name, and then enter a password for the database administrator. Click **Next**.

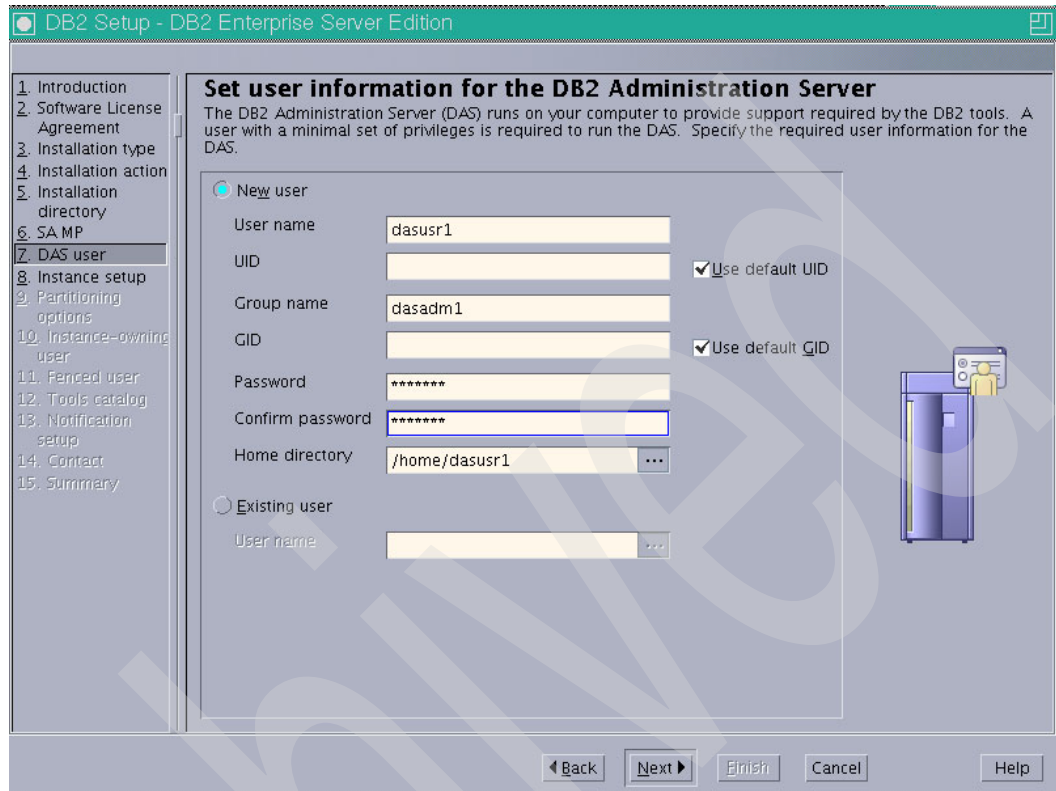


Figure 3-5 User settings for the database administrator

8. In the Set up a DB2 instance panel (Figure 3-6), select **Create a DB2 instance** and click **Next**.

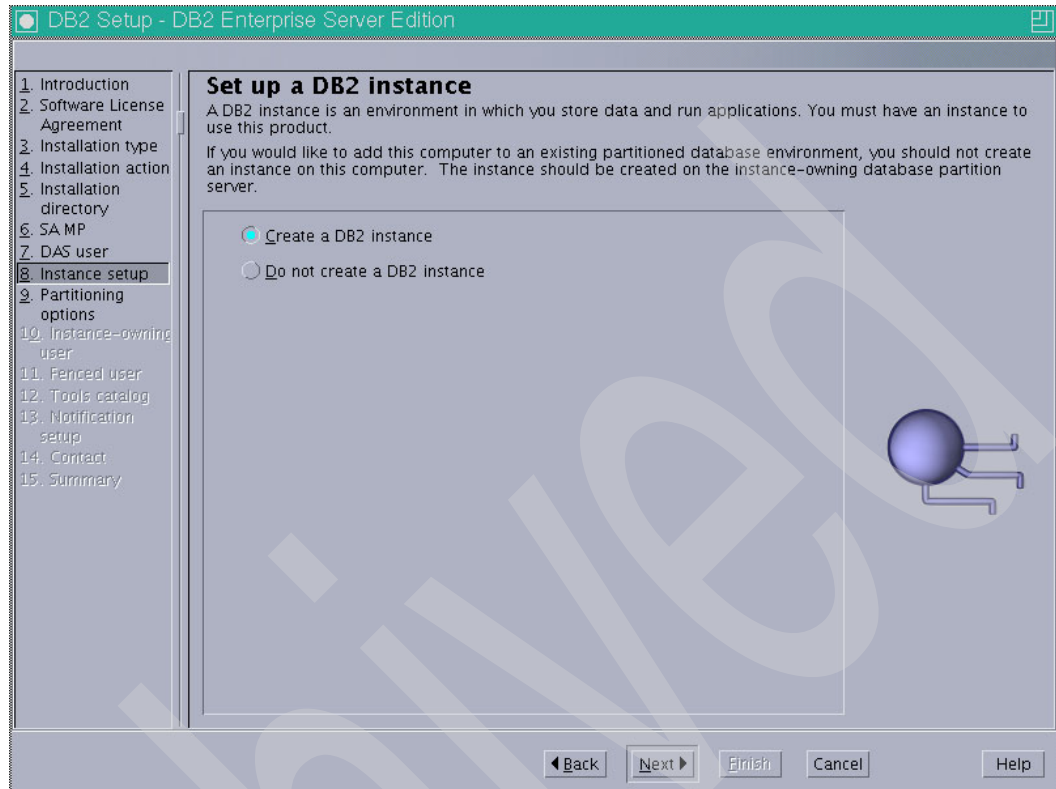


Figure 3-6 DB2 instance setup decision

- In the Set up partitioning options for the DB2 instance panel (Figure 3-7), select **Single partition instance** and click **Next**.

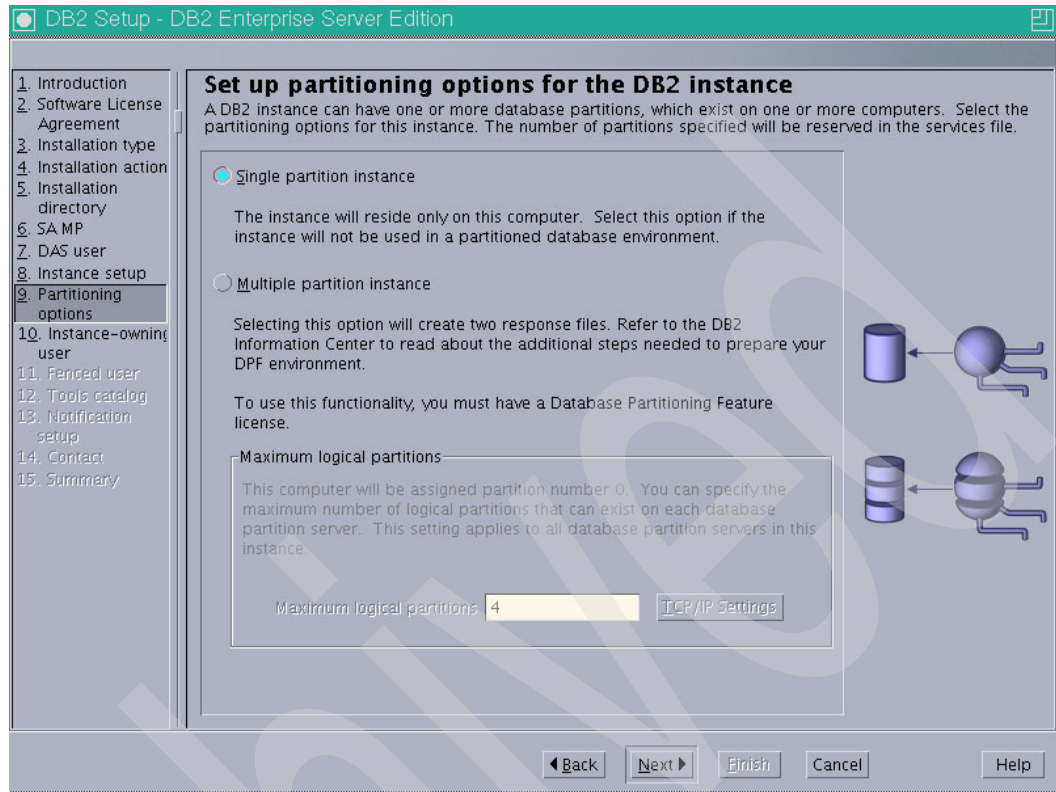


Figure 3-7 Partitioning options for the DB2 instance

10. In the Set user information for the DB2 instance owner panel (Figure 3-8), enter the user ID, group name, and password of the DB2 instance owner, and click **Next**.

The screenshot shows the 'DB2 Setup - DB2 Enterprise Server Edition' window. The main title is 'Set user information for the DB2 instance owner'. Below the title, there is a descriptive text: 'Specify the instance-owning user information for the DB2 instance. DB2 will use this user to perform instance functions, and will store instance information in the user's home directory. The name of the instance will be the same as the user name.'

The 'New user' section is selected with a radio button. It contains the following fields and options:

- User name:
- UID:   Use default UID
- Group name:
- GID:   Use default GID
- Password:
- Confirm password:
- Home directory:  ...

The 'Existing user' section is unselected. It contains a 'User name' field with a dropdown arrow:  ...

At the bottom of the window, there are five buttons: 'Back', 'Next', 'Finish', 'Cancel', and 'Help'.

Figure 3-8 DB2 instance owner definitions

11. Set up the Fenced user and customize the user setting (not shown here). Verify your settings and click **Next**.

Finally you see the Setup Complete window (Figure 3-9).

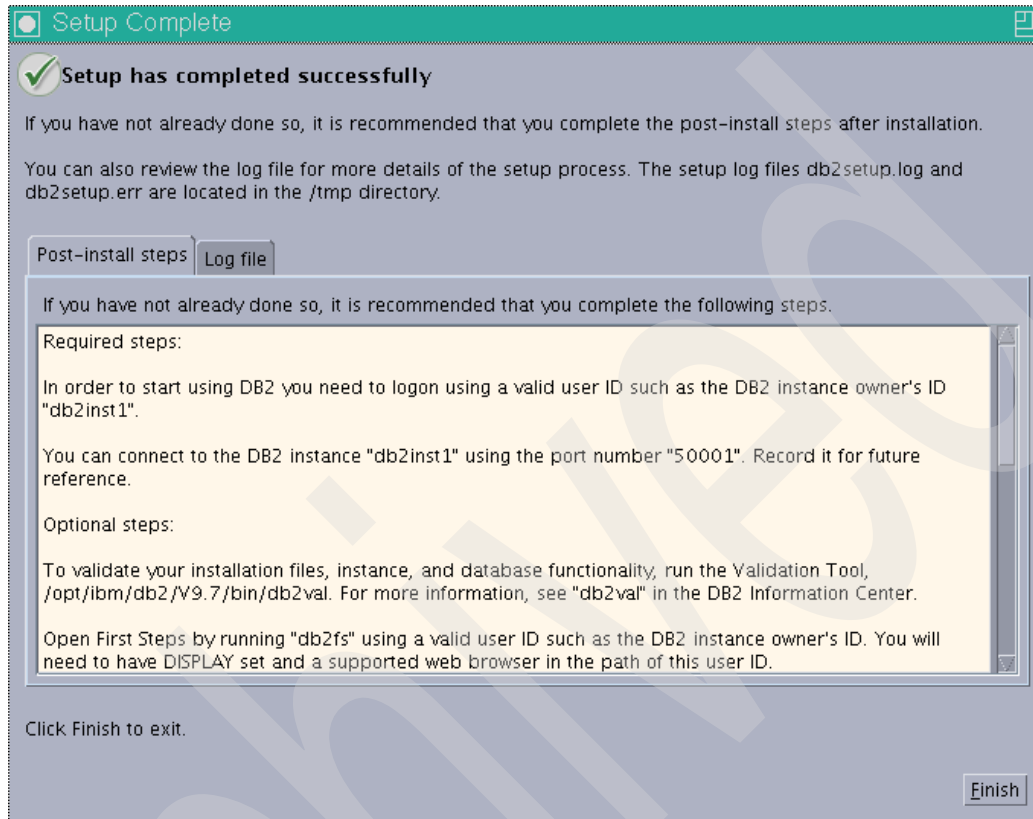


Figure 3-9 Completed setup showing the Post-install steps view

For installations with a root user, DB2 database products are installed in the `/opt/ibm/db2/V9.7` directory by default.

For installations on a system where `/opt/ibm/db2/V9.7` directory is already being used, the DB2 database product installation path adds `_xx` to the directory name, where `xx` are digits, starting at 01. The digits increase depending on how many DB2 copies you have installed. You can also specify your own DB2 database product installation path.

For non-root installations, DB2 database products are always installed in the `$HOME/sqllib` directory, where `$HOME` represents the home directory of the non-root users.

If you have National Language Packs, you can install them by running the `./db2setup` command from the directory where the National Language Pack resides, for example, `./n1pack/db2setup`, after the DB2 database product is installed.

### Verifying the installation

Next you want to verify your installation. You can build the SAMPLE database that comes with DB2 to test your installation.

1. Enter following command as a root user:

```
/opt/ibm/db2/V9.7/bin/db2saml
```

This command creates and populates the SAMPLE database in your installed DB2.



2. Switch to a user that is authorized to work with the database. As an instance owner, user ID `db2inst1` is authorized to connect to the database. Enter the following commands as a root user:

```
su - db2inst1
db2 connect to sample
```

Figure 3-10 shows an example of a successful connection.

```
db2inst1@lnxdb2:~/sql/lib/bin> db2 connect to sample

Database Connection Information

Database server          = DB2/LINUXZ64 9.7.0
SQL authorization ID    = DB2INST1
Local database alias    = SAMPLE
```

Figure 3-10 Successful connection to the *SAMPLE* database

Now, you have successfully verified the DB2 setup with instance `db2inst1`.

You can define multiple instances with different behaviors. In each instance, define databases to work with. In this book, we use the default instance with the name *db2inst1*. In this instance, define a database with the name *BIGLNxDB* that will host the ported DB2 data from the DB2 server on z/VM or on z/VSE.

Part of DB2 for Linux on System z is the IBM DB2 Connect™ functionality, which provides access to remote databases on host systems such as DB2 Server for VSE & VM.

In the following sections, you configure two scenarios. In 3.2.2, “Configuring DB2 for Linux on System z as a server” on page 51, you use DB2 for Linux on System z with your existing z/VSE applications. In 3.2.3, “Configuring DB2 for Linux on System z for access to remote databases” on page 52, you access DB2 data on DB2 Server for VM and DB2 Server for VSE by using DB2 Connect functionality.

### 3.2.2 Configuring DB2 for Linux on System z as a server

Configure the communication protocols on the DB2 server for Linux on System z to accept inbound requests from remote DB2 clients, such as the DB2 application requester in z/VSE or z/VM.

After you install DB2 on Linux, log on with the `db2inst1` user ID and enter the following command:

```
db2 get database manager configuration
```

Look for the variable TCP/IP service name. If the server is configured for the TCP/IP protocol, the output should contain a line similar to the following example:

```
TCP/IP Service name          (SVCENAME) = db2c_db2inst1
```

No additional settings are needed. You can continue with the setup as described in 3.2.3, “Configuring DB2 for Linux on System z for access to remote databases” on page 52.

When you need to configure TCP/IP for a DB2 instance:

- ▶ Ensure that TCP/IP is functional on Linux and on the DB2 client system that initiates a connection.
- ▶ Identify either a connection service name and connection port, or just a connection port.

If you know the port, update the database manager configuration file with the service name parameter (svcname) by entering the following commands:

```
db2 update database manager configuration using svcname 50001
db2stop
db2start
```

### **Configuring a connection service name and connection port**

Update the service name (svcname) parameter in the database manager configuration file at the server. When a connection service name is specified, the services file must be updated with the same service name, a port number, and the protocol. The service name is arbitrary but must be unique within the etc/services file in Linux.

Use the value db2c\_db2inst1 for the service name. If you use DB2 9.7 for Linux, UNIX, and Windows, Enterprise Server Edition, in a partitioned format, ensure that the port number does not conflict with the port numbers that are used by the fast communication manager (FCM). In a partitioned database environment, the FCM handles most communication between database partitions. To enable the FCM at a database partition and allow communication with other database partitions, create a service entry in the partition's services file of the etc directory. The FCM uses the specified port to communicate. In some installations, the entries are generated automatically, which you can verify by looking in the etc/services file.

The connection port number must be unique within the etc/services file. As a value for the port number and protocol, we used 5000/tcp.

Update the database manager by entering the following command:

```
db2 update database manager configuration using svcname db2c_db2inst1
```

### **Configuring a connection port only**

Update the service name parameter (svcname) in the database manager configuration file at the server with a port number. You do not need to update the etc/services file. If you are using DB2 server on Linux in a partitioned format, ensure that the port number does not conflict with the port numbers that are used by the FCM or any other applications on the system. In this example, for the port number, we used the value 50001.

### **Defining the DB2COMM variable**

Set the communication protocol for the current DB2 instance, db2inst1. When you set the database manager to start the connection managers for the TCP/IP communication protocols, enter the following commands:

```
db2set DB2COMM=tcPIP
db2stop
db2start
```

## **3.2.3 Configuring DB2 for Linux on System z for access to remote databases**

Make additional definitions to access a remote DB2 database on z/VM or z/VSE from DB2 on Linux on System z. In this example, we want to access the existing database SQLDS on z/VSE from DB2 on Linux by using the VSEDB2 local name in Linux.

Enter the following commands at a command prompt to catalog the remote database:

1. Define a TCP/IP node to your system where the DB2 VSE database resides:

```
db2 catalog tcpip node <nodename> remote <host name or ipaddr> server <svcname or portnumber> remote_instance <instance-name> ostype <operating-system-type>
```

See the following example:

```
db2 catalog tcpip node vselxnd remote 172.16.0.6 server 446 ostype VSE remote_instance db2inst1
```

2. Catalog the VSE database to your Linux:

```
db2 catalog db <dbname> as <alias> at node <nodename> authentication dcs
```

See the following example:

```
db2 catalog db sqlds as vsedb2 at node vselxnd authentication dcs
```

3. Catalog the VSE database as a host or Database Connection Service (dcs):

```
db2 catalog dcs db <dbname> as <target dbname>
```

See the following example:

```
db2 catalog dcs db vsedb2 as sqlds
```

4. Enter the following commands to activate the new directory changes:

```
db2 terminate  
db2start
```

Test your definitions:

1. Connect to the z/VSE database VSEDB2:

```
db2 connect to vsedb2 user db2admin using myvsepw
```

If the definition is correct, you see the following connection messages:

```
Database Connection Information
```

```
Database server          = SQL/DS VSE 7.5.0  
SQL authorization ID     = DB2ADMIN  
Local database alias     = VSEDB2
```

2. Execute a **bind** to issue select requests. Use the `ddcsvse.lst` bind file from your installation library in Linux (`sqllib/bnd`) for the bind:

```
bind sqllib/bnd/@ddcsvse.lst blocking all sqlerror continue messages vse.msg  
grant public
```

3. Select data from VSEDB2. To test whether you can select data from VSEDB2, type a select statement as follows:

```
select * from SQLDBA.EMPLOYEE
```

Some result rows are displayed.

4. Reset the connection by entering the following command:

```
db2 connect reset
```

### 3.2.4 Defining the rehosting database

Keep in mind the differences between z/VSE and Linux. z/VSE is an EBCDIC system, and Linux uses ASCII. For z/VSE applications, the DB2 data is stored in predefined fixed-length host variables, which is why you must use a single-byte character database in Linux to rehost the data stored in DB2 on z/VSE.

The default format of a database in Linux is UNICODE. Do not use a UNICODE database in Linux as a rehosting database for DB2 on z/VSE because UNICODE data is variable in length and can produce unpredictable errors in applications on z/VSE. Host variables that are used in DB2 applications on z/VSE are fixed length.

Create the rehosting database BIGLNxDB in instance *db2inst1* as single byte character database. To create this database, use a corresponding code set such as in the following example:

```
db2 create database biglnxdb using codeset IS08859-1 territory En_US
DB20000I The CREATE DATABASE command completed successfully.
```

## 3.3 Setting up a federated database environment

DB2 Server for VSE & VM stores the data in an EBCDIC collating sequence, while DB2 server on Linux uses an ASCII collating sequence by default to store the data. Alphanumeric columns that contain numbers and letters sort differently in DB2 on Linux than under DB2 on z/VSE, which affects programs that rely on EBCDIC ordering. For more information about ASCII EBCDIC considerations, see 2.5.2, “ASCII and EBCDIC considerations” on page 37.

We defined an ASCII database BIGLNxDB in Linux on System z and created a federated database with an EBCDIC collating sequence for z/VM and z/VSE applications to solve this issue. This database is called VSELNxDB.

You can set up a federated environment by using the DB2 command interface or the DB2 Control Center graphical interface. Use the DB2 Control Center from a DB2 installation on a workstation. First define all databases from Linux into the DB2 Control Center by using the DB2 Configuration Assistant on the workstation.

In this section, we show how to set up a federated DB2 database environment on Linux on System z. In Chapter 4, “DB2 data migration and application dependencies” on page 105, we provide the customizations to migrate the data from DB2 VSE to DB2 Linux.

Figure 3-11 shows the federated environment that we used for z/VSE applications.

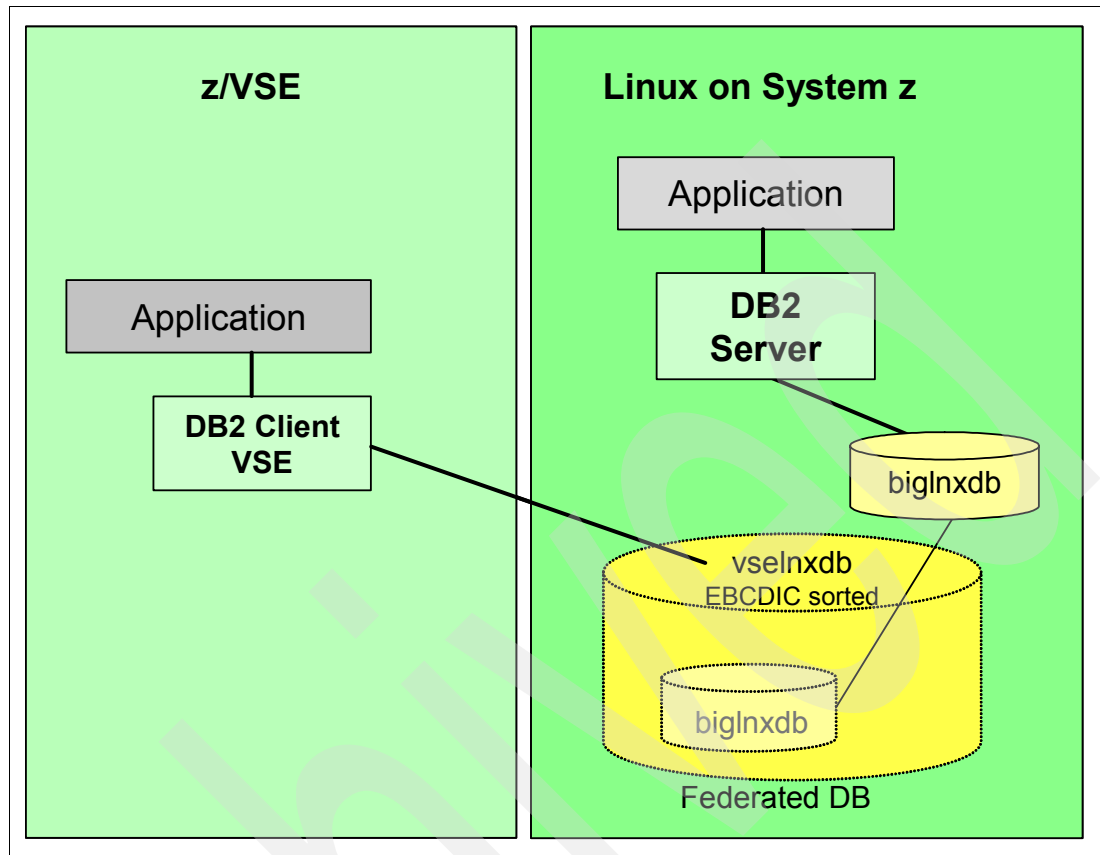


Figure 3-11 Federated access for EBCDIC considerations

### 3.3.1 Setting up a federated database

To create a federated database with a user-defined EBCDIC collating sequence, we use the sample program `db_udcs.c` to create the VSELNXDB database with an EBCDIC International Coded Character Set Identifier (CCSID) 500 standard collating sequence. You can find this program in “The `db_udcs.c` program” on page 180.

To set up the federated database:

1. Use the `db_udcs.c` sample program to create the VSELNXDB database in DB2 on Linux with the EBCDIC International CCSID 500 collating sequence.

You can use one of the ASCII-EBCDIC translation tables defined in the `include` directory of your installation folder as header files. The `db_udcs.c` program, as shown in “The `db_udcs.c` program” on page 180, uses the `sql819a.h` file. This header file is also available in “The `sql819a.h` header file” on page 182.

Figure 3-12 shows how your database tree should look if you use the DB2 Control Center.

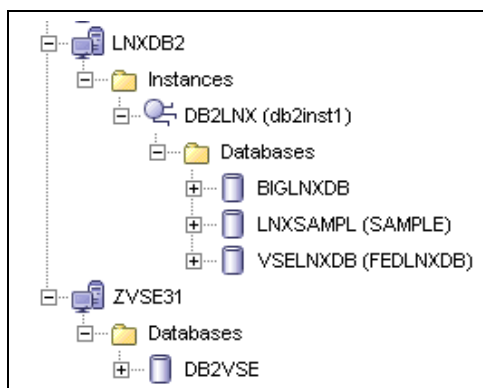


Figure 3-12 Database tree

2. Update the DB2 database manager configuration for federated database support:

db2 update dbm cfg using federated yes

If you use the DB2 Control Center:

- a. Right-click the **db2inst1** instance and select **Configure Parameters**.
- b. In the DBM Configuration window (Figure 3-13), set the **FEDERATED** parameter to **yes**.

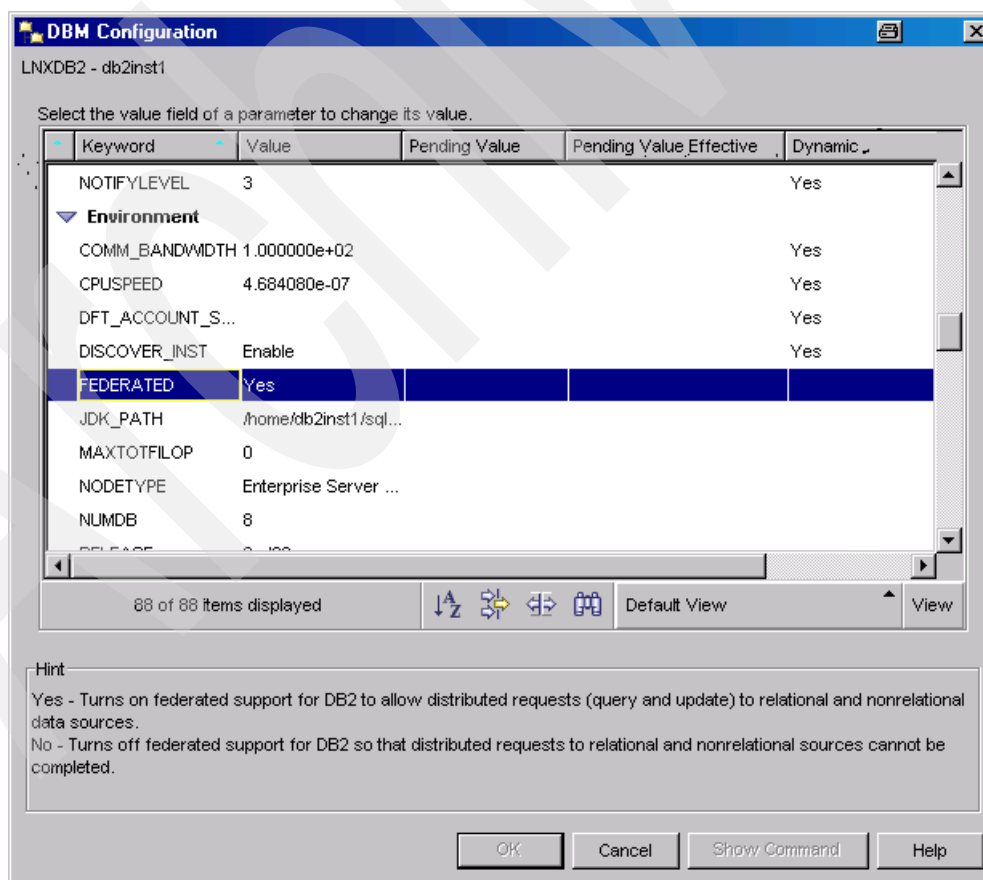


Figure 3-13 Database Manager parameter setup in DB2 Control Center

3. Create a DRDA wrapper for federated access:

```
db2 connect to vselnxdb
db2 create wrapper DRDA
```

If you use the DB2 Control Center:

- In the expanded tree for VSELNXDB (Figure 3-12 on page 56), right-click **DB2 Federated database objects** and select **Create Wrapper**.
- In the Create Wrapper window (Figure 3-14), on the Wrapper page, for Data source, select **DB2**, and for Wrapper name, type DRDA.

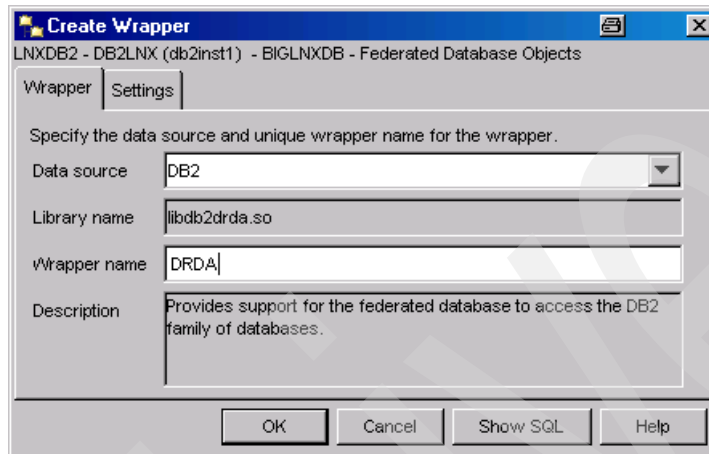


Figure 3-14 Create Wrapper window

4. Create the database server for BIGLNxDB that will be accessed with VSELNXDB as a federated database:

```
db2 create server biglnxdb type db2/luw version '9.7' wrapper drda authid
"db2inst1" password "mylinpw" options( add dbname 'BIGLNxDB', password 'Y')
```

If you use DB2 Control Center:

- In the expanded DRDA tree, right-click **Server Definitions** and select **Create**.
- Select **Discover** to view all the databases that are cataloged in the database directory in DB2 on Linux.

- c. In the Create Server Definitions window (Figure 3-15), select **BIGLNxDB**, and clear the other names. For BIGLNxDB, under Type, enter DB2/UDb, and for Version, enter 9.7. Click **Properties**.

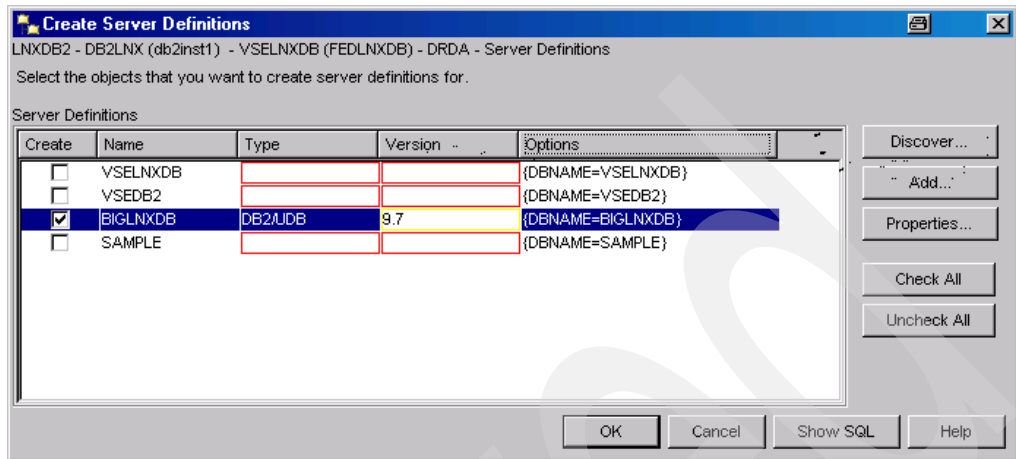


Figure 3-15 Create Server Definitions window

- d. In the Server Definition Properties window (Figure 3-16):
  - i. On the Server Definition page, for User ID, type db2inst1, and for Password, type my1inpw to access the BIGLNxDB database.

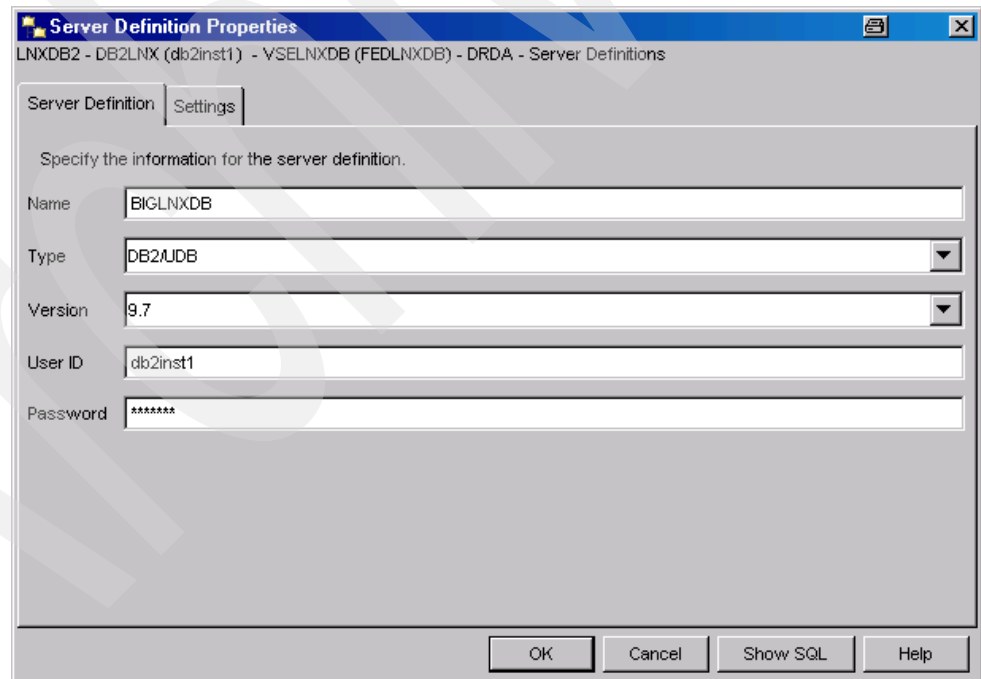


Figure 3-16 Server Definition Properties window



- ii. Click the **Settings** tab (Figure 3-17). Select **PASSWORD** and select option **Y**.

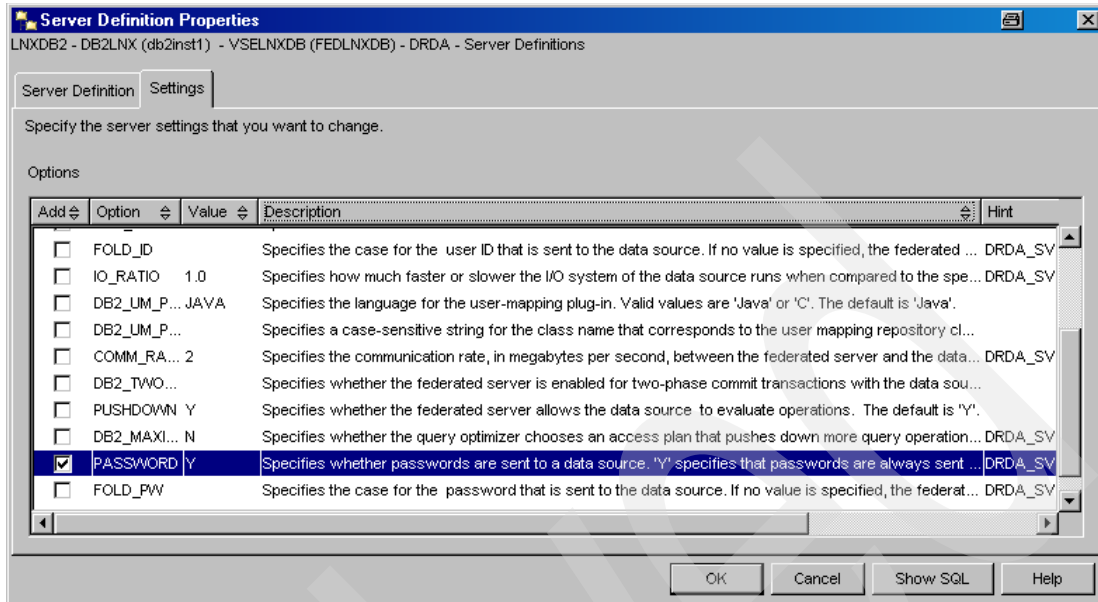


Figure 3-17 Server definition properties for password

- iii. Click **OK**.
5. Create a user mapping that maps your DB2 VSELNXDB database user ID to a user ID and password on the BIGLNxDB database.

Type the following **db2** command to define the user mapping:

```
db2 create user mapping for "db2inst1" server "biglnxdb" options ( add remote_authid 'db2inst1', add remote_password 'mylinpw')
```

If you use the DB2 Control Center:

- a. Right-click **User Mappings** and select **Create**.
- b. In the Create User Mappings window (Figure 3-18), on the Users page, select the user ID you want to map to the remote user ID.

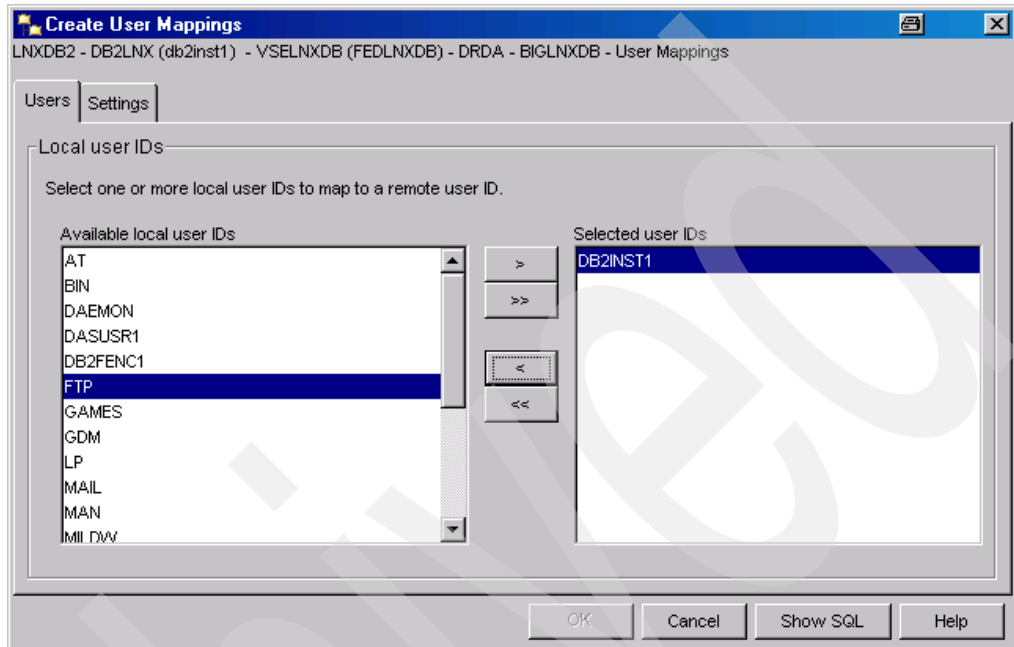


Figure 3-18 Creating user mappings

- c. Click the **Settings** tab (Figure 3-19) and enter the remote user and password. In our case, they are db2inst1 and mylinpw.

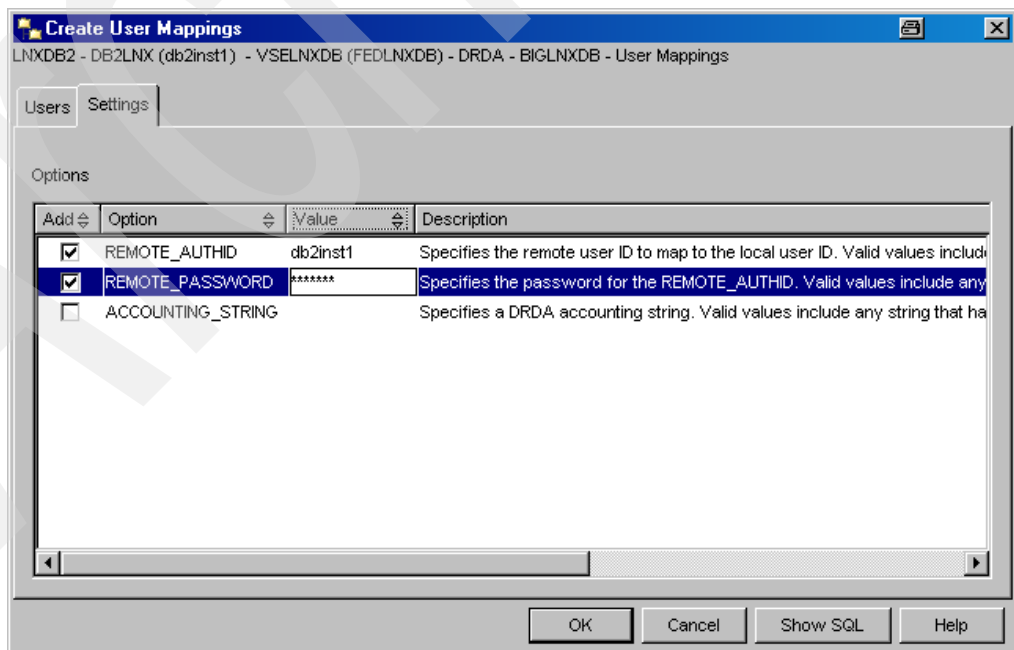


Figure 3-19 Creating user mappings for remote authentication

- iv. Click **OK**.

6. Test the federated access from VSELNXDB database to the BIGLNxDB database server. You can do the test by using the DB2 special mode called *passthru*.

a. Connect to the VSELNXDB database. For this test, we created the wptest table in BIGLNxDB. Do a first select and you see a message similar to the following example:

```
SQL0204N "DB2INST1.WMTEST" is an undefined name.
```

b. Set passthru to the BIGLNxDB database and select a table that is only available on BIGLNxDB, such as wptest. Use the following db2 commands:

```
db2 connect to vselnxdb
db2 select * from wptest
db2 set passthru biglnxdb
db2 select * from wptest
db2 set passthru reset
db2 connect reset
```

Then you see the result data.

c. To work transparently with the BIGLNxDB tables in VSELNXDB, define table nicknames for all tables of BIGLNxDB. You can define them with commands such as the following examples:

```
create nickname db2inst1.employee for biglnxdb.db2inst1.employee;
create nickname db2inst1.project for biglnxdb.db2inst1.project;
create nickname db2inst1.wptestfed for biglnxdb.db2inst1.wptest;
```

You can also define them with the DB2 Control Center:

- i. In the expanded tree of the federated database objects, right-click **Nicknames** and select **Create**.
- ii. In the Create Nicknames window (Figure 3-20), click **Discover** to see all remote tables. Select the nicknames that you want to create and adjust their names.

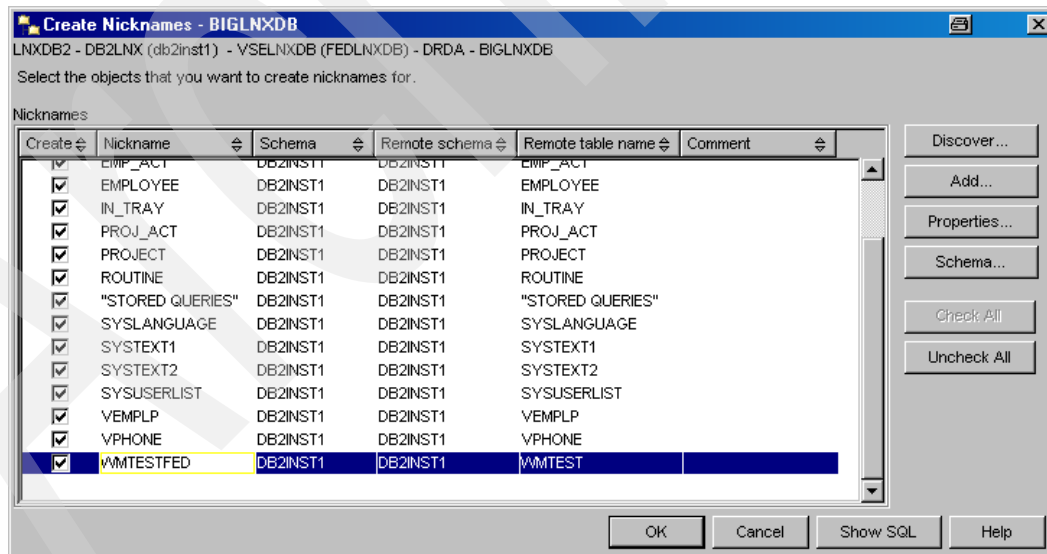


Figure 3-20 Create Nicknames window

iii. Click **OK**. The nicknames for these tables are created.

Now you can directly access all the tables in BIGLNxDB from VSELNXDB and the WMTEST table as WMTESTFED.

Figure 3-21 shows how the database tree looks in the DB2 Control Center after performing this setup.

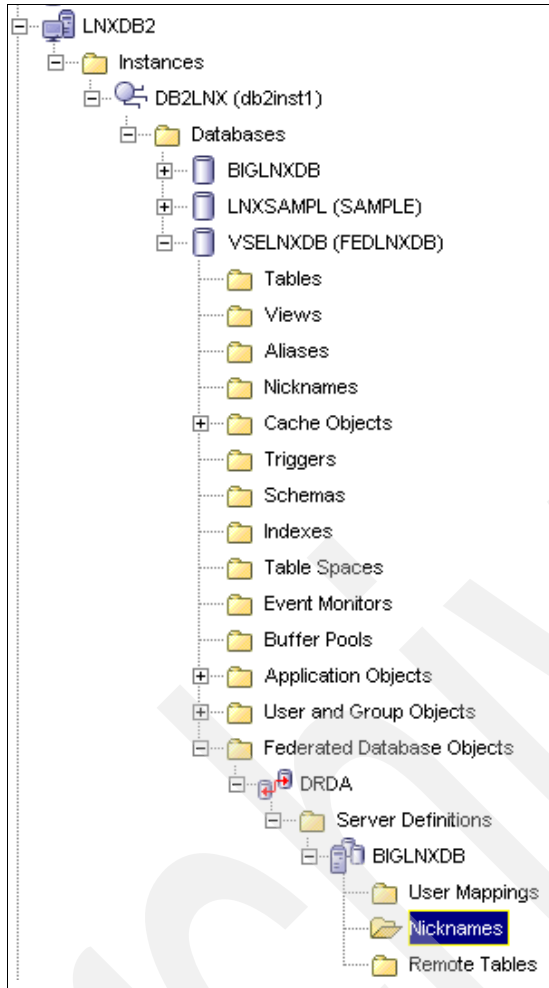


Figure 3-21 Database tree for federated database VSELNXDB and database BIGLNKDB

### 3.4 Setting up the network

For the network connection between z/VSE and Linux on System z, you can choose one of the following options:

- ▶ HiperSockets  
This is the preferred connection.
- ▶ Shared OSA  
Use this option in installations with small z/VSE CPU capacity.
- ▶ VSWITCH  
Use this option with z/VM V5R4 and later versions for a secure, flexible internal network.

Figure 2-4 on page 30 shows an overview of the network options that you can use.

In this book, we describe the HiperSockets connections between z/VSE and Linux on System z. In addition, we show connections from Linux and z/VSE through z/VM to the external local area network (LAN).

For the network parameter overview, see Figure 3-1 on page 41.

### 3.4.1 An overview of HiperSockets

HiperSockets is a Licensed Internal Code (LIC) function that emulates the Logical Link Control (LLC) layer of an OSA-Express Queued Direct Input/Output (QDIO) interface. It provides high-speed TCP/IP connectivity between virtual servers that run within different LPARs on a System z server. HiperSockets eliminates the need for any physical cabling or external networking connections between these virtual servers.

In a HiperSockets network, the system copies data synchronously from the output queue of the sending TCP/IP device to the input queue of the receiving TCP/IP device. It uses the memory bus to copy the data through an I/O instruction. The HiperSockets network copies with the speed of the CPU, meaning that a lower CPU capacity influences the speed of HiperSockets communication. This is not the case with shared OSA, because OSA works with its own clock speed, independent of the CPU speed, and it works asynchronously.

Another difference between a HiperSockets and a shared OSA is the lack of a built-in bridge. A HiperSockets network only works within a single System z server and needs other adapters to connect externally to other servers.

The following operating systems support HiperSockets:

- ▶ z/VSE
- ▶ Linux on System z
- ▶ z/VM
- ▶ z/OS

### 3.4.2 Configuring the z/VSE network

In this section, we show you the z/VSE network configuration for HiperSockets, shared OSA, and Ethernet networks. You can have multiple concurrent IP addresses operating on the same system and the same IP stack. In this case, use DEFINE MASK to specify which path to use depending on the case.

We used the HIPSO link definition for HiperSockets communication from z/VSE to Linux on System z. With the OSAFE link definition, we used the OSA interface to connect z/VSE to the external Ethernet network. The addresses 500 and 501, as well as data path 502, are used for HiperSockets communication. D00 and D01, as well as data path D02, are used for communications with the external Ethernet. Example 3-1 shows the related entries in member IPINIT00.L.

*Example 3-1 HiperSockets and OSA definitions for z/VSE*

```
*-----*
*      DEFINE THE MAIN COMMUNICATION IP ADDRESS      *
*-----*
SET IPADDR = 9.152.86.244
SET MASK = 255.255.252.000
*-----*
*      DEFINE THE COMMUNICATION LINKS      *
*-----*
```

```

*          Define Link for Hipersockets or VSWITCH Communication          *
*-----*
DEFINE LINK,ID=HIPSO,TYPE=OSAX,DEV=(0500,0501),DATAPATH=0502, -
          IPADDR=172.16.0.6
*-----*
*          Define Link for shared OSA or internet communication          *
*-----*
DEFINE LINK,ID=OSAFE,TYPE=OSAX,DEV=(0D00,0D01),DATAPATH=0D02,MTU=1492
*-----*
*          DEFINE ROUTINE INFORMATION          *
*-----*
*          Define Route for Hipersockets or VSWITCH Communication          *
*-----*
DEFINE ROUTE,ID=TOLNX,LINKID=HIPSO,IPADDR=172.16.0.0
DEFINE MASK,ID=MHIPSO,NETWORK=172.16.0.0,MASK=255.255.255.0
*-----*
*          Define Route for shared OSA or internet communications          *
*-----*
DEFINE ROUTE,ID=ALL,LINKID=OSAFE,IPADDR=0.0.0.0, -
          GATEWAY=9.152.86.245
*-----*

```

For the complete IPINIT00.L member that we used, see “The IPINIT00.L TCP/IP configuration member in z/VSE” on page 170.

### 3.4.3 Configuring the Linux network

Use the YaST2 tool to define the network in Linux. Keep in mind that you can only define a network if addresses for that network are assigned to Linux.

To view the possible Linux networks:

1. Start the YaST Control Center by entering the **yast2** command.
2. In the YaST Control Center window (Figure 3-22), click **Network Devices**, and select **Network Card**.

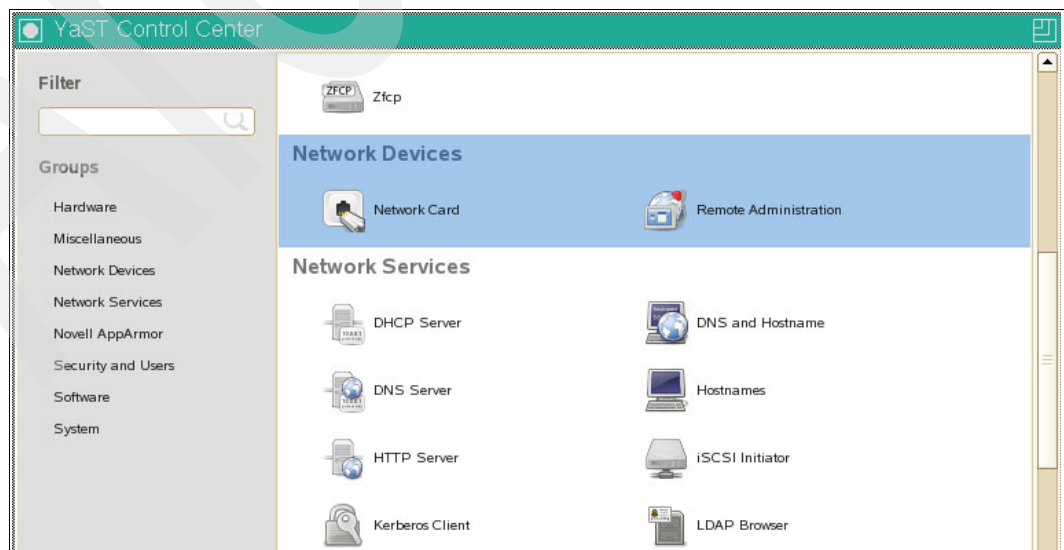


Figure 3-22 Network setup in the YaST Control Center

3. In the Network Setup Method panel (Figure 3-23), select **Traditional Method with ifup**. Then click **Next**.

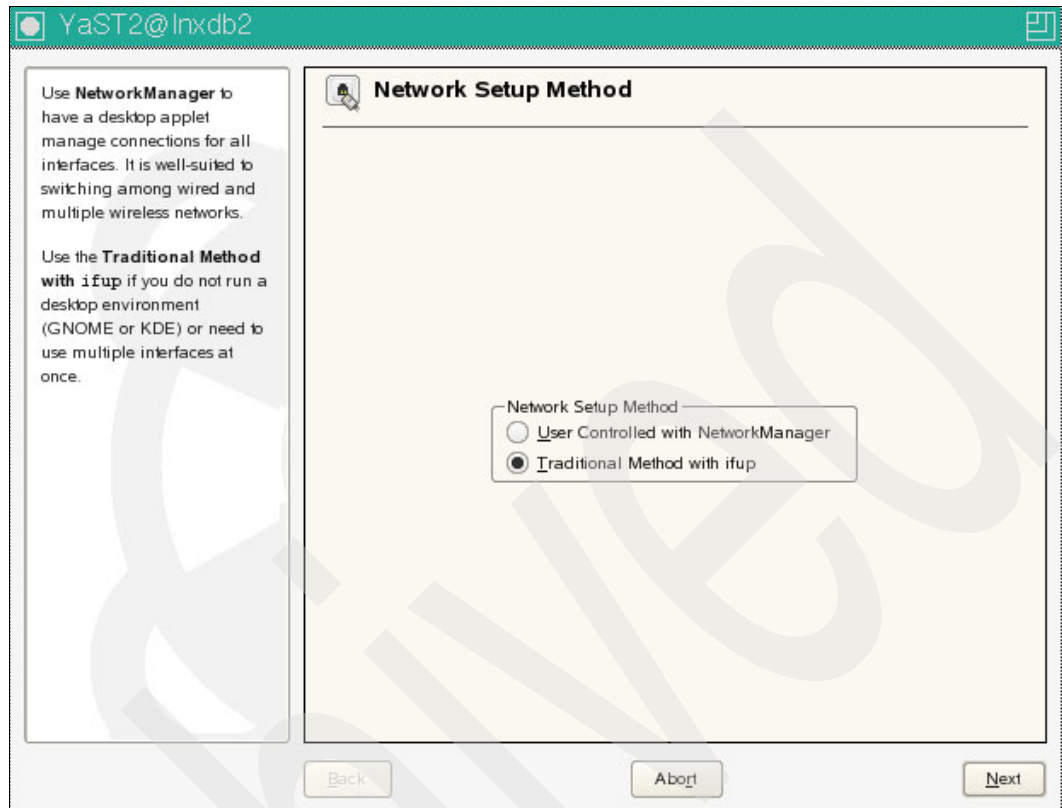


Figure 3-23 YaST2 Network Setup Method panel

Figure 3-24 shows the possible networks for the Linux environment.

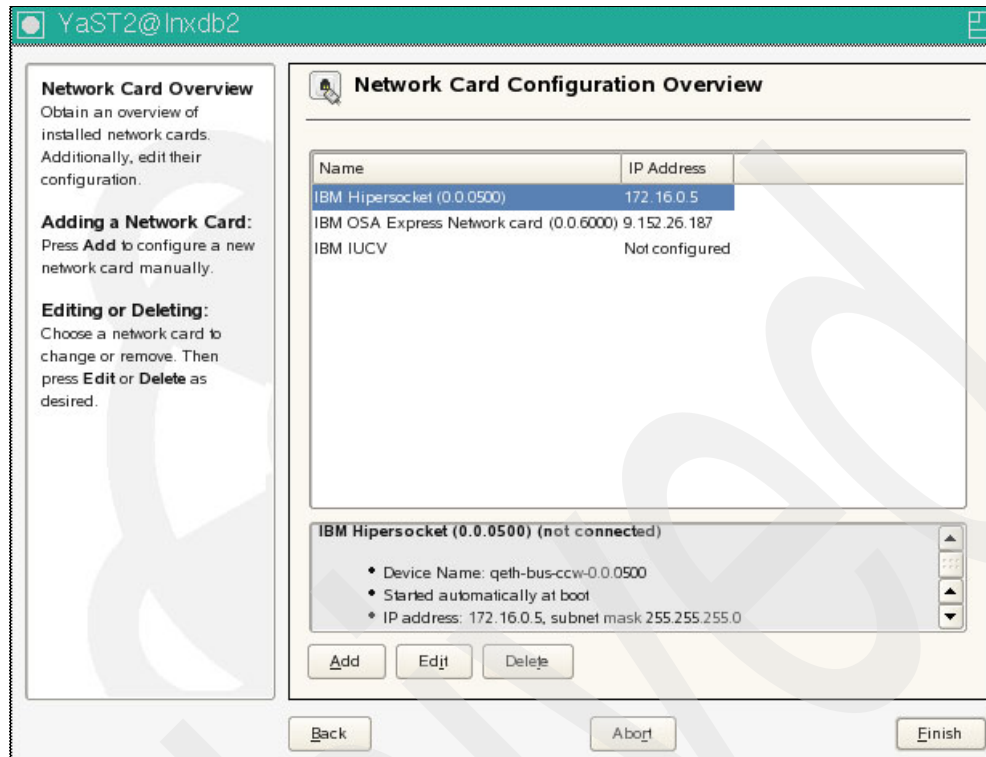


Figure 3-24 Network Card Configuration Overview panel

Now you can start to configure the HiperSockets and OSA networks.

## Configuring HiperSockets

To configure the HiperSockets network for connection to z/VSE:

1. From the Network Card Configuration Overview panel (Figure 3-24), select **IBM HiperSocket**.
2. In the Network Address Setup window (Figure 3-25 on page 67), under Detailed Settings, configure the host name, default gateway, and network card.
  - a. Select **Hostname and Name Server**.



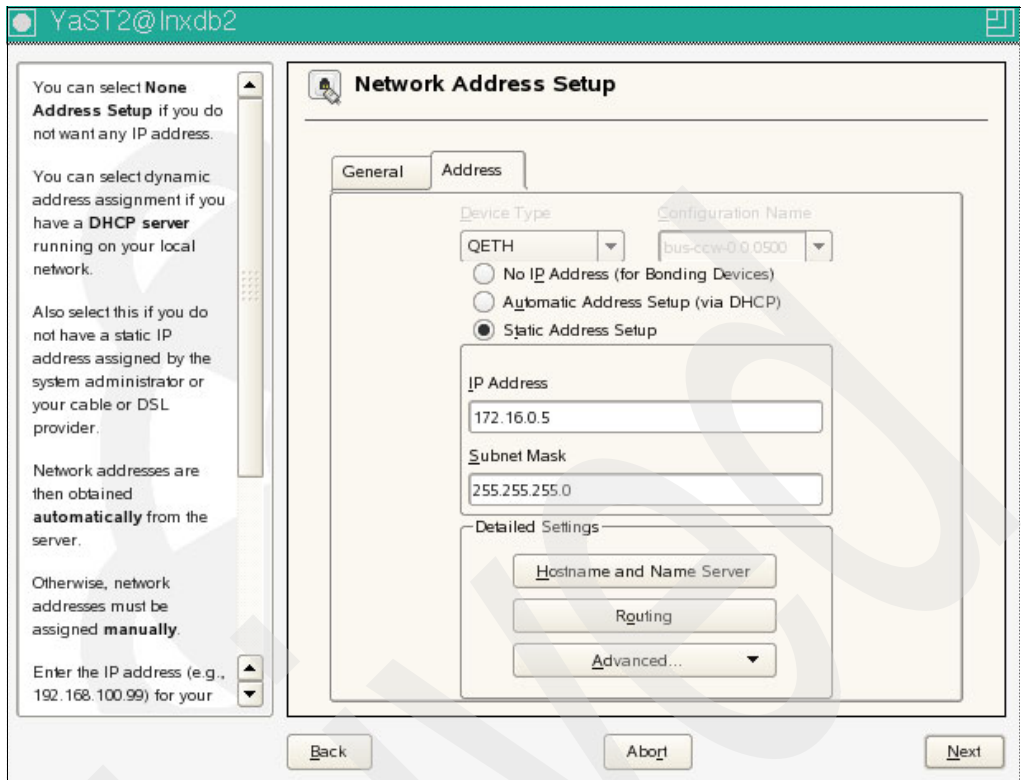


Figure 3-25 Network Address Setup panel

- b. In the Hostname and Name Server Configuration window (Figure 3-26), enter the host name and name server settings. Then click **OK**.

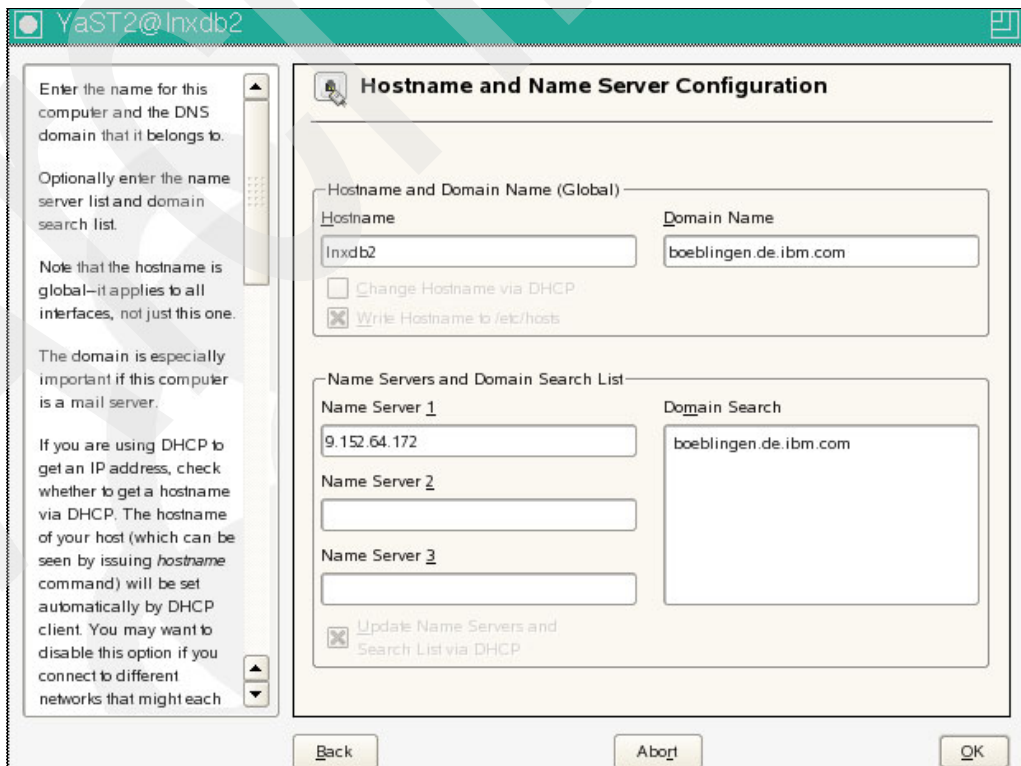


Figure 3-26 Hostname and Name Server Configuration panel

- c. After you return to the Network Address Setup panel (Figure 3-25 on page 67), select **Advanced** and **Hardware Details**.
- d. In the Manual Network Card Configuration panel (Figure 3-27), specify the hardware configuration name for the HiperSockets configuration. Then click **OK**.

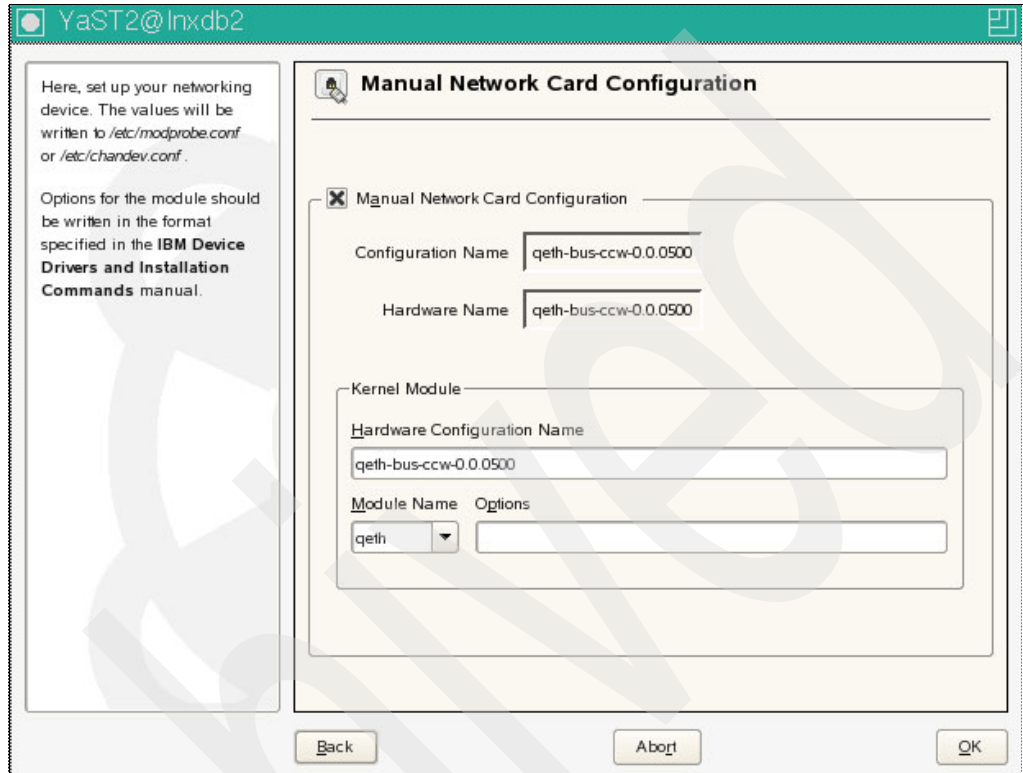


Figure 3-27 Manual Network Card Configuration panel

- e. When you return to the Network Address Setup panel (Figure 3-25 on page 67), select **Advanced** and **S/390**.

- f. In the S/390 Network Card Configuration window (Figure 3-28), specify the port name and channel IDs. Then click **Next**.

YaST2@lnxdb2

**S/390 Network Card Configuration**

S/390 Device Settings

Port Name

Options

Enable IPA Takeover

Enable Layer 2 Support

Layer2 MAC Address

00:00:00:00:00:00

CHAN IDS

0.0.0500 0.0.0501 0.0.0502

Back Abort Next

Figure 3-28 S/390 Network Card Configuration panel

You have now completed the HiperSockets network configuration. For more details about HiperSockets for Linux, see Chapter 5, “Linux support”, in *HiperSockets Implementation Guide*, SG24-6816.

## Configuring the network with the OSA card

To configure the external connection through OSA:

1. In the Network Card Configuration Overview panel (Figure 3-29), select the **IBM OSA Express Network card** interface and click **Edit**.

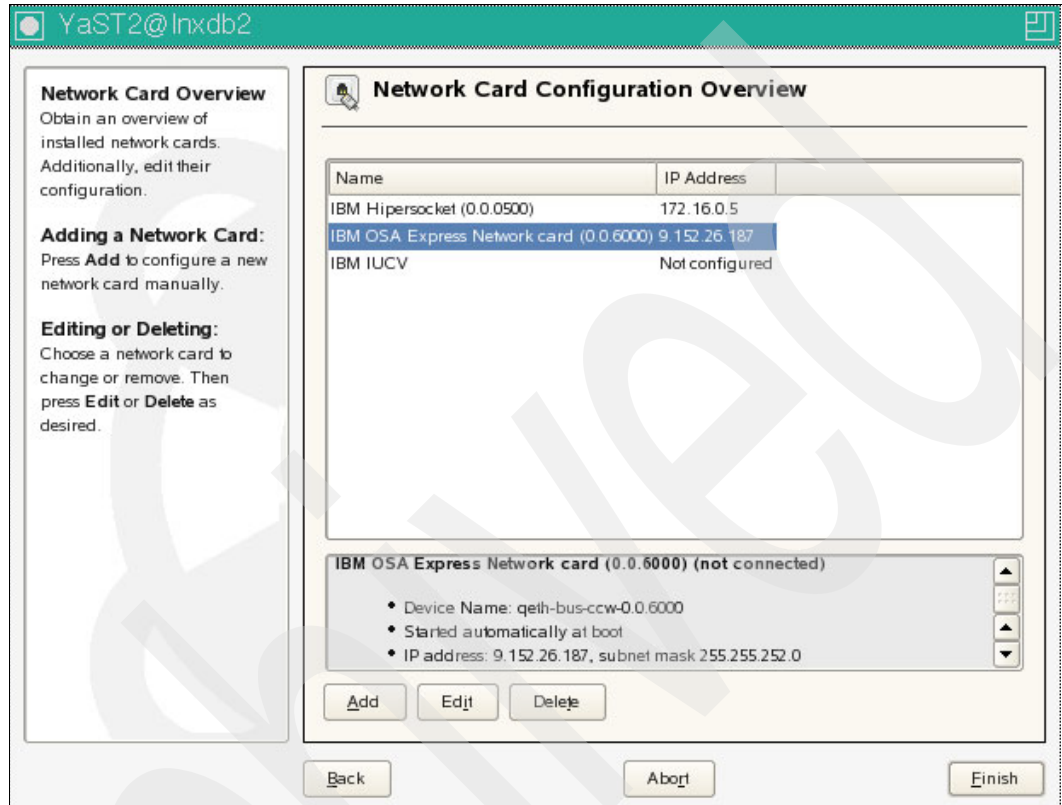


Figure 3-29 Network Card Configuration Overview panel

2. In the Network Address Setup panel (Figure 3-30):
  - a. Enter the IP address and mask for OSA.
  - b. Under Detailed Settings, select **Hostname and Name Server**.

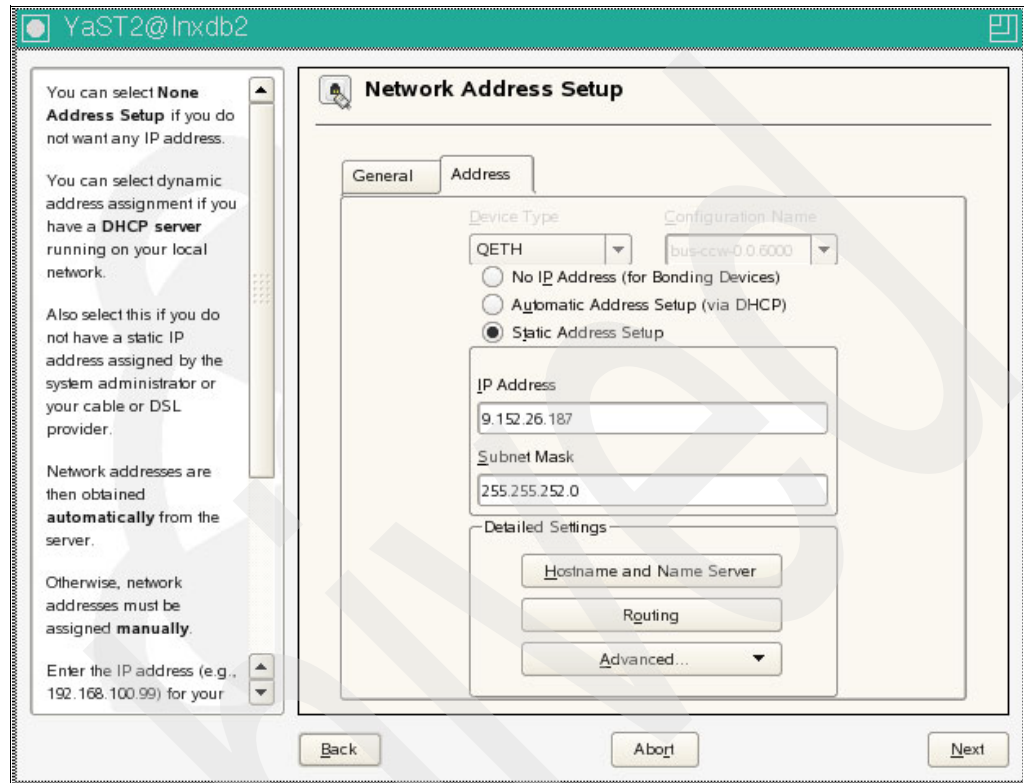


Figure 3-30 Network Address Setup panel for OSA connections

- c. In the Hostname and Name Server Configuration panel (Figure 3-31), after the setup, click **OK**.

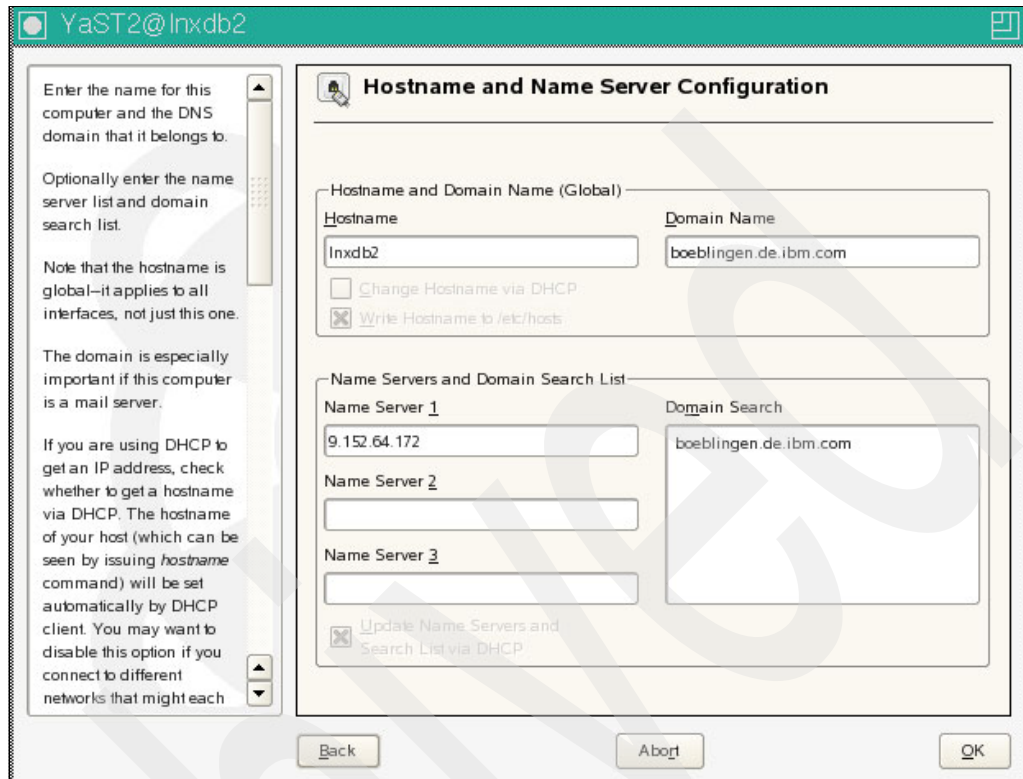


Figure 3-31 Hostname and Name Server Configuration panel

- d. When you return to the Network Address Setup panel (Figure 3-30 on page 71), click **OK** to finish your configuration.

With this configuration, you can communicate internally with z/VSE and access the external network through the OSA card.

### 3.4.4 HiperSockets specifics

If you are using a HiperSockets network between your z/VSE system and DB2 running on Linux on System z, pay attention to the MTU size. For HiperSockets, the MTU size is defined in the I/O subsystem configurations (IOCDs). You can choose between 16 KB, 24 KB, 32 KB, and 64 KB.

The frame size, that is the MTU plus the link layer headers, and MTU size are determined by `chparm` parameter of the IOCDs. The MTU size can be determined from a given frame size, such as in the following example:

$$\text{MTU size} = \text{frame size} - 8 \text{ KB}$$

Select the MTU size to suit the workload. If the application is mostly sending packets smaller than 8 KB, an MTU size of 8 KB is sufficient. If the application is capable of sending big packets, a larger MTU size increases throughput and saves CPU cycles. Use an MTU size of 56 KB only for streaming workloads with packets greater than 32 KB. HiperSockets does not require checksumming because it is a memory-to-memory operation. The default is `sw_checksumming`.

To save CPU cycles, switch off checksumming as follows:

- ▶ For SUSE Linux Enterprise Server 10, in the `/etc/sysconfig/hardware/hwcfg-qeth-bus-ccw-0.0.F200` file, add the following line:  
`QETH_OPTIONS="checksumming=no_checksumming"`
- ▶ For SUSE Linux Enterprise Server 11, in the `/etc/udev/rules.d/51-qeth-0.0.f200.rules` file, add the following line:  
`ACTION=="add", SUBSYSTEM=="ccwgroup", KERNEL=="0.0.f200", ATTR{checksumming}="no_checksumming"`
- ▶ For Red Hat, in the `/etc/sysconfig/network-scripts/ifcfg-eth0` file, add the following line:  
`OPTIONS="checksumming=no_checksumming"`

The setting of `rmem/wmem` under Linux on System z determines the minimum, default, and maximum window size. Linux window size settings are system wide and apply to all network devices. Applications can use `setsockopt` to adjust the window size individually, which has no impact on other network devices. HiperSockets and OSA devices have contradictory demands. More than 10 parallel OSA sessions suffer from a send window size greater than 32 KB. The suggested default size of the send or receive window for HiperSockets MTU 8 KB and 16 KB is also adequate for OSA devices. As a rule of thumb, the default send window size must be twice the MTU size, as in the following examples:

- ▶ MTU 8KB  
`sysctl -w net.ipv4.tcp_wmem="4096 16384 131072"`  
`sysctl -w net.ipv4.tcp_rmem="4096 87380 174760"`
- ▶ MTU 16KB  
`sysctl -w net.ipv4.tcp_wmem="4096 32768 131072"`  
`sysctl -w net.ipv4.tcp_rmem="4096 87380 174760"`
- ▶ MTU 32KB  
`sysctl -w net.ipv4.tcp_wmem="4096 65536 131072"`  
`sysctl -w net.ipv4.tcp_rmem="4096 87380 174760"`
- ▶ MTU 56KB  
`sysctl -w net.ipv4.tcp_wmem="4096 131072 131072"`  
`sysctl -w net.ipv4.tcp_rmem="4096 131072 174760"`

### 3.4.5 Network specifics in Linux distributions

To increase network performance, you can specify a buffer count up to 128. The default inbound buffer count is 16. You can check the actual buffer count by using the `lsqeth -p` command. Networking tests showed that the default buffer count of 16 limits the throughput of a HiperSockets connection with 10 parallel sessions. A buffer count of 128 leads to 8 MB of memory consumption. One buffer consists of sixteen 4 KB pages, which yields 64 KB as shown by the following equation:

$$128 \times 64 \text{ KB} = 8 \text{ MB}$$

The inbound buffer count can be set in the appropriate configuration file as follows:

- ▶ For SUSE Linux Enterprise Server 10, in the `/etc/sysconfig/hardware/hwcfg-qeth-bus-ccw-0.0.F200` file, add the following line:  
`QETH_OPTIONS="buffer_count=128"`
- ▶ For SUSE Linux Enterprise Server 11, in the `/etc/udev/rules.d/51-qeth-0.0.f200.rules` file, add the following line:  
`ACTION=="add", SUBSYSTEM=="ccwgroup", KERNEL=="0.0.f200", ATTR{buffer_count}="128"`
- ▶ For Red Hat, in the `/etc/sysconfig/network-scripts/ifcfg-eth0` file, add the following line:  
`OPTIONS="buffer_count=128"`

If you share an OSA Express adapter in QDIO mode among several LPARs, switch on priority queuing as follows:

- ▶ For SUSE Linux Enterprise Server 10, in the `/etc/sysconfig/hardware/hwcfg-qeth-bus-ccw-0.0.F200` file, add the following line:  
`QETH_OPTIONS="priority_queueing=no_prio_queueing:0"`
- ▶ For SUSE Linux Enterprise Server 11, in the `/etc/udev/rules.d/51-qeth-0.0.f200.rules` file, add the following line:  
`ACTION=="add", SUBSYSTEM=="ccwgroup", KERNEL=="0.0.f200", ATTR{priority_queueing}="no_prio_queueing:0"`
- ▶ For Red Hat, in the `/etc/sysconfig/network-scripts/ifcfg-eth0` file, add the following line:  
`OPTIONS="priority_queueing=no_prio_queueing:0"`

**Note:** Priority queuing on one LPAR can impact the performance on all other LPARs that share the same OSA card.

Carefully choose the MTU size. Set it to the maximum size supported by all hops on the path to the final destination to avoid fragmentation. You can check the MTU size by using the `tracpath destination` command. If the application sends chunks less than or equal to 1400 bytes, use MTU 1492, which can handle up to 1400 bytes of user data, plus protocol overhead. If the application can send bigger chunks, use MTU 8992. Sending packets larger than 1400 bytes with MTU 8992 will increase throughput and save CPU cycles. TCP uses the MTU size for the window size calculation, not the actual application send size. Use MTU 8992 for VSWITCH. Networking by using VSWITCH is a synchronous operation that requires SIGA for every packet. No packing mechanism exists as it does for OSA cards.

You can change the following system-wide `sysctl` settings temporarily by using the `sysctl` command or permanently in the appropriate `/etc/sysctl.conf` configuration file.

You can increase the device queue from the default of 1000 to at least 2500 by using `sysctl` as follows:

```
sysctl -w net.core.netdev_max_backlog = 2500
```

You can adapt the inbound and outbound window size to suit the workload. Use the following values for OSA devices:

```
sysctl -w net.ipv4.tcp_wmem="4096 16384 131072  
sysctl -w net.ipv4.tcp_rmem="4096 87380 174760
```



The system-wide window size applies to all network devices. Applications can use the **setsockopt** command to adjust the window size, which has no impact on other network devices. Only more than 10 parallel sessions benefit from default and maximum window sizes, as specified previously. A big window size can be advantageous for up to five parallel sessions.

By default, the system starts numerous daemons, which may or may not be necessary. Check whether **selinux** and **auditing** are required for the Linux system. If not, switch off these settings.

For example, you can use following commands to find **selinux** and **auditd**:

```
dmesg | grep -i "selinux"  
chkconfig -list | grep -i "auditd"
```

In the `/etc/zipl.conf` file, specify the following settings:

```
parameters="audit_enable=0 audit=0 audit_debug=0 selinux=0 ....."  
chkconfig --set auditd off
```

Check which daemons are running by entering the following command:

```
chkconfig --list
```

The following daemons not required under Linux on System z:

- ▶ alsasound
- ▶ avahi-daemon
- ▶ resmgr
- ▶ postfix

A controlled environment without an external network connection usually does not require a firewall. The Linux firewall (**iptables**) serializes the network traffic because it can use just one CPU. By default, this firewall is enabled in Novell SUSE. If there is no business need, switch off the firewall as follows:

```
chkconfig -set SuSEfirewall2_init off  
chkconfig -set SuSEfirewall2_setup off  
chkconfig -set iptables off
```

## 3.5 Setting up DB2 Server for VSE Client Edition

Before you can install DB2 Server for VSE Client Edition, prepare the environment. In this section, we include the steps for preparing z/VSE, CICS, and DB2.

### 3.5.1 Preparing z/VSE

To prepare your z/VSE system for the installation and execution of the DB2 workload:

1. Assign to z/VSE a tape drive that is related to the type of distribution tape that you ordered.
2. Ensure that the SYS001 work file size is at least one cylinder on an IBM 3390 DASD or 960 blocks on an FBA device.

During the installation process, input is read from **SYSIPT** and source members from the DB2 Server for VSE Client Edition sublibrary. Output is written to the printer file, **SYSLST**, and the **SYS001** work file.

3. Ensure that your partitions have additional storage, which is required for the installation of DB2 Server for VSE Client Edition. The installation requires the following additional storage in your partition:
  - 2 MB to compile and run application programs
  - 6 MB to support DRDA batch requester
  - 8 MB for the CICS system to support ISQL

For detailed information about capacity planning for storage requirements, see *DB2 Server for VSE System Administration*, SC09-2981.

### **DASD requirements for the distribution library**

The approximate space requirement is 30,000 library blocks, with 1 KB each. Therefore, on a 3390 disk, you need about 70 cylinders.

### **Restoring the DB2 Server for VSE Client Edition distribution library**

The DB2 Server for VSE Client Edition distribution library comes either as a separate DB2 tape or as part of z/VSE Extended Base Tape. If you obtain DB2 as a separate tape, DB2 Server for VSE Client Edition is shipped on a non-stacked V2 format tape.

Use the VSE Interactive Interface, product installation dialogs in z/VSE, fastpath 111 for preparation, and fastpath 112 for installation.

For more information about installation on z/VSE, see *z/VSE Installation*, SC33-8302.

### **Licensing**

Your DB2 product has a 90-day trial license with a key, which you activate at installation time. Enable the key in USERBG. Find a skeleton in Library 59 called SKUSERBG. The line with IVALPKEY is already inserted. Remove the comment signs to get the following statement:

```
// EXEC IVALPKEY,PARM='PRODUCT=DB2 KEY=0000-1111-2222-3333-4444 CUSTINF*  
0=C111-111-1111'
```

### **Customizing TCP/IP**

You must have installed and configured TCP/IP for VSE. If you switch the protocol for the online application requester, use the SQLGLOB file and specify T as the communication protocol for the referencing application, to enable the TCP/IP communication protocol.

Batch applications that access remote servers always use the TCP/IP protocol, and the SQLGLOB file communications protocol parameter is ignored.

## **3.5.2 Preparing the CICS system**

To prepare the CICS system for DB2 Server for VSE Client Edition:

1. Enable CICS journaling. Use the SKJOURN and DFHJCTSP skeletons in Library 59.
2. Change the DFHSIT table. For journaling, enter JCT=SP.
3. Update the CICS startup job control:
  - a. At SYSPARM, specify the ID of the TCP/IP that you use. The default is 00.
  - a. Add your DB2 installation library to the LIBDEF search chain.
  - b. Define the labels for SQLBIND and SQLGLOB.
  - c. Add the ASSIGN statements. For journaling, enter SYS019, and for DB2, enter SYS098.

Example 3-2 shows the changes in the CICS startup job for DB2.

*Example 3-2 DB2 settings in the CICS startup job*

---

```
// JOB CICS START
// OPTION LOG, SYSPARM='00'
// LIBDEF PHASE,SEARCH=(PRD2.DB2750C,.....)
// UPSI 1
*
*
*
// DLBL SQLBIND,'DB2.BIND.MASTER',,VSAM=VSESPUC,DISP=(OLD,KEEP)
// DLBL SQLGLOB,'DB2.SQLGLOB.MASTER',,VSAM=VSESPUC,DISP=(OLD,KEEP)
// DLBL ARITRAC,'TRACE.FILE1',
// EXTENT ,VOLXXX,1,0,nnn,120
// ASSGN SYS007,cnn
*
*
// ASSGN SYS019,DISK,VOL=DOSRES,SHR
*-- FOR CICS JOURNALING
// ASSGN SYS098,SYSLST
*-- FOR CICS T/S
// EXEC DFHSIP,SIZE=DFHSIP,PARM='DSALIMIT=6M...'
*
*
*
* *
```

---

4. Add the following JCL statement to the CICS startup JCL or the batch JCL, where *xx* must match the *xx* specified in the *ID=xx* parameter in the corresponding TCP/IP startup job:

```
//OPTION SYSPARM='xx'
```

Multiple TCP/IP partitions can be active at the same time with different IDs. In this way, the application requesters can route the TCP/IP function request to the correct TCP/IP server in case more than one TCP/IP server is running in the same z/VSE system.

5. Give the z/VSE partition running TCP/IP a higher priority than the partition that is running the DB2 for VSE database server and CICS.
6. Define the files SQLGLOB and SQLBIND to CICS. You can use CEDA DEF FILE in CICS command mode.

For details about the definitions, see the ARIS75FC.A member in the PRD2.DB2750C sublibrary. For this installation, see also “File definitions in CICS” on page 173.

7. Update the security definitions for the DB2 transactions.

**Security definition:** The description of the security definition is related to the IBM provided Basic Security Manager (BSM).

To define the security rules for the DB2 supplied transactions, use Interactive Interface fastpath 281. Select option 1 to list the transaction profiles and then choose one, for example CIRB. Use option 6 to see who is authorized on the access list for this transaction. In this installation, the default of GROUP01 is authorized for the DB2 transactions.

8. Add the users that will work with these transactions to GROUP01. Use fastpath 282 to access the group maintenance dialog. Use option 6 to see the user IDs that are connected

to this group. Use option 1 to add the new users that need access to the DB2 transactions to GROUP01. Activate your security changes in BSM with fastpath 283, BSM Security Rebuild.

9. Restart CICS to use the new settings.

### 3.5.3 Preparing DB2

In this step, customize the environment variables that are used in the flow of the DB2 preparation and installation steps.

The Job Manager ARISIMGC controls the steps for preparation and installation. It uses the contents of the job list control tables and the variables stored in the Z-type member ARISIVRR. The job lists for the preparation steps are controlled from the ARISITCP member and for the installation steps in ARISITCI. All Z-type members are located in the DB2 installation library, which by default is PRD2.DB2750C.

To prepare the variables that control the installation in your environment:

1. Punch out the Z-type member ARISIVRR from your installation library. To punch out the Z-type member ARISIVRR, you can use the z/VSE command mode of Interactive Interface option 6.

**Punch out:** This term refers to the action of copying data to another machine readable location for future use.

To punch out the ARISIVRR.Z member to your primary library, enter the following line:

```
LIBRP PRD2.DB2750C ARISIVRR.Z ARISIVRR
```

Instead of the command mode, you can use the punch job shown in Example 3-3. This job punches out the ARISIVRR member in the PUNCH queue. You can copy the member from the punch queue to your primary library by using option 4 of the Interactive Interface PUNCH QUEUE panel.

*Example 3-3 Punch job*

---

```
* $$ JOB JNM=PUNCH,DISP=D,CLASS=0
// JOB PUNCH
* * * * *
* PUNCH OUT MEMBER *
* * * * *
// EXEC LIBR,PARM='MSHP'
ACCESS SUBLIB=PRD2.DB2750C
PUNCH ARISIVRR.Z
/*
/&
* $$ E0J
```

---

2. Modify the parameters in the ARISIVRR member. See “The ARISIVRR.Z parameter file for the DB2 setup” on page 175 for details about this installation.

**Important:**

- ▶ Make sure that you preserve the structure of the ARISIVRR member.
- ▶ You do not need to update parameters for the steps that you do not plan to execute.

### 3. Re-catalog ARISIVRR to your installation library.

After successful modification of the ARISIVRR control table, punch it back to the installation library of DB2 client by using a job such as the one in Example 3-4.

#### Example 3-4 Punch back job

---

```
* $$ JOB JNM=ARISCAT,DISP=D,CLASS=0
// JOB ARISCAT
// EXEC LIBR,PARM='MSHP'
ACCESS S=PRD2.DB2750C
* $$ SLI ICCF=(ARISIVRR),LIB=(12)
/*
/&
* $$ E0J
```

---

## Setting up the DBNAME directory

Update your database directory to insert the entry for the remote database VSELNXDB on Linux.

The DBNAME entry consists of several lines, each of which has the form `KEYWORD=value`. The first line *must* have the TYPE of the database, and the second line *must* have the DBNAME of the database. Blank lines are ignored. If the first two non-blank characters of a line are asterisk (\*), forward slash asterisk (/\*), or double hyphen (--), then the line is ignored and can be used for comments.

The following mandatory keywords are among those that are available for a remote DBNAME directory entry:

**TYPE=REMOTE** Defines a remote database server.  
**DBNAME =<dbname>** Specifies the database name.  
**TCPPORT=<nnnnn>** Defines which port the target database server is listening on for incoming TCP/IP connections.

For the target host information, use either of the following keywords:

**IPADDR=<n.n.n.n>** Defines the IP address of the target database server.  
**TCPHOST=<hostname>** Defines the host name of the target database server.

The following keywords are optional:

**CONNPOOL=<YIN>** Defines whether you want to enable connection pooling.  
**PWUPPER=<YIN>** Defines password conversion to uppercase.  
**PWDENC=<YIN>** Defines whether you want to enable password encryption.  
**SYSDEF=<YIN>** Defines this database as the default database.  
**PARTDEF=<partition>** Defines the default partition.  
**IMLUSER=<userid>** Defines the user ID to be used for implicit connection.

For more detailed information, see “Setting up the DBNAME directory,” in *DB2 Server for VSE System Administration*, SC09-2981.

For this installation:

1. Punch the A-type source member ARISDIRD.
2. Update the directory as shown in Example 3-5 on page 80.
3. Re-catalog the A-type source member ARISDIRD.

The definitions that you have in the database directory should now contain the entry for your local database and the entry for the remote database on Linux. Make the remote database VSELNXDB the default database in your system, as shown in Example 3-5.

*Example 3-5 Database directory entries*

---

```
/* DEFINITIONS FOR THE LOCAL DATABASE NAME
*
TYPE=LOCAL
DBNAME=SQLDS
APPLID=SYSARIO0
*
/* DEFINITIONS FOR THE REMOTE DATABASE NAME
TYPE=REMOTE
DBNAME=vselnxdb
ALIAS=vselnxdb
TCPPORT=50001
IPADDR=172.16.0.5
CONNPOOL=Y
PWUPPER=N
PWDENC=N
IMPLUSER=db2inst1
SYSDEF=Y
*
```

---

You can re-catalog the ARISDIRD member with a job such as ARISCAT, which is shown in Example 3-4 on page 79.

Now you can build a job by using member ARISIMGC.Z from the distribution library, which starts the execution of the job manager, as shown in Example 3-6.

*Example 3-6 Starting the job manager that controls the installation*

---

```
* $$ JOB JNM=ARISIMGC,DISP=L,CLASS=4,NTFY=YES
* $$ LST CLASS=V,DISP=H
// JOB ARISIMGC JCL TO START MANAGER EXEC
// LIBDEF *,SEARCH=(PRD2.DB2750C)
// EXEC REXX=ARISIMGC
/*
/&
* $$ EOJ
```

---

For CLASS, specify any available partition to run the job manager.

**Background partition:** The ARISSTDZ Z-type member must run in the background partition (BG). Therefore, for the preparation step, do not submit the job manager to run in the background partition. Run ARISIMGC in foreground partition number 4 (F4) and select BG for the partition to be used for the preparation jobs.

The jobs that are controlled by the job manager ARISIMGC customize the DB2 environment in the following preparation steps for installing DB2 Server for VSE Client Edition.

### **Defining the CICS programs and transactions**

You must define the CICS programs and transactions if you will use ISQL or online CICS transactions. The job name ARIS75JD is used to define resources, specifically programs and transactions, to the CICS System Definition (CSD) file. File definitions for online DB2 for VSE support are not included. The expected return code is 4.

### **Setting up the SQLGLOB and BIND files**

You must set up the SQLGLOB and BIND files. The Online Resource Adapter (ORA) and the Batch Resource Adapter (Batch RA) get the default configuration settings for connecting to a database server from the SQLGLOB file. The SQLGLOB file is used to contain DB2 Server for VSE Client Edition global variables. (For more information, see *DB2 Server for VSE System Administration*, SC09-2981.) When you reference the SQLGLOB file, after it is defined, code the DISP=(OLD,KEEP) parameter on the DLBL.

### **Updating labels for the new data sets**

You must update the labels for the new data sets. The job name ARISSTDJ adds the labels for the new data sets to the standard label area.

### **Summary of the DB2 preparation steps**

Example 3-7 shows the dialog for the preparation steps with the job manager on a z/VSE console.

#### *Example 3-7 DB2 preparation for installation dialog*

---

```
r rdr,arismgc
AR 0015 1C39I COMMAND PASSED TO VSE/POWER
F1 0001 1R88I OK : 1 ENTRY PROCESSED BY R RDR,ARISIMGC
F4 0001 1Q47I F4 ARISIMGC 00495 FROM (MILD) , TIME=14:01:52
F4 0004 // JOB ARISIMGC JCL TO START JOB MANAGER
        DATE 11/08/2009, CLOCK 14/01/52
F4 0004 *****
F4 0004          PREPARE FOR INSTALLATION/MIGRATION PROCESS
F4 0004 *****
F4 0004 ENTER INSTALLATION LIBRARY NAME (PRD2.DB2750C default) Q Exit
F4-0004
4
F4 0004 YOU HAVE SELECTED PRD2.DB2750C
F4 0004 PRESS ENTER TO CONTINUE OR ENTER ANOTHER LIBRARY NAME
F4 0004                                          Q Exit
F4-0004
4
F4 0004 ENTER PARTITION NUMBER: (default 4) Q Exit
F4-0004
4 0
F4 0004 YOU HAVE SELECTED 0
F4 0004 PRESS ENTER TO CONTINUE OR ENTER ANOTHER PARTITION NUMBER
F4 0004                                          Q Exit
F4-0004
4
F4 0004 PLEASE SELECT ONE OF THE FOLLOWING :
F4 0004 FOR PREPARATION.... ENTER (P)
F4 0004 FOR INSTALLATION... ENTER (I)
F4 0004                                          Q Exit
F4-0004
4 p
```

```

F4 0004 PREPARING TO INSTALL DB2 FOR VSE CLIENT EDITION ....
F4 0004
F4 0004 DO YOU WANT TO EXECUTE ALL JOBS? {Y|N-default}          Q Exit
F4-0004
4 y
F4 0004
F4 0004 ***** JOB ARIS75JD *****
F4 0004 *   DEFINE DB2750C PROGRAMS AND TRANSACTIONS
F4 0004 *****
F4 0004 JOB ARIS75JD IS OPTIONAL.
F4 0004 WOULD YOU LIKE TO RUN IT? {Y|N-Default}
F4 0004
F4-0004
4 y
F4 0001 1Q47I  F4 ARIS75JD 00493 FROM (MILD) , TIME=14:01:55
F4 0004 // JOB ARIS75JD DEFINE DB2750C PROGRAMS AND TRANSACTIONS
        DATE 11/08/2009, CLOCK 14/01/55
F4 0004 EOJ ARIS75JD  MAX.RETURN CODE=0004
        DATE 11/08/2009, CLOCK 14/01/56, DURATION  00/00/00
F4 0004 EOJ NO NAME
        DATE 11/08/2009, CLOCK 14/01/56          F4 0004
F4 0004 ***** JOB ARIS758D *****
F4 0004 *   DEFINE THE SQLGLOB FILE
F4 0004 *****
BG 0001 1Q47I  BG ARIS758D 00496 FROM (MILD) , TIME=14:03:06
BG 0000 // JOB ARIS758D
        DATE 11/08/2009, CLOCK 14/03/06
BG 0000 * *****
BG 0000 * ARIS758D: DEFINE VSAM CLUSTER FOR THE SQLGLOB FILE
BG 0000 * *****
BG 0000 * *****
BG 0000 * STEP 2: INITIALIZE THE SQLGLOB FILE WITH A DUMMY RECORD
BG 0000 * *****
BG 0000 EOJ ARIS758D  MAX.RETURN CODE=0000
        DATE 11/08/2009, CLOCK 14/03/07, DURATION  00/00/00
BG 0000 EOJ NO NAME
        DATE 11/08/2009, CLOCK 14/03/07
        DATE 11/08/2009, CLOCK 14/03/07
BG 0001 1Q34I  BG WAITING FOR WORK
F4 0004 *****
F4 0004 * Job ARIS758D  executed successfully
F4 0004 *****
F4 0004
F4 0004 ***** JOB ARISGDEF *****
F4 0004 *   CREATE SQLGLOB DEFAULTS
F4 0004 *****
BG 0001 1Q47I  BG ARISGDEF 00497 FROM (MILD) , TIME=14:03:08
BG 0000 // JOB ARISGDEF -- CREATE GLOBAL SQLGLOB DEFAULTS
        DATE 11/08/2009, CLOCK 14/03/08
BG 0000 * *****
BG 0000 * Errors detected during SQLGLOB file default setting will be      *
BG 0000 * reported by a 4 digit non-zero return code that has the format    *
BG 0000 * XXXY, where 'Y' represents the action that was taken, and        *
BG 0000 * 'XXX' the error code that was returned.                            *
BG 0000 * The following lists the return codes with their explanations:     *

```



```

BG 0000 *      Y=1 - Check the VSAM SQLGLOB file label *
BG 0000 *      XXX=100 - Label not found *
BG 0000 *      XXX=125 - Unexpected error *
BG 0000 *      Y=2 - Generate ACB and RPL for SQLGLOB file *
BG 0000 *      XXX=175 - I/O error *
BG 0000 *      XXX=200 - Get storage failed *
BG 0000 *      XXX=225 - Free storage failed *
BG 0000 *      Y=3 - Open a VSAM file *
BG 0000 *      Y=4 - Erase data from a VSAM file *
BG 0000 *      Y=5 - Write data to a VSAM file *
BG 0000 *      Y=6 - Close a VSAM file *
BG 0000 *      XXX=150 - End of file *
BG 0000 *      XXX=175 - I/O error *
BG 0000 *      XXX=200 - Get storage failed *
BG 0000 *      XXX=225 - Free storage failed *
BG 0000 *      XXX=250 - Invalid call *
BG 0000 * *****
BG 0000 * ARISGDEF STEP 1 -- EXECUTE ARICGDEF TO STORE SQLGLOB DEFAULTS *
BG 0000 * *****
BG 0000 EOJ ARISGDEF MAX.RETURN CODE=0000
          DATE 11/08/2009, CLOCK 14/03/09, DURATION 00/00/01
BG 0000 EOJ NO NAME
          DATE 11/08/2009, CLOCK 14/03/09
BG 0001 1Q34I BG WAITING FOR WORK
F4 0004 *****
F4 0004 * Job ARISGDEF executed successfully
F4 0004 *****
F4 0004
F4 0004 ***** JOB ARIS757D *****
F4 0004 * DEFINE VSAM CLUSTER FOR THE BIND FILES
F4 0004 *****
F4 0004 JOB ARIS757D IS OPTIONAL.
F4 0004 WOULD YOU LIKE TO RUN IT? {Y|N-Default}
F4 0004
F4-0004 Q Exit
4 y
BG 0001 1Q47I BG ARIS757D 00498 FROM (MILD) , TIME=14:03:21
BG 0000 // JOB ARIS757D
          DATE 11/08/2009, CLOCK 14/03/21
BG 0000 * *****
BG 0000 * STEP 1: DEFINE VSAM CLUSTER FOR THE SQLBIND FILE
BG 0000 * *****
BG 0000 * *****
BG 0000 * STEP 2: INITIALIZE THE SQLBIND FILE WITH A DUMMY RECORD
BG 0000 * *****
BG 0000 EOJ ARIS757D MAX.RETURN CODE=0000
          DATE 11/08/2009, CLOCK 14/03/22, DURATION 00/00/00
BG 0000 EOJ NO NAME
          DATE 11/08/2009, CLOCK 14/03/22
BG 0001 1Q34I BG WAITING FOR WORK
F4 0004 *****
F4 0004 * Job ARIS757D executed successfully
F4 0004 *****
F4 0004
F4 0004 ***** JOB ARIS759D *****

```

```

F4 0004 *   DEFINE VSAM CLUSTER FOR THE BINDWKF FILE
F4 0004 *****
F4 0004 JOB ARIS759D IS OPTIONAL.
F4 0004 WOULD YOU LIKE TO RUN IT? {Y|N-Default}
F4 0004
F4-0004
4 y
BG 0001 1Q47I  BG ARIS759D 00499 FROM (MILD) , TIME=14:03:30
BG 0000 // JOB ARIS759D
        DATE 11/08/2009, CLOCK 14/03/30
BG 0000 * *****
BG 0000 * ARIS759D: DEFINE VSAM CLUSTER FOR THE BINDWKF FILE
BG 0000 * *****
BG 0000 EOJ ARIS759D MAX.RETURN CODE=0000
        DATE 11/08/2009, CLOCK 14/03/30, DURATION 00/00/00
BG 0000 EOJ NO NAME
        DATE 11/08/2009, CLOCK 14/03/30
BG 0001 1Q34I  BG WAITING FOR WORK
F4 0004 *****
F4 0004 * Job ARIS759D executed successfully
F4 0004 *****
F4 0004
F4 0004 ***** JOB ARISIQBD *****
F4 0004 * ISQL BIND FILE CONVERSION
F4 0004 *****
F4 0004 JOB ARISIQBD IS OPTIONAL.
F4 0004 WOULD YOU LIKE TO RUN IT? {Y|N-Default}
F4 0004
F4-0004
4 y
BG 0001 1Q47I  BG ARISIQBD 00500 FROM (MILD) , TIME=14:03:43
BG 0000 // JOB ARISIQBD -- ISQL BIND FILE CONVERSION
        DATE 11/08/2009, CLOCK 14/03/43
BG 0000 * *****
BG 0000 * Errors detected during ISQL bind file generation will be      *
BG 0000 * reported by a 4 digit non-zero return code that has the format *
BG 0000 * XXXY. The following lists the return codes with their        *
BG 0000 * explanations:
BG 0000 *   Y=0 - Get the storage for bind file record (GETVIS)          *
BG 0000 *       XXX=200 - Get storage failed                               *
BG 0000 *   Y=1 - Check the BIND file label (LABEL)                      *
BG 0000 *       XXX=100 - Label not found                                  *
BG 0000 *       XXX=125 - Unexpected error                                 *
BG 0000 *   Y=2 - Generate ACB and RPL for BIND file (GENCB)            *
BG 0000 *       XXX=175 - I/O error                                         *
BG 0000 *       XXX=200 - Get storage failed                               *
BG 0000 *       XXX=225 - Free storage failed                             *
BG 0000 *   Y=3 - Open a BIND file (OPEN)                                  *
BG 0000 *   Y=4 - Erase a BIND record (ERASE)                              *
BG 0000 *   Y=5 - Write data to a BIND file (PUT)                          *
BG 0000 *   Y=6 - Close a BIND file (CLOSE)                                *
BG 0000 * (Following applies to Y=3, 4, 5, and 6)                          *
BG 0000 *       XXX=150 - End of file                                       *
BG 0000 *       XXX=175 - I/O error                                         *
BG 0000 *       XXX=250 - Invalid call                                     *

```

```

BG 0000 *          XXX=200 - Get storage failed *
BG 0000 *          XXX=225 - Free storage failed *
BG 0000 *          Y=7 - Open a PACKAGE file      (OPEN) *
BG 0000 *          Y=8 - Read data from a PACKAGE file (GET) *
BG 0000 *          Y=9 - Close a PACKAGE file      (CLOSE) *
BG 0000 *          (Following applies to Y=7, 8, and 9) *
BG 0000 *          XXX=175 - I/O error *
BG 0000 *          XXX=250 - Invalid call *
BG 0000 *          XXX=200 - Get storage failed *
BG 0000 *          XXX=300 - Wrong record length *
BG 0000 *          XXX=325 - Internal error *
BG 0000 *          XXX=360 - Error retrieving file for SYSIPT read *
BG 0000 *          XXX=370 - End of extents encountered while *
BG 0000 *          attempting to write to disk *
BG 0000 * *****
BG-0000 // PAUSE
0
BG 0000 EOJ ARISIQBD MAX.RETURN CODE=0000
          DATE 11/08/2009, CLOCK 14/14/12, DURATION 00/10/29
BG 0000 EOJ NO NAME
          DATE 11/08/2009, CLOCK 14/14/12
BG 0001 1Q34I BG WAITING FOR WORK
F4 0004 *****
F4 0004 * Job ARISIQBD executed successfully
F4 0004 *****
F4 0004
F4 0004 ***** JOB ARISSTDL *****
F4 0004 * ADD NEW LABELS TO STANDARD LABELS
F4 0004 ARISSTDL MUST BE EXECUTED IN PARTITION BG
F4 0004 IF BACKGROUND PARTITION IS BUSY THEN SKIP THIS STEP
F4 0004 AND RESUBMIT ARISIMGC TO RERUN THS STEP IN BG.
F4 0004 ENTER 'YES' TO RUN THIS STEP. ANY OTHER OPTION WILL SKIP
F4 0004 THIS STEP.
F4-0004
4
F4 0004 *****
F4 0004 JOB ARISSTDL IS OPTIONAL.
F4 0004 WOULD YOU LIKE TO RUN IT? {Y|N-Default}
F4 0004
F4-0004
4 y
BG 0001 1Q47I BG ARISSTDL 00501 FROM (MILD) , TIME=14:16:02
BG 0000 // JOB ARISSTDL
          DATE 11/08/2009, CLOCK 14/16/02
BG 0000 *
BG 0000 *          Standard Label Update must run in BG
BG 0000 *
BG 0000 *          PK61416 Feb/08 - For label BINDWKF, omit disposition parameter that
BG 0000 *          defaults to DISP=(NEW,DELETE,DELETE).It ensures
BG 0000 *          that bind work file is used as temporary file.
BG 0000 *
BG-0000 // PAUSE
0
BG 0000 1L63I LABEL VSEUCAT NOT FOUND
BG 0000 1L63I LABEL SQLGLOB NOT FOUND

```

Q Exit

```

BG 0000 1L63I LABEL VSEBIND NOT FOUND
BG 0000 1L63I LABEL SQLBIND NOT FOUND
BG 0000 1L63I LABEL VSEWBND NOT FOUND
BG 0000 1L63I LABEL BINDWKF NOT FOUND
BG 0000 EOJ ARISSTDL
      DATE 11/08/2009, CLOCK 14/16/31, DURATION 00/00/28
BG 0000 EOJ NO NAME
      DATE 11/08/2009, CLOCK 14/16/31
BG 0001 1Q34I BG WAITING FOR WORK
F4 0004 READ CONSOLE FOR DETAILS.
F4 0004 EOJ ARISIMGC MAX.RETURN CODE=0000
      DATE 11/08/2009, CLOCK 14/16/32, DURATION 00/14/40
F4 0001 1Q34I F4 WAITING FOR WORK

```

---

Example 3-8 shows the standard labels that are added after the preparation steps.

*Example 3-8 Standard labels defined for DB2 Server for VSE Client Edition*

---

```

// OPTION STDLABEL=ADD
// DLBL VSEUCAT, 'VSESP.USER.CATALOG' , ,VSAM
// DLBL SQLGLOB, 'DB2.SQLGLOB.MASTER' ,0,VSAM,CAT=VSEUCAT,DISP=(OLD,KEEP)
// DLBL VSEBIND, 'VSESP.USER.CATALOG' , ,VSAM
// DLBL SQLBIND, 'DB2.BIND.MASTER' ,0,VSAM,CAT=VSEBIND,DISP=(OLD,KEEP)
// DLBL VSEWBND, 'VSESP.USER.CATALOG' , ,VSAM
// DLBL BINDWKF, 'DB2.BIND.WORKF' ,0,VSAM,CAT=VSEWBND

```

---

### 3.5.4 Installing DB2 Server for VSE Client Edition

After you complete the preparation steps, you can start the job manager again for the installation steps. The jobs in the following sections are run during the DB2 Server for VSE Client Edition installation procedure.

#### Setting up the default user ID and password

You must set up the default user ID and password. The job name ARIS75CZ generates a type A member of ARISICON that contains an SQL CONNECT statement. The user ID and password for the CONNECT statement are read from the GLOBAL parameter section DBUSRID and DBPASSWORD in the installation section in the type Z member ARISIVRR. The generated ARISICON.A member is placed in the DB2 installation sublibrary that is read by other job steps such as ARIS360D, ARIS130D, ARIS110D, and ARIS063D.

#### Optional: Link-editing the DB2 Server for VSE online support components

You can link-edit the online support components for DB2 Server for VSE. The job name is ARIS75BD, and the job control member ARIS75BD is used to link-edit the online support components. For details, see 3.9, “DRDA interfaces in z/VSE” on page 102.

#### Optional: Link-editing the DB2 Server for VSE client batch

You can link-edit the client batch for DB2 Server for VSE. The job name is ARIS75LD. The job ARIS75LD generates the batch resource adapter for a remote database environment by link-editing the required object modules and producing the ARIRBARM.PHASE for batch.

## Installing optional components

The installation of the optional components involves execution of the ARIS75DU and ARIS75ED jobs. The ARIS75DU job prepares the batch utility DBSU and facilitates the installation steps that follow. The steps use this utility to update the tables in the default database, which is identified by the keyword SYSDEF=Y in the DBNAME directory that is built in step 1 on page 79 of the installation.

The ARIS75ED job contains the job control statements to load the optional components, such as ISQL, on the default database. You can omit specific optional components by removing the procedures in the ARIS75ED job that creates that component. In this installation, you need the step with ARIS120D (in Example 3-16 on page 94) to create the ISQL package.

By default, the assembler interface is linked for best performance in the ARIS75BD job.

## Summary of installing DB2 Server for VSE Client Edition

Example 3-9 shows the dialog for the installation steps.

### *Example 3-9 Installation steps for DB2 Server for VSE Client Edition*

```
r rdr,arismgc
AR 0015 1C39I COMMAND PASSED TO VSE/POWER
F1 0001 1R88I  OK : 1 ENTRY PROCESSED BY R RDR,ARISIMGC
F4 0001 1Q47I  F4 ARISIMGC 00505 FROM (MILD) , TIME=14:42:31
F4 0004 // JOB ARISIMGC JCL TO START JOB MANAGER
      DATE 11/08/2009, CLOCK 14/42/31
F4 0004 *****
F4 0004      PREPARE FOR INSTALLATION/MIGRATION PROCESS
F4 0004 *****
F4 0004 ENTER INSTALLATION LIBRARY NAME (PRD2.DB2750C default) Q Exit
F4-0004
4
F4 0004 YOU HAVE SELECTED  PRD2.DB2750C
F4 0004 PRESS ENTER TO CONTINUE OR ENTER ANOTHER LIBRARY NAME
F4 0004                                                    Q Exit
F4-0004
F4-0004
4
F4 0004 ENTER PARTITION NUMBER: (default 4)                Q Exit
F4-0004
4 0
F4 0004 YOU HAVE SELECTED  0
F4 0004 PRESS ENTER TO CONTINUE OR ENTER ANOTHER PARTITION NUMBER
F4 0004                                                    Q Exit
F4-0004
4
F4 0004 PLEASE SELECT ONE OF THE FOLLOWING :
F4 0004 FOR PREPARATION....  ENTER (P)
F4 0004 FOR INSTALLATION...  ENTER (I)
F4 0004                                                    Q Exit
F4-0004
4 i
F4 0004 DO YOU WANT TO EXECUTE ALL JOBS? {Y|N-default}      Q Exit
F4-0004
4 y
F4 0004
F4 0004 ***** JOB ARIS75CZ *****
```

```

F4 0004 * GENERATE DEFAULT SQL CONNECT STATEMENT
F4 0004 *****
F4 0004 JOB ARIS75CZ IS OPTIONAL.
F4 0004 WOULD YOU LIKE TO RUN IT? {Y|N-Default}
F4 0004
F4-0004
4 y
BG 0001 1Q47I  BG ARIS75CZ 00506 FROM (MILD) , TIME=14:43:01
BG 0000 // JOB ARIS75CZ
        DATE 11/08/2009, CLOCK 14/43/01
BG 0000 EOJ ARIS75CZ  MAX.RETURN CODE=0000
        DATE 11/08/2009, CLOCK 14/43/01, DURATION  00/00/00
BG 0001 1Q34I  BG WAITING FOR WORK
F4 0004 *****
F4 0004 * Job  ARIS75CZ  executed successfully
F4 0004 *****
F4 0004
F4 0004 ***** JOB ARISBDID *****
F4 0004 * SETUP THE DBNAME DIRECTORY
F4 0004 *****
F4 0004 JOB ARISBDID IS OPTIONAL.
F4 0004 WOULD YOU LIKE TO RUN IT? {Y|N-Default}
F4 0004
F4-0004
4 y
BG 0001 1Q47I  BG ARISBDID 00507 FROM (MILD) , TIME=14:43:10
BG 0000 // JOB ARISBDID -- DBNAME DIRECTORY SERVICE GENERATION
        DATE 11/08/2009, CLOCK 14/43/10
BG 0000 * *****
BG 0000 *
BG 0000 * THIS JCL EXECUTES STEPS TO GENERATE THE DBNAME DIRECTORY SERVICE *
BG 0000 * ROUTINE (ARICDIRD.PHASE). *
BG 0000 * *
BG 0000 * STEP 1 EXECUTES THE PROCEDURE ARICCDID FOR MIGRATION *
BG 0000 * OR ARICBDID FOR INSTALLATION TO READ THE DBNAME *
BG 0000 * DIRECTORY SOURCE MEMBER ARISDIRD.A FROM THE PRODUCTION LIBRARY, *
BG 0000 * AND GENERATES THE ASSEMBLER VERSION OF ARISDIRD ON SYSPCH. *
BG 0000 * *
BG 0000 * *****
BG 0000 * *****
BG 0000 * ARISBDID  STEP 1 -- BUILD ASSEMBLER VERSION OF ARISDIRD *
BG 0000 * *****
BG 0000 * SQL/DS DBNAME DIRECTORY BUILT SUCCESSFULLY
BG 0000 EOJ ARISBDID  MAX.RETURN CODE=0002
        DATE 11/08/2009, CLOCK 14/43/11, DURATION  00/00/00
BG 0000 EOJ NO NAME
        DATE 11/08/2009, CLOCK 14/43/11
BG 0001 1Q34I  BG WAITING FOR WORK
F4 0004 *****
F4 0004 * Job  ARISBDID  executed successfully
F4 0004 *****
F4 0004
F4 0004 ***** JOB ARIS75BD *****
F4 0004 * LINK EDIT DB2 SERVER ONLINE SUPPORT COMPONENTS
F4 0004 *****

```

Q Exit

Q Exit

```

F4 0004 JOB ARIS75BD IS OPTIONAL.
F4 0004 WOULD YOU LIKE TO RUN IT? {Y|N-Default}
F4 0004
F4-0004
4 y
BG 0001 1Q47I BG ARIS75BD 00508 FROM (MILD) , TIME=14:43:24
BG 0000 // JOB ARIS75BD LINK EDIT DB2 FOR VSE ONLINE SUPPORT COMPONENTS
DATE 11/08/2009, CLOCK 14/43/24
BG 0000 * *****
BG 0000 * ARIS090D: LINK EDIT ONLINE RA CONTROL WITH ASM INTERFACE
BG 0000 * *****
BG 0000 * *****
BG 0000 * ARIS140D: LINK EDIT ISQL
BG 0000 * *****
BG 0000 * *****
BG 0000 * ARIS150D: LINK EDIT ISQL ITRM TERMINAL TRANSACTION
BG 0000 * *****
BG 0000 * *****
BG 0000 * ARIS160D: LINK EDIT ISQL ITRM TERMINAL EXTENSION PROGRAM
BG 0000 * *****
BG 0000 EOJ ARIS75BD MAX.RETURN CODE=0000
DATE 11/08/2009, CLOCK 14/43/27, DURATION 00/00/03
BG 0000 EOJ NO NAME
DATE 11/08/2009, CLOCK 14/43/27
BG 0001 1Q34I BG WAITING FOR WORK
F4 0004 *****
F4 0004 * Job ARIS75BD executed successfully
F4 0004 *****
F4 0004
F4 0004 *****
F4 0004 JOB ARIS75LD IS OPTIONAL.
F4 0004 WOULD YOU LIKE TO RUN IT? {Y|N-Default}
F4 0004
F4-0004
4 y
BG 0001 1Q47I BG ARIS75LD 00509 FROM (MILD) , TIME=14:43:36
BG 0000 // JOB ARIS75LD
DATE 11/08/2009, CLOCK 14/43/36
BG 0000 * *****
BG 0000 * ARIS75LD: LINK EDIT BATCH RA WITH DRDA CODE WITH ASM INTERFACE
BG 0000 * *****
BG 0000 EOJ ARIS75LD MAX.RETURN CODE=0000
DATE 11/08/2009, CLOCK 14/43/41, DURATION 00/00/04
BG 0001 1Q34I BG WAITING FOR WORK
F4 0004 *****
F4 0004 * Job ARIS75LD executed successfully
F4 0004 *****
F4 0004
F4 0004 *****
F4 0004 JOB ARIS75DU IS OPTIONAL.
F4 0004 WOULD YOU LIKE TO RUN IT? {Y|N-Default}
F4 0004
F4-0004
4 y
BG 0001 1Q47I BG ARIS75DU 00510 FROM (MILD) , TIME=14:43:47

```

Q Exit

Q Exit

Q Exit

```

F4 0004 *****
BG 0000 // JOB ARIS75DU                PREPARE DBSU UTILITY
        DATE 11/08/2009, CLOCK 14/43/47
BG 0000 * *****
BG 0000 * ARIS75SL: DB2 SERVICE/PRODUCTION LIBRARY DEFINITION
BG 0000 * *****
BG 0000 * *****
BG 0000 * ARIS040D: CREATE THE DBS SQL PACKAGE
BG 0000 * *****
BG 0000 ARI4027W THERE ARE 90 DAY(S) LEFT IN THE EVALUATION PERIOD
BG 0000          FOR THE PRODUCT DB2/VSE.
BG 0000 EOJ ARIS75DU MAX.RETURN CODE=0004
        DATE 11/08/2009, CLOCK 14/44/42, DURATION 00/00/54
BG 0000 EOJ NO NAME
DATE 11/08/2009, CLOCK 14/44/42
F4 0004 EOJ ARISIMGC MAX.RETURN CODE=0000
        DATE 11/08/2009, CLOCK 14/44/43, DURATION 00/02/12
F4 0001 1Q34I  F4 WAITING FOR WORK

```

---

### 3.5.5 Verifying the installation

Now you can verify the installation by using the batch and online utilities. Run a batch job and use DBSU and ISQL as explained in the following sections.

#### Performing a batch test

To test the installation of the DB2 Server for VSE Client Edition for batch, submit the JCL as shown in Example 3-10. If DBSU was not installed previously, submit the preprocess DBSU job ARIS75DU to a target server that is listed in the DBNAME directory.

*Example 3-10 DB2 VSE client test for batch*

---

```

* $$ JOB JNM=TSTCBAT,DISP=D,CLASS=0
// JOB TESTCLNT
// LIBDEF *,SEARCH=(PRD2.DB2750C)
// EXEC ARIDBS,SIZE=AUTO
CONNECT DB2INST1 IDENTIFIED BY MYLINPW;
SET ERRORMODE CONTINUE;
SET AUTOCOMMIT ON;
CREATE TABLE CHKINSTTAB (VAR1 INT, VAR2 CHAR(20));
INSERT INTO CHKINSTTAB VALUES (100, 'TESTCLIENT');
INSERT INTO CHKINSTTAB VALUES (200, 'SUCCESSFUL');
SELECT * FROM CHKINSTTAB;
/*
/&
* $$ EOJ

```

---



You see the result of the job from Example 3-10 in the VSE/POWER list queue as shown in Example 3-11.

*Example 3-11 Listing of the DB2 VSE client test in batch*

```
// JOB TESTCLNT                                     DATE
11/09/2009, CLOCK 07/08/59
// LIBDEF *,SEARCH=(PRD2.DB2750C)
// EXEC ARIDBS,SIZE=AUTO
1S54I PHASE ARIDBS IS TO BE FETCHED FROM PRD2.DB2750C
ARI0801I DBS Utility started: 11/09/09 07:09:00.
        AUTOCOMMIT = OFF ERRORMODE = OFF
        ISOLATION LEVEL = REPEATABLE READ
ARI2906I The only valid isolation level is CS
        when the DRDA protocol is used.
        Isolation level CS is now in effect.
-----> CONNECT "DB2INST1" IDENTIFIED BY *****;
ARI8004I User DB2INST1 connected to server SAMPLE.
ARI0500I SQL processing was successful.
ARI0505I SQLCODE = 0 SQLSTATE = 00000 ROWCOUNT = 0
-----> SET ERRORMODE CONTINUE;
ARI0827I ...Begin command execution: ERRORMODE = CONTINUE
-----> SET AUTOCOMMIT ON;
ARI0815I ...AUTOCOMMIT Processing Mode = ON
ARI8997I ...Begin COMMIT processing.
ARI0811I ...COMMIT of any database changes successful.
-----> CREATE TABLE CHKINSTTAB (VAR1 INT, VAR2 CHAR(20));
ARI0500I SQL processing was successful.
ARI0505I SQLCODE = 0 SQLSTATE = 00000 ROWCOUNT = 0
ARI8997I ...Begin COMMIT processing.
ARI0811I ...COMMIT of any database changes successful.
-----> INSERT INTO CHKINSTTAB VALUES (100, 'TESTCLIENT');
ARI0500I SQL processing was successful.
ARI0505I SQLCODE = 0 SQLSTATE = 00000 ROWCOUNT = 1
ARI8997I ...Begin COMMIT processing.
ARI0811I ...COMMIT of any database changes successful.
-----> INSERT INTO CHKINSTTAB VALUES (200, 'SUCCESSFUL');
ARI0500I SQL processing was successful.
ARI0505I SQLCODE = 0 SQLSTATE = 00000 ROWCOUNT = 1
ARI8997I ...Begin COMMIT processing.
ARI0811I ...COMMIT of any database changes successful.
-----> SELECT * FROM CHKINSTTAB;
SELECT * FROM CHKINSTTAB
PAGE          1
          VAR1 VAR2
-----
          100 TESTCLIENT
          200 SUCCESSFUL
ARI0850I SQL SELECT processing successful: Rowcount = 2
ARI8997I ...Begin COMMIT processing.
ARI0811I ...COMMIT of any database changes successful.
ARI0802I End of command file input.
ARI0809I ...No errors occurred during command processing.
ARI0808I DBS processing completed: 11/09/09 07:09:00.
1S55I LAST RETURN CODE WAS 0000
```

If you see an -805 error message in your output listing instead of the result shown in Example 3-11, your bind process did not work properly.

Verify the ARIS040D.PROC member in your DB2 installation library for the following statement that SQLDBA.ARIDSQ is mentioned as shown:

```
// EXEC      PGM=ARIPRPA,SIZE=AUTO,                                X  
              PARM='USERID=&UID/&PWD,PREPNAME=SQLDBA.ARIDSQ,PREPFILE=X  
              (ARIPREPF.A)'
```

If your member looks different, correct it. Then rerun the job step ARIS75DU by using the job manager ARISIMGC as shown in Example 3-12.

*Example 3-12 Rerunning the bind job for the batch mode of the DB2 VSE client*

```
r rdr,arismgc  
AR 0015 1C39I COMMAND PASSED TO VSE/POWER  
F1 0001 1R88I  OK : 1 ENTRY PROCESSED BY R RDR,ARISIMGC  
F4 0001 1Q47I  F4 ARISIMGC 00578 FROM (MILD) , TIME=10:24:22  
F4 0004 // JOB ARISIMGC JCL TO START JOB MANAGER  
      DATE 11/09/2009, CLOCK 10/24/22  
F4 0004 *****  
F4 0004      PREPARE FOR INSTALLATION/MIGRATION PROCESS  
F4 0004 *****  
F4 0004 ENTER INSTALLATION LIBRARY NAME (PRD2.DB2750C default) Q Exit  
F4-0004  
4  
F4 0004 YOU HAVE SELECTED  PRD2.DB2750C  
F4 0004 PRESS ENTER TO CONTINUE OR ENTER ANOTHER LIBRARY NAME  
F4 0004                                          Q Exit  
F4-0004  
4  
F4 0004 ENTER PARTITION NUMBER: (default 4)          Q Exit  
F4-0004  
4 0  
F4 0004 YOU HAVE SELECTED  0  
F4 0004 PRESS ENTER TO CONTINUE OR ENTER ANOTHER PARTITION NUMBER  
F4 0004                                          Q Exit  
F4-0004  
4  
F4 0004 PLEASE SELECT ONE OF THE FOLLOWING :  
F4 0004 FOR PREPARATION....  ENTER (P)  
F4 0004 FOR INSTALLATION...  ENTER (I)  
F4 0004                                          Q Exit  
F4-0004  
4 i  
F4 0004 DO YOU WANT TO EXECUTE ALL JOBS? {Y|N-default}  Q Exit  
F4-0004  
4 n  
F4 0004 IF YOU WANT TO START FROM A SPECIFIC JOB ENTER 1  
F4 0004 IF YOU WANT TO EXECUTE ONLY ONE STEP      ENTER 2 (Default)  
F4 0004                                          Q Exit  
F4-0004
```

```
4 2
F4 0004 ENTER THE JOB NAME YOU WANT TO EXECUTE (ex. ARIS75JD):
F4 0004 Q Exit
F4-0004
4 aris75du
```

---

Rerun the batch test job as shown in Example 3-10 on page 90. It should finish with RC=0.

### Performing an online test

To test the DB2 online client, enable the DRDA protocol for communications between the DB2 Server for VSE and remote databases. Select the interface and link the appropriate phase (linkbook). For details, see 3.9, “DRDA interfaces in z/VSE” on page 102.

Use the interactive utility ISQL. ISQL runs as a CICS transaction and is invoked similar to how any other transaction is invoked. Before you can test online applications and ISQL, install the CICS group DB2750C on your system:

1. Install the CICS group in CICS command mode as follows:

```
CEDA I G(DB2750C)
```

2. Use a CICS terminal, for example, the Interactive Interface PF6=ESCAPE(U), and start the DB2 online support.

Example 3-13 shows the start of the DB2 online support and the feedback messages that use a CICS terminal.

#### *Example 3-13 Enabling three connections to the BIGLNxDB remote database*

---

```
cirb mylinpw,3,db2inst1,0,ameng,vselnxdb
ARI0410I Resource Adapter ARI00LRM is enabled.
ARI0450I DB2 Server for VSE online support has an
entry point of 0230BD10. RMGL at 005851E4.
DRDA online support entry point at 02600000.
ARI0454I Connections to BIGLNxDB established.
RMCV at 00614040.
ARI0458I The default server is BIGLNxDB.
```

---

3. If you enabled connection pooling with CONNPOOL=Y, as shown in Example 3-5 on page 80, in the CIRB command, specify the number of connections that you will establish. In Example 3-13, we started three concurrent connections.
4. Start ISQL from a CICS terminal by using the ISQL command.
5. When you see the ISQL sign-on panel, enter your user ID, password, and optionally a target database name. Example 3-14 shows the ISQL messages of the successful start.

#### *Example 3-14 ISQL startup messages*

---

```
ARI7399I The ISQL default profile values are in effect.
ARI7079I ISQL initialization complete.
ARI7080A Please enter an ISQL command or an SQL statement.
```

---

6. On the same panel, enter some ISQL statements.

Example 3-15 shows a select statement and its result. You can see the rows that are inserted by the batch test program as shown in Example 3-10 on page 90.

*Example 3-15 ISQL select statement with result*

---

```
select * from chkinsttab
VAR1  VAR2
-----
          100  TESTCLIENT
          200  SUCCESSFUL
* End of Result *** 2 Rows Displayed ***Cost Estimate is 1***
```

---

7. Use the END statement to end the display. If you want to terminate the ISQL session, enter EXIT.

An -805 error message is an indication that the bind process did not work properly for ISQL. Run ARIS75ED by using the ARISIMGC job manager as shown in Example 3-12 on page 92. In that process, replace ARIS75DU with ARIS75ED. Example 3-16 shows the start and result of ARIS75ED.

*Example 3-16 Bind of ISQL components*

---

```
F4 0004 4 aris75ed
F4 0004 ***** JOB ARIS75ED *****
F4 0004 * INSTALL DATABASE COMPONENTS (ISQL, FIPS FLAGGER)
F4 0004 *****
BG 0001 1Q47I  BG ARIS75ED 00579 FROM (MILD) , TIME=10:29:11
BG 0000 // JOB ARIS75ED  INSTALL DATABASE COMPONENTS
          DATE 11/09/2009, CLOCK 10/29/11
BG 0000 * *****
BG 0000 * ARIS75SL: DB2 SERVICE/PRODUCTION LIBRARY DEFINITION
BG 0000 * *****
BG 0000 * *****
BG 0000 * ARIS080D: GRANT SCHEDULE AUTHORITY TO DBCCICS
BG 0000 * *****
BG 0000 ARI4027W THERE ARE 90 DAY(S) LEFT IN THE EVALUATION PERIOD
BG 0000          FOR THE PRODUCT DB2/VSE.
BG 0000 * *****
BG 0000 * ARIS110D: CREATE/GRANT ISQL TABLES/AUTHORIZATIONS
BG 0000 * *****
BG 0000 ARI4027W THERE ARE 90 DAY(S) LEFT IN THE EVALUATION PERIOD
BG 0000          FOR THE PRODUCT DB2/VSE.
BG 0000 * *****
BG 0000 * ARIS120D: RELOAD THE ISQL PACKAGE
BG 0000 * *****
BG 0000 ARI4027W THERE ARE 90 DAY(S) LEFT IN THE EVALUATION PERIOD
BG 0000          FOR THE PRODUCT DB2/VSE.
BG 0000 * *****
BG 0000 * ARIS130D: COMPLETE THE INSTALLATION OF ISQL
BG 0000 * *****
BG 0000 ARI4027W THERE ARE 90 DAY(S) LEFT IN THE EVALUATION PERIOD
BG 0000          FOR THE PRODUCT DB2/VSE.
BG 0000 * *****
BG 0000 * ARIS360D: RELOAD ARIFCRD
BG 0000 * *****
```

### 3.5.6 Considerations for an implicit connection

Set the following database manager configurations parameters (**dbm cfg**) on the DB2 on Linux database for implicit connections from DB2 Server for VSE batch and online application requesters:

- ▶ AUTHENTICATION=CLIENT
- ▶ TRUST\_ALLCLNTS=DRDAONLY
- ▶ TRUST\_CLNTAUTH=YES

Enter the following db2 commands on DB2 on Linux for System z to set these **dbm cfg** parameters:

```
Update dbm cdf using authentication client
Update dbm cfg using trust_allclnts drdaonly
Update dbm cfg using trust_clntauth yes
db2stop
db2start
```

An implicit connection for batch workload avoids the specification of a user ID and password in every batch job. The implicit user ID must be specified in the DBNAME directory by using **IMPLUSER**. The implicit connection is established to the default database that is defined in the DBNAME directory if no explicit **CONNECT** statement is in the application program.

Therefore, the z/VSE batch applications do not need a **CONNECT** statement in the application program if the following statements are *true*:

- ▶ The parameter **IMPLUSER=userid** is defined in the DBNAME directory.
- ▶ *//* **SETPARM USERID = userid** is defined in the JCL that is invoking the application program.

For example, if you must implicitly connect to the BIGLNxDB Linux database on System z from any z/VSE batch application program that does not have any explicit **CONNECT** statement coded, the following DBNAME directory is built with the **IMPLUSER** keyword:

```
TYPE=REMOTE
DBNAME=VSELNXDB
TCPPORT=50001
IPADDR=172.16.0.5
SYSDEF=Y
IMPLUSER=DB2INST1
```

The batch application program connects to VSELNXDB with user ID DB2INST1 and executes the SQL requests.

### 3.5.7 Connection pooling in an online resource adapter

When a **CONNECT** statement is processed, a TCP/IP link and a database link are established. Establishing and destroying the TCP/IP links between the online resource adapter and the remote server were found to be expensive operations. You can avoid these operations during SQL **CONNECT** if a pool of TCP/IP links is created at the time of enabling the online resource manager. At the time of SQL **CONNECT**, only a database link must be established.

Destroying these links can be deferred until CIRT/CIRR when the server is being disabled. You can optionally choose this connection pooling mechanism for a particular server that is connected over TCP/IP by using a DBNAME entry option field CONNPOOL.

Either set to *high* or disable the DB2\_SERVER\_CONTIMEOUT configuration parameter in DB2 Linux to use connection pooling.

For example, if you want to enable the connection pooling feature to the remote database BIGLNxDB, the DBNAME directory must be built with the CONNPOOL keyword as follows:

```
TYPE=REMOTE
DBNAME=BIGLNxDB
TCPPORT=50000
IPADDR=172.16.0.5
SYSDEF=Y
CONNPOOL=Y
```

**CONNPOOL:** The default value for CONNPOOL in the DBNAME is Y. If the CONNPOOL keyword is skipped, by default, it is set to Y.

Enter the following DB2 command on DB2 for Linux on System z to set the **dbm cfg** parameter DB2\_SERVER\_CONTIMEOUT:

```
update dbm cfg using DB2_SERVER_CONTIMEOUT 0
```

This command inhibits DB2 on Linux from dropping the connection after a certain time of inactivity.

## 3.6 DB2 Server for VSE

If you want to use the federated migration process for data and tables in Linux, make the DB2 data on z/VSE accessible from Linux. Therefore, ensure that both TCP/IP and DRDA are active for DB2 Server for VSE.

### 3.6.1 Preparing the application server to use TCP/IP

To allow the application server to use TCP/IP:

1. Install and configure TCP/IP for VSE.
2. Make sure that you have installed Language Environment VSE and the libraries are accessible. For the batch application requester and the online (CICS) application requester on z/VSE, this library is the PRD2.SCEEBASE library. The minimum support level of Language Environment VSE is Version 1 Release 4.
3. Add the TCP/IP for VSE library in the LIBDEF search chain. The library is PRD1.BASE on both the online application requester and the batch application requester. If you have not ordered the TCP/IP product from IBM directly, the product library is usually PRD2.TCPIP.

4. Ensure that the search order of the C runtime library and the TCP/IP library provides a good TCP/IP functionality.

The following example LIBDEF statements indicate the possible combinations in batch and online partitions:

- Installation of the TCP/IP product from IBM:

```
//LIBDEF*,SEARCH=(PRD1.BASE,PRD2.SCEEBASE,...)
```

- Installation of the TCP/IP product from a vendor:

```
//LIBDEF*,SEARCH=(PRD2.TCPIP,PRD1.BASE,PRD2.SCEEBASE...)
```

5. Add the following JCL statement to the CICS startup JCL or the batch JCL, where *xx* matches the *xx* that is specified in the ID=*xx* parameter in the corresponding TCP/IP startup job:

```
//OPTION SYSPARM='xx'
```

Multiple TCP/IP partitions can be active at the same time with different IDs, which is how the application requesters can route the TCP/IP function request to the correct TCP/IP server if more than one TCP/IP server is running in the same z/VSE system. If you do not specify `//OPTION SYSPARM`, the default ID is 00.

6. Give the z/VSE partition running TCP/IP a higher priority than the partition that is running the DB2 for VSE database server and CICS.

Make sure that TCP/IP support is started at system initialization time. The DB2 server uses it if TCP/IP for VSE is available. You must make known the listening port of the DB2 application server.

You can customize the listening port in the following ways:

- ▶ Specify the port number in the DBNAME directory of the destination database with the TCPSPORT parameter. See “Setting Up the DBNAME Directory” in Chapter 2, “Planning for Database Generation,” in *DB2 Server for VSE System Administration*, SC09-2981.

Specifying the port number in the DBNAME directory is our preferred method. The database administrator maintains the directory that resides in a VSE library. Because more than one DB2 Server for VSE can run on the same z/VSE system, ensure that they do not use the same TCP/IP port. Otherwise, the users will be connected to the wrong database. Identify the port numbers in the database directory, which makes it easier for you to ensure that different servers are using different ports.

- ▶ Use the new DB2 initialization parameter, TCPSPORT, to specify the port number to listen on. See *DB2 Server for VSE & VM Operation*, SC09-2986, for a detailed description of the TCPSPORT parameter.

Use this method when you initially test TCP/IP support or when TCP/IP support must be enabled, but the DBNAME directory cannot be updated. The disadvantage of this method is that another application might be using the same port. If this occurs, you receive an error message during initialization showing a BIND failure with return code 1115 indicating that the port was already in use by another application.

- ▶ Use the default, well-known port number 446, if it is not already in use.

This method is the least desirable. If no port number is specified in the DBNAME directory for the application server or you did not specify TCPSPORT as the initialization parameter, the default is used. The default port is called *ddm-rdb*, and the associated port number is 446. It is the well-known port assignment for relational databases in z/VSE and z/VM. If you use this default, you do not need extra configuration for TCP/IP for VSE and the DB2 application server. The disadvantage is that only one application server on the z/VSE system can use the definition.

If you initiated TCP/IP support, the TCP/IP agent is created for the application server to handle all TCP/IP-related functions. If the TCP/IP agent detects a TCP/IP function failure, TCP/IP support for the application server is disabled. You can restart TCP/IP support for the application server without recycling the application server by using the START TCPIP operator command. (See *DB2 Server for VSE & VM Operation*, SC09-2986, for a detailed description of the START TCPIP command.) If you set the TCPRETRY parameter to Y, the restart is done automatically by the database manager.

### 3.6.2 Preparing the application requester to use TCP/IP

If you use the DB2 Server for VSE & VM, you can use the included application requester to access remote DB2 databases. The application requester works in context of the calling application. Customize it for online applications in CICS and for batch applications. The instructions for preparing the application requester for TCP/IP are the same as the instructions in 3.5, “Setting up DB2 Server for VSE Client Edition” on page 75. Also, see *DB2 Server for VSE System Administration*, SC09-2981, for defining the CICS definitions to use DRDA support.

### 3.6.3 Enabling DRDA for DB2 Server for VSE

Enable the DRDA protocol for communications between the DB2 Server for VSE and remote databases. Based on the selected interface, link the appropriate phase (linkbook). For details, see 3.9, “DRDA interfaces in z/VSE” on page 102.

### 3.6.4 DB2 block size considerations in z/VSE and z/VM

A *query block* is the basic unit of transmission for query and result data. A requester can specify the size of query blocks in the `qryblkksz` instance variable of the commands that can return data.

The commands that specify `qryblkksz` are OPNQRS, EXCSQLSTT, and CNTQRS. Specifying the query block size enables the requester, which can have resource constraints, to control the amount of data that is returned at any one time.

DRDA defines two types of blocking:

**Exact blocking-every** The query block must be exactly the same size. The only exception to this rule is the last query block in the reply chain, which can be smaller.

**Flexible blocking-each** The query block can be a different size, depending on the size of the row or result set that is returned. The specified query block size is used as an initial size, and the query block can expand beyond that size, if necessary, to complete the fetch operation.

Adjust the query block size so that it matches the MTU size of the network minus 40 bytes. DB2 sends the contents of a result set in larger chunks, called *blocks*. When the system sends data over the network, each block is divided into as many network packets as needed to transport the complete block.

If the block size is, for example, 32768 bytes (32K), and the network's MTU size is 1492 bytes, every packet holds up to 1452 bytes of data. The value 1452 is the result of the following equation:

$$(1492 - 20 \text{ bytes IP header}) - 20 \text{ bytes TCP header} = 1452$$



**Byte size:** The size 1492 is a typical Ethernet MTU size.

Based on this value, the system needs 23 packets to send the complete block. This is based on the following equation:

$$32768 / 1452 = 23$$

The first 22 packets contain 1452 bytes of data, and the last packet contains only 824 bytes.

For a real Ethernet network, this example does not cause any problems. However, for a HiperSockets network, when running DB2 on Linux on System z, with an MTU size of 32K (32768 bytes) for example, the block size of 32K is most inefficient. Here, a network packet can carry up to 32728 bytes based on the following equation:

$$(32768 - 20 \text{ bytes IP header}) - 20 \text{ bytes TCP header}$$

Therefore, the block is sent in two packets. The first packet contains 32728 bytes. However, the second packet contains only 40 bytes, which is a great waste of resources.

For use with HiperSockets, either reduce the BLOCK SIZE so that a complete block fits into a packet (that is 32K minus about 40 bytes), or increase the MTU size (that is to 64K). Instead of reducing the BLOCK SIZE, you can also increase the BLOCK SIZE to about 60K, so that it takes two almost complete network packets to transport the complete block.

The block size considerations are most important when you transfer large result sets. If you transfer less data, the result set might not even occupy a full block. In this case, the previous calculations might not be valid, but should still not cause any harm.

You can change the block size by updating the QRYBLKSIZE parameter in the SQLGLOB file for z/VSE and the SQLINIT parameter for z/VM.

For more information, see *DB2 Server for VSE & VM Database Administration*, SC09-2888.

## 3.7 DB2 Server for VM Client Edition

DB2 Server for VM Client Edition is only required if you want local z/VM applications to access the remote DB2 database on Linux on System z. To install DB2 Server for VM Client Edition, you must have Virtual Machine Serviceability Enhancements Staged/Extended (VMSES/E) installed on your z/VM. The product is installed and serviced by using the new user ID 5697F75C, which is the IBM suggested user ID. Although you can override this user ID, create a Product Parameter File (PPF) override.

DB2 Server for VM Client Edition requires the following virtual machines in order to run:

- ▶ A DB2 Server for VM Client Edition client machine  
This machine owns the DB2 Server for VM Client Edition service, production, and SFS directories. We refer to it in this book as the SQLCLNT machine, but you can define it with any valid machine identifier (user ID).
- ▶ A DB2 Server for VM Client Edition user machine  
One or more of these machines is required. The default user machine is SQLUSER.

### 3.7.1 Setting up the communication directory

Create an entry in the communications directory, COMDIR, to enable the DB2 application requester function to access the remote DB2 database by using TCP/IP. Set up COMDIR to provide the host and service names. The following example shows the TCP/IP COMDIR entry:

```
:nick.LNXVM1 :service.DB2LNX
              :host.LNXDB2
              :security.PGM
              :userid.DB2INST1
              :password.MYLINPW
              :dbname.BIGLNxDB
```

For more information, see *DB2 Server for VM System Administration*, SC09-2980.

**Important:** Before any database maintenance is applied on the database client machine, run the following ARIRDBMA EXEC command to indicate that DRDA code must be built after maintenance:

```
ARIRDBMA DRDA (AR=Y)
```

For more details about the installation steps, see Chapter 6, “Installing DB2 for VM Client Edition,” in *Program Directory for DB2 Server for VM Client Edition*, GI11-8336.

## 3.8 DB2 Server for VM

If you want to communicate with remote databases, enable the TCP/IP protocol for the DB2 server on z/VM. Also install the DRDA code for the following connections:

- ▶ Inbound connections to enable access from remote databases
- ▶ Outbound connections for use with the application requester to access remote databases such as DB2 on Linux

For details about the DRDA setup, see Chapter 15, “Using a DRDA Environment,” in *DB2 Server for VM System Administration*, SC09-2980.

### 3.8.1 Preparing the application server to use TCP/IP

To activate the application server to use TCP/IP:

1. Install and configure TCP/IP for VM.
2. Access the TCP/IP client data disk. This disk is often defined as TCPMAINT 592. The disk contains the TCPIP DATA file and the ETC SERVICES file, which are necessary for successful TCP/IP support initialization.
3. Create the ARICTCP MODULE and store it on the production disk. You can find the instructions for creating the ARICTCP MODULE in *Program Directory for DB2 Server for VM*, GI10-4998.
4. Make sure that a C runtime library is available. You can use the SCEERUN LOADLIB library that is provided with z/VM or the C runtime library that is provided with Language Environment for VM.

5. Optional: Update the ETC SERVICES file with the RESID of the application server and the port number to use.

The system initialization invokes TCP/IP. If TCP/IP for VM is available, the server uses it. The application server must be able to determine the port number to listen on for connections.

You can configure the port number in any of the following ways:

- ▶ In the ETC SERVICES file on the TCP/IP maintenance disk, you associate a port number with the RESID of the application server.

This method is the *preferred* method. The TCP/IP administrator maintains this file on the TCP/IP client data disk. Because many application servers can run on the same VM system, ensure that they do not use the same TCP/IP port because only one server can use a port. Identifying the port numbers in this file makes it easier to ensure that different servers are using different ports. The name that you use must be the application server's RESID. You can determine the name by using the SHOW INITPARM command on the application server.

The entry in the ETC SERVICES file consists of a service name, port number, protocol, and an optional comment. The only protocol that DB2 Server for VM recognizes is TCP. The entries in the ETC SERVICES file are case sensitive. When you use the RESID as the search criteria, it is entirely in uppercase. The TCP/IP services use these entries, which is why you must define it all in uppercase. Similarly, the protocol is also part of the search criteria and is specified entirely in lowercase. Define it in lowercase for the search to be successful.

Table 3-4 shows an example of entries for two DB2 Server for VM application servers.

Table 3-4 Databases and their corresponding port or protocol

RESID	Value	Description
SQLPROD	6100/tcp	Production database
SQLTEST	6200/tcp	Test database

- ▶ You have specified the port number to listen on at the new TCPPOINT initialization parameter.

This method of port identification uses the new initialization parameter, TCPPOINT. The advantage of this method is that the ETC SERVICES file does not need to be updated. This is helpful when you initially test TCP/IP support or when enable TCP/IP support, but the ETC SERVICES file cannot be updated. The disadvantage of this method is that it is possible that another application might be using the same port.

- ▶ You have used the well-known port number 446 with the service name ddm-rdb. You might need to define ddm-rdb in the ETC SERVICES file.

This method is the least desirable one. If there is no entry for the RESID in the ETC SERVICES file or TCPPOINT was not specified, there is a well-known port assignment for relational databases on z/VSE and z/VM. The port assignment is called *ddm-rdb*, and the port number is 446. This name must be defined in the ETC SERVICES file. The file that is shipped with TCP/IP for VM has this definition, which has the advantage that no extra configuration is necessary to TCP/IP for VM and the application server. The disadvantage of this method is that only one application server on that VM system can use that definition.

You can use TCP/IP-related functions if TCP/IP support is initiated for the application server that you use. If the TCP/IP agent detects any TCP/IP function failure, TCP/IP support for the application server is disabled. You can restart TCP/IP support for the application server

without recycling the application server. Use the TCP/IP START operator command. (For a detailed description of this command, see *DB2 Server for VSE & VM Operation*, SC09-2986.) The database manager automatically restarts TCP/IP if the TCPRETRY parameter is set to Y.

### 3.8.2 Preparing the application requester to use TCP/IP

Enable the application requester for TCP/IP in the DB2 Server for VM to access a remote database. The instructions for preparing the application requester for TCP/IP is the same as discussed in 3.7, “DB2 Server for VM Client Edition” on page 99.

## 3.9 DRDA interfaces in z/VSE

DRDA is an IBM architecture with which you can use databases on different systems with different architectures. With DRDA, your databases communicate with other databases by using TCP/IP or Advanced Program-to-Program Communication (APPC) as the underlying protocol. In this book, we use DRDA with TCP/IP only.

By using the DRDA protocol, you can accomplish the following tasks:

- ▶ Access data that resides in DB2 Server for VSE & VM application servers from other DB2 systems that are equipped with the DRDA application requester function. If you use platforms, such as Linux, UNIX, AIX®, OS/400®, z/OS, or Windows, you can run applications that use DRDA remote unit of work or DRDA distributed unit of work processing.
- ▶ Use the DB2 Server for VSE & VM application requester function for remote unit of work access to work with data on other DB2 application servers. You can access data that might otherwise remain unavailable. You do not need any additional components such as DB2 Connect.

**Requirement for sufficient free space:** The re-linked components with DRDA code build new phases in z/VSE that need more space than the components without DRDA code. The corresponding libraries must have enough free space.

### 3.9.1 DB2 Server for VSE & VM interface considerations

You can compile applications in z/VSE to use one of the following DRDA interfaces that are available with DB2 Server for VSE & VM:

- ▶ Language Environment C interface  
The Language Environment C interface uses the Language Environment and adds significant code path length to the DB2 VSE application requester.
- ▶ CSI assembler interface  
The CSI assembler interface uses the assembler interface to perform TCP/IP functions and avoids the overhead that is needed by the Language Environment C interface.
- ▶ EZASMI assembler interface  
The EZA assembler interface (EZASMI) is available only with the releases DB2 Server for VSE & VM V7R5 and DB2 Server for VSE Client Edition V7R5. Similar to the CSI assembler interface, the EZASMI assembler interface avoids the overhead of the Language Environment C interface.

If you use the DB2 Server for VSE, you can use the Language Environment C interface or the EZASMI assembler interface for TCP/IP functions. For the DB2 Server for VSE Client Edition, you can use the Language Environment C, CSI Assembler, or EZASMI assembler interface.

**Important:** Always use the EZASMI assembler interface.

Looking at Table 3-5, you can choose the job control member that you need to enable DRDA support with the Language Environment C, CSI assembler, or EZASMI TCP/IP interface on the application server and application requester. In the installation described in this book, we used ARIS75ZD, ARIS75LF, and ARIS75Z.

Table 3-5 DB2 components and associated job members for corresponding TCP/IP interfaces

Application server or application requester	TCP/IP interface	Job control member
Application server	Language Environment C	ARIS752D
Application server	EZASMI assembler interface	ARIS752Z
Online application requester	CSI assembler interface	ARIS755D
Online application requester	Language Environment C	ARIS75UD
Online application requester	EZASMI assembler interface	ARIS75ZD
Batch application requester	CSI assembler interface	ARIS75LD
Batch application requester	Language Environment C	ARIS75LB
Batch application requester	EZASMI assembler interface	ARIS75LF

You have two linkbooks for the online resource adapter:

- ▶ If you link-edit the shipped online resource adapter phase, you automatically use the first linkbook.
- ▶ To complete the enablement of the DB2 Server for VSE online DRDA application requester support, use the second linkbook.

For more information about the DRDA interfaces, see *DB2 Server for VSE System Administration*, SC09-2981.

### 3.9.2 PROTOCOL parameter considerations

With the PROTOCOL parameter, you specify the types of protocols that the application server can process and the types of protocol that the application requester can use. For the application requester, you specify the PROTOCOL parameter in the SQLINIT EXEC. The PROTOCOL parameter has three options on the application requester:

- ▶ SQLDS
- ▶ AUTO
- ▶ DRDA

With PROTOCOL(SQLDS), you can only connect to DB2 servers on z/VM or z/VSE. The DB2 Server for VSE application requester function, for example, can only connect to another DB2 Server for VSE application server. When you specify PROTOCOL(AUTO) or PROTOCOL(DRDA), the DB2 Server for VSE application requester can connect to DB2 Server for VSE application servers and other DB2 servers on other platforms.

On the application server, you specify the `PROTOCOL` parameter in the `SQLSTART EXEC`. The `PROTOCOL` parameter has two options, `SQLDS` and `AUTO`, on the application server. When you specify `PROTOCOL=SQLDS`, the DB2 Server for VM application server allows access from DB2 Server for VM application requesters only. The application requesters and application servers can be in either a local or remote environment. When you specify `PROTOCOL=AUTO`, the DB2 Server for VM application server allows access from DB2 Server for VM application requesters and from distributed DB2 systems, such as DB2 on Linux.

Specify the `AUTO` option on the DB2 Server for VM application server, because it has many advantages. For example, with this option, the application server can receive both `SQLDS` protocol and `DRDA` protocol, from both DB2 Server for VM application requesters and remote DB2 application requesters. If you specify the `AUTO` option on the DB2 application requester on z/VM, it makes the necessary handshakes for DB2 server on z/VM and remote DB2 servers.

If you specify `PROTOCOL(DRDA)` on the application requester, the `DRDA` protocol is forced for connections, even if the target is a DB2 server on z/VSE or z/VM. The `DRDA` option is useful when you are using a DB2 requester on z/VSE or z/VM and a remote DB2 server on an ASCII platform such as Linux. You can also use the `DRDA` option to test SQL extensions, such as a larger block size, that are available only in a `DRDA` protocol environment.

Using the `PROTOCOL` parameter affects the `CCSID` conversion. If either the application requester or application server specifies `SQLDS` for the `PROTOCOL` parameter, the application requester default `CCSIDs` are ignored, and the application server `CCSIDs` are assumed.

Application requester `CCSIDs` are used if you have one of the following situations:

- ▶ Both the application server and the application requester specify the `AUTO` option.
- ▶ The application server is started with the `PROTOCOL=AUTO` option, and the application requester specifies `PROTOCOL(DRDA)` on the `SQLINIT EXEC`.

## DB2 data migration and application dependencies

If you want to use existing applications on z/VSE or z/VM to access the database on Linux on System z, you must move the data to the new database on Linux on System z. This is the step that your database administrators probably think of first. This step can be time consuming to perform, depending on the data migration method and the amount of data involved.

An additional step that you must do to rehost the database is to migrate the database access packages that are used from a database management system (DBMS) to access data. Move or regenerate the database packages and the package authorizations.

In this chapter, we describe an efficient way to move data from a DB2 Server for VSE & VM to a DB2 database on Linux on System z. In addition, we discuss the following topics:

- ▶ “Data migration” on page 106
- ▶ “Migrating the database access packages” on page 111
- ▶ “Application considerations” on page 114
- ▶ “Moving to production” on page 120

## 4.1 Data migration

Migrate all data from a DB2 Server for VSE & VM to DB2 for Linux on System z. You can perform this migration by using one of two methods. The first method is a traditional one in which your DB2 database administrator uses the utilities of DB2 Server for VSE & VM and DB2 9.7 for Linux, UNIX, and Windows, Enterprise Server Edition. This method entails using the DB2 Server for VSE & VM commands DBSU DATAUNLOAD and DBSU DATALOAD or the DB2 for Linux commands EXPORT and IMPORT. The DBSU DATALOAD option requires a Distributed Relational Database Architecture (DRDA) connection to load the data. This data is transmitted in blocks, where QRYBLKSIZE determines the block size. The data is inserted into the DB2 for Linux database, which creates a network overhead and incurs all of the logging overhead of each insert.

The second method is an efficient way that entails using DB2 for Linux federation to migrate all data from a DB2 Server for VSE & VM to DB2 for Linux on System z. This method is the one we describe in this book. Figure 4-1 shows a scenario where BIGLNxDB is used as a federated database. VSEDB2 and VMDB2 are defined as remote databases. In this scenario, you can copy the data from the federated VSEDB2 database into the BIGLNxDB federated database.

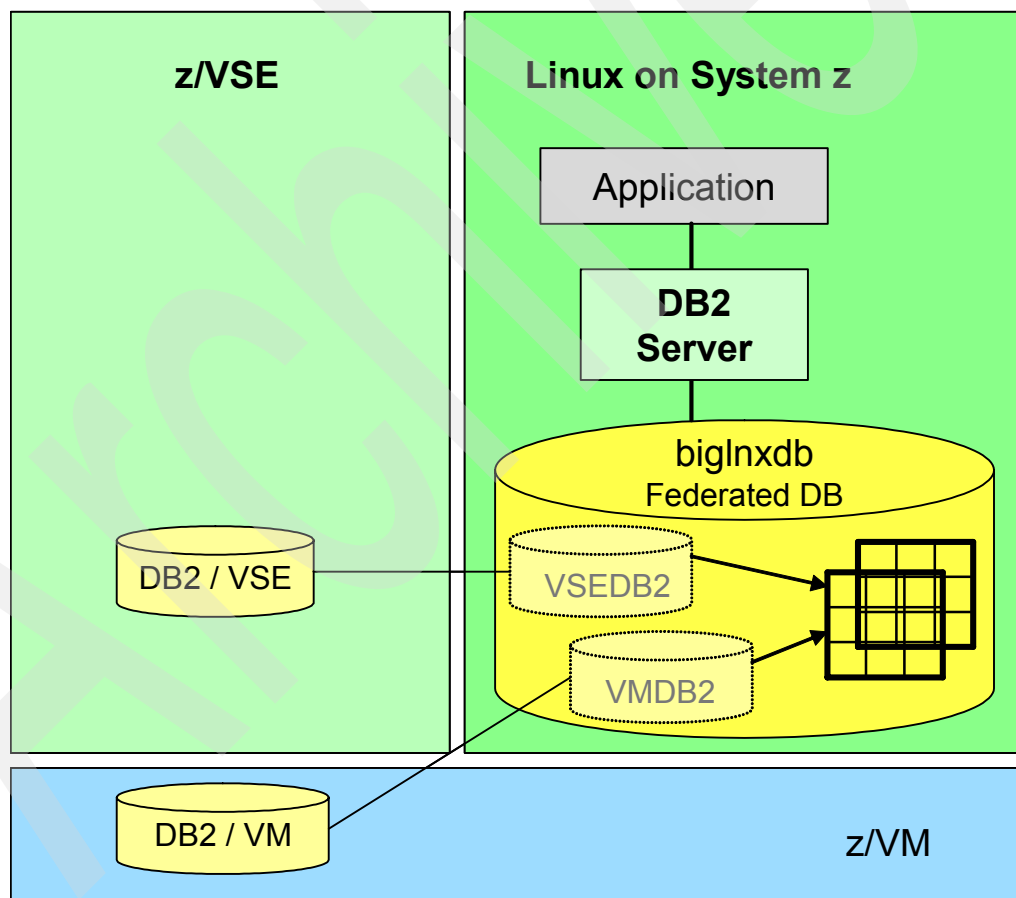


Figure 4-1 Federated data migration to DB2 on Linux



## 4.1.1 Prerequisites for data migration

For the data migration, consider the following environmental characteristics:

- ▶ Ensure that DB2 Server for VSE & VM Version 7 Release 5 or later is installed and is running on a z/VM or z/VSE system.
- ▶ Ensure that DRDA is enabled on the DB2 Server for VSE & VM server:
  - For DB2 Server for VM, start the server with the `PROTOCOL=``AUTO` parameter.
  - For DB2 Server for VSE, start the server with the `RMTUSERS=``n`, where `n > 0` and indicates the number of concurrent remote users.

If your database initialization fails because of these parameters, the DRDA code has not been enabled. For more information about how to enable the DRDA protocol, see 3.9, “DRDA interfaces in z/VSE” on page 102.

- ▶ Ensure that TCP/IP support for DB2 Server for VSE & VM is enabled. Verify that the database server can accept TCP/IP connections from remote clients and the z/VM or z/VSE requesters can use TCP/IP to connect to the remote database server. If TCP/IP support is not enabled, see 3.6, “DB2 Server for VSE” on page 96.
- ▶ Ensure that DBSU and ISQL packages are installed on the target DB2 Server for VSE & VM server. See Chapter 3, “Planning for Database Migration,” in *DB2 Server for VM System Administration*, SC09-2980, or *DB2 Server for VSE System Administration*, SC09-2981, for instructions about how to create DBSU and ISQL packages.
- ▶ Ensure that the SQLDBDEF packages are installed on the DB2 Server for VSE & VM. For instructions about how to load the SQLDBDEF package and use the SQLDBDEF utility on z/VM, see Appendix G, “Service and Maintenance Utilities,” in *DB2 Server for VM System Administration*, SC09-2980. On z/VSE, see the same appendix, but in the book *DB2 Server for VSE System Administration*, SC09-2981.
- ▶ Use DB2 9.7 for Linux, UNIX, and Windows, Enterprise Server Edition or a later version, to have the cursor load function available. DB2 for Linux on System z exists only as the Enterprise Server Edition.

### Restrictions:

- ▶ With DB2 Server for VSE & VM, you can use schemas and table names that begin with SYS and SQL. However, such prefixes are not allowed in DB2 for Linux.
- ▶ With DB2 Server for VSE & VM, you can use VARCHAR (32767) or LONG VARCHAR, which has a maximum length of 32767. This is greater than the maximum that is allowed by DB2 for Linux. You might need to modify *Create table definitions* in these cases to avoid the potential for data truncation.

## 4.1.2 Migrating data from DB2 Server for VSE & VM

In this section, we demonstrate the data migration steps on an implementation example with DB2 Server for VSE & VM. Before you begin, gather information about the DB2 Server for VSE & VM and DB2 for Linux. For this example, use the system definitions shown in Table 3-1 on page 40, Table 3-2 on page 40, and Table 3-3 on page 41.

To migrate the data:

1. Decide if you want to use an existing DB2 for Linux database or create a new one to be the target database for the data from DB2 Server for VSE or DB2 Server for VM. In this example, we use the existing database called BIGLNxDB.

2. Configure the DB2 for Linux database to be enabled for federated database support. Verify the FEDERATED parameter by entering the following command:

```
db2 get database manager configuration
```

Look for FEDERATED variable, which should look similar to the following line, meaning that the federated option is enabled:

```
Federated Database System Support          (FEDERATED) = YES
```

If the federated option is enabled, continue with step 3.

If you need to make changes, connect to BIGLNxDB with a user ID that has SYSADM authority. Update the database manager configuration to allow federated database support. You can perform these actions by entering the following commands:

```
db2 connect to biglnxdb user db2inst1
db2 update dbm cfg using federated yes
```

3. Configure the DB2 for Linux database to accept incoming TCP/IP connections from remote requesters:

```
db2 get database manager configuration
```

In the results of this command, look for the TCP/IP Service name variable:

```
TCP/IP Service name          (SVCENAME) = db2c_db2inst1
```

The server is configured for the TCP/IP protocol with the service name db2c\_db2inst1. You do not need additional settings. Continue with step 4.

If you need to make changes, update the SVCENAME with db2c\_db2inst1 in the database manager configuration, if necessary. For details, see 3.2.2, “Configuring DB2 for Linux on System z as a server” on page 51.

Verify that the service name db2c\_db2inst1 exists in the etc/services file in Linux. Set the DB2COMM DB2 for Linux environment variable for TCP/IP communications. Restart the DB2 for Linux instance to apply these changes. You can perform these actions by entering the following commands:

```
db2 update dbm cfg using svcename db2c_db2inst1
db2set db2comm=TCPIP
db2stop
db2start
```

4. Configure the DB2 Server for VSE & VM database to accept incoming TCP/IP connections and use DRDA to be accessible from remote requesters. The DRDA code for the DB2 Server for VSE & VM server must be enabled. For details, see 3.9, “DRDA interfaces in z/VSE” on page 102.
5. Configure DB2 for Linux on System z to access the DB2 Server for VSE & VM remote database and test the connection.

Enter the necessary **catalog** commands for the DB2 for Linux client to connect to the DB2 Server for VSE & VM server. Enter the **db2 terminate** command to apply the new directory changes.

To access DB2 on z/VSE, set up DB2 Linux as explained in 3.2.3, “Configuring DB2 for Linux on System z for access to remote databases” on page 52.

Test whether you can connect to the DB2 Server for VSE by entering the following command:

```
db2 connect to VSEDB2 user db2admin using myvsepw
```

You should see the following information:

#### Database Connection Information

```
Database server      = SQL/DS VSE 7.5.0
SQL authorization ID = DB2ADMIN
Local database alias = VSEDB2
```

To access DB2 on z/VM, enter the following commands on Linux:

```
db2 catalog tcpip node tcpvmnod remote MYVMSYS server 8030 system VMDB2 ostyle
VM
db2 catalog db tcpvmnod as VMDB2 at node tcpvmnod authentication dcs
db2 catalog dcs db tcpvmnod as VMDB2
db2 terminate
```

At this point, test whether you can connect to the DB2 Server for VM:

```
db2 connect to VMDB2 user db2admin using myvmpw
```

You can use the database connection information that is returned to verify that you have successfully connected to the DB2 Server for VSE & VM database.

6. On the DB2 for Linux server, use BIGLNxDB as the federated server. Create a DRDA wrapper for federated access.

Sign on to the DB2 for Linux server with a user ID that has SYSADM authority. You can use the DB2 Control Center for graphical setup as demonstrated in 3.3, "Setting up a federated database environment" on page 54, or you can use commands.

Enter the **create wrapper** commands on Linux:

```
db2 connect to BIGLNxDB
db2 create wrapper DRDA
```

7. On the DB2 for Linux server, create a server for federated access to the remote DB2 Server for VSE & VM server.

You can use the DB2 Control Center for graphical setup as demonstrated in 3.3, "Setting up a federated database environment" on page 54, or you can use commands.

For z/VSE, enter the following commands on Linux:

```
db2 create server vsedb2 type db2/vse version '7.5' wrapper drda authid
"db2admin" password "myvsepw" options (add dbname 'vsedb2', password 'Y')
```

For z/VM, enter the following commands on Linux:

```
db2 create server vmdb2 type db2/vm version '7.5' wrapper drda authid
"db2admin" password "myvmpw" options (add dbname 'vmdb2', password 'Y')
```

8. On the DB2 for Linux server, create a user mapping for federated access to the remote DB2 Server for VSE & VM server.

You can use the DB2 Control Center for graphical setup as shown in 3.3, "Setting up a federated database environment" on page 54, or you can use commands.

Create a user mapping that maps your DB2 for Linux user ID to a user ID and password on the DB2 server system.

For z/VSE, enter the following commands on Linux:

```
db2 create user mapping for "db2inst1" server "vsedb2" options (remote_authid
'db2admin', add remote_password 'myvsepw')
```

For z/VM, enter the following commands on Linux:

```
db2 create user mapping for "db2inst1" server "vmdb2" options (remote_authid
'db2admin', add remote_password 'myvmpw')
```

9. Test the federated access from DB2 for Linux to the remote DB2 Server for VSE & VM database server.

Perform a select from a table where you are sure that it exists only on the DB2 Server for VSE & VM database and not on your local DB2 for Linux database. A system catalog table is a good choice.

```
db2 connect to biglnxdb
db2 set passthru vsedb2
db2 select * from SYSTEM.SYSOPTIONS
db2 set passthru reset
db2 connect reset
```

Figure 4-2 shows how your environment should look in DB2 Control Center.

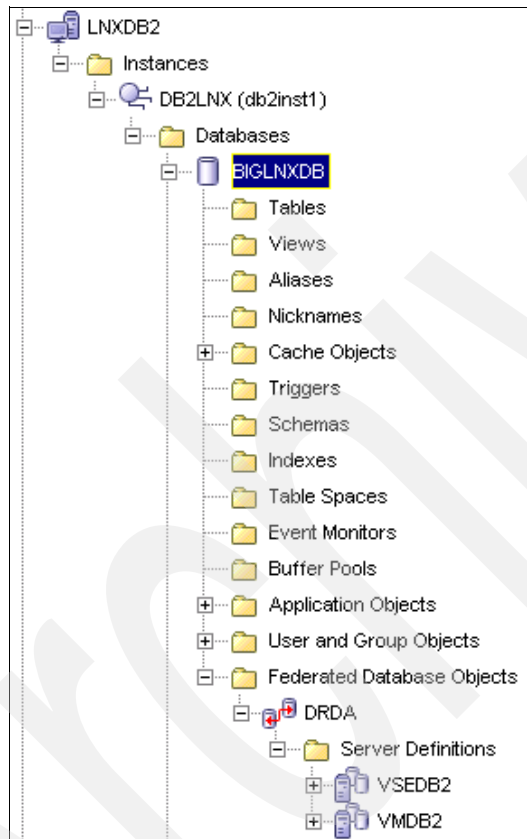


Figure 4-2 Object tree federated database BIGLNxDB and connections to VSEDB2 and VMDB2

10. Determine which tables on the DB2 Server for VSE & VM server you want to migrate to the DB2 for Linux server.
11. For each table identified:
  - a. Create a nickname in Linux for the remote table on DB2 Server for VSE & VM. You can use commands or the DB2 Control Center for a graphical setup as shown in 3.3, “Setting up a federated database environment” on page 54.
  - b. Create a new table in BIGLNxDB, whose definition matches the table definition on the remote DB2 Server for VSE & VM. Use SQLDBDEF to export table definitions from z/VSE or z/VM and import them in DB2 on Linux.
12. Declare a cursor for SELECT from the nickname created in step 11 a.

13. From DB2 for Linux, enter the `load` command, specifying the cursor in step 12, and set the target to the new table that we created in step 11 b.

Because there are a significant number of tables that you must handle with similar commands for each table, you can use some tools to aid with this task.

We assume that you want to copy all of the tables that belong to the DBSPACE named SDBS on the DB2 Server for VSE server. You can use the `SQLDBDEF` utility on DB2 Server for VSE to access the table definitions. Then use the `DBSU` utility to create the new tables on the DB2 for Linux server.

You can also use the small C program `crtnick.sqc` that can run on DB2 for Linux to access the DB2 Server for VSE & VM server and obtain the list of table names. This program creates a DB2 for Linux command file that creates the necessary nicknames. It also creates a second command file that declares the cursors to perform the `load` command.

You can find the source code for the `crtnick.sqc` program in “The `crtnick.sqc` C program” on page 183. Because this program uses embedded SQL, perform the prepare and bind processes to the target DB2 database BIGNXDB and then compile it on DB2 for Linux on System z. The program runs as a DB2 for Linux client that accesses data from the remote DB2 Server for VSE & VM. It creates two output files in the directory where the program was started:

- ▶ The `crtnickname.ddl` file contains the `create nickname` commands.
- ▶ The `crtnload.ddl` file contains the declare cursor and load statements.

This program is basic and does not perform extensive error checking and input parameter handling. The output file names are hardcoded in numerous places. The SQL `SELECT` statement is hardcoded for our example. You can adapt this sample to suit your own needs.

After all the data is in the new database on Linux, you can use SQL queries to verify the data. Now you can drop the `VSEDB2` and `VMDB2` servers from the federated database BIGNXDB. To verify the behavior of your z/VSE applications and the queries, build access packages, as explained in the following section, for the applications in the DB2 for Linux server.

## 4.2 Migrating the database access packages

For applications that work with databases, execute the bind process before you use such applications. The bind process has BIGNXDB specified as the target database.

The bind process creates access packages to efficiently access the database. These packages are required in the database server on DB2 for Linux on System z. Therefore, the existing packages on DB2 Server for VSE & VM must be regenerated or transferred to DB2 for Linux. Whenever possible, recreate these packages on the DB2 for Linux database server by using the batch or online bind process on z/VSE or z/VM, and avoid the transfer of existing packages. These steps guarantee the use of the latest optimizer technology in DB2 for Linux.

**Bind file support:** Bind file support for batch applications is available only in the DB2 Server for VM and VSE Version 7 Release 5 or later.

## 4.2.1 Creating access packages

*Binding* is the process of creating packages in the target database by using the bind file as input. Preprocessing with the BIND option and binding are complementary to each other. You can invoke binding multiple times to bind the packages to one or more databases. Ensure that the bind file is not corrupted or modified to achieve proper results. You can create packages with binding only for COBOL, C, PL/I, and assembler preprocessed programs.

In z/VM, you can use the SQLBIND exec to perform binding. SQLBIND exec cannot be used if the DB2 server is running in single user mode to bind to DB2 Server for VM.

For each application server specified, the SQLBIND exec performs the following actions:

1. Establishes a link to the application server
2. Binds the bind file to the application server to create the package
3. Displays summary messages that show the results for this binding
4. Creates two files, which contain a summary and output of the binding, with file names that are the same as the bind file and the file type BINDLST and BINDOUT

See “Binding to Create Package” in Chapter 4, “Preprocessing and Running a DB2 Server for VM Program,” in *DB2 Server for VSE & VM Application Programming*, SC09-2889.

If you run z/VSE batch or CICS applications in z/VSE against DB2 application servers that do not run on z/VM or z/VSE, perform the following actions:

- ▶ When you preprocess your application, specify the BIND option so that a bind file is created.
- ▶ For CICS applications and batch applications, compile and link-edit the output from the preprocessor to create a z/VSE phase.

**CBND transaction:** Use the CBND transaction or batch binder to generate a package in each DB2 application server that the application will access. The CBND transaction uses the bind file that was created in the preprocessor step.

## 4.2.2 Creating a package by using CBND and batch binding

Binding can be performed through an online binder (CICS transaction CBND) or batch binder (program ARIPBIN). Both CBND and batch binder require the bind file to create successfully during preprocessing. The same bind file can be used as input for CBND or ARIPBIN.

The output of the preprocessing with the BIND option is called *bind record* or *bind file* in z/VSE. The bind record contains parameters and modified SQL statements for each preprocessed program. During the preprocessing step in z/VSE, one bind record per program is written to the KSDS VSAM bind file as a series of records. The bind records for multiple programs can be written to the VSAM bind file, unlike z/VM that creates one CMS bind file for each program.

You can use UNLOAD and RELOAD commands in DB2 on z/VSE and z/VM to export the packages from DB2 on z/VSE or z/VM and import them into DB2 for Linux.

### 4.2.3 Binding online applications using CBND

If you generate a bind record after preprocessing, you can use it to create a package in a remote or local application server, which enables the CICS online application program to access a DRDA server. To convert the SQL statements inside a bind file into a package, use the CBND CICS transaction. You can use CBND to bind all applications whose bind file was generated by the preprocessor to DB2 for Linux.

To create a package locally, for a FORTRAN application, you still need to run the batch preprocessor. Executing the preprocessor with the BIND option and executing CBND are complementary. The preprocessor creates and stores the bind file in a VSAM file, and CBND reads the bind file back into an application server. If you are using the CBND command, you must be the owner of the program whose bind file you are binding to a remote server or a database administrator.

A second level transaction, CB2D, is required for CBND to complete the bind process. The CB2D transaction is invoked internally by CBND through an EXEC CICS START command. One or more CB2D transactions can be started by one CBND transaction based on the number of application servers that is specified in the CBND DBLIST input parameter. Each CB2D is responsible for connecting to a target database and creating a package. This transaction must be defined during installation to fully enable the CBND function.

### 4.2.4 Binding batch applications by using the batch binder

For batch binder, the same rules apply as for the online binder. If you generate a bind record, you can use it to create a package in a remote or local application server. Use the batch binder to bind all applications whose bind record was generated by the preprocessor, to DB2 for Linux. Executing the preprocessor with the BIND option and executing batch binder are complementary.

To perform binding, you can invoke the batch binder by calling batch binding phase ARIPBIN in your job control language (JCL). Use the same DLBL statement in the batch binder JCL as for SQLBIND that you used at the time of preprocessing the program. Binder does not require a bind work file. The batch binder gets the parameters through the PARM parameters.

Example 4-1 shows a batch binding job that takes the bind records from the master file SQLBIND and generates the NEW package.

*Example 4-1 Batch binder for bind records from the SQLBIND file*

---

```
// JOB BATCH BINDER
// LIBDEF *,SEARCH=(STSPVT.DEVELOP,STSPVT.DB2750)
// ASSGN SYSIPT,YSRDR
// DLBL VSEUCAT,'VSEMCH.USER.CATALOG',,VSAM
// DLBL SQLGLOB,'V750.DB2750.GLOB.FILE',,VSAM,CAT=V750CAT, X
DISP=(OLD,KEEP)
// DLBL SQLBIND,'V750.DB2750.BIND.FILE',,VSAM,CAT=V750CAT, X
DISP=(OLD,KEEP)
// EXEC PGM=ARIPBIN,SIZE=AUTO,PARM='NEW'
/*
/ &
```

---

**Note:** When you use the JCL shown in Example 4-1, specify the correct LIBDEF statement and library. The following example shows a typical invocation of the ARIPBIN phase with bind parameters:

```
// EXEC ARIPBIN,SIZE=AUTO,PARM='PACKAGE(VSEAPP.PACK),DB(DBNAME),X  
USERID(USER1/PASSWD1),OWNER(OWNER1),NEW'
```

For detailed information about binding parameters, see “Creating a Package Using CBND and Batch Binding” in Chapter 5, “Preprocessing and Running a DB2 Server for VSE Program,” in *DB2 Server for VSE & VM Application Programming*, SC09-2889.

## 4.2.5 Package loading and rebinding

You produce the packages with the DB2 bind and store them in the DBMS. When the application server loads the package, it checks that the package is still valid. A package might not be valid if one of its dependencies, such as an index that the package uses, has been dropped.

Packages are also invalidated when primary keys and referential constraints are added to, dropped from, activated, or deactivated on tables on which the modules depend. The following rules apply:

- ▶ If a primary key is added, dropped, activated, or deactivated, all packages that have a dependency on the parent table are invalidated, including any tables that have a foreign key relationship with the parent table.
- ▶ If a foreign key is added, dropped, activated, or deactivated, all packages that have a dependency on the dependent table or parent table are invalidated.

The system has an internal change management facility that keeps track of whether packages are valid. If a package is valid, the system begins running the program. If the package is not valid, the system tries to re-create it. The original SQL statements are stored with the package when you preprocess the program. The system uses them to automatically bind the program again. It does this dynamically, that is, while it is running. If the rebinding works, a new package is created and stored in the database and the system then continues execution of the program. If the rebinding does not work, an error code is returned to the program in the SQLCA, and the program stops running.

A successful rebinding has no negative effect on your program except for a slight delay in processing your first SQL statement. To minimize this delay, you can use the DBSU REBIND PACKAGE command to rebind the invalid package after it is invalidated, but before it is executed. For more information and commands to create a valid bind package, see *DB2 Server for VSE & VM Database Services Utility*, SC09-2983.

DB2 Server for VSE & VM V7R5 includes bind files for DBSU and the ISQL utilities ARIDSQLB and ARISQBD. The system-defined stored procedure bind files are also available with the product. You can use these bind files directly to create the packages.

## 4.3 Application considerations

If you use applications that are designed to work with a relational database, you use the logic of the SQL statements to select the required data. That means the applications use a standard interface to relational databases. Therefore, the applications can be easily directed



to access data on a DB2 for Linux on System z relational database instead of accessing data on DB2 Server for VSE or DB2 Server for VM.

The applications on z/VSE or z/VM have some unique characteristics that you need to consider. For example, in applications, you typically code a `CONNECT` statement to a database to authenticate yourself with a user ID and password. Each user has certain rights in a database. Some applications do not use explicit `CONNECT` statements to connect to a database. For these applications, make additional definitions to allow them to access the data.

Another example is when you run applications, and code page conversion is done between the DB2 Linux server and the DB2 Server for VSE & VM. This conversion occurs because of the nature of z/VM and z/VSE to be EBCDIC systems and of Linux on System z to be an ASCII system. The data receiver is responsible for the correct data conversion. The communication between the databases on z/VSE and Linux uses the DRDA protocol to transfer the data. There might be issues, such as collating sequences and sort orders, that you must consider. For more information, see 2.5.2, “ASCII and EBCDIC considerations” on page 37.

You must also consider the compatibility of SQL statements, which we discuss in 4.3.4, “Compatibility of DB2 VSE with DB2 for Linux, UNIX, and Windows” on page 117.

### 4.3.1 Implicit connect setup

If you use applications that are not doing an explicit `CONNECT` to a database, they use the implicit connect mechanism. For these applications, adapt the database manager configuration in DB2 for Linux to enable the authentication to take place in z/VSE instead of DB2 for Linux.

After you set the database manager configuration parameters (`dbm cfg`) on DB2 for Linux database, allow implicit connections from DB2 Server for VSE batch and online application requesters to DB2 for Linux as follows:

```
AUTHENTICATION=CLIENT
TRUST_ALLCLNTS=DRDAONLY
TRUST_CLNTAUTH=YES
```

By using implicit connect for batch workload, you do not have to specify a user ID and password in every batch job. For batch applications, you also specify the implicit user ID in the `DBNAME` directory. In this directory, you define other server parameters such as the IP address and port number.

Therefore, z/VSE batch applications do not need a `CONNECT` statement in the application program in the following situations:

- ▶ The parameter `IMPLUSER=<userid>` is defined in the `DBNAME` directory.
- ▶ The parameter `// SETPARM USERID = <userid>` is defined in the JCL that invokes the application program.

### 4.3.2 Connection pooling

If z/VSE applications work with DB2 on Linux, a performance issue can be the high number of connects that occur. To mitigate this problem, connection pooling was introduced in DB2 Server for VSE Client Edition. A connect process establishes a physical TCP/IP connection and a logical connection between DB2 on z/VSE and DB2 on Linux. The logical connection is established if you initiate a database `CONNECT` statement.

If you have established connection pooling, the logical connection uses a physical connection from the pool, which saves on the overhead of establishing a physical connection for every logical connection. Establishing and closing the physical TCP/IP connection are time consuming operations. You can avoid these operations during SQL CONNECT time if you created a pool of TCP/IP connections at the time of enabling the online resource manager. Use this pool only when running SQL CONNECT. By using connection pooling, you do not have to destroy the connection when running COMMIT RELEASE, switching databases, or ending a transaction. Instead, the connect is returned to the pool to be reused by another user.

With connection pooling, you can defer the destruction of these links until CIRT/CIRR when the server is disabled. You can choose this connection pooling mechanism for a particular server that is connected over TCP/IP. Use the CONNPOOL option entry in the DBNAME directory.

The number of connections in the pool can be determined based on the maximum number of concurrent application connections that are established to the DB2 server on Linux. To avoid having applications wait for connections, ensure that you have 60% to 70% of the maximum number for concurrent application connections specified in CONNPOOL. The system releases connections only when the transaction terminates or issues COMMIT or ROLLBACK with RELEASE option. Connections are not relinquished on a simple COMMIT or ROLLBACK.

The connection pooling feature is targeted for users who frequently switch between databases or use CONNECT in their applications. In these cases, performance improvements have been reported at up to 80%. Applications that stay CONNECTED with the remote server through the day with just one CONNECT statement are not good candidates for connection pooling because they might lock up the connection for the entire day.

### Connection timeout

The DB2\_SERVER\_CONTIMEOUT parameter was introduced with DB2 for Linux, UNIX, and Windows V9R1. With DB2\_SERVER\_CONTIMEOUT, you can adjust the time, in seconds, that an agent waits for a client's connection request before terminating the connection. In most cases, you do not need to adjust this registry variable. However, if DB2 clients are constantly timing out by the server at connect time, you can set a higher value for DB2\_SERVER\_CONTIMEOUT to extend the timeout period. If an invalid value is set, the default value is used. A value of 0 indicates no timeout. This registry variable is not dynamic. The default is 180 seconds, but you can specify 0 to 32767 seconds.

**Tip:** Disable this timeout function in DB2 for Linux, UNIX, and Windows if you use connection pooling. Otherwise a bad link (-933) condition might occur.

You can only disable the parameter if you have installed the service that is available at the following address:

<http://www-01.ibm.com/support/docview.wss?uid=swg1LI72085>

After you install this service, you can disable the timeout by entering the following command:

```
DB2_SERVER_CONTIMEOUT=0
```

### 4.3.3 Fetch and insert blocking

With blocking, you can improve performance, because data is sent between your program and the database manager in blocks of rows rather than one row at a time. This process reduces overhead from communications between the application server and the requester. Most applications that do multiple row insertions or retrievals benefit from blocking.

To use blocking, specify the BLOCK parameter during DB2 preprocessing of the program. You can block homogenous insert statements in the program by specifying the IBLOCK parameter while preprocessing the program.

You gain performance by using the following cursors:

- ▶ Insert cursors that use OPEN, PUT, and CLOSE statements
- ▶ Fetch cursors that use OPEN, FETCH, and CLOSE statements

If you use the DRDA protocol, the application requester can set the block size for fetch and insert blocking.

You can set the block size for fetch and insert blocking from a minimum value of 512 bytes to a maximum value of 32 KB minus 1 byte (32767 bytes) by using either the QRYBLKSIZE option of the SQLGLOB file in z/VSE or the QRYBLKSIZE option of the SQLINIT EXEC in z/VM.

#### Multirow buffered insert

The multirow buffered insert feature enables the DB2 Server for VM client and z/VSE batch application requesters to support buffered inserts. Users have the option of blocking their PUT SQL statements by using the existing BLOCK preprocessing parameter. They also have the option of blocking INSERT statements by using a new IBLOCK preprocessor parameter.

**Blocking:** Blocking is only useful if the block size is sufficiently large that many rows can be blocked and fit in the specified block size.

### 4.3.4 Compatibility of DB2 VSE with DB2 for Linux, UNIX, and Windows

SQL is the query language for DB2 databases. Because the syntax for DB2 on z/VSE differs from DB2 for Linux, UNIX, and Windows, full compatibility does not exist between the query languages for these databases.

#### SQL characteristics in DB2 Server for VSE & VM

An SQL statement cannot be greater than 8192 bytes. This limit is based on the 8K buffer that is used to process each SQL statement. To bypass the limitation, move more than one column in a single line and avoid qualifiers for unique columns.

SQL statements can be slow because of a high number of result rows. You can optimize the response time with the OPTIMIZED FOR n ROWS parameter in DB2 for Linux.

Table 4-1 on page 118 lists the functions that are supported by DB2 Server for VSE & VM and their compatibility with DB2 for Linux on System z. Although most of the SQL statements will work for both platforms, not all are supported because of inherent differences in the DRDA levels that are supported. Consider alternate SQL statements based on the requirement, error messages, and level of support.

Table 4-1 Functions supported by DB2 Server for VSE & VM and DB2 for Linux on System z

Function	Compatibility
AVG	Function works the same.
COUNT	Function works the same.
MAX	Function works the same.
MIN	Function works the same.
SUM	Function works the same.
CHAR(time-expression)	Function works the same if the date-time format chosen is the same.
CHAR(timestamp-expression)	Function works the same if the date-time format chosen is the same.
CHAR(date-expression)	Function works the same if the date-time format chosen is the same.
CHAR(decimal-expression)	Function is not working the same. For DB2 Server for VSE & VM, a leading blank is added if the decimal expression is positive. DB2 for Linux, UNIX, and Windows requires you to add a trailing blank if the decimal expression is positive.
DATE	Function works the same if the date format chosen for the database is the same for both.
DAY	Function works the same.
DAYS	Function works the same.
DECIMAL	Function works the same.
DIGITS	Function works the same.
FLOAT	Function works the same except for the display format.
HEX	Function returns an EBCDIC character for DB2 Server for VSE & VM or ASCII character for DB2 on Linux, UNIX, and Windows.
HOUR	Function works the same.
INTEGER	DB2 Server for VSE & VM supports only numeric arguments, while DB2 on Linux, UNIX, and Windows supports numeric, character-string, and date and time expressions as arguments.
LENGTH	Function works the same.
MICROSECOND	Function works the same.
MINUTE	Function works the same.
MONTH	Function works the same.
SECOND	Function works the same.
STRIP	Function works the same.
SUBSTR	Function works the same.
TIME	Function works the same if the time format chosen for the database is the same for both.
TIMESTAMP	Function works the same.

Function	Compatibility
TRANSLATE	Function works the same.
VALUE	Function works the same.
YEAR	Function works the same.

For further information about SQL considerations, see *DB2 Server for VSE & VM SQL Reference*, SC09-2989.

### Extended dynamic SQL considerations

Keep in mind the following considerations for extended dynamic SQL:

- ▶ Only DB2 Server for VSE & VM supports extended dynamic SQL statements. Extended dynamic SQLs do not work against DB2 for Linux when used with parameter markers.
- ▶ The DBSU Dataload facility does buffered inserts. This feature is not enabled to be used with online user applications including ISQL on z/VSE and z/VM.
- ▶ The size of the buffer that is used to block inserts depends on the QRYBLKSIZE parameter in the SQLINIT for VM and SQLGLOB in z/VSE.
- ▶ Tailor your application programs to use the multirow buffered insert feature. All homogenous insert statements are executed immediately one after the other and can exploit the advantages of blocking.
- ▶ Homogenous insert statements are defined as a set of insert statements that access either the same DB2 table or the same set of columns in the same sequence in that table.
- ▶ In the following scenarios, the buffer is sent to the DB2 for Linux, UNIX, and Windows server:
  - An SQL COMMIT statement is encountered after a set of homogenous insert statements.
  - The buffer is full with a set of homogenous insert statements.
  - A non-insert SQL statement is encountered after a set of homogenous SQL insert statements.
- ▶ The following restrictions apply for the multirow buffered insert feature:
  - A COMMIT must be issued after a batch of INSERTs. Otherwise, data integrity issues can occur.
  - The AUTOCOMMIT option is disabled for buffered INSERTs.
  - A batch of INSERTs followed by a non-COMMIT statement might result in an incorrect SQLCODE for the non-COMMIT statement that follows the INSERTs.

### 4.3.5 Transferring database object permissions

When you move object permissions from the original DB2 Server for VSE database to the database on DB2 for Linux on System z, you must adjust it with DB2 for Linux security concepts of groups and roles.

Privileges and authorities granted to groups are not considered when creating views, materialized query tables (MQTs), SQL routines, triggers, and packages that contain static SQL. To avoid this restriction, use roles instead of groups. By using roles, user can create database objects with their privileges acquired through roles, which are controlled by the DB2

database system. Groups and users are controlled externally from the DB2 database system, for example, by an operating system, in this case Linux on System z.

For more information about the roles, see the following papers in IBM developerWorks®:

- ▶ *New features in DB2 9.5 to help your business grow*  
<http://www.ibm.com/developerworks/db2/library/techarticle/dm-0706see/>
- ▶ *Implement DB2 for Linux, UNIX, and Windows trusted contexts and roles in a Web application*  
<http://www.ibm.com/developerworks/db2/library/techarticle/dm-0803eak1e/index.html>

For details about roles and groups in DB2, see the “AUTH\_LIST\_AUTHORITIES\_FOR\_AUTHID table function” topic in the DB2 for Linux, UNIX, and Windows Information Center at the following address:

<http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.sql.rtn.doc/doc/r0052898.html>

## 4.4 Moving to production

After the data migration, take time to test all applications, with intensive tests in a production-type environment. Therefore, it is important that you set up a coexistence environment between the DB2 Server for VSE & VM environment and the environment on DB2 for Linux on System z for these tests. Such an environment gives you the opportunity to test production with the DB2 VSE setup and to perform pre-production tests in parallel to verify the same results.

The easiest way to enable a coexisting environment is to enable, on DB2 Server for VSE & VM, the capture and apply features on DB2 for Linux on System z. The changes in DB2 Server for VSE & VM are transferred based on a predefined time to the DB2 database on Linux that keeps the databases synchronized. You can also perform parallel testing and tuning on the DB2 for Linux on System z database. To establish this environment, you use the traditional replication DB2 method. If the test is successful, you can confidently switch your production work to the DB2 for Linux database.

Figure 4-3 illustrates the DB2 coexistence pre-production setup scenario.

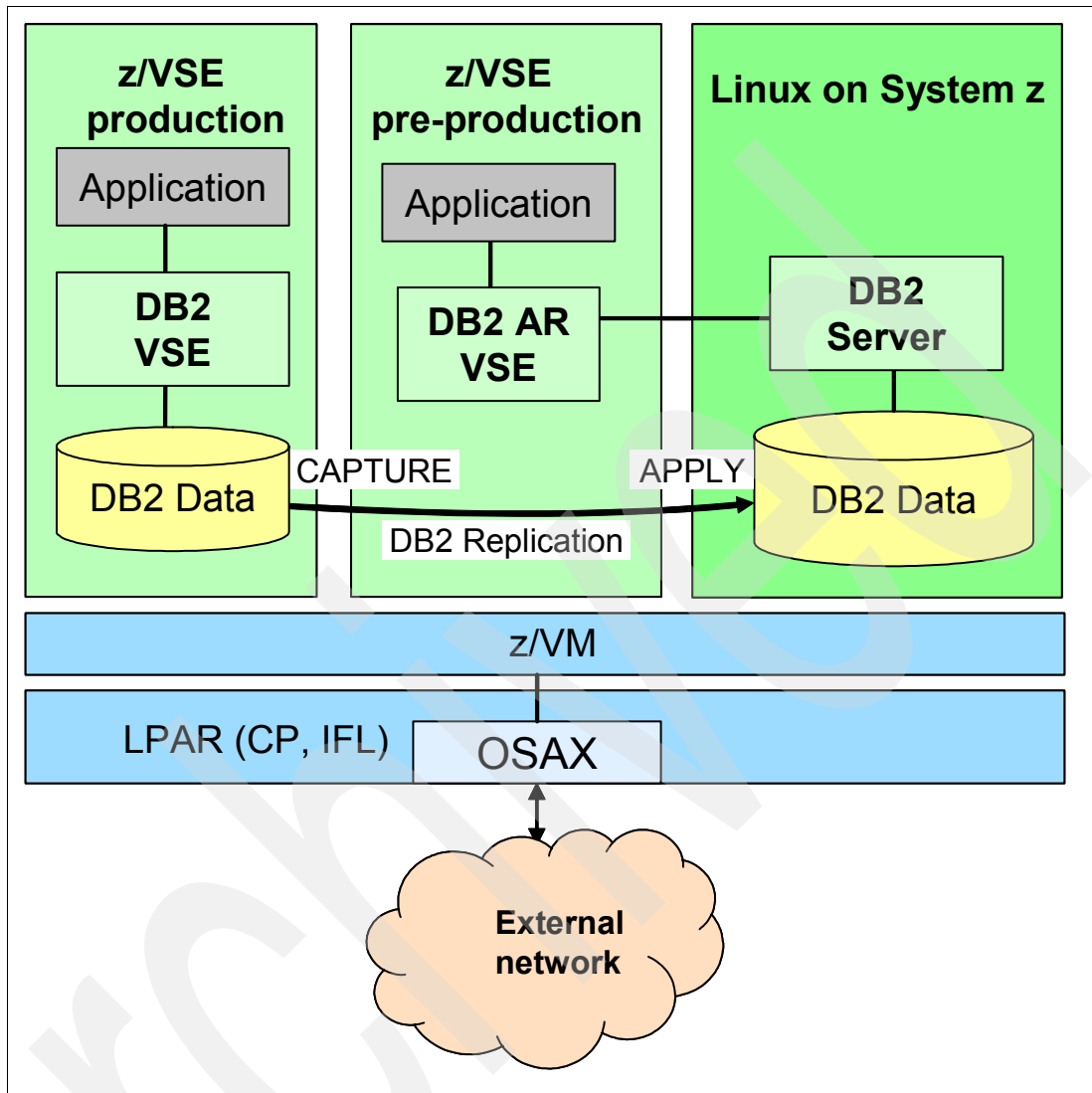


Figure 4-3 DB2 coexistence pre-production scenario

Archived



## Monitoring and tuning

The purpose of monitoring and tuning is to obtain optimal performance of your workload. Performance refers to the way in which a computer system behaves in response to a particular workload. It is measured in terms of system response time, throughput, and resource utilization. Performance is also affected by the resources that are available on the system and how well those resources are used and shared.

In this chapter, we discuss the impact of different layers or subsystems regarding performance. We show how to find possible bottlenecks and tune them. We begin this chapter by providing general information about performance based on best practice activities in monitoring and tuning.

Specifically we discuss the following topics:

- ▶ “Best practice activities for monitoring and tuning” on page 124
- ▶ “Performance activities” on page 126
- ▶ “Linux system monitoring and tuning” on page 127
- ▶ “DB2 for Linux on System z monitoring and tuning” on page 135
- ▶ “Network monitoring and tuning” on page 143
- ▶ “z/VSE system monitoring and tuning” on page 148
- ▶ “DB2 Server for VSE & VM monitoring and tuning” on page 161
- ▶ “Monitoring z/VSE applications and tuning recommendations” on page 163

## 5.1 Best practice activities for monitoring and tuning

Keep in mind the following guidelines when developing an overall approach to performance tuning:

- ▶ Remember the law of diminishing returns.  
The greatest performance benefits usually come from your initial efforts.
- ▶ Do not tune just for the sake of tuning.  
Tune to relieve identified constraints. If you tune resources that are not the primary cause of performance problems, you can make subsequent tuning work more difficult.
- ▶ Consider the whole system.  
You cannot tune one parameter or resource in isolation. Before you make an adjustment, consider how the change will affect the system as a whole. Performance tuning requires trade-offs among various system resources. For example, you might increase buffer pool sizes to achieve improved I/O performance, but larger buffer pools require more memory, and that might degrade other aspects of performance.
- ▶ Change one parameter at a time.  
Do not change more than one factor at a time. Even if you are sure that all the changes will be beneficial, you will have no way to assess the contribution of each change.
- ▶ Measure and configure by levels.  
Tune one of the following levels of your system at a time:
  - Hardware
  - Operating system
  - Application server and requester
  - Database manager
  - SQL statements
  - Application programs
- ▶ Check for both hardware and software problems.  
Some performance problems can be corrected by applying service to your hardware, your software, or both. Do not spend excessive time monitoring and tuning your system before applying service to the hardware or software.
- ▶ Understand the problem before you upgrade your hardware.  
Even if it seems as though additional storage or processor power might immediately improve performance, take the time to understand where the bottlenecks are. You might spend money on additional disk storage, only to find that you do not have the processing power or the channels to exploit it.
- ▶ Put fallback procedures in place before you start tuning.  
If tuning efforts result in unexpected performance degradation, reverse the changes that are made before you attempt an alternative approach. Save your original settings so that you can easily undo changes that you do not want to keep.

The performance improvement process is an iterative approach to monitoring and tuning aspects of performance. Depending on the results of this performance monitoring, you will adjust the configuration of the database server and make changes to the applications that use the database server.

Any performance improvement process includes the following fundamental steps:

1. Define the performance objectives.
2. Establish performance indicators for the major constraints in the system.
3. Develop and execute a performance monitoring plan.
4. Continually analyze monitoring results to determine which resources require tuning.
5. Make one adjustment at a time.

If an application or a set of applications does not run as fast as expected, you must become familiar with the affected systems and components. A distributed environment, such as z/VSE connected to a DB2 database server running on Linux on System z, requires a look at the entire picture rather than at each site or component separately.

Figure 5-1 shows the scenario and the components that are involved.

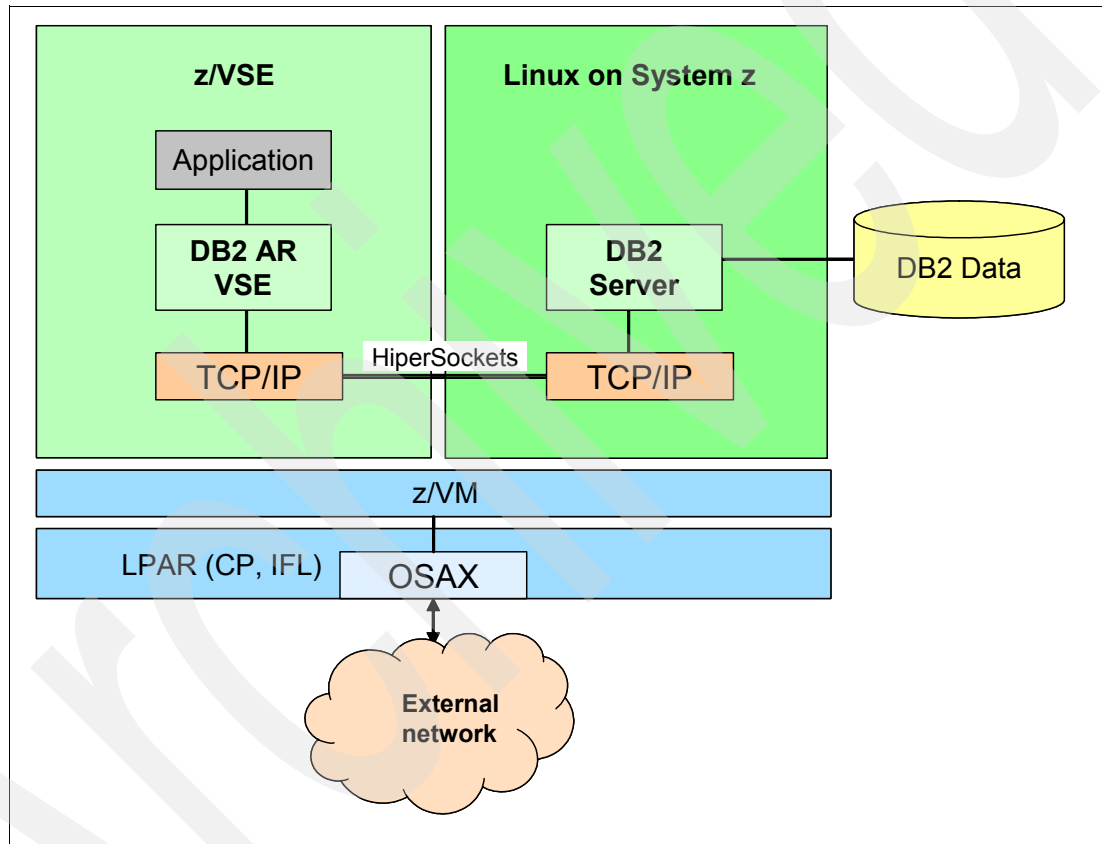


Figure 5-1 z/VSE and DB2 for Linux on System z

The data and requests flow through several different software layers and services on its way from the z/VSE application to the database server on Linux on System z and back. Each layer adds a certain amount of overhead and processing time. To learn why such an application does not perform as expected, you might need to look at each layer, which includes the z/VSE application itself, the DB2 application requester, the TCP/IP stack on both systems, the network, and DB2 for Linux on System z. In addition to the layers that are directly involved, you also must look at both operating systems, z/VSE and Linux on System z.

## 5.2 Performance activities

To tune such an environment, execute the performance activities in the following sections. The steps are usually repeated multiple times, until the system performs well enough or until there is nothing more to tune.

### 5.2.1 Collecting data

The data collection step involves the usage of various performance monitor tools and optionally tracing tools on all components that are affected.

In most situations, when a performance problem is reported, no monitored data is available to substantiate the claim. One of the first steps you must do is to establish a procedure to capture all relevant monitoring and tracing data. It is important to have consistent data from all components from the same test run, so that the different reports can be correlated based on timestamps.

Part of this step proves that the monitoring and tracing tools work as expected and produce the kind of output that is needed for further analysis. This part is sometimes frustrating for the participants, because they do not see any direct outcome. However, it is important to ensure that you obtain all data that is required for later analysis.

Be aware that using tracing tools might slow down processing. Also be aware of any other systems that are running concurrently on the same system, for example, other systems running in logical partitions (LPARs) or z/VM guests. Concurrent activities can take away processing power from your test system because of shared CPUs. Therefore, monitor all systems that share resources, including the test system.

### 5.2.2 Analyzing the data

The data analysis step uses the data that was collected in the previous step and analyzes it. Depending on the performance problem, concentrate on unusual behavior that catches your attention such as delays, high CPU utilization, long wait times, high water marks, and so on. When you start, concentrate on the big things. The longer you work on data analysis, the more details you will have to look into.

From common experiences, the effort to analyze data is based on the A-B-C method:

A - Concentrate on the essentials:	20% work for 80% results
B - More detailed analysis:	30% work for 15% results
C - Analyze all details:	50% work for 5% results

This method can help to correlate the quality and effort of a project.

Especially when a lot of network communication is involved, it might be useful to look at a TCP packet trace and look closer at the protocol that is used by the systems. Look for certain requests that flow over the network, and try to correlate that with information you see in the performance monitors, for example, high CPU utilization or long wait after receiving a certain request.

Also consider implications that are caused by other systems or applications that share the same resources. In many cases, the test system, for example, running in an LPAR or z/VM guest, does not get the full CPU power. Usually at least one production system is running concurrently in another LPAR or z/VM guest that might take away a reasonable amount of CPU power because it has a higher priority or z/VM share value.

### 5.2.3 Tuning the systems

Based on the results from analyzing the data, you tune the different parts of the environment. The goal is to change settings that influence the behavior when analyzing the data but in a positive way. Change only one parameter at a time to prove it has helped by performing another test run.

Only change those settings that have a good chance of influencing the findings from analyzing the data. If you are unsure about whether and how settings influence the whole system, do not change them. Before you make any configuration changes, make a backup of the current settings. Also, carefully document every change, so that you have a record of what you changed.

It might be helpful to strip down the test case to test a specific setting that can be run much easier and more often. Be aware that stripping down a test case can influence the behavior of the test. Good tuning results on the stripped down test case may or may not be as helpful on the full test case.

### 5.2.4 Running the test case

Next you re-run the test case to check whether the tuning has helped. Before re-running the test case, again activate the performance monitor and tracing tools. Make sure that the preconditions of the test case are the same on all test runs. Also ensure that the amount of data being processed is the same, which might require a complete restoration of the data before you re-run the test case.

## 5.3 Linux system monitoring and tuning

Document every run for performance purposes in a performance log book. Make only one change at a time to the system under investigation, because multiple changes can have opposite effects. For every subsequent run, document the expectations prior to the experiment. Test runs without expectations usually do not benefit tuning.

For all test runs, plan and use a common performance data collection. Even though the interest in a specific test run might be in a special area, always collect the full set of performance data. Too much randomness, which is good for system test, is bad for performance and tuning. Conditions and test workload for a test run should be repeatable. You might need to implement a “warmup” to achieve repeatability.

When looking at the results, take nothing for granted. Every behavior has a reason, and this reason must be known. To avoid side effects from other resource consuming systems (such as processors, processor caches, memory, OSA and FICON/Fibre Channel Protocol (FCP) ports, switches, and storage server) as much as possible, the best strategy is to run on a system without other workloads active during monitoring.

To simplify test case execution and time savings, some scripting for automatic data collection helps. For a valid performance test, the duration should be long enough to produce 50 to 100 snapshots of performance data. The data collection interval needs to be adapted accordingly.

For Linux system monitoring, you can use tools that are included into the Linux distribution, or you can use additional specialized monitoring tools to monitor Linux and the major applications in Linux such as DB2.

In the sections that follow, we describe which internal Linux facilities you can use for monitoring and tuning.

### 5.3.1 Monitoring with `sysstat` utilities

The `sysstat` utilities are a collection of performance monitoring tools for Linux. They contain the following parts:

<b>sadc</b>	Data gatherer that stores data in a binary file.
<b>sar</b>	Reporting tool that reads a binary file and converts it to readable output.
<b>mpstat</b>	Processor utilization.
<b>iostat</b>	I/O utilization.

All `sysstat` packages of version 7.0.0. or later include the processor “steal time.” This performance data is essential when running on Linux on System z and especially when Linux is running as a guest under z/VM. The installation of the `sysstat` package and configuration depend on the Linux distribution and crontab settings. By default, the data is collected in `/var/log/sa`. You can obtain more information about your Linux system by using the `man sar` command.

To collect test-case-specific data by using `sysstat`:

1. Activate `sadc` with a sample rate of 20 seconds. The sample rate depends on the duration of the measured workload. You should have a total 50 to 100 measured samples. Enter the following command to start `sadc`:

```
/usr/lib/sa/sadc 20 sadc_bin.out &
```

2. Execute the workload for about 30 minutes.
3. Stop `sadc` after the test by entering the following command:

```
kill <PID of sadc>
```

4. Generate the `sar_ascii.out` report file from the binary file:

```
/usr/bin/sar -A -f sadc_bin.out 1>sar_ascii.out 2>&1
```

The test document with performance data is the `sar_ascii.out` file.

### 5.3.2 Disk monitoring with the `iostat` utility

You can use the `iostat` utility to monitor current disk utilization. For example, to see a report with a monitor interval of 20 seconds with 30 repetitions (count):

1. Enter the following command:

```
iostat 20 30 -dmx > iostat.out
```

2. Start the workload or test case that you want to measure.
3. Compress the output file by entering the command:

```
tar cvfz iostat.tgz iostat.out
```

The output file can be large if several hundred disks are attached to the Linux system.

### 5.3.3 Disk statistics monitoring with the dasd statistics utility

The **dasd statistics** utility monitors the activities of the DASD driver and the storage subsystem. The output gives information about the total number of I/O requests, a histogram of request sizes, and various histograms on times spent in I/O execution.

You can use the **dasd statistics** utility to collect data for specific disks:

1. Optional: To reset the counters, enter the following command:

```
echo set off > /proc/dasd/statistics
```

2. Enable the **dasd statistics** utility:

```
echo set on > /proc/dasd/statistics
```

3. Run the workload.

4. To look at the statistics, enter the following command:

```
cat /proc/dasd/statistics
```

5. To print the statistics for each device, enter one command per device as in the following example:

```
tunedasd -P /dev/dasdxy  
tunedasd -P /dev/dasdxz  
...  
echo set off > /proc/dasd/statistics
```

### 5.3.4 SCSI disk statistics for the FCP

Statistical data on FCP devices can be collected starting with SUSE Linux Enterprise Server 9 SP3 and kernel version 2.6.5-7.283 or later and with SUSE Linux Enterprise Server 10 with kernel version 2.6.16.21-0.8 or later. The information contains histograms of read and write I/O request sizes and latencies for several phases of the I/O execution.

To perform the data collection:

1. Set the `CONFIG_STATISTICS=y` parameter in the kernel configuration file.

If the `debugfs` file system is mounted in the `/sys/kernel/debug/` directory, all the statistics data that is collected is in the `/sys/kernel/debug/statistics/` directory. The names of these subdirectories consist of `zfc<device-bus-id>` for an adapter and `zfc<device-bus-id>-<WWPN>-<LUN>` for a logical unit number (LUN). Each subdirectory contains two files, `data` and `definition`.

2. Switch on the data gathering for each device:

```
echo on=1 > definition
```

3. Switch off the data gathering again:

```
echo on=0 > definition
```

By default, the data gathering is turned off.

4. Reset the collected data to 0:

```
echo data=reset > definition
```

### 5.3.5 Linux process monitoring with the top utility

The **top** command prints system data for each process. It shows an overview of the currently running system processes. The **top** command consumes CPU resources.

In this section, we explain how to collect **top** data in batch mode where the non-default column “wait state” is included. First, make some modifications to the default settings. Adapt the “top” repetition frequency to the experiment, duration, and an additional display column for process wait states that helps later in the analysis. To have these settings in place for batch run for minimal system overhead, save them in a local file.

To prepare and run the test with the **top** performance data collection:

1. For the preparation, use the **top** interactive mode with a command prompt as shown in Example 5-1.

*Example 5-1 The top command prompts*

---

```
top
d          The delay time in seconds between screen updates
1          One sample per second
<enter>
f          You will see a screen specifying the field order on the top line,
           possible fields and short descriptions. Use the corresponding
           letter to enable or disable a field.
<enter>
W          Write current setup to ~/.toprc. This is the way to write a top
           configuration file.
q          Quit
```

---

The **top** command creates a `.toprc` file (Example 5-2).

*Example 5-2 Contents of the .toprc file*

---

```
Def fieldscur=AEHIOQTWKNMbcdfgjplrsuvYzX
  winflags=62777, sortindx=10, maxtasks=0
  summc1r=1, msgsc1r=1, headc1r=3, taskc1r=1
```

---

The uppercase letter “Y” in the fields definition displays information about the process wait state. A lowercase “y” indicates that this column is not shown.

2. Start the **top** command in batch mode as follows:

```
top -b -n 300 1>top.log 2>&1 &
```

The **top** command starts the collection of 300 samples, one per second at a run time of five minutes. Do not collect data for too long because the size of the `top.log` file increases. As soon as the test workload is started, start the data collection immediately, or better, start both the test workload and the data collection in parallel in a shell script.

After the test case is finished, compress the log file by entering the following command:

```
tar cvfz top.tgz top.log
```

### 5.3.6 Monitoring with the oprofile utility

The **oprofile** utility profiles all running code on Linux systems, providing a variety of statistics. By default, kernel mode and user mode information is gathered for configurable events. The System z hardware currently does not provide hardware performance counters. Instead the timer interrupt is used. Enable the `hz_timer`. The rate of the jiffy Linux interrupt



defines the timer value. This value cannot be changed by the user. The Novell SUSE distributions include the **oprofile** utility on the distribution CDs.

**The oprofile utility for Red Hat users:** At the time of writing this document, in the Red Hat distribution, you must download the oprofile utility and install it yourself.

Example 5-3 shows how to use the **oprofile** utility.

*Example 5-3 The oprofile utility*

---

```
sysctl -w kernel.hz_timer=1
gunzip /boot/vmlinuz-2.6.5-7.201-s390x.gz
opcontrol --vmlinuz=/boot/vmlinuz-2.6.5-7.201-s390x
opcontrol --start
<DO TEST>
opcontrol --shutdown
opreport
any new test to run? If yes
opcontrol --reset
opcontrol --start
...
opcontrol -start
```

---

In the following examples, we show the output with different granularities.

The **opreport** command gives an overview, as shown in Example 5-4, that includes how often the program counter is displayed in the kernel and each module during the **oprofile** sample period.

*Example 5-4 Output of the opreport command*

---

```
>opreport
CPU: CPU with timer interrupt, speed 0 MHz (estimated)
Profiling through timer interrupt
      TIMER:0|
samples|    %|
-----|
140642 94.0617 vmlinuz-2.6.16.46-0.4-default    <=== kernel
 3071  2.0539 libc-2.4.so                        <=== glibc
 1925  1.2874 dbench                            <=== application
 1922  1.2854 ext3                              <=== file system
 1442  0.9644 jbd                               <=== journaling
  349  0.2334 dasd_mod                          <=== dasd driver
  152  0.1017 apparmor                          <=== security
    6  0.0040 oprofiled
    5  0.0033 bash
    5  0.0033 ld-2.4.so
    1  6.7e-04 dasd_eckd_mod
    1  6.7e-04 oprofile
```

---

Example 5-5 shows the output of the `opreport -l` command, which gives a more detailed view than the `opreport` command. Without further specification, the `opreport -l` command resolves only the kernel symbols. Attempts to find the respective symbol files in other modules leads to warnings. Note that the statistics are not consistent, because they list kernel hits per symbol and all hits outside the kernel in one number as in Example 5-4 on page 131.

Example 5-5 Output of the `opreport -l` command

---

```

>opreport -l
warning: /apparmor could not be found.
warning: /dasd_eckd_mod could not be found.
warning: /dasd_mod could not be found.
warning: /ext3 could not be found.
warning: /jbd could not be found.
warning: /oprofile could not be found.
CPU: CPU with timer interrupt, speed 0 MHz (estimated)
Profiling through timer interrupt
samples %      app name                symbol name
130852  87.5141 vmlinux-2.6.16.46-0.4-default cpu_idle
1922    1.2854 ext3                      (no symbols)          <=== almost idle
1442    0.9644 jbd                        (no symbols)          <=== unresolved symbols
734     0.4909 vmlinux-2.6.16.46-0.4-default memcpy
662     0.4427 libc-2.4.so               strchr
619     0.4140 dbench                    next_token
567     0.3792 vmlinux-2.6.16.46-0.4-default do_gettimeofday
536     0.3585 vmlinux-2.6.16.46-0.4-default __link_path_walk
525     0.3511 vmlinux-2.6.16.46-0.4-default copy_to_user_std
435     0.2909 libc-2.4.so               strstr
413     0.2762 dbench                    child_run
349     0.2334 dasd_mod                  (no symbols)
347     0.2321 vmlinux-2.6.16.46-0.4-default _spin_lock
328     0.2194 vmlinux-2.6.16.46-0.4-default sysc_do_svc
285     0.1906 dbench                    all_string_sub
283     0.1893 vmlinux-2.6.16.46-0.4-default __d_lookup
251     0.1679 vmlinux-2.6.16.46-0.4-default __find_get_block
231     0.1545 libc-2.4.so               ___strtol_l_internal
216     0.1445 dbench                    vsnprintf
209     0.1398 vmlinux-2.6.16.46-0.4-default filldir64
205     0.1371 vmlinux-2.6.16.46-0.4-default memset
196     0.1311 vmlinux-2.6.16.46-0.4-default _atomic_dec_and_lock
166     0.1110 vmlinux-2.6.16.46-0.4-default strchr
155     0.1037 libc-2.4.so               memmove
152     0.1017 apparmor                  (no symbols)
148     0.0990 libc-2.4.so               readdir
147     0.0983 vmlinux-2.6.16.46-0.4-default __brelse
146     0.0976 vmlinux-2.6.16.46-0.4-default generic_file_buffered_write
144     0.0963 vmlinux-2.6.16.46-0.4-default generic_permission
140     0.0936 vmlinux-2.6.16.46-0.4-default __getblk
140     0.0936 vmlinux-2.6.16.46-0.4-default kmem_cache_free

```

---

In Example 5-6, we extended the `oprofile -l` command to include the symbol files from the modules that are displayed with hits in Example 5-4 on page 131. This command resolves the warnings from Example 5-5 on page 132 and gives a consistent list of mostly hit symbols in all areas.

Example 5-6 Output of the extended `oprofile -l` command

---

```
>oprofile -l
--image-path=/lib/modules/2.6.16.46-0.4-default/kernel/fs/ext3/,/lib/modules/2.6.16.46-0.4-defau
lt/kernel/fs/jbd/,/lib/modules/2.6.16.46-0.4-default/kernel/drivers/s390/block/,/lib/modules/2.6
.16.46-0.4-default/kernel/security/apparmor/,/lib/modules/2.6.16.46-0.4-default/kernel/arch/s390
/oprofile
CPU: CPU with timer interrupt, speed 0 MHz (estimated)
Profiling through timer interrupt
samples %      image name          app name          symbol name
130852 87.5141 vmlinux-2.6.16.46-0.4-default vmlinux-2.6.16.46-0.4-default cpu_idle
734      0.4909 vmlinux-2.6.16.46-0.4-default vmlinux-2.6.16.46-0.4-default memcopy
662      0.4427 libc-2.4.so          libc-2.4.so       strchr
619      0.4140 dbench              dbench            next_token
567      0.3792 vmlinux-2.6.16.46-0.4-default vmlinux-2.6.16.46-0.4-default do_gettimeofday
536      0.3585 vmlinux-2.6.16.46-0.4-default vmlinux-2.6.16.46-0.4-default __link_path_walk
525      0.3511 vmlinux-2.6.16.46-0.4-default vmlinux-2.6.16.46-0.4-default copy_to_user_std
435      0.2909 libc-2.4.so          libc-2.4.so       strstr
413      0.2762 dbench              dbench            child_run
361      0.2414 ext3.ko             ext3              ext3_get_block_handle
347      0.2321 vmlinux-2.6.16.46-0.4-default vmlinux-2.6.16.46-0.4-default __spin_lock
328      0.2194 vmlinux-2.6.16.46-0.4-default vmlinux-2.6.16.46-0.4-default sysc_do_svc
285      0.1906 dbench              dbench            all_string_sub
283      0.1893 vmlinux-2.6.16.46-0.4-default vmlinux-2.6.16.46-0.4-default __d_lookup
251      0.1679 vmlinux-2.6.16.46-0.4-default vmlinux-2.6.16.46-0.4-default __find_get_block
231      0.1545 libc-2.4.so          libc-2.4.so       __strtol_l_internal
226      0.1511 ext3.ko              ext3               ext3_try_to_allocate
223      0.1491 dasd_mod.ko         dasd_mod          dasd_smallocc_request
216      0.1445 dbench              dbench            vsnprintf
209      0.1398 vmlinux-2.6.16.46-0.4-default vmlinux-2.6.16.46-0.4-default filldir64
205      0.1371 vmlinux-2.6.16.46-0.4-default vmlinux-2.6.16.46-0.4-default memset
196      0.1311 vmlinux-2.6.16.46-0.4-default vmlinux-2.6.16.46-0.4-default
_atomic_dec_and_lock
188      0.1257 ext3.ko              ext3               ext3_new_inode
166      0.1110 vmlinux-2.6.16.46-0.4-default vmlinux-2.6.16.46-0.4-default strchr
157      0.1050 jbd.ko               jbd                journal_init_dev
155      0.1037 libc-2.4.so          libc-2.4.so       memmove
148      0.0990 libc-2.4.so          libc-2.4.so       readdir
147      0.0983 vmlinux-2.6.16.46-0.4-default vmlinux-2.6.16.46-0.4-default __brelse
146      0.0976 vmlinux-2.6.16.46-0.4-default vmlinux-2.6.16.46-0.4-default
generic_file_buffered_write
144      0.0963 vmlinux-2.6.16.46-0.4-default vmlinux-2.6.16.46-0.4-default generic_permission
140      0.0936 vmlinux-2.6.16.46-0.4-default vmlinux-2.6.16.46-0.4-default __getblk
140      0.0936 vmlinux-2.6.16.46-0.4-default vmlinux-2.6.16.46-0.4-default kmem_cache_free
```

---

### 5.3.7 Linux on System z system check

If you want to do an overall system check, IBM provides a script to collect important system data from Linux. The generation of this information is done by the `dbginfo.sh` script, which is included in the `s390-tools` package. This information is written to the `/tmp/DBGINFO-[date]-[time]-[hostname].tgz` file. `[date]` and `[time]` are the date and time when debug data is collected. `[hostname]` indicates the host name of the system from which the data was collected. The output files are compressed into a `.tgz` file.

The `.tgz` file can be sent to problem determination or performance analysis experts as initial material for investigation. The file contains a consistent view of versions, processors, memory, modules, peripheral devices, error messages, and configuration files. If your system runs as a z/VM guest, it contains some z/VM guest specific information.

The script does not read user data and does not modify any settings or add anything to the system. You have to run the `dbginfo.sh` script from the user root. Otherwise parts of the script cannot be executed because of missing privileges.

### 5.3.8 Additional tools for Linux and z/VM monitoring

You can use the following additional tools, among others, for z/VM and Linux on System z monitoring:

► IBM Tivoli® OMEGAMON® XE on z/VM and Linux

This tool gives you the ability to view data that is collected from multiple systems in one, flexible interface called the *Tivoli Enterprise Portal*. With Tivoli OMEGAMON XE on z/VM and Linux, you can monitor z/VM data obtained from the Performance Toolkit for VM. You can simultaneously monitor the performance data for the Linux on System z systems running on z/VM.

The Tivoli OMEGAMON XE on z/VM was enhanced for z/VM V5R4 with the following functions:

- Reporting of new data for control unit cache, DASD cache, virtual disks, virtual switches, and spin locks
- A default workspace to the z/VM systems in the navigation tree
- Enhanced functionality for modifications of the network workspace

► Performance Toolkit for VM

The Performance Toolkit for VM provides capabilities to monitor and report performance data for a z/VM system. It includes the following functions:

- Management of multiple z/VM systems, local or remote
- Post-processing of Performance Toolkit for VM history files and VM monitor data captured by the MONWRITE utility
- The ability to view performance monitor data using either Web browsers or PC-based 3270 emulator graphics
- TCP/IP performance reporting

In addition to analyzing VM performance data, the Performance Toolkit for VM processes Linux performance data that is obtained from the Resource Management Facility (RMF) Linux tool, `rmfpms`. The Linux performance data obtained from RMF can be viewed and printed similarly to the way z/VM data is presented.

- ▶ Tools from Velocity Software, Inc.

Velocity Software provides performance monitoring products for z/VM. Performance monitoring includes network data collection, analysis, and display of performance data for z/VM and Linux on System z. To learn more about the tools that are offered by Velocity Software, see the Velocity Software Web site at the following address:

<http://www.velocitysoftware.com/product.html>

For more information about monitoring tools, see “Online resources” on page 188.

## 5.4 DB2 for Linux on System z monitoring and tuning

DB2 for Linux on System z includes different tools to monitor and tune the activities of the database and the queries. Database monitoring is a vital activity for the maintenance of the performance and health of your database management system. To facilitate monitoring, DB2 collects information from the database manager, its databases, and any connected applications. You can use this information to perform the following tasks:

- ▶ Forecast hardware requirements based on database usage patterns.
- ▶ Analyze the performance of individual applications or SQL queries.
- ▶ Track the usage of indexes and tables.
- ▶ Determine the cause of poor system performance.
- ▶ Assess the impact of optimization activities, such as altering database manager configuration parameters, adding indexes, or modifying SQL queries.

Consider using performance monitoring tools, such as the following DB2 internal tools:

- ▶ DB2 Design Advisor
- ▶ DB2 Workload Manager
- ▶ DB2 Explain
- ▶ DB2 Snapshot™ monitor

### 5.4.1 DB2 Design Advisor

The DB2 Design Advisor tool can help you significantly improve your workload performance. You can use it to identify all of the objects that are needed to improve the performance of your workload. To launch DB2 Design Advisor, you use the `db2adv` command.

Given a set of SQL statements in a workload, the DB2 Design Advisor generates recommendations for the following areas:

- ▶ New indexes
- ▶ New clustering indexes
- ▶ New materialized query tables (MQTs)
- ▶ Conversion to multidimensional clustering (MDC) tables
- ▶ The redistribution of tables

You can decide whether the DB2 Design Advisor should implement some or all of these recommendations immediately. You can also schedule them to run at a later time.

The DB2 Design Advisor can help you from the planning phase to a workload-based tuned database. While designing your database, use DB2 Design Advisor to generate design alternatives in a test environment for indexing, MQTs, MDC tables, or database partitioning.

In partitioned database environments, you can use the DB2 Design Advisor in the following ways:

- ▶ Determine an appropriate database partitioning strategy before loading data into a database.
- ▶ Assist in upgrading from a single-partition database to a multipartition database.
- ▶ Assist in migrating from another database product to a multipartition DB2 database.

### **Workload performance tuning**

After your database is set up, you can use the DB2 Design Advisor to help you in the following ways:

- ▶ Improve the performance of a particular statement or workload.
- ▶ Improve general database performance by using the performance of a sample workload as a gauge.
- ▶ Improve the performance of the most frequently executed queries, as identified, for example, by the activity monitor.
- ▶ Determine how to optimize the performance of a new query.
- ▶ Use the Health Center in DB2 and respond to recommendations regarding shared memory utility or sort heap problems with a sort-intensive workload.
- ▶ Find objects that are not used in a workload.

## **5.4.2 DB2 Workload Manager**

With DB2 Workload Manager (WLM), you can perform the following monitoring tasks:

- ▶ Analyze the workload on your system to help design your initial DB2 Workload Manager configuration.
- ▶ Track and investigate the behavior of your system by obtaining types of operational information with which you can perform the following tasks:
  - Analyze system performance degradation.
  - Diagnose activities that are taking too long to complete.
  - Investigate agent contention.
  - Isolate poorly performing queries.

Information is available for activities, service classes, workloads, work classes, threshold queues, and threshold violations.

- ▶ Exercise control over the execution environment by canceling queued activities that you expect will cause problems or cancel running activities that you have diagnosed as negatively impacting the system.
- ▶ Apply the real-time monitoring with table functions. Real-time monitoring data includes information about work that is currently running on the system, statistics, and metrics. This is done for work that has been performed on the system to help you to determine usage patterns and resource allocation and identify problem areas. Use the DB2 table functions to obtain this operational information.
- ▶ Use historical monitoring with WLM event monitors. DB2 Workload Manager uses event monitors to capture information that might be of use in the future or for historical analysis.

You can use the DB2 Workload Manager to keep your system tuned and to automate tasks based on rules in WLM.

You can set up the following items, among others, by using WLM:

- ▶ Available monitoring data  
Monitoring data is available from workloads, service subclasses and service superclasses, work classes, and threshold queues. You can use this data to diagnose and correct problems and for performance tuning.
- ▶ DB2 Workload Manager stored procedures  
You can use stored procedures to cancel an activity, capture details about an activity, reset the statistics on DB2 Workload Manager objects, and set client information at the data server.
- ▶ Statistics for DB2 Workload Manager objects  
Statistics are maintained for DB2 Workload Manager objects including service classes, work classes, workloads, and threshold queues. These statistics reside in memory and can be viewed in real time by using the DB2 Workload Manager statistics table functions. The statistics can be collected and sent to a statistics event monitor where they can be viewed later for historical analysis.
- ▶ Monitoring metrics for DB2 Workload Manager  
Monitoring metrics provide data about the health of, and query performance on, your DB2 data server. They can be used as input to a tool from an independent software vendor (ISV) or in conjunction with additional scripting that you provide to analyze the metrics returned.
- ▶ Workload management table functions and snapshot monitor integration  
You can use the DB2 Workload Manager table functions with the snapshot monitor table functions when performing problem determination or performance tuning.
- ▶ Monitoring threshold violations  
When a DB2 Workload Manager threshold is violated, a threshold violation record is written to the active THRESHOLD VIOLATIONS event monitor, if one exists.
- ▶ Collecting data for individual activities  
You can use an ACTIVITIES event monitor to collect data for individual activities that run in your system. The data collected includes such items as statement text and compilation environment. The data can be used to investigate and diagnose problems, and it can be used as input to other tools, such as the DB2 Design Advisor.
- ▶ Importing activity information into the DB2 Design Advisor  
You can import activities that are collected by an activities event monitor into the DB2 Design Advisor to help you make decisions about the database objects that are accessed by these activities.
- ▶ Cancelling activities  
If an activity is consuming too many resources, or is running too long, you can cancel it. Cancelling an activity is gentler than forcing the application that submitted the activity. A cancelled activity returns SQL4725N to the user, but does not end the connection or affect any other user activity. Forcing the application ends both the connection and user activities.
- ▶ Guidelines for capturing information based on rules and thresholds  
DB2 Workload Manager provides functions to define guidelines for capturing information about a specific DB2 workload and for investigating rogue activities in your DB2 database.

► Workload management performance modelling

You can use WLM as a coordinator for a time-based workload. You can also use WLM to coordinate workload tasks based on elapse time of the workload entities. WLM schedules parallel workloads based on the amount of work tasks.

### Capturing information about an activity for later analysis

You can use workload management features to capture information about an activity for later analysis. In this section, we show an example of capturing information about an event to tune SQL queries.

Let us assume that you have a stored procedure called MYSCHEMA.MYSLOWSTP, which is running slower than usual. You begin to receive complaints about this situation and decide to investigate the cause of the slowdown. If investigating the problem while the stored procedure is running is impractical, you can capture information about the stored procedure activity and any activities nested in it.

Assuming that you have an active activities event monitor called DB2ACTIVITIES, you can create a work class for CALL statements that apply to the schema of the MYSCHEMA.MYSLOWSTP stored procedure. Then you can create a work action to map the CALL activity and all nested activities to a service class that has activity collection enabled. The CALL activity, and any activities nested in it, are sent to the event monitor.

Example 5-7 shows the Data Definition Language (DDL) entries that are required to create the DB2 Workload Manager objects.

*Example 5-7 DDL for workload management*

---

```
CREATE SERVICE CLASS SC1;
CREATE WORKLOAD WL1 APPLNAME ('DB2BP') SERVICE CLASS SC1;
CREATE SERVICE CLASS PROBLEMQUERIESSC UNDER SC1 COLLECT ACTIVITY DATA ON
COORDINATOR WITH DETAILS;

CREATE WORK CLASS SET PROBLEMQUERIES
(WORK CLASS CALLSTATEMENTS WORK TYPE CALL ROUTINES IN SCHEMA MYSCHEMA);

CREATE WORK ACTION SET DATABASEACTIONS FOR SERVICE CLASS SC1 USING WORK CLASS SET
PROBLEMQUERIES
(WORK ACTION CAPTURECALL ON WORK CLASS CALLSTATEMENTS MAP ACTIVITY WITH NESTED TO
PROBLEMQUERIESSC);
```

---

After the MYSCHEMA.MYSLOWSTP stored procedure runs, you can run the query, shown in Example 5-8, to obtain the application handle, the unit of work identifier, and the activity identifier for the activity.

*Example 5-8 Select statement for correlating queries*

---

```
SELECT AGENT_ID,
       UOW_ID,
       ACTIVITY_ID
FROM ACTIVITY_DB2ACTIVITIES
WHERE SC_WORK_ACTION_SET_ID = (SELECT ACTIONSETID
                              FROM SYSCAT.WORKACTIONSETS
                              WHERE ACTIONSETNAME = 'DATABASEACTIONS')
AND SC_WORK_CLASS_ID = (SELECT WORKCLASSID
                       FROM SYSCAT.WORKCLASSES
                       WHERE WORKCLASSNAME = 'CALLSTATEMENTS')
```



```
AND WORKCLASSETID = (SELECT WORKCLASSETID FROM
SYSCAT.WORKACTIONSETS WHERE ACTIONSETNAME = 'DATABASEACTIONS');
```

---

Assuming that the captured activity has an application handle of 1, a unit of work identifier of 2, and an activity identifier of 3, the results are generated as shown in Example 5-9.

*Example 5-9 Result for a query of correlation of activity and unit of work*

---

AGENT_ID	UOW_ID	ACTIVITY_ID
=====	=====	=====
	1	2 3

---

By using this information, you can issue the query against the ACTIVITY\_DB2ACTIVITIES and the ACTIVITYSTMT\_DB2ACTIVITIES tables, shown in Example 5-10, to determine where the activity spent its time.

*Example 5-10 Performance query for an identified activity*

---

```
WITH RAH (LEVEL, APPL_ID, PARENT_UOW_ID, PARENT_ACTIVITY_ID,
UOW_ID, ACTIVITY_ID, STMT_TEXT, TIME_CREATED, TIME_COMPLETED) AS
(SELECT 1, ROOT.APPL_ID, ROOT.PARENT_UOW_ID,
ROOT.PARENT_ACTIVITY_ID, ROOT.UOW_ID, ROOT.ACTIVITY_ID,
ROOTSTMT.STMT_TEXT, ROOT.TIME_CREATED, ROOT.TIME_COMPLETED
FROM ACTIVITY_DB2ACTIVITIES ROOT, ACTIVITYSTMT_DB2ACTIVITIES ROOTSTMT
WHERE ROOT.APPL_ID = ROOTSTMT.APPL_ID AND ROOT.AGENT_ID = 1
AND ROOT.UOW_ID = ROOTSTMT.UOW_ID AND ROOT.UOW_ID = 2
AND ROOT.ACTIVITY_ID = ROOTSTMT.ACTIVITY_ID AND ROOT.ACTIVITY_ID = 3
UNION ALL
SELECT PARENT.LEVEL +1, CHILD.APPL_ID, CHILD.PARENT_UOW_ID,
CHILD.PARENT_ACTIVITY_ID, CHILD.UOW_ID,
CHILD.ACTIVITY_ID, CHILDSTMT.STMT_TEXT, CHILD.TIME_CREATED,
CHILD.TIME_COMPLETED
FROM RAH PARENT, ACTIVITY_DB2ACTIVITIES CHILD,
ACTIVITYSTMT_DB2ACTIVITIES CHILDSTMT
WHERE PARENT.APPL_ID = CHILD.APPL_ID AND
CHILD.APPL_ID = CHILDSTMT.APPL_ID AND
PARENT.UOW_ID = CHILD.PARENT_UOW_ID AND
CHILD.UOW_ID = CHILDSTMT.UOW_ID AND
PARENT.ACTIVITY_ID = CHILD.PARENT_ACTIVITY_ID AND
CHILD.ACTIVITY_ID = CHILDSTMT.ACTIVITY_ID AND
PARENT.LEVEL < 64
)
SELECT UOW_ID, ACTIVITY_ID, SUBSTR(STMT_TEXT,1,40),
TIMESTAMPDIFF(2, CHAR(TIME_COMPLETED - TIME_CREATED)) AS
LIFE_TIME
FROM RAH
ORDER BY UOW_ID, ACTIVITY_ID;
```

---

Example 5-11 shows the results of the query.

*Example 5-11 Query result that shows the impact area*

UOW_ID	ACTIVITY_ID	STMT_TEXT	LIFE_TIME
2	3	CALL SLOWPROC	1000
2	4	SELECT COUNT(*) FROM ORG	1
2	5	SELECT * FROM MYHUGETABLE	999

The results indicate that the stored procedure is spending most of its time querying the MYHUGETABLE table. The next step is to investigate the changes to the MYHUGETABLE table that might cause queries running against it to slow down.

When many stored procedures run simultaneously, greater overhead is incurred when performing the analysis. To solve this problem, you can create a workload and service class for running a stored procedure that is issued by a specific authorization identifier, a specific application, or both. You can then use the preceding method to analyze the behavior of the stored procedure.

### 5.4.3 SQL and XQuery Explain facility

The DB2 Explain facility provides detailed information about the access plan that the optimizer chooses for an SQL or XQuery statement. The information describes the decision criteria that is used to choose the access plan and can help you tune the statement or your instance configuration to improve performance.

More specifically, explain information can help you in the following ways:

- ▶ Understand how the database manager accesses tables and indexes to satisfy your query.
- ▶ Evaluate your performance-tuning actions. After altering a statement or making a configuration change, examine the new explain information to determine how your action has affected performance.

The captured information includes the following details:

- ▶ The sequence of operations that were used to process the query
- ▶ Cost information
- ▶ Predicates and selectivity estimates for each predicate
- ▶ Statistics for all objects that were referenced in the SQL or XQuery statement at the time that the explain information was captured
- ▶ Values for host variables, parameter markers, or special registers that were used to re-optimize the SQL or XQuery statement

You can invoke the explain facility by issuing the EXPLAIN statement, which captures information about the access plan chosen for a specific explainable statement and writes this information to explain tables. Create the explain tables prior to issuing the EXPLAIN statement. You can also set CURRENT EXPLAIN MODE or CURRENT EXPLAIN SNAPSHOT, which are special registers that control the behavior of the explain facility.

For privileges and authorities that are required to use the explain utility, see the description of the EXPLAIN statement. The EXPLAIN authority can be granted to an individual who requires access to explain information but not to the data that is stored in the database. This authority is a subset of the database administrator authority and has no inherent privilege to access data stored in tables.

To display explain information, you can use either a command-line tool or Visual Explain. The tool that you use determines how you set the special registers that control the behavior of the explain facility. For example, if you expect to use Visual Explain only, you need only capture the snapshot information. If you expect to perform detailed analysis with one of the command-line utilities or with custom SQL or XQuery statements against the explain tables, capture all explain information.

The **db2exp1n** command describes the access plan that is selected for SQL or XQuery statements. You can use this tool to obtain a quick explanation of the chosen access plan when explain data is not captured. For static SQL and XQuery statements, the **db2exp1n** command examines the packages that are stored in the system catalog. For dynamic SQL and XQuery statements, the **db2exp1n** command examines the sections in the query cache.

The explain tool is in the bin subdirectory of your instance sqllib directory. If the **db2exp1n** command is not in your current directory, it must be in a directory that appears in your PATH environment variable.

The **db2exp1n** command uses the db2exp1n.bnd, db2exsrv.bnd, and db2exdyn.bnd files to bind itself to a database the first time the database is accessed.

For more information about DB2 Explain, see the IBM DB2 Database for Linux, UNIX, and Windows Information Center at the following address:

<http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp>

#### 5.4.4 Snapshot monitor

You can use the snapshot monitor to capture information about the database and any connected applications at a specific time. Snapshots are useful for determining the status of a database system. When taken at regular intervals, they are also useful for observing trends and foreseeing potential problems. Some of the data from the snapshot monitor is obtained from the system monitor. The data available from the system monitor is determined by system monitor switches.

The system monitor accumulates information for a database only while it is active. If all applications disconnect from a database and the database deactivates, then the system monitor data for that database is no longer available. You can keep the database active until your final snapshot has been taken either by starting the database with the ACTIVATE DATABASE command or by maintaining a permanent connection to the database.

Use the following snapshot commands to analyze your database behavior:

- ▶ db2 get snapshot for all applications;
- ▶ db2 get snapshot for all bufferpools;
- ▶ db2 get snapshot for all databases;
- ▶ db2 get snapshot for all on biglnxdb;
- ▶ db2 get snapshot for dynamic sql on biglnxdb;
- ▶ db2 get snapshot for database manager;

**Important:** Enter all the snapshot commands at the end of a test.

For more information about DB2 monitoring, see the “Database monitoring” topic in the DB2 for Linux Information Center at the following address:

<http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=/com.ibm.db2.luw.admin.mon.doc/doc/c0001138.html>

## 5.4.5 DRDA tracing of activity with DB2 for Linux

From your z/VSE applications, a Distributed Relational Database Architecture (DRDA) data stream is sent to DB2 on Linux. Along this path, several layers can influence the behavior of your application.

To trace the data flow, enter the following commands to gather the information and visualize it:

1. Before starting the test, turn on the trace by entering the following command:

```
db2trc on -f trace.dmp
```

The trace will remain on until the test finishes.

2. After the test finishes, stop the trace by entering the following command:

```
db2trc off
```

3. Enter the following command after the test is done and the trace is stopped:

```
db2trc flw trace.dmp trace.flw
```

4. Format the trace.dmp file:

```
db2trc fmt trace.dmp trace.fmt
```

5. Convert the trace.dmp file into a format for a graphical tool, such as Wireshark, which we explain in 5.5.2, “Network activity in Linux on System z” on page 145:<sup>1</sup>

```
db2trc fmt -c trace.dmp trace.fmc
```

In some situations, there is no timestamp information for the trace entries. In those situations, use the **db2drdat** command as follows:

```
db2drdat on -i -l=length_of_buffer_in_bytes  
db2drdat off -t=trace_output_file_name -f
```

The trace\_output\_file will already be formatted. Trace a short scenario to see if the timestamps are included in the trace.

**Note:** The **db2drdat** command does not write to a file while tracing. Therefore, the buffer must be as large as possible to trace for a whole job run.

## 5.4.6 Additional DB2 monitoring tools

Two other monitoring tools are DB2 Performance Expert and the Tivoli Monitoring for Databases DB2 Agent tool.

### DB2 Performance Expert

DB2 Performance Expert is a tool for monitoring, analyzing, and tuning the performance of DB2 and its applications. DB2 Performance Expert can help you in the following tasks:

- ▶ Determining whether an index can improve performance
- ▶ Reviewing sort performance
- ▶ Checking the need to reorganize tables
- ▶ Ensuring that sufficient DB2 agents are available to handle the workload
- ▶ Resolving lock conflicts
- ▶ Examining frequently used SQL statements from the package cache
- ▶ Analyzing buffer pools
- ▶ Monitoring system health

<sup>1</sup> See <http://www.wireshark.org/>.

For more information about how to analyze and tune your database workload, see *DB2 Performance Expert for Multiplatforms V2.2*, SG24-6470.

### **Tivoli Monitoring for Databases: DB2 Agent**

Tivoli Monitoring for Databases uses DB2 Agent monitors on the systems where you use it for DB2 workload monitoring. It monitors the availability and performance information of DB2 servers.

DB2 Agent provides historical data that you can use to track trends and troubleshoot problems. You install the monitoring agent for DB2 on Linux with root control.

The monitoring agent for DB2 is installed with the DB2 product. It is intended for users of IBM Optim™ Database Administrator. The monitoring agent for DB2 is restricted to supplying monitoring information to the Optim Database Administrator Health and Availability monitoring feature only.

For more information about DB2 Agent, see *Tivoli Monitoring for Databases: DB2 Agent User's Guide*, SC32-9449.

## **5.5 Network monitoring and tuning**

Network monitoring and tuning ensures that data flows fast and efficient through the network. The design and structure of the network has high impact on the end-to-end performance.

Network performance can be measured by using two parameters:

- ▶ *Throughput* defines how many bytes flow through the network in a given time. Usually it is measured in bytes, kilobytes, or megabytes per second (MBps). A higher throughput value is better.
- ▶ *Latency* defines how much time a block of data, for example a packet, needs to reach the final receiver. Typically it is measured in milliseconds (ms). A lower latency value is better.

Both parameters affect end-to-end performance of distributed applications. A high throughput is more important for large file transfers. A short latency time is more important for transactional type of communication, where small amounts of data are transferred back and forth between the client and the server.

For a distributed DB2 application, both parameters can be important. A high throughput is needed to transport large result sets of SQL queries in a short amount of time, for example, `SELECT * FROM TABLE` or mass `INSERTs`. A short latency is needed for fast processing of smaller SQL queries that transport only small amounts of data, such as `UPDATE` or `DELETE` statements.

For best network performance, the client and the server should be as close to each other as possible. If the data must be transported over hundreds of miles through a wide area network (WAN), at a minimum, the latency will suffer. For best performance, the client and the server should reside on the same System z server, connected through a high performance network such as HiperSockets or virtual LANs.

If network performance is not as expected, or if you want a close look at the data and requests that are flowing through the network, you can use a network sniffer and capture the packets and analyze them. Using network sniffers and analyzing network traces is a more advanced task that requires advanced knowledge in network protocols such as IP, TCP, and DRDA.

Be sure to look at your network behavior from the z/VSE side and from the Linux side.

## 5.5.1 Network trace in z/VSE

You can use the following TCP/IP commands and tools to view TCP/IP connections from a z/VSE perspective:

- ▶ The TCP/IP for VSE QUERY CONNECTIONS command, which displays the currently established or listening sockets and connections

With the information from Example 5-12, you can determine the number of TCP/IP connections that are currently established and to what address. This example also shows which z/VSE partition owns the connection.

*Example 5-12 TCP/IP connections and the owning applications*

---

```
F7 0097 IPN253I << TCP/IP Connections >>
F7-0101 IPN300I Enter TCP/IP Command
F7 0097 21FF: IPT353I >21 0.0.0.0, 0 TCP Listen
F7 0097 21FF: IPT345I Open by phase FTPDAEMN in F7
F7 0097 21FD: IPT353I >21 9.152.2.70, 4349 TCP Established
F7 0097 21FD: IPT345I Open by phase FTPDAEMN in F7
F7 0097 21FD: IPT341I Start: 13:40:20 Idle: 0.000 Duration: 0.693
F7 0097 21FD: IPT360I Socket RECV 256 bytes, Duration: 0.000
F7 0097 1CFD: IPT353I >2893 0.0.0.0, 0 TCP Listen
F7 0097 1CFD: IPT345I Open by phase IESVCSRV in R1
F7 0097 0B4C: IPT353I >80 0.0.0.0, 44565 TCP Listen
F7 0097 0B4C: IPT345I Open by phase DFHSIP in F2
F7 0097 0034: IPT353I >1435 0.0.0.0, 0 TCP Listen
F7 0097 0034: IPT345I Open by phase DFHSIP in F2
...
```

---

- ▶ IPTraceTool, which converts a packet trace that is captured by TCP/IP for VSE into the trace format as a Generic Network Capture Document (.cap) file

This file is an open standard and can be opened by using graphical tools, such as the open source tool Wireshark,<sup>2</sup> for detailed analysis. You can download the IPTraceTool from the z/VSE home page at the following address:

<http://www.ibm.com/servers/eserver/zseries/zvse/>

- ▶ The TCP/IP for VSE DEFINE TRACE command, which starts a tracing operation for a specified IP address

All data traffic to and from this address is saved in memory and can be dumped with the DUMP TRACES command. You can trace network packets (IBBLOKs) and socket requests. Reading such traces is a more advanced task and might require help from a support person.

- ▶ The See-TCP/IP product from CSI International, which can help you to analyze the activity and performance of your z/VSE and TCP/IP systems from a PC graphical user interface

You can find more information about See-TCP/IP at the CSI International Web site at the following address:

<http://www.csi-international.com/products/zvse/See-TCPIP/See-TCPIP.htm>

---

<sup>2</sup> See <http://www.wireshark.org/>.

- ▶ The z/IPMon product from illustro Systems International, LLC, which delivers all the functions that you need to have a comprehensive look at both the real-time and historical activities in your z/VSE TCP/IP systems

You can find more information about z/IPMon at the illustro Systems Web site at the following address:

<http://www.illustro.com/zipmon>

## 5.5.2 Network activity in Linux on System z

The following TCP/IP commands and tools are available to look at TCP/IP connections on Linux on System z:

- ▶ The **netstat** command, which is available on many systems, including Linux, UNIX, and Windows

This command shows the currently active connections and listening sockets.

- ▶ The **tcpdump** command, which is available on UNIX and Linux systems

With this command, you can capture network packets (network sniffer). The **tcpdump** tool can either print out each packet or save it into a file that you can later open in a tool such as Wireshark.

- ▶ Wireshark (formerly Ethereal), with which you can capture network packets (network sniffer) and read capture data from other sources, such as **tcpdump**

Wireshark has comprehensive network protocol analysis functions. It can analyze (work with) almost all available protocols, including the DRDA protocol. It can display the DRDA buffers that are sent over the network in ASCII and in the EBCDIC format. Analyzing such network traces is a more advanced task that might require help from a person with advanced network skill.

Bad network performance can be caused by frequent packet drops and retransmissions of packets. You can only learn about packet drops if you have a packet trace from both sides at the same time. Therefore, it makes sense to capture a packet trace on both the client and the server side, in our case on z/VSE and on Linux on System z.

Packet traces give good information about the timing of requests. Provided that you have some knowledge about the protocols used (IP, TCP, and DRDA), you see when a TCP connection was established, when the first DRDA request was sent, when the DRDA reply was received, and so on. You can also see from the data within the DRDA request the SQL statement that was executed (by the sequence number in the package that is used). All of this information can help you to determine which SQL statement took longer than expected. You can then look more closely at the SQL statement in question and try to optimize it, or look at the application source code and try to optimize the statement.

First, you can determine the overall network traffic for each interface configured in your network configuration by entering the following command before and after the test:

```
ifconfig -a
```

You can have internal HiperSockets networks and Ethernet connections to the external network. Example 5-13 shows the results for the HiperSockets interface.

*Example 5-13 HiperSockets overview with interface hsi0*

---

```
hsi0    Link encap:Ethernet HWaddr 02:00:00:00:00:02
        inet addr:172.16.0.5 Bcast:172.16.0.255 Mask:255.255.255.0
        inet6 addr: fe80::ff:fe00:2/64 Scope:Link
        UP BROADCAST RUNNING NOARP MULTICAST MTU:16384 Metric:1
        RX packets:0 errors:0 dropped:0 overruns:0 frame:0
        TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:0 (0.0 b) TX bytes:296 (296.0 b)
```

---

Example 5-14 shows the results for the Ethernet network interface.

*Example 5-14 Ethernet network traffic on interface eth0*

---

```
eth0    Link encap:Ethernet HWaddr 00:04:AC:7C:EA:20
        inet addr:9.152.26.187 Bcast:9.152.27.255 Mask:255.255.252.0
        inet6 addr: fe80::4:ac00:247c:ea20/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST MTU:1492 Metric:1
        RX packets:2458597 errors:0 dropped:0 overruns:0 frame:0
        TX packets:2158104 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:1722948267 (1643.1 Mb) TX bytes:144003571 (137.3 Mb)
```

---

If you issued **ifconfig -a** before and after your test workload, in both examples, you can calculate the sent and received IP packets in packets parameters TX and RX. An increase in the errors and dropped counts after the test might indicate a resource shortage.

If you want to analyze the IP packet traces from z/VSE and from the Linux side, you can use the following procedures:

► Analyzing IP packet traffic on Linux:

- a. Enter the **tcpdump** command from a window shell.
- b. Replace **hsi0** with the interface name that you want to use.
- c. Print the packet information to the file specified in **<outfile>**.

```
tcpdump -i hsi0 -XX -s 0 -w <outfile>
```

The **tcpdump** command runs until you stop it by pressing Ctrl+C. The **<outfile>** can then be used as input for Wireshark.

► Analyzing IP packet traffic from z/VSE:

- a. Define the trace for TCP/IP by entering the following command. Replace **<linux ip>** with your Linux IP, and replace **<num pkts>** with the number of packets to be traced on z/VSE. Depending on the TCP/IP partition size, limit the trace size by using the **SIZE** parameter. A size of 4000 is acceptable. If the size is too high, you might experience GETVIS problems in the TCP/IP partition, in which case you must reduce the size.

```
DEFINE TRACE,ID=DRDA,IP=<linux ip>,SIZE=<num pkts>
```

- b. When the test case is finished, enter the following TCP/IP commands:

```
DUMP TRACES
SEGMENT
DELETE TRACE,ID=DRDA
```



The trace is written to SYSLST (job output of TCP/IP partition) when using the DUMP TRACES command.

- c. Download the TCP/IP listing in text format to your PC. You can analyze the listing by using the IPTraceTool, which converts the trace data into the Wireshark format. For more information about this tool, see “Network trace in z/VSE” on page 144.

Figure 5-2 shows an example packet trace of DRDA communication. The selected packet number 8 is a Prepare SQL Statement request (PRPSQLSTT) for the SQL statement `SELECT * FROM NAMES`. The following packets contain various DRDA requests and responses needed to execute that SQL statement. In packet number 15, the result set is returned as part of the QRYDTA portion of the DRDA response. In the Time and Delta column of the packet list, you see the times (absolute and relative) of the packets. You can use this information to correlate it with other events that you have monitored.

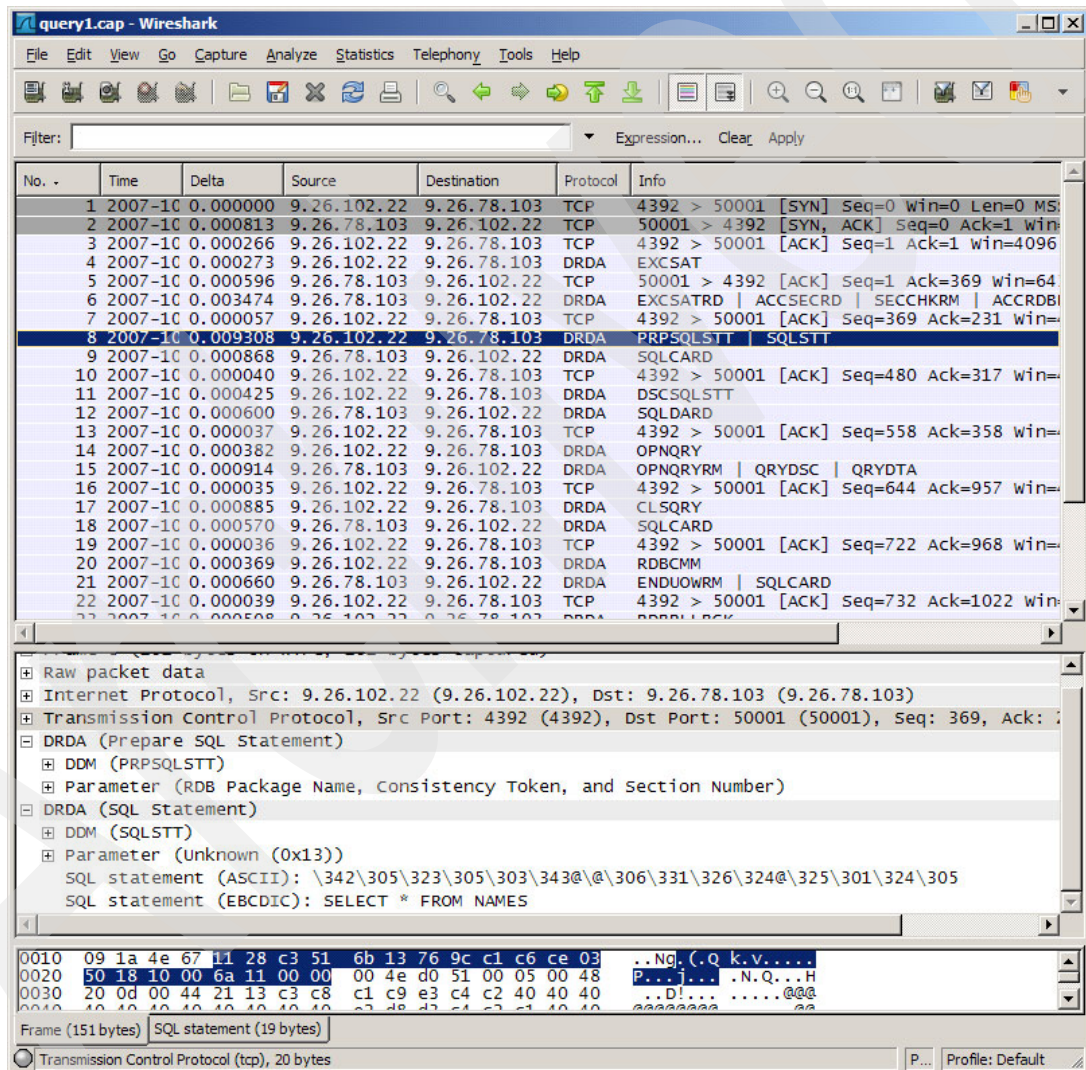


Figure 5-2 Network trace as displayed in Wireshark (source: Wireshark.org)

### 5.5.3 HiperSockets considerations

If you are using a HiperSockets network between your z/VSE system and DB2 running on Linux on System z, make sure that you use a proper MTU size. For HiperSockets, the MTU size is defined in the I/O subsystem configurations (IOCP). You can choose between 16K, 24K, 32K, and 64K. In general, a large MTU size is beneficial. However, because of possible kernel memory constraints in Linux on System z, do not use an MTU size larger than 32K.

### 5.5.4 DB2 BLOCK SIZE considerations

Adjust the DB2 BLOCK SIZE so that it matches the MTU size of the network minus 40 bytes. DB2 sends the contents of a result set in larger chunks, called *blocks*. When the data is sent over the network, each block is divided into as many network packets as needed to transport the complete block.

For example, if the block size is 32768 bytes (32K), and the network's MTU size is 1492 (typical Ethernet MTU size), every packet can hold up to 1452 bytes based on the following equation:

$$(1492 - 20 \text{ bytes IP header}) - 20 \text{ bytes TCP header} = 1452 \text{ bytes of data}$$

This results in 23 packets to send the complete block:

$$32768 / 1452 = 23$$

The first 22 packets contain 1452 bytes of data, and the last packet contains only 824 bytes of data.

For a real Ethernet network, this example does not cause any problems. However, for a HiperSockets network, when running DB2 on Linux on System z, with an MTU size of 32K (32768 bytes), for example, the block size of 32K is most inefficient. Here, a network packet can carry up to 32728 bytes:

$$(32768 - 20 \text{ bytes IP header}) - 20 \text{ bytes TCP header} = 32728$$

The block is sent out in two packets. The first packet contains 32728 bytes, and the second packet contains only 40 bytes, which is a great waste of resources.

For use with HiperSockets, either reduce the BLOCK SIZE so that a complete block fits into a packet (32K minus about 40 bytes) or increase the MTU size to 64K. Instead of reducing the BLOCK SIZE, you can also increase the BLOCK SIZE to about 60K, so that it takes two almost complete network packets to transport the complete block.

The block size considerations are most important for transferring large result sets. If fewer amounts of data are transported, the result set might not even occupy a full block. Therefore, while the previous calculations might not be valid, the resulting sizes will not cause any harm.

## 5.6 z/VSE system monitoring and tuning

If an application or a set of applications does not run as fast as expected, you must obtain a good understanding about the availability and utilization of various z/VSE resources. Your application might not run as fast as it should, because your system runs CPU constrained, memory constrained, or I/O constrained, or it might be limited by other resources, such as network speed or locked resources. Also, the underlying virtualization layer, such as z/VM or LPAR, might have an impact on performance.

## 5.6.1 z/VSE system monitoring facilities

To determine the cause of the bottleneck, use a z/VSE system monitoring facility. You can use the z/VSE commands and tools presented in the following sections to monitor z/VSE resources.

### System activity dialogs using SYSA fastpath 361 and 362

Figure 5-3 shows an example of the DISPLAY SYSTEM ACTIVITY panel of the Interactive Interface. It contains real-time performance information about the z/VSE system, including CPU, static and dynamic partitions, and I/O.

```

IESADMDA          DISPLAY SYSTEM ACTIVITY          15 Seconds  15:02:31
*--- SYSTEM (CPUs: 1 / 0 ) ---* *----- CICS : DBDCCICS -----*
|CPU      : 0% I/O/Sec: 1 | |No. Tasks: 25,817 Per Second : * |
|Pages In : 0 Per Sec: * | |Dispatchable: 0 Suspended : 3 |
|Pages Out: 0 Per Sec: * | |Curr. Active: 4 MXT reached: 0 |
*-----*
Priority: Z,Y,S,R,P,C,BG,FA,F9,F8,F6,F5,F4,F2,F7,FB,F3,F1

ID S JOB NAME PHASE NAME ELAPSED CPU TIME OVERHEAD %CPU I/O
F1 1 POWSTART IPWPOWER 378:35:01 21.57 9.70 15,562
F3 3 VTAMSTRT ISTINCVT 378:34:58 21.41 9.73 17,030
FB B SECSERV BSTPSTS 378:35:02 .10 .04 824
*F7 7 TCPIP00 IPNET 378:34:58 34.23 15.53 14,975
F2 2 CICSICCF DFHSIP 378:34:58 217.81 96.94 18,424
F4 4 <=WAITING FOR WORK=> .00 .00 2
F5 5 <=WAITING FOR WORK=> .00 .00 2
F6 6 <=WAITING FOR WORK=> .00 .00 2
F8 8 <=WAITING FOR WORK=> .00 .00 2
F9 9 <=WAITING FOR WORK=> .00 .00 2
FA A <=WAITING FOR WORK=> .00 .00 2
BG 0 <=WAITING FOR WORK=> .00 .00 2
PF1=HELP 2=PART.BAL. 3=END 4=RETURN 5=DYN.PART 6=CPU
  
```

Figure 5-3 Display System Activity panel (361)

With fastpath 362, you see the DISPLAY CHANNEL AND DEVICE ACTIVITY panel (Figure 5-4).

```

IESADMSIOS          DISPLAY CHANNEL AND DEVICE ACTIVITY          Page 01 of 05
DEVICE ADDRESS RANGE FROM: 000 TO: FFF          Seconds 15:04:03

      DEVICE      PART      JOB      DEVICE I/O
              ID      NAME      REQUESTS

      009          F1      POWSTART      1244
                   F3      VTAMSTRT        23
                   FB      SECSERV          6
                   F7      TCPIP00      14383
                   F2      CICSICCF        146
                   R1      STARTVCS      12001
      00C          F1      POWSTART        159
      120          F7      TCPIP00          3
      121          F7      TCPIP00         11
      122          F7      TCPIP00          2
      150          F1      POWSTART      4954
                   F3      VTAMSTRT        444

PF1=HELP          3=END          4=RETURN
                  8=FORWARD
  
```

Figure 5-4 Display Channel and Device Activity panel (362)

Figure 5-3 and Figure 5-4 on page 149 show real-time information without any history facility for backtracking.

### QUERY TD command

Example 5-15 shows the output of the QUERY TD command. The output shows information about CPU usage on the console. For more information about this command, see 5.6.3, “CPU considerations” on page 157.

*Example 5-15 The QUERY DT command*

---

```

query td
AR 0015 CPU STATUS SPIN_TIME NP_TIME TOTAL_TIME NP/TOT
AR 0015 00 ACTIVE 0 424862 597001 0.711
AR 0015
-----
AR 0015 TOTAL 0 424862 597001 0.711
AR 0015
AR 0015 NP/TOT: 0.711 SPIN/(SPIN+TOT): 0.000
AR 0015 OVERALL UTILIZATION: 0% NP UTILIZATION: 0%
AR 0015
AR 0015 ELAPSED TIME SINCE LAST RESET: 1363073307
AR 0015 1I40I READY

```

---

### SIR SMF command

You use the SIR SMF command to collect and display I/O-related performance information by using the Subsystem Measurement Facility (SMF), as shown in Example 5-16. For more information about this command, see 5.6.5, “I/O considerations” on page 159.

*Example 5-16 The SIR SMF command*

---

```

sir smf=on
AR 0015 1I40I READY
....
sir smf
AR 0015 DEVICE I/O-CNT QUEUED CONNECT DISCONN TOTAL
AR 0015 msec/SSCH msec/SSCH msec/SSCH msec/SSCH
AR 0015
AR 0015 150 1668 0.190 0.342 0.241 0.775
AR 0015 151 1723 0.190 0.341 0.134 0.667
AR 0015 152 468 0.188 0.343 0.557 1.090
AR 0015 1I40I READY
sir smf=off
AR 0015 1I40I READY

```

---

### SIR MON command

The SIR MON command collects and displays internal monitor counters, such as SVC counters and system services counters, as shown in Example 5-17.

*Example 5-17 The SIR MON command*

---

```

sir mon=on
AR 0015 MONITORING HAS BEEN ACTIVATED
....
sir mon
AR 0015 MONITORING REPORT
AR 0015 (BASED ON A 0000:00:18.595 INTERVAL)

```

```

AR 0015                                SVC SUMMARY REPORT
AR 0015 EXCP      =      3771 FCH-$$B =          8 LOAD      =      116
AR 0015 WAIT      =      3966 SETIME  =          26 SVC-0B   =          2
AR 0015 SVC-0C    =          229 SVC-0D  =         230 SYSIO    =         111
AR 0015 EXIT IT   =          36 SETIME  =          23 SVC-1A   =          2
AR 0015 WAITM     =          36 COMREG  =        1048 GETIME   =          8
AR 0015 POST      =          98 DYNCLASS =          1 SVC-31   =         32
AR 0015 HIPROG    =          2 TTIMER   =          3 SVC-35   =        298
AR 0015 INVPART   =          1 GETVIS   =         608 FREEVIS  =        553
AR 0015 CDLOAD    =          1 RUNMODE  =          1 SECTVAL  =       3737
AR 0015 SETLIMIT  =          2 EXTRACT  =          6 GETVCE   =         28
AR 0015 EXTENT    =          1 FASTSVC  =        9149 SECHECK  =         58
AR 0015 (UN)LOCK  =        572 MSAT     =          8 VIO      =          1
AR 0015 SVC-75    =          53 PRODID  =          4 SVC-83   =         18
AR 0015 SVC-84    =        215 SVC-85   =          1
AR 0015                                SVC-X'6B' DETAIL REPORT
AR 0015 FC-02     =          73 FC-03   =         214 FC-06   =         402
AR 0015 FC-07     =          4 FC-08   =          69 FC-09   =         278
AR 0015 FC-0A     =        222 FC-0D   =          36 FC-0E   =         528
AR 0015 FC-0F     =          1 FC-1B   =        107 FC-26   =          1
AR 0015 FC-31     =          1 FC-3F   =          11 FC-40   =          11
AR 0015 FC-46     =       3736 FC-47   =       3736 FC-4D   =          19
AR 0015 FC-4F     =          3 FC-50   =          1 FC-51   =          19
AR 0015 FC-59     =          1 FC-61   =        171 FC-62   =          2
AR 0015 FC-67     =          3 FC-73   =          52 FC-79   =          1
AR 0015 FC-7D     =         62 FC-7E   =          4 FC-86   =         64
AR 0015 FC-89     =          2 FC-90   =          42 FC-96   =         12
AR 0015 FC-9F     =        428 FC-AA   =          1 FC-B6   =         36
AR 0015                                SVC-X'75' DETAIL REPORT
AR 0015 FC-60     =         24 FC-64   =          29
AR 0015                                MVS-SVC'S DETAIL REPORT
AR 0015 SVC-01    =          95 SVC-02   =          87 SVC-22   =          5
AR 0015 SVC-23    =          7 SVC-2E   =          5 SVC-2F   =         28
AR 0015 SVC-6B    =          6
AR 0015                                0000 PERFORMANCE COUNTER DETAILS
AR 0015 CNT-00    =       21929 CNT-01   =       21931 CNT-02   =       21931
AR 0015 CNT-03    =       17983 CNT-08   =        3910 CNT-09   =          2
AR 0015 CNT-0B    =       5665 CNT-0C   =          4 CNT-0D   =       9169
AR 0015 CNT-0E    =      30109
AR 0015                                BOUND STATE DETAIL REPORT
AR 0015 BND-49    =          1 BND-50   =          2 BND-57   =          18
AR 0015 BND-80    =        125 BND-82   =       5394 BND-85   =        125
AR 0015                                PROGRAM-CALL BASED SERVICES
AR 0015 PC-13     =          49
AR 0015 1I40I READY
sir mon=off
AR 0015 MONITORING HAS BEEN DEACTIVATED

```

---

## z/VSE job accounting exit

The z/VSE job accounting exit prints performance related information, such as CPU and I/O usage, to SYSLST after each job step (Example 5-18). If you want to use this exit, see the SKJOBACC skeleton in ICCF library 59.

*Example 5-18 Output of the z/VSE job accounting exit*

JOBNAME	=	H2SORT	USER INFO	=		EXEC NAME	=	SORT
DATE	=	07/03/09	PART ID	=	F6			
START	=	10:57:42	STOP	=	10:57:44	DURATION	=	2.400 SEC
CPU	=	0.037 SEC				PAGEIN SINCE IPL	=	0
OVERHEAD	=	0.010 SEC				PAGEOUT SINCE IPL	=	0
TOTAL CPU	=	0.047 SEC						
UNIT	=	F15	UNIT	=	FEC	UNIT	=	009
SIO	=	57	SIO	=	12	SIO	=	3
UNIT	=	FEE	UNIT	=	B53	UNIT	=	F16
SIO	=	52	SIO	=	394	SIO	=	22

## VSE Navigator tool

The VSE Navigator tool collects real-time performance information about the z/VSE system, including CPU, static and dynamic partitions, and I/O, similar to the system activity dialogs of the Interactive Interface. The VSE Navigator shows one main panel and provides access to four subpanels. The main panel (Figure 5-5) contains general performance information about overall CPU utilization, paging activity, and CICS task and storage statistics. If you save the collected information, you can use it for future analysis or archiving.

You can download the VSE Navigator tool from the VSE home page.

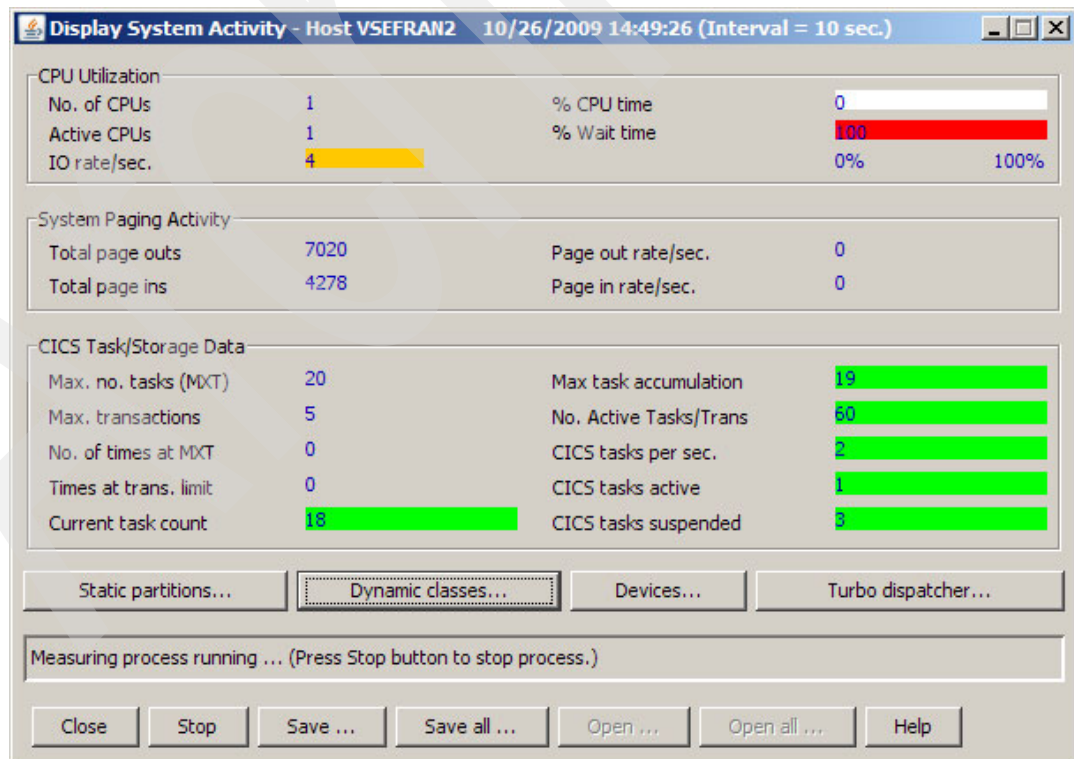


Figure 5-5 Display System Activity main panel



The four subpanels are available by clicking the Static partitions, Dynamic classes, Devices, and Turbo dispatcher buttons.

When you click the *Static partitions button*, the Static Partition Activity panel (Figure 5-6) opens. This panel shows performance related information for each static partition and a graph indicating the CPU usage of each static partition.

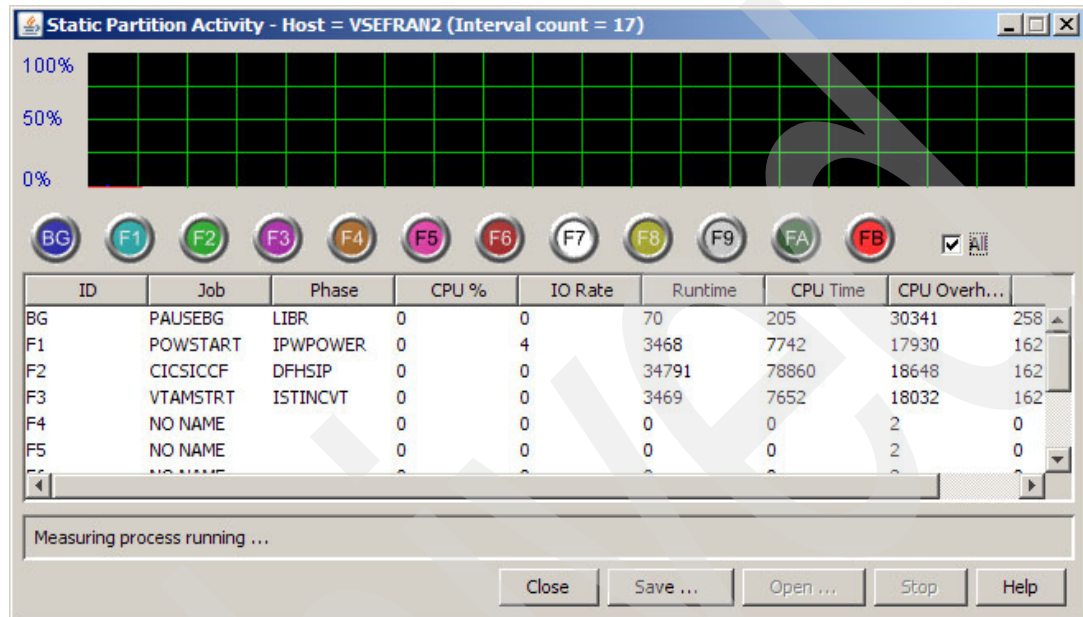


Figure 5-6 Static Partition Activity panel

When you click the *Dynamic classes button*, the Dynamic Classes panel (Figure 5-7) opens. This panel shows performance related information for each dynamic class and active dynamic partition, as well as a graph indicating the CPU usage of each dynamic partition.

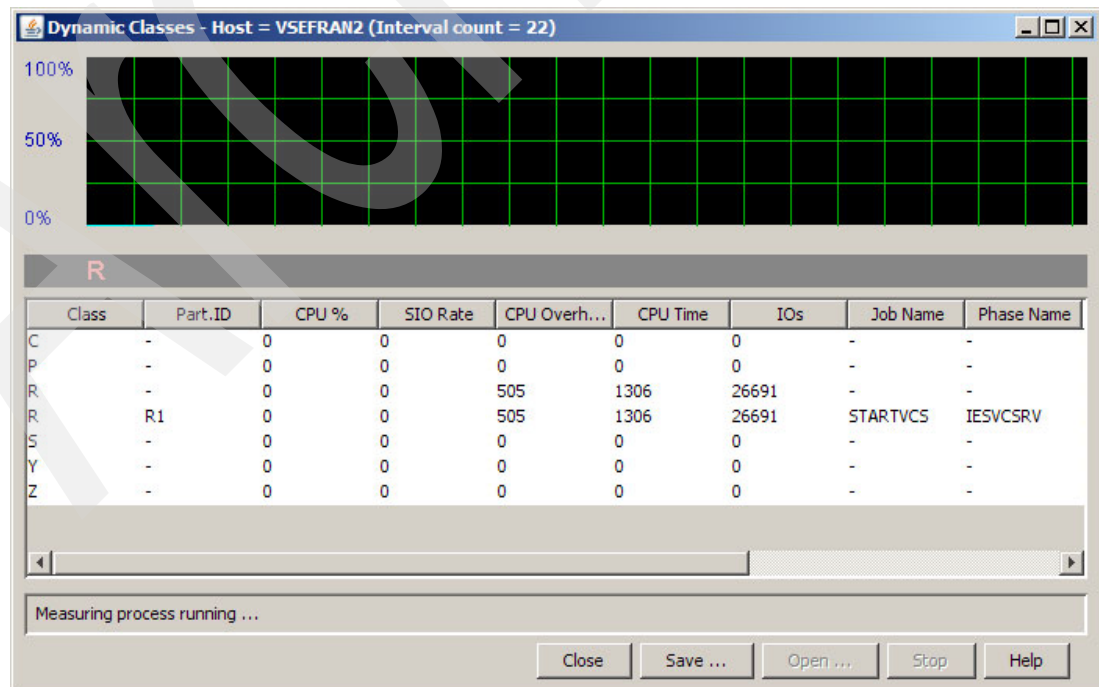


Figure 5-7 Dynamic Classes panel

When you click the *Turbo Dispatcher button*, the Turbo Dispatcher Status panel (Figure 5-8) opens. This panel shows information about the CPUs and a graph indicating the overall system CPU usage.

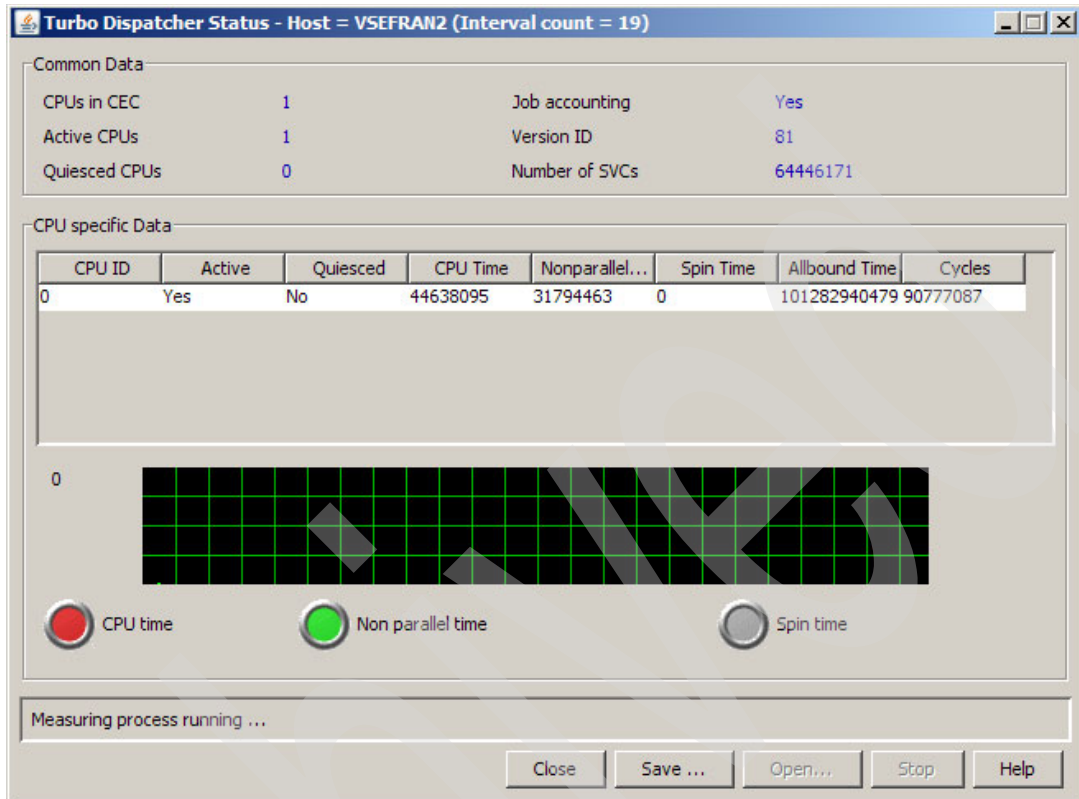


Figure 5-8 Turbo Dispatcher Status panel

When you click the *Devices button*, a window (not shown) opens that shows a list of devices and performance counters for each device.

### VSE Health Checker tool

The VSE Health Checker is a diagnosis tool to retrieve, display, and analyze performance relevant data from a z/VSE system. The information that is displayed, as shown in Figure 5-9 on page 155, is based on a snapshot of data from z/VSE at a certain point in time.

You can download this tool from the z/VSE home page at the following address:

<http://www.ibm.com/servers/eserver/zseries/zvse/>



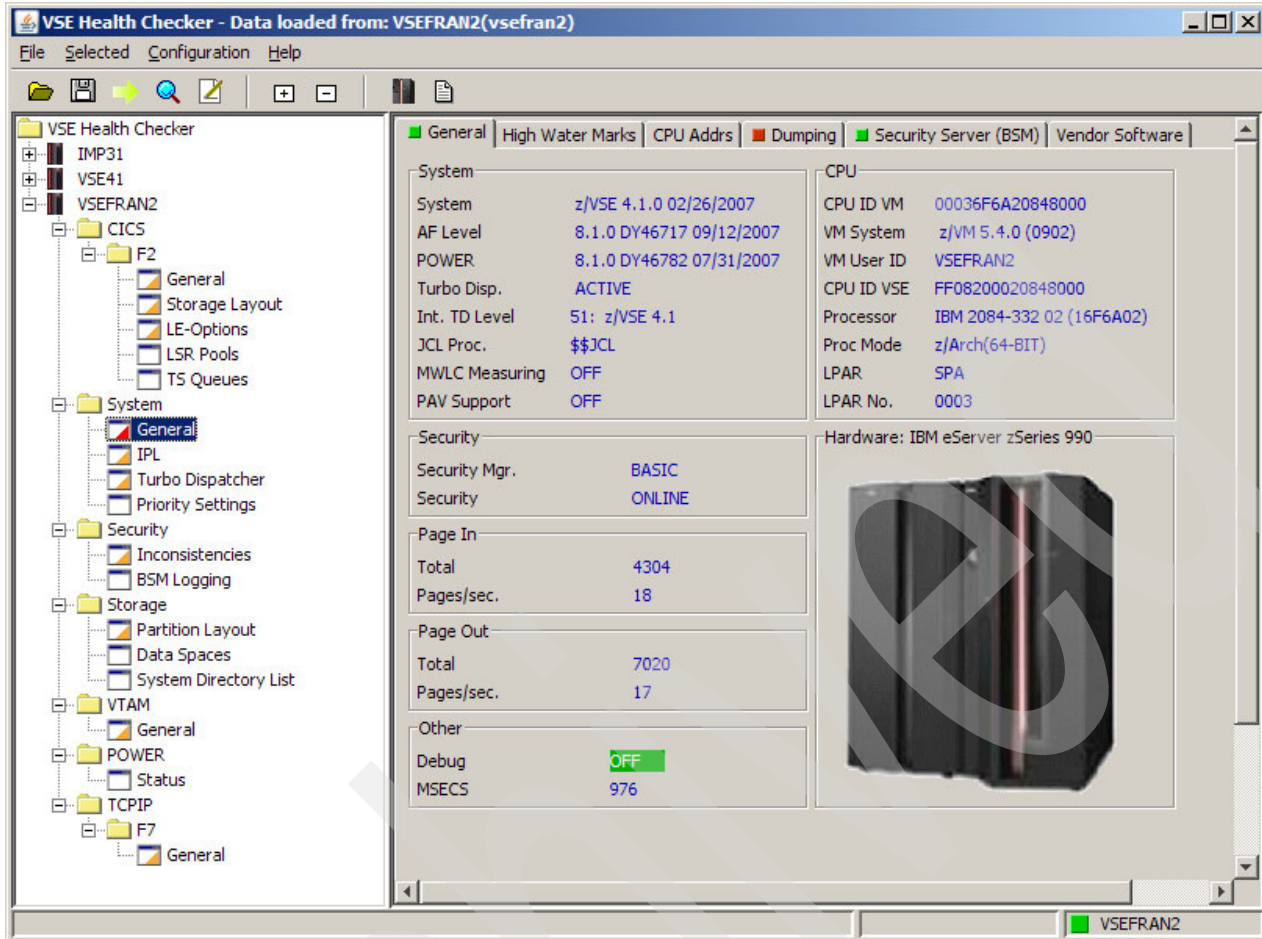


Figure 5-9 The main panel of the VSE Health Checker showing general information about the z/VSE system

## MAP and GETVIS commands

The MAP and GETVIS commands show information about memory usage and partition sizes on the console. You can use the MAP command to see an overview, as shown in Example 5-19.

Example 5-19 Output of the MAP command

```
map
AR 0015 SPACE AREA      V-SIZE  GETVIS  V-ADDR  UNUSED NAME
AR 0015  S  SUP          716K      0          0      $$$SUPI
AR 0015  S  SVA-24      1888K    1748K    B3000    768K
AR 0015  0  BG V         1280K    4864K    500000   45056K PAUSEBG
AR 0015  1  F1 V         1024K    4096K    500000   0K POWSTART
AR 0015  2  F2 V         2048K    49152K   500000   0K CICSIKCF
AR 0015  3  F3 V          600K    14760K   500000   0K VTAMSTRT
AR 0015  4  F4 V         2048K    18432K   500000   0K
AR 0015  5  F5 V          768K     256K    500000   0K
AR 0015  6  F6 V          256K     256K    500000   0K
AR 0015  7  F7 V         1024K    19456K   500000   0K TCPIP00
AR 0015  8  F8 V         2048K    49152K   500000   0K
AR 0015  9  F9 V          256K     256K    500000   0K
AR 0015  A  FA V         256K     256K    500000   0K
AR 0015  B  FB V         256K     256K    500000   0K SECSERV
```

```

AR 0015 S SVA-31 7588K 6748K 3700000
AR 0015 DYN-PA 15360K
AR 0015 DSPACE 6400K
AR 0015 SYSTEM 1088K
AR 0015 AVAIL 55744K
AR 0015 TOTAL 270336K <-----'
AR 0015 1I40I READY

```

---

Example 5-20 shows the storage information for partition BG.

*Example 5-20 Output of the GETVIS command*

---

```

getvis bg,detail
AR 0015 GETVIS USAGE      BG-24      BG-ANY      BG-24      BG-ANY
AR 0015 AREA SIZE:      4,864K      4,864K
AR 0015 USED AREA:      48K        48K MAX. EVER USED:      56K        56K
AR 0015 FREE AREA:      4,816K      4,816K LARGEST FREE:      4,816K      4,816K
AR 0015 SUMMARY REPORT
AR 0015 SUBPOOL      REQUEST <---BG-24-AREA--- ---BG-ANY-AREA-->
AR 0015 INLG0000      12K        OK
AR 0015      00641000-00643FFF
AR 0015 Default      4K        OK
AR 0015      00640000-00640FFF
AR 0015 SUBPOOL TOTALS      16K        OK
AR 0015 1I40I READY

```

---

## 5.6.2 Stand-alone monitor tools and products

Various stand-alone monitor tools and products, such as the following examples, are available:

- ▶ CPUMON from IBM (free of charge)  
<http://www.ibm.com/servers/eserver/zseries/zvse/downloads/tools.html#cpumontool>
- ▶ ASG-TMON from Allen System Group, Inc.  
[http://www.asg.com/products/product\\_details.asp?code=TVE&id=112](http://www.asg.com/products/product_details.asp?code=TVE&id=112)
- ▶ z/VM Performance Toolkit for monitoring z/VM guests  
<http://www.vm.ibm.com/related/perfkit/>
- ▶ CA Explore Performance Management  
<http://ca.com/us/products/product.aspx?id=7978>
- ▶ MINIMON from The Minimon People  
<http://www.minimon.com/software.htm#Minimon>

In the following sections, we explain what to look for in the areas of CPU, Memory, and I/O.

### 5.6.3 CPU considerations

In general, the available CPU processing power is dispatched over all active tasks based on their status and priorities. You can display and adjust partition priorities by using the PRTY command.

Example 5-21 shows that partition F1 (running VSE/POWER) has the highest priority and dynamic class Z has the lowest priority. You can also build a partition balancing group by placing an equal sign (=) between two or multiple partitions.

*Example 5-21 The PRTY command*

---

```
PRTY
AR 0015 PRTY Z,Y,S,R,P,C,BG,FA,F9,F8,F6,F5,F4,F2,F7,FB,F3,F1
AR 0015
AR 0015 1I40I  READY
```

---

In Example 5-22, F9, F8, and F6 have equal priorities and build a partition balancing group.

*Example 5-22 Partition balancing group*

---

```
PRTY Z,Y,S,R,P,C,BG,FA,F9=F8=F6,F5,F4,F2,F7,FB,F3,F1
AR 0015 PRTY Z,Y,S,R,P,C,BG,FA,F9=F8=F6,F5,F4,F2,F7,FB,F3,F1
AR 0015
AR 0015 SHARE F9= 100, F8= 100, F6= 100
AR 0015 1I40I  READY
```

---

You can also set the relative shares of each partition in the group as shown in Example 5-23.

*Example 5-23 Relative shares in a partition balancing group*

---

```
PRTY SHAR,F9=300,F8=200,F6=50
AR 0015 PRTY Z,Y,S,R,P,C,BG,FA,F9=F8=F6,F5,F4,F2,F7,FB,F3,F1
AR 0015
AR 0015 SHARE F9= 300, F8= 200, F6= 50
AR 0015 1I40I  READY
```

---

Use the partition priority settings to ensure that important applications get enough CPU power if you are running CPU constrained. However, in most cases, priorities only make a difference if you are running CPU constrained. As long as enough unused CPU power is available, every partition, even the one with the lowest priority, gets its part of the CPU power.

A general rule, set a server partition, that is a partition that acts as a server for other partitions, to a higher priority than the partitions that are using the services from the server partition. Therefore, the TCP/IP partition or partitions normally should be set to a higher priority than the applications that use TCP/IP services, such as a partition running DB2 applications.

Although z/VSE supports up to ten active processors, do not use more than three CPUs per z/VSE system. Only run with two or three CPUs if your workload is able to exploit the additional CPUs. Turning on more CPUs than the workload can exploit causes additional overhead with no benefit. To exploit additional CPUs, run at least the number of CPUs plus one concurrent active batch or CICS partitions.

One z/VSE application running in a partition can only exploit the power of one CPU. Even if more than one CPU is active, one partition does not benefit from additional CPUs, which is important for applications running under CICS (CICS TS 1.1 and CICS/VSE 2.3). One CICS

region runs in a partition and, therefore, can only get the power of one CPU. All applications running within that CICS region only get a portion of the power of one CPU.

Because of these considerations, z/VSE runs best on a fast single CPU system. To learn if your workload can exploit more than one CPU, enter the QUERY TD command as follows:

```
SYSDEF TD,RESETCNT      reset the counters
// run your workload
QUERY TD                display the counters
```

Example 5-24 shows the output of using the QUERY TD command.

*Example 5-24 Output from running the QUERY TD command*

---

```
query td
AR 0015 CPU STATUS SPIN_TIME NP_TIME TOTAL_TIME NP/TOT
AR 0015 00 ACTIVE 0 428 688 0.622
AR 0015
-----
AR 0015 TOTAL 0 428 688 0.622
AR 0015
AR 0015 NP/TOT: 0.622 SPIN/(SPIN+TOT): 0.000
AR 0015 OVERALL UTILIZATION: 2% NP UTILIZATION: 1%
AR 0015
AR 0015 ELAPSED TIME SINCE LAST RESET: 27641
AR 0015 1I40I READY
```

---

In addition to the total used CPU time (TOTAL\_TIME) in milliseconds and the CPU utilization in percent, the value for the non-parallel part of the CPU time (NP\_TIME) is important. The amount of non-parallel CPU time is an indication the number of CPUs that the workload can exploit. The value for NP/TOT shows the percentage of the non-parallel part. It is also called the *nonparallel share* (NPS). In the previous example, an NPS of 0.622 means that 62.2% of the workload is non-parallel CPU time.

You can use the following equation to calculate the number of CPUs that a workload can use:

$$\text{Maximum CPUs exploited} = 0.8 / \text{NPS}$$

In Example 5-24, the workload can exploit  $0.8 / 0.622 = 1.3$  CPUs. Therefore, a second CPU does not bring much benefit in this case.

**Tip:** Our observation is workload dependent. Use the QUERY TD command as shown multiple times during heavy workload times. You might discover that, at times, your workload can exploit two CPUs, for example, but at other times, only one CPU can be exploited.

## 5.6.4 Memory analysis

High memory consumption can also be the reason for bad performance. In z/VSE, most areas, such as partitions, have a predefined size. If the application running in the partition exceeds the predefined size, it is unable to obtain more memory, and therefore, it will not negatively affect other applications running in the system. Instead your application might fail with an appropriate error message. However, if the application is running under CICS, it might allocate huge amounts of memory, affecting other CICS applications negatively.

Memory consumption can cause a bottleneck if the actual amount of memory used by the applications (also called the *working set*) exceeds the amount of real memory. When this

happens, the z/VSE system must perform paging to the page data set to satisfy the increased memory usage. Page I/Os can dramatically affect performance.

Provide enough real storage to your z/VSE system so that no or only minimal page I/Os occur. You can check for page-I/O activity by using either a performance monitor or the SIR command. The output of the SIR command shown in Example 5-25 contains values for total page-in I/Os and page-out I/Os.

*Example 5-25 The SIR command output of a page I/O*

---

```

sir
...
AR 0015 PGIN  TOT.= 0000002877      EXP.AVRGE.= 0000000029/SEC
AR 0015 PGOUT TOT.= 0000004493
AR 0015      UNC.= 0000001724      EXP.AVRGE.= 0000000002/SEC
AR 0015      PRE = 0000002769      EXP.AVRGE.= 0000000007/SEC
...

```

---

To learn about the memory layout of your z/VSE system, use the following commands:

- MAP** Shows the memory layout of the system and static partitions.
- MAP Fn** Shows the memory settings for partition Fn.
- GETVIS SVA,DETAIL** Shows the system GETVIS statistics, including details.
- GETVIS Fn,DETAILS** Shows the GETVIS statistics for partition Fn.
- PDISPLAY DYNC** Shows the memory layout of dynamic partitions.

Use the following Interactive Interface panels to view storage layout information:

- ▶ Display Storage Layout, by using SYSA fastpath 363
- ▶ Display CICS TS Storage, by using SYSA fastpath 364

## 5.6.5 I/O considerations

I/O is not a major factor for a typical DB2 workload that uses the DB2 client to communicate with a DB2 server over the network. However, if your applications also access other resources, such as files, clusters, or tapes, I/O might still play a role in the overall performance picture.

The first step is to determine the type of resource that is causing a performance problem. Therefore, a monitor tool is required. Most performance monitor tools provide a view for I/O activity per device address (CUU), per file, or both. Check for those files or devices that have the most activity or those that have extremely high I/O response times.

If a performance monitor is not available, use the SIR SMF command as shown in Example 5-26.

*Example 5-26 The SIR SMF command*

---

```

SIR SMF=ON          Turn SIR SMF monitoring on
SYSDEF TD,RESETCNT  Reset the counters
// run the workload
SIR SMF or SIR SMF,VSE  Display the counters
SIR SMF=OFF         Turn SIR SMF monitoring off

```

---

Example 5-27 shows the output that the SIR SMF command produces.

*Example 5-27 Output of the SIR SMF command*

---

```
sir smf
AR 0015 DEVICE I/O-CNT      QUEUED      CONNECT      DISCONN      TOTAL
AR 0015                               msec/SSCH    msec/SSCH    msec/SSCH    msec/SSCH
AR 0015
AR 0015 150          61          0.188        0.360        0.000        0.549
AR 0015 151          521         0.198        0.369        0.000        0.568
AR 0015 152          10          0.179        0.358        0.000        0.563
AR 0015 1I40I  READY
```

---

You see one line per device address (CUU) that contain the number of I/Os performed (I/O-CNT) and the times in milliseconds for QUEUED, CONNECT, DISCONN, and TOTAL. Look at the total time, which is the time that the I/O request took from starting the I/O (SSCH, Start Subchannel) until it completed.

The SIR SMF command shows I/O times monitored by the I/O subsystem. The SIR SMF,VSE command shows I/O times that are monitored by z/VSE.

## 5.6.6 z/VM and LPAR considerations

Today, z/VSE systems run in an LPAR or in a z/VM guest. Both the LPAR hypervisor and z/VM represent a virtualization layer. That is, most resources that are available to your z/VSE system are not real resources, but virtual resources, unless they are dedicated.

In a virtualized environment, over-committing resources is a common practice. This means, for example, that your z/VSE system sees more real storage than exists. In most cases, the system never uses the full memory that it seems to have available. The virtualization layer provides the system with the required resources if needed, for example, by paging out other memory areas. In general, over-committing resources can reduce system performance if it is overdone. Monitor the virtualization layer (z/VM) to check if it is doing heavy paging because of memory over-commitment.

Similar to memory, the CPUs that a guest system sees are only virtual, unless they are dedicated. This means that the guest system might not get the full power of a CPU, because the underlying virtualization layer assigns only a portion of the real CPUs to the guest, based on the priorities, shares and capping, and the CPU consumption of other systems that run concurrently.

Use a monitor on the virtualization layer, such as the z/VM Performance Toolkit, to check how much of the available real CPU power is distributed to the guest system. If another system that runs concurrently takes away most of the CPU power, your applications might not run as fast as expected. In this case, your applications still work as expected. However, you might need to adapt the shares for the affected z/VM guests to give your system enough CPU power.

## 5.7 DB2 Server for VSE & VM monitoring and tuning

You can use various configuration options and tuning parameters to optimize the performance of your application server and application requester.

A range of tools for monitoring performance is available in both z/VM and in z/VSE. Each tool covers a particular area or a different level of the overall system. In the following sections, we give an overview of the various tools that are available.

### 5.7.1 Operating system tools for monitoring database workload behavior

The following operating system tools are available to help you monitor the behavior of your database workload:

- ▶ The *CP Monitor subsystem* measures the performance of the VM operating system and its resources. The VM/Performance Reporting Facility (VM/PRF) product creates usage and historical reports from those measurements. You can control the amount and nature of the data collection, based on the analysis that you want to do. To create reports from the collected data, you can use VM/PRF to produce standard reports. This facility contains reports that are helpful in monitoring the overall DASD I/O performance of your database. The CP monitor subsystem is included with the VM system. VM/PRF is an optional product from IBM.
- ▶ The *CP INDICATE USER and QUERY TIME commands* measure the resources that are consumed by your database virtual machine. The command output includes measurements of system paging, database manager DASD I/O, and CPU load. This is a z/VM CP command.
- ▶ The *Real Time Monitor VM/ESA® (RTM VM/ESA)* provides online performance monitoring. Data is typically gathered in short intervals, usually one to three minutes. You can use this tool to capture system level data about your system and the database machine. It is an optional product from IBM.
- ▶ The *CICS Monitoring Facility* measures the performance of CICS under VSE and CICS/SPARS/VSE creates historical reports. Both are optional products from IBM.
- ▶ The *CIRD transaction* displays a snapshot of the links between CICS and your DB2 application server. It is provided with the DB2 Server for VSE & VM base product.

### 5.7.2 DB2 Server for VSE & VM tools

You can use the following tools for DB2 Server for VSE & VM to perform system monitoring and tuning:

- ▶ Whenever the application server starts, it automatically displays its *initialization parameters*, which describe how the server is configured. This facility is included with the DB2 Server for VSE & VM base product.
- ▶ The *DB2 Server for VSE & VM system catalog* contains information about dbspaces, tables, indexes, keys, packages, authorities, and other objects in the database. Much of the information is used by the database manager when deciding how to retrieve data from the database.
- ▶ The *SHOW operator commands*, which are available with DB2 Server for VSE & VM base product, show the status of the application server. For example, these commands can monitor user activity, locking, log usage, and storage.
- ▶ The *COUNTER operator command* measures the performance of your application server by recording how often significant events occur in the database manager. These events

relate to workload, locking, and database manager storage (buffer pools). This command is included in the DB2 Server for VSE & VM base product.

- ▶ *IBM DB2 Control Center for VSE and VM* automates DBA functions such as archiving, recovery, adding dbextents, deleting dbextents, adding dbspaces, startup, shutdown, startup parameter changing, dbspace reorganizations, catalog index reorganizations, and database monitoring. Any of these functions can be initiated immediately by an automated user (local or remote), or they can be scheduled to execute at any specified date and time, or repetitive execution interval.
- ▶ *DB2 Server for VSE & VM*
  - The *accounting facility* records how much CPU time is consumed and how many buffer pool accesses were done while a user was signed onto the application server.
  - The *trace facility* records the sequence of events that occur in different components of the database manager. For example, you can trace the sequence of locks that lead up to a deadlock.

While the accounting facility and trace facility can be useful in diagnosing performance problems, use them very sparingly. Both tools consume a great amount of system resources and can severely affect overall performance when they are activated.

For more information about the accounting facility, see the *DB2 Server for VSE System Administration*, SC09-2981, or *DB2 Server for VM System Administration*, SC09-2980. For more information about the trace facility, see *DB2 Server for VSE & VM Operation*, SC09-2986.

- ▶ The *SHOW TARGETWS operator command of the DB2 Server DSS Directory Service Server* measures the amount of main and expanded storage that your database machine is currently using. This command is included with the DB2 Server DSS feature. See *DB2 Server for VSE & VM Operation*, SC09-2986.
- ▶ The *COUNTER POOL operator command of the DB2 Server DSS Directory Service Server* measures the performance of individual storage pools, the internal dbspaces, and the directory. This command is included with the DB2 Server DSS feature. See *DB2 Server for VSE & VM Operation*, SC09-2986.

For detailed information about monitoring, improving performance, and tuning, see *DB2 Server for VSE & VM Performance Tuning Handbook*, GC09-2987.

### Online documentation for database tuning

For additional information about database tuning, consult the following online documentation:

- ▶ DB2 explain function to see the access plan for SQL queries  
<http://publib.boulder.ibm.com/infocenter/db2luw/v8/index.jsp?topic=/com.ibm.db2.udb.doc/admin/c0005135.htm>
- ▶ DB2 and DRDA trace (db2trc)  
<http://publib.boulder.ibm.com/infocenter/db2luw/v8/index.jsp?topic=/com.ibm.db2.udb.doc/core/r0002027.htm>
- ▶ DB2 snapshot  
<http://publib.boulder.ibm.com/infocenter/db2luw/v8/index.jsp?topic=/com.ibm.db2.udb.doc/admin/c0006003.htm>



## 5.8 Monitoring z/VSE applications and tuning recommendations

If an application or a set of applications does not run as fast as expected, it is important to know which SQL queries the applications are performing. Some SQL queries can cause a high load on the database server, while other SQL queries can cause a huge amount of data to be sent over the network. Both can affect performance of your applications.

In addition to SQL queries, your applications can access other resources, such as local files. Access to those resources can also affect the performance.

Applications can run inside CICS or in batch. In both cases, you might need different tools to monitor the applications and their activity with the database.

### 5.8.1 Monitoring connections

Two types of connections are possible:

- ▶ *Logical connections* are established by means of a DB2 CONNECT command. This command performs a handshake over a physical connection. The handshake process includes the authentication process to verify user IDs and passwords.
- ▶ *Physical connections* are realized by establishing a TCP/IP connection to the database server. Establishing a physical connection is, in most cases, more expensive than establishing a logical connection. It usually involves TCP/IP name lookup, connection establishment, and handshaking. Use the DB2 connection pool function to reduce the cost of connection establishment for physical connections. If you use a connection pool, physical connections are used on a free basis and not assigned statically to any logical connection.

For both types of connections, several tools are available to monitor and tune them.

#### Monitoring logical connections

To monitor logical connections, look at the database server snapshots. Look for such values as currently active connections, maximum allowed connections, number of worker agents, and so on.

Check whether enough connections and workers are available to support the number of clients and connections from clients that you are using. If you are using a connection pool, make sure that the pool contains enough connections. If insufficient connections are available, some clients might have to wait for a connection to become free. This can dramatically affect the performance of your client applications.

#### Monitoring physical connections

To monitor physical connections, in most cases, you can use tools, such as commands provided by the TCP/IP stacks or network sniffer tools, to monitor TCP/IP communication. Using network sniffers is a more advanced task that requires good knowledge of network protocols such as IP, TCP, and DRDA. We discuss network monitoring in detail in 5.5, “Network monitoring and tuning” on page 143.

## 5.8.2 Authentication considerations

As part of an SQL CONNECT statement, the database server performs authentication to the underlying operating system such as Linux. Depending on how your applications are written, a huge amount of CONNECT statements being executed can exist. Some CICS programs perform a CONNECT statement every time a new program is invoked. Performing a CONNECT too often can add a certain amount of overhead.

If you upgrade the Linux on System z operating system from SUSE Linux Enterprise Server 9 to SUSE Linux Enterprise Server 10, as part of the migration to SUSE Linux Enterprise Server 10, the default password encryption algorithm has changed from DES to BLOWFISH. While the DES algorithm can be accelerated by the CPACF hardware crypto feature of System z, the BLOWFISH algorithm cannot be accelerated. In a such situation, the changed algorithm creates a huge amount of overhead and dramatically increases the CONNECT duration. The solution is to manually change the encryption algorithm to DES.

## 5.8.3 Batch application monitoring

To monitor a batch application, use a monitoring tool. The following tools that are part of z/VSE System package and vendor tools can be helpful:

- ▶ System Activity Dialogs (SYSA fastpath 361 and 362), which display real-time performance information about the z/VSE system, including CPU, static and dynamic partitions, and I/O
- ▶ The QUERY TD command, which shows information about CPU usage on the console
- ▶ The SIR SMF command with which you can collect and display I/O related performance information using the SMF
- ▶ The SIR MON command with which you can collect and display internal monitor counters, such as SVC counters and system services counters.
- ▶ VSE Job Accounting exit (see skeleton SKJOBACC in ICCF library 59), which prints performance related information, such as CPU and I/O usage, to SYSLST after each job step
- ▶ VSE Navigator tool, which displays and collects real-time performance information about the z/VSE system, including CPU, static and dynamic partitions, and I/O, similar to the System Activity Dialogs
- ▶ VSE Health Checker tool, which is a diagnosis utility to retrieve, display, and analyze performance relevant data from a z/VSE system  
This tool is based on snapshot data.
- ▶ MAP and GETVIS commands, which display information about memory usage and partition sizes on the console
- ▶ Messages issued by the application itself or by z/VSE, for example, job duration
- ▶ Source code debugger, such as the IBM Debug Tool, to follow the program flow
- ▶ A z/VSE performance monitor that shows I/O requests, I/O service times, CPU usage, job duration, and so on per job step

For more details and additional tool descriptions, see 5.6, “z/VSE system monitoring and tuning” on page 148.

## 5.8.4 CICS application monitoring

To monitor a CICS application, the following CICS internal tools and vendor product tools can be helpful:

- ▶ CICS Statistics (STAT transaction, see the DFH0STAT skeleton in the ICCF library 59)
- ▶ CICS built-in tools such as CEMT INQUIRE to view certain CICS resources
- ▶ Source code debugger (IBM Debug Tool) to follow the program flow
- ▶ CEDF/CEDX (Execution Diagnostic Facility) to step through the CICS commands of your program
- ▶ CICS AUX traces for a deep look into what is happening

This tool produces a lot of output and can be difficult to read and interpret. You might need help from IBM Support for when using this tool.

- ▶ A CICS performance monitor that shows I/O requests, I/O service times, CPU usage, duration, and so on per transaction

You can use the same tools that are mentioned in 5.6, “z/VSE system monitoring and tuning” on page 148.

## 5.8.5 SQL query tuning

To understand which SQL queries an application uses, in addition to consulting the application’s source code, look into the package that was produced by the DB2 preprocessor in the database when you compiled your programs. The package contains information about all SQL queries that the application might execute.

To see how expensive a particular SQL query is, you can use the DB2 explain facility. In DB2 Linux, the explain facility shows the access path that is defined by DB2 and helps you to analyze how the data will be accessed and how you can improve the statement’s performance.

This information helps you to understand how individual SQL statements are executed. You can tune the statements with more efficient SQL. Changing your database manager configuration to improve performance, for example, adds indexes for statements that use table scan indexed access.

You collect and use explain data to understand how the database manager accesses tables and indexes to satisfy your query. You also use this data to evaluate your performance-tuning actions.

When you change some aspect of the database manager, the SQL statements, or the database, examine the explain data to understand how your action has changed performance. The captured data includes the following information:

- ▶ Sequence of operations to process the query
- ▶ Cost information
- ▶ Predicates and selectivity estimates for each predicate
- ▶ Statistics for all objects referenced in the SQL statement at the time that the explain information is captured
- ▶ Values for the host variables, parameter markers, or special registers used to re-optimize the SQL statement

As an example, your analysis might show that an index is no longer being used as part of the access path. By using the catalog statistics information, you might notice that the number of index levels (NLEVELS column) is now substantially higher than when the query was first bound to the database.

You might then choose to perform one of the following actions:

- ▶ Reorganize the index.
- ▶ Collect new statistics for your table and indexes.
- ▶ Gather explain information when rebinding your query.

After you perform one of these actions, examine the access plan again. If the index is used again, performance of the query might no longer be a problem. If the index is still not used or if performance is still a problem, perform a second action and examine the results. Repeat these steps until the problem is resolved.

### **Evaluation of performance tuning efforts**

You can take a number of actions to help improve query performance, such as adjusting configuration parameters, adding containers, collecting fresh catalog statistics, and so on.

After you make a change in any of these areas, you can use the explain facility to determine the impact, if any, that the change has on the access plan chosen. For example, if you add an index or MQT based on the index guidelines, the explain data can help you determine whether the index or materialized query table is used as you expected.

Although the explain output provides information with which you can determine the access plan that was chosen and its relative cost, the only way to accurately measure the performance improvement for a query is to use benchmark testing techniques.

For details about the DB2 explain facility, see the “Explain facility” topic in the DB2 Database for Linux, UNIX, and Windows Information Center at the following address:

<http://publib.boulder.ibm.com/infocenter/db2luw/v9/topic/com.ibm.db2.udb.admin.doc/doc/c0005134.htm>

## **5.8.6 DB2 applications**

Applications that are already written for use with a database, for example, DB2 Server for VSE, in most cases run unchanged with a remote database. You might need to tune only a few applications that use inefficient SELECT statements or cause huge amounts of data to be transferred between the application and the database server. Such applications have already been written with the relational concepts in mind, so that they avoid VSAM style sequential processing. For a compatibility list of functions, see 4.3.4, “Compatibility of DB2 VSE with DB2 for Linux, UNIX, and Windows” on page 117.

## **5.8.7 Best practices for designing high-performance DB2 applications**

If you develop applications that effectively work with a DB2 database, consider the following best practices:

- ▶ Let the database determine which rows to update through a WHERE clause. Do not sequentially read all rows and perform the decision in your application code. If you need custom code to be executed to perform specific processing, use stored procedures instead, which run inside the database server.
- ▶ For all SELECT statements, make sure that they return small result sets, containing only those rows that your application needs. Especially SELECT statements with open ended

WHERE clauses (WHERE field >= value) can create huge result sets. If you know that there is a good chance that you do not fetch all rows from the result set, you can use the OPTIMIZE FOR n ROWS clause or FETCH FIRST n ROWS ONLY to tell the DB2 optimizer that you plan to retrieve only a certain number of rows.

- ▶ Transfer as little data as possible with every statement or result set:
  - If no further rows are required from a result set, stop fetching rows and close the result set. This method does not save on overhead for creating the huge result set, but at least it saves on overhead for transferring the remaining rows. Note that the result set contents are transferred in larger chunks (DB2 BLOCK SIZE).
  - Only include those columns into a SELECT statement that you really need. Do *not* use SELECT \* FROM table, but instead specify the columns of interest.
- ▶ Check every SQL statement that your application executes through the DB2 Explain feature (see Application Monitoring and Tuning in this feature). Make sure that you build all required indexes.
- ▶ Avoid unnecessary SORTs whenever possible. If you do not need to fetch the rows from a result set in a specific order, do not specify the SORTED BY clause.

Most of these tuning tips require larger changes in the affected applications. In some cases, you might even need to redesign part of the application.

You must adapt the applications to use the features and functions of a relational database in order to make them perform well. Instead of using traditional VSAM style sequential processing, use relational concepts. Let the database, and not your application, do the work, such as selecting the affected rows.

### 5.8.8 Tuning considerations for migrated VSAM applications

In most customer situations, the applications that show unacceptable performance have been migrated over from a pure VSAM application to a DB2 application. Often, the first migration step involved changing them from pure VSAM to use DB2 Server for VSE. If the next step of rehosting the database is done to move the data to DB2 on Linux on System z with remote access through the DB2 client functionality on z/VSE, some special effects can appear.

While this migration path might seem smooth and easy at first sight, it contains pitfalls and points to consider. If you do not pay attention to these pitfalls, you will end up with an application that might work well function-wise, but that does not perform as expected. We have seen cases where batch jobs run only a few minutes on a local DB2 database and run for hours after they are migrated to access a DB2 on Linux on System z. In several cases, the bad performance is a result of poor application design and migration, *not* the fault of DB2 on Linux on System z.

#### Differences between VSAM and a database

VSAM is an access method, not a database. Therefore, applications that are optimized for VSAM perform different types of processing than database applications. A VSAM application reads through a VSAM file sequentially. Based on the contents of a field or multiple fields within each record, determines if that record must be processed, for example, needs updating. VSAM is highly optimized for sequential processing. It contains such features as read ahead, which in many cases results in good performance. VSAM also provides keyed access with KSDS type clusters for locating and reading a specific record. VSAM maintains an index for keyed access.

In contrast to VSAM, a database is optimized to execute SQL queries and perform complex searches (WHERE clauses) on a huge set of rows at once. A database contains various

indexes that allow fast access to specific rows (records). For every SQL query, the database creates an access plan that controls how the rows that match the query are found the most efficient way. In contrast to VSAM, a database can update multiple rows in one statement (UPDATE ... WHERE ...). While a VSAM application has to locate and read each of the records to update, before it can perform the update on the record, a database application can do the same in just one single SQL statement. It is obvious that the database can perform such an update more efficiently than a VSAM application.

The problem with most migrated z/VSE applications is that they do not use the capabilities of a database, such as the update processing explained previously. Instead, such migrated applications still work the same way as they did when it was a pure VSAM application. Instead of using an UPDATE SQL statement that affects all rows to update, the applications first read the table sequentially (SELECT \* FROM table), the same way as it did with VSAM. Next, the application fetches every single row from the result set and look at some fields to decide whether that row needs to be updated. If the row is to be updated, the application performs an UPDATE statement that only updates that single row. Now, in most cases, the update statement causes the result set of the SELECT statement to be no longer valid. Therefore, the application has to perform the SELECT statement once again, this time with a WHERE clause to start at the last processed row.

From a function perspective, the processing described in the previous section results in the same rows getting updated. However, it is obvious that such processing with SQL can never be as fast as the original VSAM processing. It can also not be as fast as a single UPDATE statement that affects all rows to update at once. Unfortunately the processing involves multiple SQL statements to be executed, some of which can be expensive in CPU and time. Such a SELECT statement virtually returns a result set that contains all rows of the table (or all remaining rows starting at the last processed row). Building the result set on the database server can be time consuming, especially when a table scan and a sort are involved. In addition, transferring the result set's contents over the network causes a lot of overhead. In contrast to that, the UPDATE statement that updates all affected rows at once does not create a large result set, nor does it transfer huge amounts of data over the network.

## Configuration examples

In this appendix, we provide configuration examples for the TCP/IP network in z/VSE, the adaptations in CICS TS for z/VSE, and the DB2 Client environment in z/VSE:

- ▶ “The IPINIT00.L TCP/IP configuration member in z/VSE” on page 170
- ▶ “File definitions in CICS” on page 173
- ▶ “The ARISIVRR.Z parameter file for the DB2 setup” on page 175

## The IPINIT00.L TCP/IP configuration member in z/VSE

Example A-1 shows the entire IPINIT00.L member of the PRD2.CONFIG z/VSE library. It contains the configuration information for TCP/IP on z/VSE.

*Example: A-1 z/VSE configuration member IPINIT00.L for TCP/IP*

```
*-----*
*
*      DEFINE THE CONSTANTS
*
*-----*
SET IPADDR = 9.152.86.244
SET MASK = 255.255.252.000
*
SET ALL_BOUND      = 30000
SET WINDOW         = 65535
SET TRANSFER_BUFFERS = 20
SET MAX_BUFFERS   = 6
SET TELNETD_BUFFERS = 20
SET RETRANSMIT    = 100
SET DISPATCH_TIME = 30
SET REDISPATCH    = 10
SET CONSOLE_HOLD  = ON
*
SET GATEWAY        = ON
SET DOWNCHECK     = OFF
*-----*
*
*      WAIT FOR VTAM STARTUP
*
*-----*
WAIT    VTAM
*-----*
*
*      DEFINE THE COMMUNICATION LINKS
*
*-----*
DEFINE LINK, ID=OSAFE, TYPE=OSAX, DEV=(0D00,0D01), DATAPATH=0D02, MTU=1492
DEFINE LINK, ID=HIPSO, TYPE=OSAX, DEV=(0500,0501), DATAPATH=0502, -
        IPADDR=172.16.0.6
SET LINK_RETRY      = 18000
*-----*
*
*      DEFINE ROUTINE INFORMATION
*
*-----*
DEFINE ROUTE, ID=ALL,      LINKID=OSAFE,      IPADDR=0.0.0.0, -
        GATEWAY=9.152.86.245
DEFINE ROUTE, ID=TOLNX, LINKID=HIPSO, IPADDR=172.16.0.0
DEFINE MASK, ID=MHIPSO, NETWORK=172.16.0.0, MASK=255.255.255.0
*-----*
*
*      DEFINE TELNET DAEMONS
*
```



```

*-----*
DEFINE TELNETD, ID=LU, TERMNAME=TELNLU, TARGET=DBDCCICS, PORT=23, COUNT=4, -
    LOGMODE=S3270, LOGMODE3=D4B32783, LOGMODE4=D4B32784, -
    LOGMODE5=D4B32785, POOL=YES
SET CONNECT_SEQUENCE =OFF
*-----*
*
*          DEFINE FTP DAEMONS
*
*-----*
DEFINE FTPD, ID=FTP, PORT=21, COUNT=2
DEFINE FTPD, ID=FTP11, PORT=21
DEFINE FTPD, ID=FTP12, PORT=21
*-----*
*
*          DEFINE HTTP DAEMONS
*
*-----*
* DEFINE HTTPD, ID=MANUS, ROOT='PRD2.TEST'
*-----*
*
*          LINE PRINTER DAEMONS
*
*-----*
DEFINE LPD, PRINTER=FAST, QUEUE='POWER.LST.A'
DEFINE LPD, PRINTER=FASTLIB, QUEUE='PRD2.SAVE'
DEFINE LPD, PRINTER=LOCAL, QUEUE='POWER.LST.A', LIB=PRD2, SUBLIB=SAVE
*-----*
*
*          AUTOMATED LINE PRINTER CLIENT
*
*-----*
* DEFINE EVENT, ID=LST_LISTEN, TYPE=POWER, CLASS=X, QUEUE=LST, ACTION=LPR
*-----*
*
*          SETUP THE FILE SYSTEM
*
*-----*
DEFINE FILESYS, LOCATION=SYSTEM, TYPE=PERM
*
DEFINE FILE, PUBLIC='IJSYSRS', DLBL=IJSYSRS, TYPE=LIBRARY
DEFINE FILE, PUBLIC='PRD1', DLBL=PRD1, TYPE=LIBRARY
DEFINE FILE, PUBLIC='PRD2', DLBL=PRD2, TYPE=LIBRARY
DEFINE FILE, PUBLIC='POWER', DLBL=IJQFILE, TYPE=POWER
*
*-----*
*
*          MESSAGE SETTINGS
*
*-----*
SET MESSAGE CRITICAL      = ON
SET MESSAGE WARNING      = ON
SET MESSAGE VITAL        = ON
SET MESSAGE IMPORTANT    = ON
SET MESSAGE INFORMATION  = ON

```

```

SET PING_MESSAGE          = ON
*-----*
*                               *
*   PERMITTED USERIDS       *
*                               *
*-----*
*-----*
*                               *
*   SECURITY SETTINGS       *
*                               *
*-----*
DEFINE SECURITY,DRIVER=BSSTISX
SET SECURITY                = ON
SET SECURITY_ARP           = OFF
SET SECURITY_IP            = OFF
SET ISOLATION              = OFF
*-----*
*                               *
*   OPERATION AND STARTUP  *
*                               *
*-----*
SET RECORD                  = OFF
*-----*
*                               *
*   SYMBOLIC NAMES        *
*                               *
*-----*
DEFINE NAME,NAME=ZVSEDB2,IPADDR=9.152.86.244
DEFINE NAME,NAME=LNxDB2,IPADDR=9.152.26.187
*-----*
*                               *
*   START HTTP SERVER     *
*                               *
*-----*
* DEFINE HTTPD,ID=MYHTTPD,ROOT=PRIMARY.SYSA
*-----*
*                               *
*   SETUP MEMBER NETWORK.L TO
*   EXECUTE ONCE THE ENGINE HAS
*   BEEN ACTIVATED       *
*                               *
*-----*
INCLUDE NETWORK,DELAY

```

---

# File definitions in CICS

For online applications and interactive SQL (ISQL), you define the SQLGLOB and SQLBIND files in CICS TS by using the online definition as shown in Example A-2.

*Example: A-2 CICS FCT definitions*

---

```

CEDA DEF FILE(          )
  OBJECT CHARACTERISTICS
  CEDA DEF File( SQLGLOB )
    File          : SQLGLOB
    Group         : DB2750C
    Description   : GLOBAL VAR IN DB2 CLIENT
  VSAM PARAMETERS
    DSName       : DB2.SQLGLOB.MASTER
    Password     :                               PASSWORD NOT SPECIFIED
    Lsrpoolid    : 01                          1-15 | None
    Catname      : VSESPUC
    DSNSharing   : Noreqs                       Noreqs | Allreqs | Modifyreqs
    STRings      : 002                          1-255
    Nsrgroup     :
    SHr4access   : Key                          Key | Rba
  REMOTE ATTRIBUTES
    REMOTESystem :
    REMOTENAME   :
    RECORDSize   :                             1-32767
+ Keylength     :                             1-255
+ DSNSharing    : Noreqs                       Noreqs | Allreqs | Modifyreqs
  STRings       : 002                          1-255
  Nsrgroup      :
  SHr4access    : Key                          Key | Rba
  REMOTE ATTRIBUTES
    REMOTESystem :
    REMOTENAME   :
    RECORDSize   :                             1-32767
    Keylength    :                             1-255
  INITIAL STATUS
    STatus       : Enabled                     Enabled | Disabled | Unenabled
    Opertime     : Startup                     Firstref | Startup
  BUFFERS
    Databuffers  : 00003                       2-32767
    Indexbuffers : 00002                       1-32767
  DATATABLE PARAMETERS
+ Table         : No                          No | Cics | User
+ INITIAL STATUS
    STatus       : Enabled                     Enabled | Disabled | Unenabled
    Opertime     : Startup                     Firstref | Startup
  BUFFERS
    Databuffers  : 00003                       2-32767
    Indexbuffers : 00002                       1-32767
  DATATABLE PARAMETERS
    Table        : No                          No | Cics | User
    Maxnumrecs   :                             16-16777215
  DATA FORMAT
    RECORDFormat : V                          V | F

```

```

OPERATIONS
  Add           : Yes           No | Yes
  Browse       : Yes           No | Yes
  DElete       : Yes           No | Yes
  REAd         : Yes           Yes | No
+ Update      : Yes           No | Yes + AUTO JOURNALLING
  JOurnal      : No            No | 1-99
  JNLRead     : None          None | Updateonly | Readonly | All
  JNLSYNCRd   : No            No | Yes
  JNLUpdate   : No            No | Yes
  JNLAdd      : None          None | Before | After | All
  JNLSYNCRw   : Yes           Yes | No
RECOVERY PARAMETERS
  RECOVery    : None          None | Backoutonly | All
  Fwdrcovlog  : No            No | 1-99

```

SQLBIND:

```

OBJECT CHARACTERISTICS                                CICS RELE
CEDA DEF File(                                     )
  File        : SQLBIND
  Group       : DB2750C
  DEScription : DB2 BIND FILE
VSAM PARAMETERS
  DSName      : DB2.BIND.MASTER
  Password    :                PASSWORD NOT SPECIFIED
  Lsrpoolid   : 01             1-15 | None
  Catname     : VSESPUC
  DSSharing   : Noreqs         Noreqs | Allreqs | Modifyreqs
  STRings     : 002           1-255
  Nsrgroup    :
  SHr4access  : Key           Key | Rba
REMOTE ATTRIBUTES
  REMOTESystem :
  REMOTENAME   :
  RECORDSize   :                1-32767
+ Keylength   :                1-255
+ DSSharing   : Noreqs         Noreqs | Allreqs | Modifyreqs
  STRings     : 002           1-255
  Nsrgroup    :
  SHr4access  : Key           Key | Rba
REMOTE ATTRIBUTES
  REMOTESystem :
  REMOTENAME   :
  RECORDSize   :                1-32767
  Keylength    :                1-255
INITIAL STATUS
  STatus      : Enabled       Enabled | Disabled | Unenabled
  Opentime    : Startup       Firstref | Startup
BUFFERS
  DATabuffers : 00003         2-32767
  Indexbuffers : 00002       1-32767
DATATABLE PARAMETERS
+ Table      : No            No | Cics | User

```

```

+ INITIAL STATUS
  Status      : Enabled          Enabled | Disabled | Unenabled
  Opentime    : Startup         Firstref | Startup
BUFFERS
  Databuffers : 00003          2-32767
  Indexbuffers : 00002        1-32767
DATATABLE PARAMETERS
  Table       : No             No | Cics | User
  Maxnumrecs  :                16-16777215
DATA FORMAT
  RECORDFormat : V             V | F
OPERATIONS
  Add         : No             No | Yes
  Browse      : Yes            No | Yes
  DElete     : No             No | Yes
  REAd       : Yes            Yes | No
+ Update     : No             No | Yes
+ AUTO JOURNALLING
  JOurnal    : No             No | 1-99
  JNLRead    : None           None | Updateonly | Readonly | All
  JNLSYNCRd  : No             No | Yes
  JNLUpdate  : No             No | Yes
  JNLAdd     : None           None | Before | After | All
  JNLSYNCRw  : Yes            Yes | No
RECOVERY PARAMETERS
  RECOVery   : None           None | Backoutonly | All
  Fwdrecovlog : No            No | 1-99

```

---

## The ARISIVRR.Z parameter file for the DB2 setup

The ARISIVRR.Z member is originally in the installation library of DB2 Server for VSE Client Edition PRD2.DB2750C. Example A-3 shows a customized version of the file for this installation. With these parameters, the ARISIMGC job prepares and installs the DB2 client for z/VSE.

*Example: A-3 Variables for DB2 client customization*

---

```

*****
* This file includes all parameters needed for Installation Process. *
* IMPORTANT:                                                         *
*   If ARISIVRR Control table structure is not preserved the       *
*   Installation EXEC will fail.                                     *
* STRUCTURE THE CONTROL TABLE                                       *
* (a) COMMENTS: all comments will start with '*' and be placed in 1st *
*               column of this file OR in column 40.                 *
* (b) 1st line will always be USERID.                               *
* (c) Control table divided into 3 parts:                             *
*     1. Parameters for Preparation Step                             *
*     2. Parameters for Installation Step                             *
*     3. Parameters for Migration Step                               *
*   IMPORTANT: Do not delete keyword 'START', it used to specify the *
*               start of new part or keyword 'END', which specifies  *
*               the end of a part.                                    *
* (d) Each part has 2 types of Parameters:                           *

```

```

*      1. Globals, which must start with keyword GLOBAL                *
*      2. Variables for each job, which start with name of a job      *
*      (ex. ARIS75JD)                                                *
* (e) Must use capital letters.                                       *
*
* Sample control table will look as follows:                          *
* | COLUMN 1 | COLUMN 46 |
* |          |          |
* |          |          |
* |          |          |
* |          |          |
* ->USERID  USERID  user1          ->* Some comment
*   * more comments
*   START OF PREPARATION STEP          * Don't delete this line
*   * more comments
*   GLOBAL  PARMNAME  PARMVALUE
*   * end of globals
*   ARIS75JD PARMNAME  PARMVALUE          * more comments
*   END OF PREPARATION STEP          * don't delete this line
*
* IMPORTANT: file must consist of 3 parts (PREPARATION/INSTALLATION/
*           MIGRATION)
*
* PARMNAME - is name that used in members for installation, and will
*           be replaced with PARMVALUE that you specify beside.
*
* DCR-ACTIVITY           - None
*
* PTR-ACTIVITY           - None
*
* Change Activity        - None
* APARNUM  mmm/yyyy - describe the change          @SandDcd
*           describe the change                    @SandDcd
*           describe the change                    @SandDcd
*****
USERID  USERID  DB2INST1          * don't delete this line
CLASS  PRINTER  V          * print class for $LST
START OF PREPARATION STEP          * don't delete this line
GLOBAL  GLOBCAT  VSESP.USER.CATALOG
GLOBAL  MASTERFILE  DB2.SQLGLOB.MASTER
GLOBAL  BINDFILE  DB2.BIND.MASTER
GLOBAL  BINDWFILE  DB2.BIND.WORKF
GLOBAL  ATYPE      CYL          * allocation unit
* PREPARATION STEP 5: DEFINE DB2 PROGRAMS AND TRANSACTIONS TO CICS
ARIS75JD  CSDCAT  VSESP.USER.CATALOG
ARIS75JD  GROUP   DB2750C
ARIS75JD  CICSFILE  CICS.CSD
ARIS75JD  TRANSEC  2
ARIS75JD  LIST     VSELIST          *
ARIS75JD  CICSTS   Y                *
ARIS75JD  ISQL     Y                *
* PREPARATION STEP 6.1: SETUP THE SQLGLOB FILE
ARIS758D  GLOBCAT  VSESP.USER.CATALOG
ARIS758D  VOLUME   SYSWK1          * CHANGE AS APPROPRIATE
ARIS758D  ATYPE1   TRACKS          * allocation unit
ARIS758D  ALLOC1  15

```

```

ARIS758D ALLOC2      5
* PREPARATION STEP 6.3: SETUP THE PREPROCESSOR BIND FILE
ARIS757D BINDCAT    VSESP.USER.CATALOG
ARIS757D VOLUME     SYSWK1                * CHANGE AS APPROPRIATE
ARIS757D ALLOC1     10
ARIS757D ALLOC2     5
* PREPARATION STEP 6.4: SETUP THE PREPROCESSOR BIND WORK FILE
ARIS759D BINDWCAT   VSESP.USER.CATALOG
ARIS759D VOLUME     SYSWK1                * CHANGE AS APPROPRIATE
ARIS759D ALLOC1     1
ARIS759D ALLOC2     1
* PREPARATION STEP 6.5: CONVERTING THE ISQL BIND FILE TO A VSAM FILE
ARISIQBD BINDCAT    VSESP.USER.CATALOG
* PREPARATION STEP 7: UPDATING STANDARD LABELS FOR THE NEW DATASETS
ARISSTDL BINDWCAT   VSESP.USER.CATALOG
ARISSTDL BINDCAT    VSESP.USER.CATALOG
END OF PREPARATION STEP                    * don't delete this line
START OF INSTALLATION STEP                 * don't delete this line
GLOBAL  CATALOG     VSESP.USER.CATALOG
GLOBAL  GLOBFILE    DB2.SQLGLOB.MASTER
GLOBAL  WDISK       SYSWK1                * WM CHANGE AS APPROPRIATE
GLOBAL  START       18000
GLOBAL  FOR         300
GLOBAL  LELIB       PRD2.SCEEBASE         * LE / C LIBRARY
GLOBAL  COMPILELIB  PRD2.DB2750C         * CHANGE AS APPROPRIATE
GLOBAL  CONF        PRD2.DB2750C         * User own config
GLOBAL  DBUSRID     DB2INST1             * WM USER ID AND PASSWORD
GLOBAL  DBPWWORD    MYLINPW              * WM FOR DEFAULT CONNECT
ARIS75CZ LIBNAME    PRD2.DB2750C         * ARISICON.A is placed
ARIS75SL CLIB      PRD2.DB2750C         * TCP AND OTHER LIBS
ARIS75SL PLIB      BSILIB.V241CFG,BSILIB.V241
* INSTALLATION STEP 1: SETUP THE DBNAME DIRECTORY
ARISBDID MIGRATION NO                    * don't change this value
ARISBDID OUTPUTB   ARISDIRD.A
ARISBDID LIB1      PRD2.DB2750C
* INSTALLATION STEP 3: LINK-EDIT DB2 ONLINE SUPPORT COMPONENTS
ARIS75BD CICS LIB  PRD1.BASE              * sublibrary for CICS comp.
ARIS75BD ARIS140D? //                    * EXECUTE=//, DON'T EXECUTE=*
ARIS75BD ARIS150D? //
ARIS75BD ARIS160D? //
* INSTALLATION STEP 5: INSTALL OPTIONAL COMPONENTS
ARIS75ED ARIS110D? //
ARIS75ED ARIS120D? //
ARIS75ED ARIS130D? //
* INSTALLATION STEP 6: RELOAD CCSID-RELATED PHASES PACKAGE
ARIS75WD SUM       N                      *
ARIS75WD DBNAME    SQLDS                  *
* INSTALLATION STEP 7: SELECTING NATIONAL LANGUAGE SUPPORT
ARIS75HZ PAGES     8192
* INSTALLATION STEP 7.1: INSTALL LANGUAGE AND ISQL HELP TEXT
ARIS75JZ LANGUAGE  AME
ARIS75JZ HELP      NO
ARIS75JZ CUU       181
ARIS75JZ DEFAULTL NO                    * select default language
ARIS380D NLSLIB    PRD2.DB275R           * see PD for explanation

```

```
ARIS380D CLIB      PRD2.DB2750C          * custom library
* INSTALLATION STEP 8: GRANT SCHEDULE AUTHORITY
ARIS75FD NAME     DBDCCICS
ARIS75FD NAMEPW   SQLDBAPW
ARIS75FD SUM      N                      *
ARIS75FD DBNAME   SQLDS                  *
* INSTALLATION STEP 9: CREATE CCSID RELATED PHASES
ARISCNVD PARMS    DB2INST1/MYLINPW/SAMPLE * CHANGE TO YOUR DBNAME
* INSTALLATION STEP 10: RUN THE SAMPLE PROGRAMS TO VERIFY INSTALLATION
ARIS6PLD PLILIB   Cmplr22.Plivse
ARIS6FTD FRTLIB   COMPILE.COMPILE
ARIS6CD  C-LIB    Cmplr22.Cvse
ARIS6C2D COBLIB   Cmplr22.Cobvse
END OF INSTALLATION STEP                * don't delete this line
START OF MIGRATION                      * don't delete this line
FOR FUTURE ENHANCEMENTS TO CLIENT
END OF MIGRATION STEP                  * don't delete this line
```

---





## Database manipulation

In this appendix, we provide the following applications for database definition and migration:

- ▶ “The db\_udcs.c program” on page 180
- ▶ “The sqle819a.h header file” on page 182
- ▶ “The crtnick.sqc C program” on page 183

## The db\_udcs.c program

Example B-1 shows the program that defines a database in Linux with an EBCDIC user-defined collated sequence.

*Example: B-1 The db\_udcs.c program*

---

```

/*****
**
** Source File Name = db_udcs.c
**
** Licensed Materials - Property of IBM
**
** Governed under the terms of the International
** License Agreement for Non-Warranted Sample Code.
**
** (C) COPYRIGHT International Business Machines Corp. 1996 - 2002
** All Rights Reserved.
**
** US Government Users Restricted Rights - Use, duplication or
** disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
**
** PURPOSE :
**     This program is an example of how APIs are implemented in order to
**     CREATE a DATABASE with a user-defined collating sequence
**
**     A user-defined collating sequence allows the user to specify the
**     collating behavior of the database. This can be used by
**     applications that require compatibility to the collating
**     behavior of host database products. For example, simulating
**     the collating behavior of DB2 VSE / VM CCSID 500 (EBCDIC International)
**     can be achieved by specifying a collating sequence that maps
**     codepage 819 characters to CCSID 500 characters when the
**     database is created.
**
** APIs USED :
**     INSTANCE ATTACH          sqlleatin()
**     CREATE DATABASE          sqllecrea()
**     DROP DATABASE           sqlledrpd()
**
** STRUCTURES USED :
**     sqllebdbdesc
**     sqlledbcountryinfo
**     sqlca
**
** OTHER FUNCTIONS USED :
**     'C' COMPILER LIBRARY :
**     stdio.h - printf
**
** external :
**     check_error :    Checks for SQLCODE error, and prints out any
**     [in util.c]      related information available.
**                     This procedure is located in the util.c file.
**
**

```

```

**      EXTERNAL DEPENDENCIES :
**      - Compiling and linking with a language C compiler supported on your platform.
**      - a database by the name of "VSELNXDB" does not exist.
**

```

```

**      Invocation of the program: db_udcs [userid password remote_nodename]
*****/

```

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sql819a.h> /* collating sequence mapping 819 to 500 */
#include <sqlutil.h>
#include <sqlenv.h>
#include "util.h"

```

```

#define CHECKERR(CE_STR)  if (check_error (CE_STR, &sqlca) != 0) return 1;

```

```

int main (int argc, char* argv[]) {
    struct sqlca sqlca;
    struct sqldbdesc db_desc;
    struct sqldbcountryinfo dbCountry;

    char userid[9];
    char passwd[19];
    char nodename[9];

    if (argc == 4) {
        strcpy (userid, argv[1]);
        strcpy (passwd, argv[2]);
        strcpy (nodename, argv[3]);

        /******
        /* ATTACH API called
        /******
        sqlcatin (nodename, userid, passwd, &sqlca);
    }
    else if (argc != 1) {
        printf("\nUSAGE: db_udcs [userid password remote_nodename]\n");
        return 1;
    }
}

```

```

strcpy(db_desc.sqldbdid, SQLE_DBDESC_2);
db_desc.sqldbccp = 0;
db_desc.sqldbcss = -1;
memcpy(db_desc.sqldbudc, sql8_819_500, SQL_CS_SZ);
db_desc.sqldbcmt[0] = 0x00;
db_desc.sqldbsgp = 0;
db_desc.sqldbnsg = 10;
db_desc.sqltsext = -1;
db_desc.sqlcatts = NULL;
db_desc.sqlusrts = NULL;
db_desc.sqltmpts = NULL;

```

```

strcpy(dbCountry.sqldbcodeset, "ISO8859-1");

```

```

strcpy(dbCountry.sqldblocale, "En_US");

printf ("CREATING the database VSELNXDB...\n");
printf ("please wait... this will take a while...\n");
/*****\
* CREATE DATABASE API called *
\*****/
sqlcrea( "VSELNXDB",
        "VSELNXDB",
        "",
        &db_desc,
        &dbCountry,
        '\0',
        NULL,
        &sqlca
        ) ;
CHECKERR("creating the database");
printf ("database VSELNXDB with a user-defined collating\n");
printf ("sequence created successfully.\n");

return 0;
}

```

---

## The sqlc819a.h header file

Example B-2 shows the header file that contains the translation table from ASCII 819 to EBCDIC International 500.

*Example: B-2 The sqlc819a.h header file*

---

```

/*****
**
** Source File Name: SQLC819A
**
** (C) COPYRIGHT International Business Machines Corp. 1987, 1997
** All Rights Reserved
** Licensed Materials - Property of IBM
**
** US Government Users Restricted Rights - Use, duplication or
** disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
**
** Function = Include File defining:
**           Collating Sequence
**           Source: CODE PAGE 819 (ISO Latin/1)
**           Target: CCSID 500 (EBCDIC International)
**
** Operating System: Windows NT
**
*****/
#ifndef SQL_H_SQLC819A
#define SQL_H_SQLC819A

#ifdef __cplusplus
extern "C" {

```

```

#endif

unsigned char sqle_819_500[256] = {
0x00,0x01,0x02,0x03,0x37,0x2d,0x2e,0x2f,0x16,0x05,0x25,0x0b,0x0c,0x0d,0x0e,0x0f,
0x10,0x11,0x12,0x13,0x3c,0x3d,0x32,0x26,0x18,0x19,0x3f,0x27,0x1c,0x1d,0x1e,0x1f,
0x40,0x4f,0x7f,0x7b,0x5b,0x6c,0x50,0x7d,0x4d,0x5d,0x5c,0x4e,0x6b,0x60,0x4b,0x61,
0xf0,0xf1,0xf2,0xf3,0xf4,0xf5,0xf6,0xf7,0xf8,0xf9,0x7a,0x5e,0x4c,0x7e,0x6e,0x6f,
0x7c,0xc1,0xc2,0xc3,0xc4,0xc5,0xc6,0xc7,0xc8,0xc9,0xd1,0xd2,0xd3,0xd4,0xd5,0xd6,
0xd7,0xd8,0xd9,0xe2,0xe3,0xe4,0xe5,0xe6,0xe7,0xe8,0xe9,0x4a,0xe0,0x5a,0x5f,0x6d,
0x79,0x81,0x82,0x83,0x84,0x85,0x86,0x87,0x88,0x89,0x91,0x92,0x93,0x94,0x95,0x96,
0x97,0x98,0x99,0xa2,0xa3,0xa4,0xa5,0xa6,0xa7,0xa8,0xa9,0xc0,0xbb,0xd0,0xa1,0x07,
0x20,0x21,0x22,0x23,0x24,0x15,0x06,0x17,0x28,0x29,0x2a,0x2b,0x2c,0x09,0x0a,0x1b,
0x30,0x31,0x1a,0x33,0x34,0x35,0x36,0x08,0x38,0x39,0x3a,0x3b,0x04,0x14,0x3e,0xff,
0x41,0xaa,0xb0,0xb1,0x9f,0xb2,0x6a,0xb5,0xbd,0xb4,0x9a,0x8a,0xba,0xca,0xaf,0xbc,
0x90,0x8f,0xea,0xfa,0xbe,0xa0,0xb6,0xb3,0x9d,0xda,0x9b,0x8b,0xb7,0xb8,0xb9,0xab,
0x64,0x65,0x62,0x66,0x63,0x67,0x9e,0x68,0x74,0x71,0x72,0x73,0x78,0x75,0x76,0x77,
0xac,0x69,0xed,0xee,0xeb,0xef,0xec,0xbf,0x80,0xfd,0xfe,0xfb,0xfc,0xad,0xae,0x59,
0x44,0x45,0x42,0x46,0x43,0x47,0x9c,0x48,0x54,0x51,0x52,0x53,0x58,0x55,0x56,0x57,
0x8c,0x49,0xcd,0xce,0xcb,0xcf,0xcc,0xe1,0x70,0xdd,0xde,0xdb,0xdc,0x8d,0x8e,0xdf
};
#ifdef __cplusplus
}
#endif

#endif /* SQL_H_SQLE819A */

```

---

## The crtnick.sqc C program

Use the crtnick.sqc (Example B-3) program to automate the migration of the database table definitions and the data load from DB2 on z/VSE to DB2 on Linux.

When you run this program against a DB2 VSE database, the following output files are created:

- ▶ The crtNickname.ddl file  
You can use this file to define the nicknames in the federated database VSELNXDB on Linux.
- ▶ The crtLoad.ddl file  
You can use this file to migrate the data from DB2 on z/VSE to DB2 on Linux.

*Example: B-3 Language C program crtnick.sqc*

```

/*****
* This program generates 2 files to create nicknames and load statements
* for relocating tables and data from DB2 VSE to Db2 Linux on System z
*****/
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <sqlenv.h>
#ifdef DB268K
/* Need to include ASLM for 68K applications */

```

```

#include <LibraryManager.h>
#endif
EXEC SQL INCLUDE SQLCA; /* :rk.1:erk. */
/*****
* BAD RETURN CODE - PUT OUT SQLCA STRUCTURE *
*****/
void putsqlca(struct sqlca sqlca)
{
char string[32]; /* work field */
printf("SQLCODE : %d\n", sqlca.sqlcode);
printf("SQLERRD1: %d\n", sqlca.sqlerrd[0]);
printf("SQLERRD2: %d\n", sqlca.sqlerrd[1]);
printf("SQLERRD3: %d\n", sqlca.sqlerrd[2]);
printf("SQLERRD4: %d\n", sqlca.sqlerrd[3]);
printf("SQLERRD5: %d\n", sqlca.sqlerrd[4]);
printf("SQLERRD6: %d\n", sqlca.sqlerrd[5]);
sprintf(string,"SQLERRM : %%-u.%-us\n",
sqlca.sqlerrml, sqlca.sqlerrml);
printf(string, sqlca.sqlerrmc);
printf("SQLERRP : %8.8s\n", sqlca.sqlerrp);
printf("SQLWARN0: %c\n", SQLWARN0);
printf("SQLWARN1: %c\n", SQLWARN1);
printf("SQLWARN2: %c\n", SQLWARN2);
printf("SQLWARN3: %c\n", SQLWARN3);
printf("SQLWARN4: %c\n", SQLWARN4);
printf("SQLWARN5: %c\n", SQLWARN5);
printf("SQLWARN6: %c\n", SQLWARN6);
printf("SQLWARN7: %c\n", SQLWARN7);
printf("SQLWARN8: %c\n", SQLWARN8);
printf("SQLWARN9: %c\n", SQLWARN9);
printf("SQLWARNA: %c\n", SQLWARNA);
printf("SQLSTATE: %s\n", sqlca.sqlstate);
return;
}
int main(int argc, char *argv[]) {
EXEC SQL BEGIN DECLARE SECTION; /* :rk.2:erk. */
char userid[9];
char passwd[19];
char db[19];
char schema[9];
char table[19];
EXEC SQL END DECLARE SECTION;
char cl[255];
FILE *crtNickname; /* Output file name that contains CREATE NICKNAME commands */
FILE *crtLoad; /* Output file name that contains DECLARE CURSOR and LOAD commands */
#ifdef DB268K
/* Before making any API calls for 68K environment, need to initialize the Library Manager */
InitLibraryManager(0,kCurrentZone,kNormalMemory);
atexit(CleanupLibraryManager);
#endif
if (argc == 4) {
strcpy (db, argv[1]);
strcpy (userid, argv[2]);
strcpy (passwd, argv[3]);
EXEC SQL CONNECT TO :db USER :userid USING :passwd; /* :rk.3:erk. */

```

```

if (sqlca.sqlcode != 0)
{
printf("CONNECT TO database\n");
putsqlca(sqlca);
}
}
else {
printf ("\nUSAGE: crtnick db userid passwd\n\n");
return 1; } /* endif */
/* Modify the select statement to restrict the number or type of tables returned */
/* Get all the tables that belong to the DBSPACE named 'SDBS' */
strcpy (c1, "SELECT STRIP(CREATOR), STRIP(TNAME) FROM SYSTEM.SYSCATALOG WHERE DBSPACENAME =
'SDBS' ORDER BY 1, 2");
EXEC SQL PREPARE S1 FROM :c1;
if (sqlca.sqlcode != 0)
{
printf("Prepare first SELECT cursor\n");
putsqlca(sqlca);
}
EXEC SQL DECLARE company_a CURSOR FOR S1;
if (sqlca.sqlcode != 0)
{
printf("Declare company_a cursor\n");
putsqlca(sqlca);
}
EXEC SQL OPEN company_a;
if (sqlca.sqlcode != 0)
{
printf("Open company_a cursor\n");
putsqlca(sqlca);
}
EXEC SQL FETCH company_a INTO :schema, :table;
if (sqlca.sqlcode < 0 ) /* This allows sqlcode 0 and 100 to be ok */
{
printf("Fetch from company_a\n");
putsqlca(sqlca);
}
/* Open for write. If the file exists, the current contents will be erased */
if( (crtNickname = fopen( "crtNickname.ddl", "w" )) == NULL )
printf( "The file 'crtNickname.ddl' was not opened\n" );
else
printf( "The file 'crtNickname.ddl' was opened\n" );
if( (crtLoad = fopen( "crtLoad.ddl", "w" )) == NULL )
printf( "The file 'crtLoad.ddl' was not opened\n" );
else
printf( "The file 'crtLoad.ddl' was opened\n" );
fprintf(crtNickname, "-- DB2 command file to create nicknames\n");
fprintf(crtNickname, "-- Use the command 'DB2 -tvfcrtNickname.ddl' to execute this command
file.\n\n");
fprintf(crtNickname, "-- Modify the following connect statement before executing this command
file.\n");
fprintf(crtNickname, "CONNECT TO <UDB Database Name> USER <userid> ;\n\n");
fprintf(crtLoad, "-- DB2 command file to declare cursor and perform load\n");
fprintf(crtLoad, "-- Use the command 'DB2 -tvfcrtLoad.ddl' to execute this command file.\n\n");

```

```

fprintf(crtLoad, "-- Modify the following connect statement before executing this command
file.\n");
fprintf(crtLoad, "CONNECT TO <UDB Database Name> USER <userid> ;\n\n");
while (sqlca.sqlcode != 100)
{
fprintf(crtNickname, "-- Create nickname for table %s.%s from database %s.\n", schema, table,
db);
fprintf(crtNickname, "CREATE NICKNAME %s FOR %s.%s.%s ;\n\n", table, db, schema, table);
fprintf(crtLoad, "-- Declare cursor and perform load for table %s.%s from database %s.\n",
schema, table, db);
fprintf(crtLoad, "DECLARE MYCURS CURSOR FOR SELECT * FROM %s ;\n", table);
fprintf(crtLoad, "LOAD FROM MYCURS OF CURSOR MESSAGES crossload.msg REPLACE INTO %s.%s ;\n\n",
schema, table);
EXEC SQL FETCH company_a INTO :schema, :table;
if (sqlca.sqlcode < 0)
{
printf("Fetch from company_a\n");
putsqlca(sqlca);
}
}
fprintf(crtNickname, "CONNECT RESET ;\n");
fprintf(crtLoad, "CONNECT RESET ;\n");
/* Close stream */
if( fclose( crtNickname ) )
printf( "The file 'crtNickname.ddl' was not closed\n" );
/* Close stream */
if( fclose( crtLoad ) )
printf( "The file 'crtLoad.ddl' was not closed\n" );
EXEC SQL CLOSE company_a;
if (sqlca.sqlcode != 0)
{
printf("Close company_a cursor\n");
putsqlca(sqlca);
}
EXEC SQL CONNECT RESET;
if (sqlca.sqlcode != 0)
{
printf("CONNECT RESET\n");
putsqlca(sqlca);
}
return 0;
}
/* end of program : crtnick.SQC */

```

---



# Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

## IBM Redbooks

For information about ordering these publications, see “How to get Redbooks” on page 189. Note that some of the documents referenced here may be available in softcopy only.

- ▶ *DB2 Performance Expert for Multiplatforms V2.2*, SG24-6470
- ▶ *HiperSockets Implementation Guide*, SG24-6816
- ▶ *IBM System z Connectivity Handbook*, SG24-5444
- ▶ *IBM System Storage DS8000: Architecture and Implementation*, SG24-6786
- ▶ *Linux on IBM System z: Performance Measurement and Tuning*, SG24-6926
- ▶ *Sharing and maintaining Linux under z/VM*, REDP-4322
- ▶ *z/VM and Linux on IBM System z: The Virtualization Cookbook for SLES9*, SG24-6695
- ▶ *z/VM and Linux on IBM System z*, SG24-7492

## Other publications

These publications are also relevant as further information sources:

- ▶ *DB2 Server for VM System Administration*, SC09-2980
- ▶ *DB2 Server for VSE System Administration*, SC09-2981
- ▶ *DB2 Server for VSE & VM Operation*, SC09-2986
- ▶ *DB2 Server for VSE & VM Performance Tuning Handbook*, GC09-2987
- ▶ *DB2 Server for VSE & VM SQL Reference*, SC09-2989
- ▶ *DB2 Server for VSE & VM Application Programming*, SC09-2889
- ▶ *DB2 Server for VSE & VM Database Administration*, SC09-2888
- ▶ *DB2 Server for VSE & VM Database Services Utility*, SC09-2983
- ▶ *Linux on zSeries zSeriesDevice Drivers, Features, and Commands*, SC33-8281
- ▶ *Program Directory for DB2 Server for VM*, G110-4998
- ▶ *Program Directory for DB2 Server for VM Client Edition*, G111-8336
- ▶ *Program Directory for DB2 Server for VSE and Guest Sharing for VSE*, G110-4999
- ▶ *Program Directory for DB2 Server for VSE and Guest Sharing for VSE, Client Edition*, G111-8337
- ▶ *Tivoli Monitoring for Databases: DB2 Agent User's Guide*, SC32-9449
- ▶ *z/VSE Installation*, SC33-8302

## Online resources

These Web sites are also relevant as further information sources:

- ▶ z/VSE home page  
<http://www.ibm.com/servers/eserver/zseries/zvse/>
- ▶ IBM developerWorks documentation that explains the ROLES
  - “New features in DB2 9.5 to help your business grow”  
<http://www.ibm.com/developerworks/db2/library/techarticle/dm-0706see/>
  - “Implement DB2 for Linux, UNIX, and Windows trusted contexts and roles in a Web application”  
<http://www.ibm.com/developerworks/db2/library/techarticle/dm-0803eakle/index.html>
- ▶ DB2 Server for VSE 750 Manuals  
[http://www.ibm.com/support/docview.wss?rs=66&context=SSEPF&dc=DA410&uid=swg27011615&loc=en\\_US&cs=utf-8&lang=en](http://www.ibm.com/support/docview.wss?rs=66&context=SSEPF&dc=DA410&uid=swg27011615&loc=en_US&cs=utf-8&lang=en)
- ▶ DB2 for Linux, UNIX, and Windows page  
<http://www.ibm.com/software/data/db2/9/>
- ▶ General Linux for System z tuning hints and tips  
<http://www.ibm.com/developerworks/linux/linux390/perf/index.html>
- ▶ SYSSTAT tool information  
[http://www.ibm.com/developerworks/linux/linux390/perf/tuning\\_how\\_tools.html#sysstat](http://www.ibm.com/developerworks/linux/linux390/perf/tuning_how_tools.html#sysstat)
- ▶ VMSTAT tool information  
[http://www.ibm.com/developerworks/linux/linux390/perf/tuning\\_how\\_tools.html#vmstat](http://www.ibm.com/developerworks/linux/linux390/perf/tuning_how_tools.html#vmstat)
- ▶ Top tool information  
[http://www.ibm.com/developerworks/linux/linux390/perf/tuning\\_how\\_tools.html#top\\_tool](http://www.ibm.com/developerworks/linux/linux390/perf/tuning_how_tools.html#top_tool)
- ▶ nmon performance: A free tool to analyze AIX and Linux performance  
[http://www.ibm.com/developerworks/aix/library/au-analyze\\_aix/](http://www.ibm.com/developerworks/aix/library/au-analyze_aix/)
- ▶ Tuning recommendations for DS8000 and Linux for System z  
[http://www.ibm.com/developerworks/linux/linux390/perf/tuning\\_rec\\_dasd\\_ds8000.html](http://www.ibm.com/developerworks/linux/linux390/perf/tuning_rec_dasd_ds8000.html)
- ▶ DB2 explain function to see the access plan for SQL queries  
<http://publib.boulder.ibm.com/infocenter/db2luw/v8/index.jsp?topic=/com.ibm.db2.udb.doc/admin/c0005135.htm>
- ▶ DB2 and DRDA trace (db2trc)  
<http://publib.boulder.ibm.com/infocenter/db2luw/v8/index.jsp?topic=/com.ibm.db2.udb.doc/core/r0002027.htm>
- ▶ DB2 snapshot monitor  
<http://publib.boulder.ibm.com/infocenter/db2luw/v8/index.jsp?topic=/com.ibm.db2.udb.doc/admin/c0006003.htm>

- ▶ IBM DB2 Database for Linux, UNIX, and Windows Information Center  
<http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp>
- ▶ Velocity Software performance tool for z/VM and Linux on System z  
<http://velocity-software.com/>
- ▶ Cognos Business Intelligence and Financial Performance Management  
<http://www-01.ibm.com/software/data/cognos/>
- ▶ Business intelligence and financial performance management solutions  
<http://www-01.ibm.com/software/data/cognos/solutions/>
- ▶ z/VM for VSE Guests  
<http://www.vm.ibm.com/vmvse/>
- ▶ z/VM resources for Linux on IBM System z  
<http://www.vm.ibm.com/linux/>
- ▶ z/VM Technical Resources  
<http://www.vm.ibm.com/techinfo/>

## How to get Redbooks

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications and Additional materials, as well as order hardcopy Redbooks, at this Web site:

[ibm.com/redbooks](http://ibm.com/redbooks)

## Help from IBM

IBM Support and downloads

[ibm.com/support](http://ibm.com/support)

IBM Global Services

[ibm.com/services](http://ibm.com/services)

Archived

# Index

## Symbols

./db2setup command 42

## Numerics

-805 error message 94

## A

A-B-C method 126  
access package 112  
accounting facility 162  
anticipatory scheduler (as scheduler) 25  
application considerations 114  
application dependencies for DB2 105  
application monitoring for z/VSE 163  
    authentication considerations 164  
application requester 17, 40, 100, 104, 161  
    TCP/IP 98, 102  
application server for TCP/IP 96, 100  
architecture 4  
ARISIVRR.Z parameter file 175  
as scheduler 25  
ASCII 37, 54  
ASCII-EBCDIC translation table 55  
ASG-TMON 156  
Assembler program 18  
asynchronous I/O 25  
atime mode 23  
authentication considerations for z/VSE application monitoring 164

## B

background partition 80  
Basic Security Manager (BSM) 77–78  
batch application monitoring 164  
batch binder for binding batch applications 113  
batch binding for package creation 112  
batch test 90, 93–94  
BIND file 81  
bind file 112  
    support 111  
bind record 112  
binding process 112  
    batch applications using batch binder 113  
    online applications using CBND 113  
BLOCK SIZE 148  
block size 98  
blocks 98, 148  
BLOWFISH algorithm 164  
BSM (Basic Security Manager) 77–78  
buffer count 74  
buffer pool 124, 162  
    sizing 14

byte size 99

## C

CA Explore Performance Management 156  
caching mode 28  
    tunedasd command 28  
CBND transaction 112  
    binding online applications 113  
    package creation 112  
central processor (CP) 12  
cfq scheduler 25  
checksumming 73  
CICS  
    application monitoring 165  
    file definitions 173  
    programs and transactions 81  
    system preparation for DB2 Server for VSE Client Edition 76  
CICS Monitoring Facility 161  
CICS System Definition (CSD) 81  
CICS Transaction Server 18  
CIRD transaction 161  
CKD rank 15  
COBOL/VSE 18  
coexistence environment 120  
collating sequence 38  
communication directory 100  
completely fair queuing scheduler (cfq scheduler) 25  
connection  
    logical 22, 163  
    monitoring 163  
    physical 21, 163  
    service name 52  
    timeout 116  
connection pool 21  
connection pooling 96, 115  
    in online resource adapter 95  
connection port 52  
CONNPOOL 96  
container striping 28  
count key data (CKD) rank 15  
COUNTER operator command 161  
COUNTER POOL operator command 162  
CP (central processor) 12  
CP INDICATE USER command 161  
CP Monitor subsystem 161  
CPU 12, 148  
    considerations for z/VSE 157  
    planning 18  
CPUMON 156  
Cross Partition Communication (XPCC) protocol 21  
CRR (connect/request/response) 31  
crtnick.sqc program 111, 183  
CSD (CICS System Definition) 81

CSI assembler interface 102

## D

DASD 14

dasd statistics utility 129

data

analysis 126

collection 126

management, modern approach 1

placement 15

data migration 106

database access packages 111

DB2 105

federated 106

from DB2 Server for VSE & VM 107

prerequisites 107

production 120

restrictions 107

database

consideration 37

ASCII and EBCDIC 37

database type 37

migration 6

object permission 119

tuning references to online documentation 162

versus VSAM 167

z/VM 17

z/VSE 17

database access package migration 111

database management systems (DBMS) 37

dataset labels 81

db\_udcs.c program 180

DB2

application dependencies 105

application monitoring and tuning 166

application requester 17

ARISIVRR.Z parameter file 175

best practices for designing high-performance applications 166

block size considerations in z/VSE and z/VM 98

container striping 28

data migration 105

DBMS in Linux 37

enterprise server setup for Linux on System z 42

I/O dependency 21

monitoring tools 135, 142

planning 11

preparing for DB2 Server for VSE Client Edition 78

VSE compatibility with DB2 for Linux, UNIX, and Windows 117

z/VSE and z/VM 17

z/VSE environment considerations 18

DB2 Agent 143

DB2 BLOCK SIZE considerations 148

DB2 Connect 51

DB2 CONNECT command 163

DB2 Control Center for VSE and VM DBA functions 162

DB2 DBMS 37

DB2 Design Advisor 135

workload performance tuning 136

DB2 Enterprise Server Edition features 2

DB2 for Linux 111

DRDA activity tracing 142

DB2 for Linux on System z

architecture 4

business value 2

configuration as a server 51

configuration for access to remote databases 52

data migration from DB2 Server for VSE & VM 106

DB2 Connect 51

installation 42

installation verification 50

monitoring and tuning 135

physical integration 6

remote database access 42, 52

z/VSE benefits 3

DB2 for Linux, UNIX, and Windows

compatibility with DB2 VSE 117

SQL dialect 8

DB2 Performance Expert 142

DB2 Portfolio 1

DB2 Server DSS Directory Service Server

COUNTER POOL operator command 162

SHOW TARGETWS operator command 162

DB2 Server for VM

PROTOCOL parameter 104

SQLBIND exec 112

TCP 101

TCP/IP for application requester 102

TCP/IP for application server 100

DB2 Server for VM Client Edition

application requester 17

communication directory 100

remote client 42

virtual machines 99

z/VM 17

z/VM application data access 17

DB2 Server for VSE 96

DRDA 98

implicit connection 95

link-editing client batch 86

link-editing online support components 86

online support components 86

TCP/IP and DRDA 96

TCP/IP for application requester 98

TCP/IP for application server 96

DB2 Server for VSE & VM

accounting facility 162

application requester 17, 98

COUNTER operator command 161

COUNTER POOL operator command 162

data migration to DB2 for Linux on System z 106

initialization parameters 161

interface considerations 102

monitoring and tuning 161

tools 161

rehosting 4

SHOW operator commands 161

SHOW TARGETWS operator command 162

SQL characteristics 117

- SQL dialect 8
- system catalog 161
- tools for monitoring database workload behavior 161
- trace facility 162
- DB2 Server for VSE Client Edition
  - application requester 17
  - ARISIVRR.Z parameter file 175
  - batch test 90
  - CICS system preparation 76
  - connection pooling 95, 115
  - DB2 application requester 17
  - DB2 preparation 78
  - distribution library restoration 76
  - implicit connection 95
  - installation 75, 86
  - installation on z/VSE 18
  - installation verification 90
  - licensing 76
  - memory sizing 20
  - online test 93
  - remote client 42
  - TCP/IP customization 76
  - z/VSE 17
  - z/VSE access 17
- DB2 Setup Launchpad 43
- DB2 Workload Manager (WLM) 136
  - activity information for later analysis 138
- db2advls command 135
- DB2COMM variable 52
- db2drdat command 142
- db2expln command 141
- DBCS (double-byte character set) 18
- DBNAME directory 79
- deadline scheduler 24
- default setup 86
- DEFINE TRACE command 144
- Devices button 154
- direct I/O 25
  - SCSI disks 26
- disk cache 15
- disk considerations for IBM System Storage 14
- disk monitoring, iostat utility 128
- disk statistics monitoring, dasd statistics utility 129
- Distributed Relational Database Architecture (DRDA)
  - activity tracing with DB2 for Linux 142
  - application requester 18
  - communication 40
  - connectivity between z/VSE and Linux on System z 9
  - enabling for DB2 Server for VSE 98
  - interfaces in z/VSE 102
  - protocol 21
- double striping 16
- double-byte character set (DBCS) 18
- DRDA (Distributed Relational Database Architecture)
  - activity tracing with DB2 for Linux 142
  - application requester 18
  - communication 40
  - connectivity between z/VSE and Linux on System z 9
  - enabling for DB2 Server for VSE 98
  - interfaces in z/VSE 102

- protocol 21
- DS6000, connection to a SAN and System z 14
- DS8000 16
  - cache 15
  - connection to a SAN and System z 14
  - PAV 14
  - storage pool striping 27
- Dynamic classes button 153

## E

- EBCDIC (Extended Binary Coded Decimal Interchange Code) 37
- ECKD (Extended Count Key Data) 14
- ECKD DASD, System z 26
- environment setup 39
  - considerations 40
  - operating systems 40
  - relational databases 40
- Ethernet 31, 148
  - network interface 146
- exact blocking-every 98
- Extended Binary Coded Decimal Interchange Code (EBCDIC) 37
- Extended Count Key Data (ECKD) 14
- extended dynamic SQL 119
- extent 28
- extent pool 15
  - balancing with ranks 16
- EZASMI assembler interface 102

## F

- fast communication manager (FCM) 52
- FB rank 15
- FCM (fast communication manager) 52
- FCP (Fibre Channel Protocol) 14
  - SCSI disk statistics 129
- FCP-attached SCSI disk 24
- federated access 57, 109
- federated data migration 106
- federated database 5, 108, 111
  - data sources 5
  - environment setup 54
  - setup 55
- federated migration process 96
- fetch and insert blocking 117
- Fibre Channel Protocol (FCP) 14
  - SCSI disk statistics 129
- Fibre Connection (FICON) 14–15
- FICON (Fibre Connection) 14–15
- FICON Express4 card in System z 26
- FICON-attached ECKD DASD 23
- file system I/O 23
- fixed block (FB) rank 15
- flexible blocking-each 98
- FORTTRAN 18

## G

- GETVIS command 155

GETVIS Fn,DETAILS command 159  
GETVIS SVA,DETAIL command 159  
guest memory 14

## H

hardware configuration definition (HCD) 14  
HATS (Host Access Transformation Services) 2  
HCD (hardware configuration definition) 14  
HiperSockets 30, 40, 62, 72, 148  
    configuration for z/VSE 66  
    interface 146  
    monitoring and tuning considerations 148  
    operating system support 63  
    overview 63  
Host Access Transformation Services (HATS) 2  
HyperPAV 15  
hypervisor 12

## I

I/O  
    asynchronous 25  
    direct 25  
    file system 23  
    processing for disk 23  
    schedulers in Linux 24  
    scheduling 24  
    z/VSE 159  
I/O configuration data set (IOCDS) 14, 72  
I/O dependency 21  
IBM System Storage  
    connectivity considerations 14  
    data placement 15  
    disk considerations 14  
IBM System Storage N series Gateways 16  
IFL (Integrated Facility for Linux) 12  
implicit connect setup 115  
implicit connection 95  
InfoSphere Federation Server 5  
    database build 6  
    logical data integration 5  
insert blocking 117  
installation, DB2 Server for VSE Client Edition 75  
Integrated Facility for Linux (IFL) 12  
IOCDS (I/O configuration data set) 14, 72  
iostat utility 128  
IPINIT00.L TCP/IP configuration member 170  
IPTraceTool 144

## J

job accounting exit 152  
Job Manager ARISIMGC 78  
journal mode 23  
journaling mode 23

## K

kernel 22, 28, 131  
    parameter  
        message queue limits 23

    semaphore limits 23  
    shared memory 22

kernel.msgmax 23  
kernel.msgmnb 23  
kernel.msgmni 23  
kernel.sem 23  
kernel.shmall 22  
kernel.shmmax 22  
kernel.shmmni 22

## L

LAN (local area network) 63  
Language Environment C interface 102  
latency 143  
LCU (logical control unit) 14–15  
licensing 76  
Linux 37  
    ASCII 54  
    considerations 22  
    database server 14  
    I/O processing for disk 23  
    I/O schedulers 24  
    journaling mode 23  
    kernel 22  
    LVM striping 16  
    monitoring and tuning 127  
    monitoring tools 134  
    multipath daemon 24  
    network configuration 64  
    network specifics for distributions 73  
    process monitoring with the top utility 130  
    throughput 29  
    YaST Control Center 64  
Linux on System z  
    central data and access hub 2  
    DB2 enterprise server setup 42  
    federated DB2 database environment setup 54  
    network activity 145  
    network connection 62  
    system check 134  
    system definitions 41  
    listening port customization 97  
LLC (Logical Link Control) 63  
local area network (LAN) 63  
logical connection 22, 163  
logical control unit (LCU) 14  
    System z 15  
logical integration 4–5  
    InfoSphere Federation Server 5  
Logical Link Control (LLC) 63  
logical partition (LPAR) 12  
    considerations for z/VSE 160  
    shared processors 14  
    virtualization options 13  
logical volume 27  
Logical Volume Manager (LVM) 16  
logical volume striping 16, 27, 29  
LPAR (logical partition) 12  
    considerations for z/VSE 160  
    shared processors 14



virtualization options 13  
LVM (Logical Volume Manager) 16

## M

mainframe disk 14  
MAP command 155, 159  
MAP Fn command 159  
Maximum Transmission Unit (MTU) 31, 148  
memory  
    analysis 158  
    guest 14  
    sizing 20  
message queue limits 23  
migration, database 6  
minidisk caching 14  
MINIMON 156  
modern data management 1  
monitoring 123–124, 128, 161  
    additional tools for DB2 142  
    batch applications 164  
    best practices 124  
    CICS applications 165  
    connections 163  
    dasd statistics utility 129  
    DB2 applications 166  
    DB2 BLOCK SIZE 148  
    DB2 for Linux on System z 135  
    DB2 Server for VSE & VM 161  
    DB2 tools 135  
    HiperSockets considerations 148  
    iostat utility 128  
    Linux process with the top utility 130  
    Linux system 127  
    logical connections 163  
    network 143  
    network activity in Linux on System z 145  
    network trace in z/VSE 144  
    oprofile utility 130  
    physical connections 163  
    stand-alone tools and products 156  
    sysstat utilities 128  
    tools for Linux and z/VM 134  
    z/VSE 148  
MTU (Maximum Transmission Unit) 31, 148  
multipath daemon 24  
multirow buffered insert 117

## N

National Language Package 42  
netstat command 145  
network  
    activity, Linux on System z 145  
    configuration, OSA 70  
    connection setup between z/VSE and Linux on System z 62  
    connection types, throughput measurements 31  
    monitoring 143  
    planning, throughput measurements 31  
    topology 29, 40

    trace in z/VSE 144  
    tuning 143  
network configuration  
    Linux 64  
    z/VSE 63  
network sniffer 143, 163  
nonparallel share (NPS) 19, 158  
noop scheduler 25  
NP\_TIME 19  
NPS (nonparallel share) 19, 158

## O

online test 93  
opreport -l command 132  
oprofile utility 130  
    Red Hat 131  
ordered mode 23  
OSA 72  
    network configuration 70  
OSA-Express QDIO interface 63  
outbound database connections 21

## P

package loading 114  
Parallel Access Volume (PAV) 14–15  
partition balancing 20  
partition size 19  
passthru mode 61  
password 86  
PAV (Parallel Access Volume) 14–15  
PDISPLAY DYNC command 159  
Performance Toolkit for VM 134  
performance tuning, evaluation 166  
physical connection 21, 163  
physical integration 4–5  
    DB2 for Linux on System z 6  
PL/I program 18  
planning  
    CPU 18  
    DB2 11  
    network 29  
priority planning 19  
priority queuing 74  
processor memory 15  
production, coexistence environment 120  
PROTOCOL parameter considerations 103  
PRTY 20  
punch out 78  
pureXML 2

## Q

query block 98  
QUERY CONNECTIONS command 144  
QUERY TD command 19, 150, 158  
QUERY TIME command 161

## R

rank, balancing with extent pools 16

- Real Time Monitor VM/ESA (RTM VM/ESA) 161
- rebinding 114
- Red Hat
  - buffer count 74
  - checksumming 73
  - oprofile utility 131
  - priority queuing 74
- Redbooks Web site 189
  - Contact us xi
- rehosting 6
  - considerations 7
  - database, UNICODE 54
  - process overview 7
  - reasons for 4
  - required documentation 10
  - required skills 9
- remote database access 52
- rmfpms tool 134
- RR (request/response) 31

## S

- SAN (storage area network) 14
- SAN Volume Controller (SVC) 16
- SCSI disk 14
  - direct I/O 26
  - statistics for FCP 129
- security definitions 77
- See-TCP/IP 144
- semaphore limit 23
- service-oriented architecture (SOA) 2
- setsockopt command 75
- shared memory, kernel parameters 22
- shared OSA 62
  - Express card 30
- SHOW operator commands 161
- SHOW TARGETWS operator command 162
- SIR command 159
- SIR MON command 150
- SIR SMF command 150, 159
- SMF (Subsystem Measurement Facility) 150
- snapshot monitor 141
- SOA (service-oriented architecture) 2
- SQL
  - characteristics in DB2 Server for VSE & VM 117
  - extended dynamic 119
  - query tuning 165
- SQL and XQuery Explain facility 140
- sql819a.h header file 182
- SQLGLOB file 81
- Static partitions button 153
- storage area network (SAN) 14
- storage pool striping 16, 27
- storage server caching mode 28
- striped volume 15
- striping 16
  - container 27–28
  - logical volume 27, 29
  - storage pool 15, 27
- Subsystem Measurement Facility (SMF) 150
- SUSE Linux Enterprise Server 10

- buffer count 74
- checksumming 73
- priority queuing 74
- SUSE Linux Enterprise Server 11
  - buffer count 74
  - checksumming 73
  - priority queuing 74
- SVC (SAN Volume Controller) 16
- swap disk 22
- SYSA fastpath 361 and 362 149
- sysctl
  - command 74
  - service 22
- sysstat utilities 128
- system activity dialogs in z/VSE 149
- system check, Linux on System z 134
- System z
  - configuration planning 11
  - ECKD DASD 26
  - FICON Express4 card 26
  - hardware options 12
  - LCU 15
  - LPAR 12
  - striped volume 15
  - supported processor types 12
  - z/VM hypervisor 12
- System z Application Assist Processor (zAAP) 12
- System z Integrated Information Processor (zIIP) 12

## T

- TCP/IP
  - application requester 98, 102
  - application server 96, 100
  - customization for VSE 76
  - DRDA communication 40
- TCP/IP for VSE
  - DEFINE TRACE command 144
  - QUERY CONNECTIONS command 144
- tcpdump command 145
- test
  - batch 90, 93–94
  - online 93
- throughput 143
  - measurements 31
- Tivoli Enterprise Portal 134
- Tivoli Monitoring for Databases, DB2 Agent 143
- Tivoli OMEGAMON XE on z/VM and Linux 134
- top command 130
- top utility 130
- TOTAL\_TIME 19
- trace facility 162
- tunedasd command 28
- tuning 123–124, 127–128, 161
  - best practices 124
  - considerations for migrated VSAM applications 167
  - data analysis 126
  - data collection 126
  - DB2 applications 166
  - DB2 BLOCK SIZE 148
  - DB2 for Linux on System z 135

- DB2 Server for VSE & VM 161
- evaluation of efforts 166
- HiperSockets 148
- Linux system 127
- network 143
- network activity in Linux on System z 145
- network trace in z/VSE 144
- performance activities 126
- recommendations for z/VSE 163
- running the test case 127
- system 127
- z/VSE 148
- Turbo Dispatcher button 154

## U

- user ID default setup 86
- user mapping 109

## V

- virtual processors 14
- virtualization
  - layer 13
  - options 13
- virtualized environment 13
- VM/Performance Reporting Facility (VM/PRF) 161
- VM/PRF (Performance Reporting Facility) 161
- volume group 15
- VSAM
  - tuning considerations for migrated applications 167
  - versus a database 167
- VSE Health Checker tool 154
- VSE Navigator tool 152
- VSE/ICCF 18
- VSE/POWER 18
- VSWITCH 30, 62

## W

- WebSphere Application Server 2
- WLM (workload manager) 136
  - activity information for later analysis 138
- working set 158
- workload manager (WLM) 136
- workload performance tuning 136
- writeback mode 23

## X

- XPCC (Cross Partition Communication) protocol 21
- XQuery Explain facility 140

## Y

- YaST Control Center 64
- YaST2 64

## Z

- z/IPMon 145
- z/VM

- consideration for z/VSE 160
- database 17
- DB2 17
- DB2 block size considerations 98
- DB2 Server for VM Client Edition 17
- hypervisor 12
- monitoring tools 134
- system definitions 40
- virtualization options 13
- virtualized environment 13
- z/VM Performance Toolkit 156
- z/VSE
  - application monitoring 163
    - authentication 164
  - benefits of using DB2 for Linux on System z 3
  - CPU considerations 157
  - CPU planning 18
  - database 17
  - DB2 17
  - DB2 block size considerations 98
  - DB2 Server for VSE Client Edition 17
  - distribution library DASD requirements 76
  - DRDA interfaces 102
  - EBCDIC 54
  - GETVIS command 155
  - HiperSockets configuration 66
  - I/O considerations 159
  - IPINIT00.L TCP/IP configuration member 170
  - job accounting exit 152
  - MAP command 155
  - memory analysis 158
  - memory sizing 20
  - monitoring and tuning 148
  - network configuration 63
  - network connection 62
  - network trace 144
  - preparing for DB2 Server for VSE Client Edition 75
  - QUERY TD command 150
  - SIR MON command 150
  - SIR SMF command 150
  - stand-alone monitoring tools 156
  - system activity dialogs 149
  - system definitions 40
  - system monitoring facilities 149
  - tuning recommendations 163
  - z/VM and LPAR considerations 160
- zAAP (System z Application Assist Processor) 12
- zIIP (System z Integrated Information Processor) 12
- Z-type member 78

Archived









# z/VSE Using DB2 on Linux for System z



**See the advantages for z/VSE applications using DB2 on Linux**

**Plan a migration of DB2 data to Linux for System z**

**Use a sample scenario for your implementation**

Data is one of the most critical and valuable assets of a business. Critical strategic decisions can be made more quickly and effectively when they are based on complete, accurate, and timely operational data. From this point of view, it is important to have an enterprise data management architecture that supports a flexible global view of the business.

Many environments today are heterogeneous with a high quantity and diversity of data. In this IBM Redbooks publication, we help enterprise architects and IT managers with these environments make decisions for a centralized database or data warehouse. We recommend a centralized data management environment on Linux on System z. We include guidance for IBM z/VSE and Linux specialists to reorganize existing IBM DB2 VSE data and build a database environment with continuous operation in Linux on System z.

We begin this book by describing the possibilities and advantages of enterprise data management and different technical ways to realize it. Then we discuss planning, which is important for setting the foundation of the architecture that is implemented. We explain the hardware considerations for capacity and performance planning. For the z/VSE system and Linux on System z, we describe considerations for operation in a logical partition (LPAR) and in a virtualized environment with IBM z/VM. In addition, we discuss the disk behavior for different workloads, storage dependencies, network connections, and DB2 database considerations.

We also guide you in customizing the DB2 server for z/VSE, z/VM, and DB2 on Linux to allow existing z/VSE and z/VM applications to access the database on Linux on System z. We include the data migration, application considerations, dependencies, compatibility, monitoring, and tuning possibilities in such an environment.

**INTERNATIONAL  
TECHNICAL  
SUPPORT  
ORGANIZATION**

**BUILDING TECHNICAL  
INFORMATION BASED ON  
PRACTICAL EXPERIENCE**

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

**For more information:  
[ibm.com/redbooks](http://ibm.com/redbooks)**

SG24-7690-00

ISBN 0738434000