DB2® Information Management Software

IBM

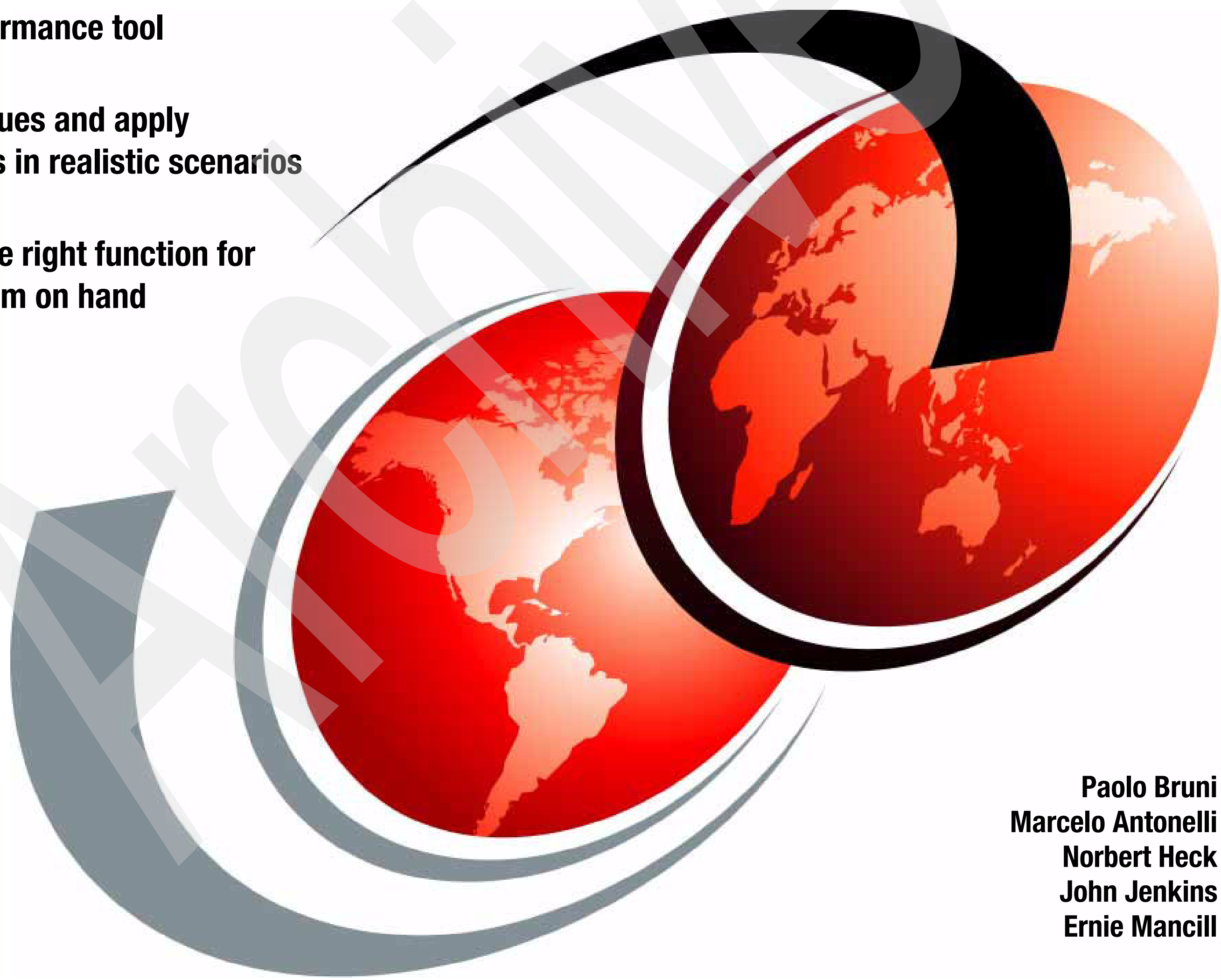
A Deep Blue View of DB2 Performance

IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS

Explore the combined functions of the new performance tool

Detect issues and apply thresholds in realistic scenarios

Choose the right function for the problem on hand



Paolo Bruni
Marcelo Antonelli
Norbert Heck
John Jenkins
Ernie Mancill

Redbooks

ibm.com/redbooks

International Technical Support Organization

A Deep Blue View of DB2 Performance IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS

April 2006

SG24-7224-00

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

First Edition (April 2006)

This edition applies to Version 3, Release 1 of IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS (program number 5655-P07), Version 2 Release 1 of IBM Performance Expert for z/OS (program number 5655-J49), and Version 3 of IBM Tivoli OMEGAMON XE for DB2 on z/OS (program number 5608-A67) for use with IBM DB2 UDB for z/OS Version 8 (program number 5625-DB2).

Contents

Figures

Tables

Examples

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 illustrates 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. 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 IBM's application programming interfaces.

Trademarks

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

- AIX®
- Candle Management Server®
- Candle Management Workstation®
- Candle®
- CandleNet Portal®
- CandleNet®
- CICS®
- CICSPlex®
- Common User Access®
- CUA®
- Database 2™
- DB2®
- DB2 Connect™
- DB2 Universal Database™
- DFSMShsm™
- DRDA®
- Enterprise Storage Server®
- eserver®
- eserver®
- e Trust® CA-Top Secret Security
- GDDM®
- GDPS®
- Geographically Dispersed Parallel Sysplex™
- IBM®
- IMS™
- MVS™
- MVS/ESA™
- OMEGAMON II®
- OMEGAMON®
- OS/390®
- Parallel Sysplex®
- PR/SM™
- QMF™
- RACF®
- Redbooks™
- Redbooks (logo) ™
- RMF™
- S/390®
- System/390®
- Tivoli®
- Tivoli Enterprise™
- VTAM®
- WebSphere®
- z/OS®
- zSeries®

The following terms are trademarks of other companies:

EJB, Java, Java Naming and Directory Interface, JDBC, JVM, J2EE, PDB, Sun, Sun Microsystems, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Visual Studio, 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

IBM® Tivoli® OMEGAMON® XE for DB2® Performance Expert on z/OS® represents the effort on converging OMEGAMON XE for DB2 and DB2 Performance Expert into one product that retains the best features of each. This new tool gives you a single, comprehensive tool to help assess the efficiency of and optimize performance from your DB2 Universal Database™ in the z/OS environment. It automates the analysis of your database performance in real time and also adds expert database analysis functions to help you maximize performance and enhance productivity.

The main functions of this tool allow you to:

- Monitor, analyze, and tune the performance of IBM DB2 Universal Database and DB2 applications on z/OS
- Improve productivity with meaningful views of performance
- Quickly and easily identify performance bottlenecks using predefined rules of thumb
- Enjoy substantial breadth and depth in monitoring DB2 environments by combining batch-reporting capabilities with real-time monitoring and historical tracking functions
- Support an enterprise-wide integrated systems management strategy activated by the IBM Tivoli OMEGAMON XE family
- Store performance data and analysis tools in a performance warehouse

The software combines the sophisticated reporting, monitoring, and buffer pool analysis features of the IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS and IBM DB2 Buffer Pool Analyzer products.

This IBM Redbook will help you install and understand the main functions of the product, clarify the differences, and point out the advantages if you had one of the pre-existing products already in use.

The redbook consists of five parts:

- Part 1, “Introduction” on page 1 contains three chapters in which we describe how IBM began work to incorporate the best functions of the Candle acquired OMEGAMON XE for DB2 with those of Performance Expert for z/OS Version 2. The result is IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS V3.1.0, a performance monitor for DB2 with the strong points of both, which integrates with the Tivoli Enterprise Portal and the Tivoli OMEGAMON family of z/OS monitoring products.
- Part 2, “Installation” on page 33 contains two chapters in which we discuss the steps necessary for a new installation of Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS in a new environment and introduce the concepts of internal and external security.
- Part 3, “What’s new for DB2 PE users” on page 85 contains six chapters in which we highlight the new functions that OMEGAMON XE for DB2 has made available for the current users of DB2 Performance Expert or Performance Monitor.
- Part 4, “What’s new for OMEGAMON users” on page 191 contains three chapters where we concentrate on the main new and enhanced functions offered by DB2 Performance Expert for a previous user of OMEGAMON for DB2. There are several other functions which can be of interest, and they are described in the still very relevant redbook *IBM DB2 Performance Expert for z/OS Version 2,* SG24-6867-01.
- Part 5, “User scenarios” on page 275 contains two chapters where we provide details on how use IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS in solving real life DB2 related issues.

The team that wrote this redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization, San Jose Center.

Paolo Bruni is a DB2 Information Management Project Leader at the ITSO, San Jose Center. He has authored several Redbooks™ about DB2 for z/OS and related tools, and has conducted workshops and seminars worldwide. During Paolo’s many years with IBM, in development and in the field, his work has been mostly related to database systems.

Marcelo Antonelli is a DB2 Specialist in Brazil. He has 20 years of experience working with DB2 for z/OS. Marcelo holds a graduate degree in System Analysis from PUCC in Campinas, S‹o Paulo. His areas of expertise include database design, system administration, and performance. Marcelo is currently supporting an outsourcing internal IBM account from EUA, as well as assisting IBM technical professionals about DB2.

Norbert Heck is a senior IT Specialist and Teamleader at IBM Germany. He holds a Ph.D. degree in Computer Science from the University of Kaiserslautern, Germany. For over 20 years with IBM, Norbert has been working as a software engineer in development, manufacturing and service organizations. In 2000, he joined the DB2 Performance Monitor development team in the Boeblingen Lab. His areas of expertise include database systems, distributed systems, object-oriented technologies, expert systems, and simulation and scheduling applications in the semiconductor and automotive industry.

John Jenkins is a senior IT Specialist with IBM in Germany. He has 18 years experience working for a variety of large companies in England and Germany as a DBA on DB2 for z/OS. Since 2003 he has been working for IBM in Frankfurt, Germany supporting an outsourcing account for a major german bank. John holds a B.Sc. degree in Electronics from the University of Southampton, England. His areas of expertise include DB2 Utilities and DB2 performance in very large sysplex environments.

Ernie Mancill is a certified IBM Data Management IT Specialist with IBM Software Group. Ernie has 29 years of experience in IT with 14 years of experience with DB2 as a Systems Programmer. He joined IBM seven years ago and is currently a member of the IBM SWG DB2 Database Tools technical sales team where he is responsible for pre and post sales technical support of the IBM DB2 Database Tools portfolio. His areas of expertise include the DB2 system, performance and database administration, as well as utilities and tools.

A photograph of the team is shown in Figure 1.

Figure 1 Left to right: John, Norbert, Paolo, Marcelo, and Ernie (photo courtesy of Whei-Jen Chen)

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

Rich Conway
Bob Haimowitz
Yvonne Lyon
Leslie Parham
Deanna Polm
Sangam Racherla
International Technical Support Organization

John Bowen
John Campbell
Steve Fafard
Paul Kenney
Gopal Krishnan
Ben Marino
Manfred Olschanowsky
Mary Petras
Dave Schwartz
Bob Shellgren
Akira Shibamiya
Maria Weinerth
Suk Wong
IBM Silicon Valley Lab

Denis Aberth
Norbert Jenninger
Holger Karn
Andrea Lutz
Michael Reichert
Konrad Rueckschloss
Ruediger Stumm

Matthias Tschaffler
Christopher Watson
Gabriele Widmer-Adams
IBM Boeblingen Lab

Mike Bracey
IBM UK

Bart Steegmans
IBM Belgium

Ernie Gilman
Joe Means
Carol So
Ed Woods
IBM Sales & Distribution

Ken Taylor
DEMOnet DB2 Support, IBM Fort Wayne

Robert Risley
IBM Tivoli Lab

Rama Naidoo
IBM Australia

Become a published author

Join us for a two- to six-week residency program! Help write an IBM Redbook dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll team with IBM technical professionals, Business Partners and/or customers.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you'll 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`

Comments welcome

Your comments are important to us!

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

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

  `ibm.com/redbooks`

- Send your comments in an e-mail to:

  `redbook@us.ibm.com`

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. QXXE Building 026
  5600 Cottle Road
  San Jose, California 95193

Part 1

Introduction

When IBM acquired Candle Corp. in June 2004, work began on incorporating the best of two product lines together:

- IBM Performance Expert for z/OS Version 2 Release 1 (program number 5655-J49)
- IBM Tivoli OMEGAMON XE for DB2 on z/OS Version 3 (program number 5608-A67)

The result is IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS V3.1.0 (program number 5655-P07), generally available since December 2005. A merged development team, made up of DB2 Performance Expert members in Germany and OMEGAMON XE for DB2 members in the U.S.A., has combined their expertise to create a performance monitor for DB2 with the strong points of both.

The converged product integrates with the Tivoli Enterprise Portal and the Tivoli OMEGAMON family of z/OS monitoring products, so you can include DB2 in your overall zSeries monitoring solution. The Tivoli OMEGAMON Dashboard Edition will even help you combine monitoring information from different Tivoli OMEGAMON products in a single display.

This part of the book contains the following chapters:

1

The new IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS

In this chapter we introduce some basic concepts and background information about IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS.

We discuss the following topics:

- DB2 performance management
- DB2 performance management products: Why needed?
- Performance management solutions from IBM
- Introducing IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS
- Which one? Performance Monitor or Performance Expert?

1.1 DB2 performance management

How you define good performance for your DB2 subsystem depends on your particular data processing needs and their priority. Performance objectives should be realistic, in line with your budget, understandable, and measurable. Common objectives could include:

- Values for acceptable response time (a duration within which some percentage of all applications have completed)
- Average throughput (the total number of transactions or queries that complete within a given time)
- System availability, including mean time to failure and the durations of down times

Objectives such as these define the workload for the system and determine the requirements for resources — processor speed, amount of storage, additional software, and so on. Often, though, available resources limit the maximum acceptable workload, which requires revising the objectives.

Presumably, your users have a say in your performance objectives. A mutual agreement on acceptable performance, between the data processing and user groups in an organization, is often formalized and called a service-level agreement. Service-level agreements can include expectations of query response time, the workload throughput per day, hour, or minute, and windows provided for batch jobs (including utilities). These agreements list criteria for determining whether or not the system is performing adequately.

Next, describe a set of preliminary workload profiles that might include a definition of the workload type in terms of its function and its volume. You are likely to have many workloads that perform the same general function (for example, order entry) and have an identifiable workload profile. Other workload types could be more ad hoc in nature, such as SPUFI or QMF™ queries.

Identify the resources required to do the work described for each set of defined workloads, including physical resources managed by the operating system (such as real storage, disk I/O, and terminal I/O) and logical resources managed by the subsystem.

For each workload type, convert the estimates of resource requirements into measurable objectives. Include statements about the throughput rates to be supported (including any peak periods) and the internal response time profiles to be achieved. Make assumptions about I/O rates, paging rates, and workloads.

Next, determine the frequency and level of detail gathered in your collection of performance data. Consider the following cost factors when planning for monitoring and tuning:

- Trace overhead
- Trace data reduction and reporting times
- Time spent on report analysis and tuning action

Inspect your performance data to determine whether performance has been satisfactory, to identify problems, and to evaluate the monitoring process. When establishing requirements and planning to monitor performance, also plan how to review the results of monitoring. Plan to review the performance data systematically. Review daily data weekly and weekly data monthly; review data more often if a report raises questions that require checking. Depending on your system, the weekly review might require about an hour, particularly after you have had some experience with the process and are able to locate quickly any items that require special attention. The monthly review might take half a day at first, less time later on. But when new applications are installed, workload volumes increased, or terminals added, allow more time for review.

The inspection and review process could use data collected by DB2 instrumentation facility traces and summarized and interpreted using the Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS Performance Reporter or the Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS Performance Warehouse.

Review the data on a gross level, looking for problem areas. Review details only if a problem arises or if you need to verify measurements.When reviewing performance data, try to identify the basic pattern in the workload, and then identify variations of the pattern. After a certain period, discard most of the data you have collected, but keep a representative sample. For example, save the report from the last week of a month for three months; at the end of the year, discard all but the last week of each quarter. Similarly, keep a representative selection of daily and monthly figures. Because of the potential volume of data, consider using Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS Performance Warehouse or Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS Performance Database to store data in a manageable form to use to track historical changes in performance. More information on this approach is available in Chapter 14, “Performance Database and Performance Warehouse” on page 247.

1.2 DB2 performance management products: Why needed?

Quality and reliability is taken seriously in most IT organizations. Many companies reflect this commitment to quality in their corporate mission statement. Service level agreements in many IT organizations act as its charter, and its ability to adhere to that agreement defines the quality of service that it provides. Poor performance of critical applications can have a negative financial impact on the organization. With IBM DB2 tools being used to perform real-time and historical performance reporting, subsequent analysis and tuning will help avoid problems and improve application efficiency, resulting in a reduction in performance degradation frequency.

Industry analysis has shown that up to 80% of IT budget expenditures are for staff and 20% are for software and hardware. Ten years ago, those numbers were reversed – 20% for staff and 80% for hardware and software. The demand for IT professionals far exceeds the supply and salaries continue to rise, but the representative IT enterprise is becoming more and more complex, requiring pockets of expertise to guarantee service levels.

Also, with companies operating remote systems at each office location, uniform support and complete/consistent application rollout are of paramount importance. In the competitive marketplace today, many organizations are faced with the challenge of having to expand their business to increase market share in order to continue to provide adequate shareholder equity. The demands of this environment place a high premium on IT organizations to be able to leverage the scalability of the zSeries® platform in order to satisfy the demands of their customers. In addition to growth related to new business/customers, the size of existing customers will also grow providing the IT organization with additional challenges.

1.3 Performance management solutions from IBM

Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS is a member of the family of the IBM Tivoli OMEGAMON integrated products, which provide an overall monitoring and tuning solution across the several components of the z platform. From the DB2 subsystem perspective, this is one in a series of complementary solutions that can provide a complete view of the DB2 performance story. Some other tools that could be considered when building a performance management tool kit are described in the following sections.

IBM DB2 Query Monitor

IBM DB2 Query Monitor gives you the ability to efficiently customize and tune your structured query language (SQL) workload and DB2 objects to ensure the effectiveness of your DB2 subsystems and improve overall performance. Users can view all the SQL statements that are open in DB2, including queries in nested cursors that help debug solutions. Key features include GUI-based reporting, viewing, and configuration capability that enables you to access consolidated data and events for DB2 subsystems. IBM DB2 Query Monitor enables the collection of query activity information at an increased levels of granularity, down to the level of the individual table and index, and related back to the exact request or to the user who is requiring the access. Enhanced integration capabilities let you launch Visual Explain, which shows the paths DB2 uses to run queries helping you select the most efficient path.

DB2 Query Monitor can also launch Tivoli OMEGAMON XE for DB2 Performance Expert in context and display threads in the active subsystem to save you navigation time. And you can invoke IBM DB2 SQL Performance Analyzer when you have discovered a problem with an SQL statement. Improvements in SQL processing let you store data in DB2, making it easier to do statistical analysis on data and create your own reports using tools such as QMF and QMF for Windows®.

IBM DB2 SQL Performance Analyzer

IBM DB2 SQL Performance Analyzer provides you with an extensive analysis of SQL queries without executing them. This analysis aids you in tuning your queries to achieve maximum performance. DB2 SQL Performance Analyzer can analyze new access paths, determine if action is needed, and estimate the costs of new paths in database resources consumed.

DB2 SQL Performance Analyzer makes it easier to reduce the escalating costs of database queries by estimating their cost prior to execution. DB2 SQL Performance Analyzer delivers an Easy Explain function that provides an alternate view of the Explain data. Comparison of old and new plans is supported, along with Retro-Explain for Access plans, helping you to find out how long queries will take and to prevent queries from running too long. It can also aid in the migration of catalog statistics to test machines for in-depth analysis of production applications.

IBM DB2 Path Checker

IBM DB2 Path Checker helps you increase the stability of your DB2 environments. It can help you discover and correct unwanted and unexpected access path changes before you are notified about them. These changes can occur as a result of binding when installing a new release of DB2, applying service, or migrating an application from one system to another.

DB2 Path Checker can efficiently scan hundreds or thousands of SQL statements and identify just the statement or statements that have experienced or will experience an access path change. It can seamlessly integrate into your change control procedures for automated protection. DB2 Path Checker can determine which DBRM will have issues if rebound, without endangering existing access paths. You can determine what will change if you migrate between subsystems of mixed releases. It can help find DBRMs for which you do not have matching Explains. Use DB2 Path Checker to display old and new SQL statements and compare them where the SQL is available from the catalog.

Visual Explain for DB2 for z/OS Version 8

Visual Explain for DB2 for z/OS Version 8 provides graphical depictions of the access plans that DB2 chooses for your SQL queries and statements. Such graphs eliminate the need to manually interpret plan table output. The relationships between database objects, such as tables and indexes, and operations, such as table space scans and sorts, are clearly illustrated in the graphs. You can use this information to tune your queries. You can also use Visual Explain to generate customized reports on explainable statements, to view subsystem parameters, and to view data from the plan table, the statement table, and the function table.

Visual Explain Version 8 has been completely reworked so that it also includes the following enhancements:

- More context-sensitive tuning suggestions are provided. You can link to these suggestions directly from the graph.
- You can also link from the graph to additional statistics and descriptions for each object or operation that is used in the access plan.
- Each graph can display multiple query blocks, so that you can view the entire access plan in one graph. In previous versions of Visual Explain, each graph displayed only one query block. In this version, you still have the option to view only one query block at a time.
- Visual Explain can be used to catalog and uncatalog databases on your local machine.
- Visual Explain will allow you to run a query and view the formatted results.
- Visual Explain can be launched from Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS with an SQL statement in context.

1.4 Introducing IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS

IBM Tivoli OMEGAMON XE for DB2 Performance Monitor/Expert on z/OS helps to proactively manage the DB2 mainframe environment and tune for optimal performance. These products bring the strength of both OMEGAMON for DB2 and DB2 Performance Monitor/Expert into a new set of DB2 monitors. The products achieve this robustness by tapping into DB2 version 8 features, such as 64-bit above-the-bar virtual addressing and long SQL statements, while advancing distributed thread tracking, workload monitoring, historical reporting, and more. The portal interface provides a single interface at the big picture and granular levels, including interaction between DB2 and other applications. This product helps to identify performance spikes and anomalies that might otherwise go unseen, take action in real time, and automate repetitive DB2 operations.

IBM Tivoli OMEGAMON XE for DB2 on z/OS has the following features:

- Monitors threads, I/O, buffer pools, SQL cache
- Views threads running stored procedures
- Monitors interaction of DB2 with subsystems such as CICS® and IMS™.
- Includes an application trace facility that tracks every step of a DB2 transaction
- Monitors DB2 Connect™. and distributed applications
- Provides in-depth analysis with the detailed reports and the Performance Warehouse with its analysis support
- Provides buffer pool analysis, object analysis, and simulation

Figure 1-1 shows the generalized product architecture for IBM Tivoli OMEGAMON for DB2 Performance Expert on z/OS V3.1.0.

For more information, please refer to Chapter 3, “Architecture” on page 17.

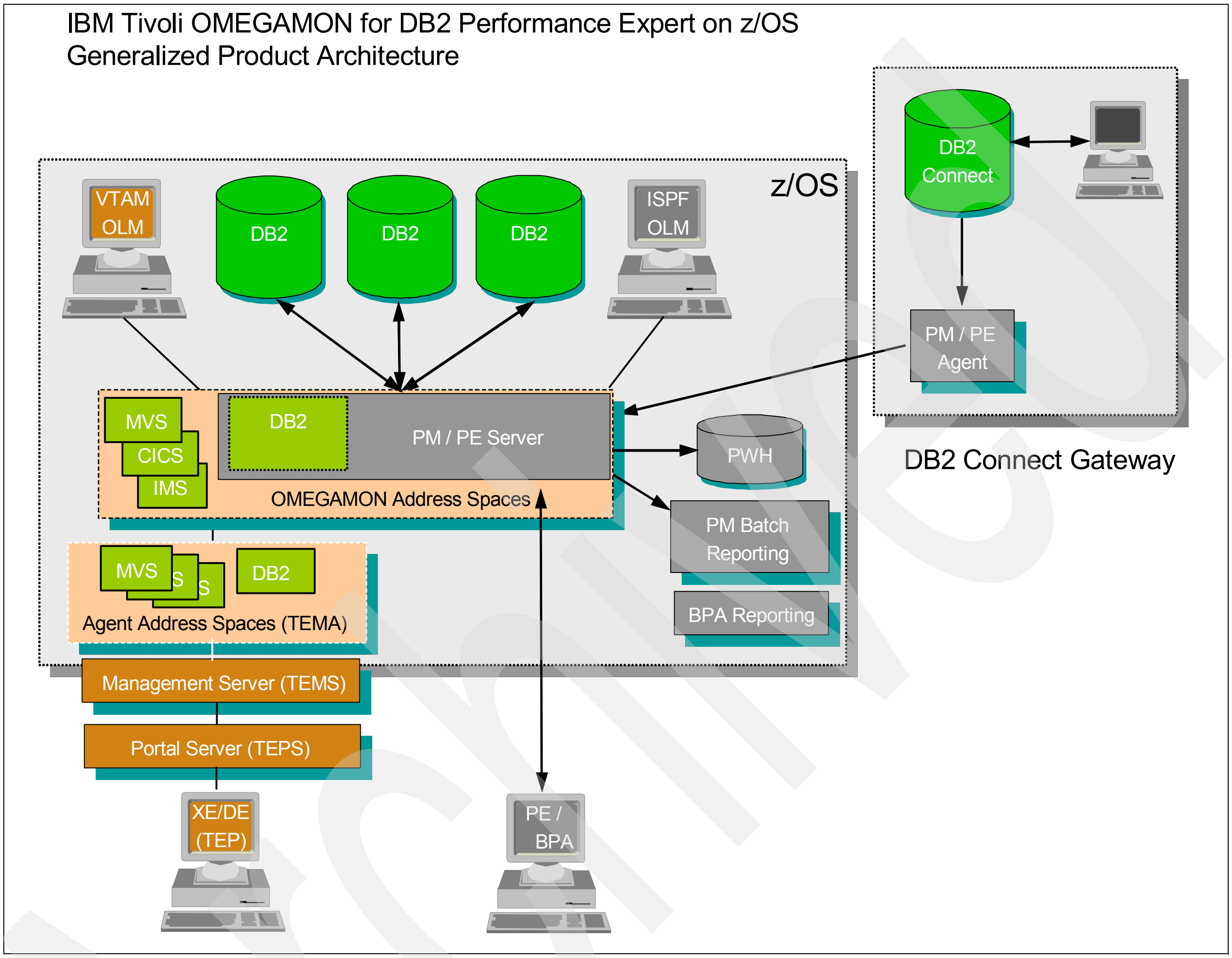


Figure 1-1 Product architecture

1.5 Which one? Performance Monitor or Performance Expert?

There are some features in Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS that are not available in Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS. With Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS, all current and future functionality related to expert analysis is shipped as part of the Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS Performance Warehouse component, as well as access to the DB2 Buffer Pool Analyzer functionality. With the stand-alone Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS, we can expect a limited set of expert analysis features and do not have access to DB2 Buffer Pool Analyzer. At the current time, IBM still continues to license the DB2 Buffer Pool Analyzer as a separate product.

In the future, functionality may be added to Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS that will not exist in the separate Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS and BPA products. All new functions relating to expert analysis will be implemented in Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS, but not necessarily in Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS. However, basic support for new DB2 versions and releases and functionality that relates to performance measurement will continue to be available in Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS until otherwise announced.

2

History of products

To understand where we are, we sometimes need to reflect back on where we have been. In this chapter we briefly review the ancestry of IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS, highlighting the evolution of both OMEGAMON for DB2 and DB2 Performance Monitor. This leads us into a discussion of the development road map and strategy used to build IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS.

We discuss the following topics:

- The OMEGAMON story
- DB2 Performance Monitor...here from the beginning
- IBM gets serious about DB2 tools
- DB2 Performance Expert
- IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS: Road map and convergence
- Converged product and entitlement

2.1 The OMEGAMON story

In this section we provide a brief overview of the OMEGAMON history.

The mid 1970s through the mid 1980s

In 1976, OMEGAMON for MVS™ was first introduced. The user interface was a simple 3270 command interface that consisted of major and minor mnemonic commands, which made its usage difficult to use for anyone but the very technical.

From the end of the 1970s till the mid 1980s, many of the capabilities that are still available today were developed. This period represents the shift in importance from data collection to data analysis:

- Exception analysis was created, allowing OMEGAMON to warn when anomalies existed
- Commands were introduced to enable the user to perform actions to resolve problems
- OMEGAMON for CICS, OMEGAMON for IMS and OMEGAMON for DB2 were introduced to provide subsystem and application monitoring
- The 3270 menu interface was released to simplify usage and enable broader audience appeal.

The late 1980s through the early 1990s

From the late 1980s through early 1990s, niche competitors started offering lower cost solutions that were easy to use. To combat this competition and to improve usability, Candle® Corporation purchased the CL technology to enable delivery of Common User Access® (CUA®) compliant 3270 interfaces.

The early 2000s

The early 2000s were turbulent times for Information Technology as the impact of Year 2000 spending was felt across all of IT:

- OMEGAMON Extended Edition (XE) and OMEGAMON Dashboard Edition (DE) were released.

IBM acquires Candle Corporation

On April 2004, IBM announced the intent to acquire Candle Corporation. On June 2004, Candle Corporation was acquired by IBM. Subsequent to IBM acquiring Candle Corporation, the IBM Tivoli division developed a road map to outline the directions for IBM Tivoli and (the former) Candle solutions that contained overlapping or similar capabilities:

- OMEGAMON XE for CICS, OMEGAMON XE for CICSPlex®, and OMEGAMON II® for CICS are combined into a single solution, IBM Tivoli OMEGAMON XE for CICS on z/OS.
- OMEGAMON XE for DB2, OMEGAMON XE for DB2Plex, and OMEGAMON II for DB2 are already combined into IBM Tivoli OMEGAMON XE for DB2 on z/OS. IBM Tivoli OMEGAMON XE for DB2 on z/OS, DB2 Performance Monitor, and DB2 Performance Expert are being merged into two solutions, IBM Tivoli OMEGAMON XE for DB2 PM and IBM Tivoli OMEGAMON XE for DB2 PE.
- OMEGAMON XE for IMS, OMEGAMON XE for IMSPlex, and OMEGAMON II for IMS are combined into a single solution, IBM Tivoli OMEGAMON XE for IMS on z/OS.
- OMEGAMON XE for Mainframe Networks, OMEGAMON II for Mainframe Networks, and IBM Tivoli Monitoring for Network Performance are combined into a single solution, IBM Tivoli OMEGAMON XE for Mainframe Networks. This entitlement will also include Netview Performance Monitor (NPM).

- OMEGAMON XE for Storage becomes IBM Tivoli OMEGAMON XE for Storage.
- OMEGAMON XE for Sysplex, OMEGAMON XE for OS/390®, and OMEGAMON II for MVS are combined into a single solution, IBM Tivoli OMEGAMON XE for z/OS. OMEGAMON XE for Unix System Services (USS) is currently a stand-alone solution. The current road map has OMEGAMON XE for USS being included in the IBM Tivoli OMEGAMON for z/OS solution in the future.
- Other OMEGAMON solutions providing capabilities associated with WebSphere® and Distributed platforms are being enhanced and merged as well. These solutions are out of scope for discussion in this redbook.

2.2 DB2 Performance Monitor...here from the beginning

DB2 Performance Monitor can trace its ancestry back to the early days of DB2. By its nature, the raw DB2 instrumentation data is very difficult to translate into a meaningful performance picture. The need for an easy to use, comprehensive performance product was quickly identified. That need was satisfied by the early versions of DB2 Performance Monitor: In its early versions, the product was strictly used as a batch oriented performance tool. So, while it provided a good view of performance data from a historical perspective, it did not provide any real-time "reactive" performance monitoring capability.

From an early stage, the DB2 product development team has placed a heavy reliance on DB2 PM accounting and statistics reporting to validate and measure incremental performance changes. This has resulted in a very close relationship between the performance measurement team at SVL and the DB2 PM development lab. One of the biggest benefits arising from this relationship is that we have seen a historical trend of very early exploitation of new releases of the DB2 for z/OS product family by DB2 PM.

Many customers have found that they could rely on the quality and accuracy of the data externalized by DB2 PM, since the DB2 Instrumentation Facility is the sole source of metrics. However, customers found that having to experience the delay of running traces, waiting for SMF to become available, and then relying on a series of batch reporting processes to begin problem determination, was sometimes unacceptable. Based on customer demand, starting with Version 3.1 of DB2 Performance Monitor, IBM developed additional functionality that provided for online monitoring capability. This functionality was enhanced by improvements in the DB2 instrumentation, which included the introduction of the monitor class traces.

Some types of performance data lend themselves well to being shown in a graphical manner. Initially, for a limited amount of information, we produced some rudimentary graphical displays using 3270 based technologies like GDDM®. As the personal computer became more prevalent in the workplace, and with the emergence of the Windows operating system, starting with DB2 Performance Monitor Version 4.1, the early workstation based client was made available.

While DB2 PM demonstrates that IBM provided an extremely rich reporting based performance solution, the prevalent view as to online performance monitoring was that there were several excellent solutions on the market place, and as such, the DB2 Performance Monitor development team focused less on this set of customer requirements.

2.3 IBM gets serious about DB2 tools

Starting in the late 1990's, IBM was becoming increasingly concerned about the overall cost of ownership on the S/390® platform. A number of customers were making platform decisions and moving workload away from the mainframe. After further investigation, it became apparent that one of the biggest factors in many of these decisions was related to the rising cost of different 3rd party software solutions.

Among the more popular and essential products were DB2 performance monitoring solutions. In order to help customers with this cost of ownership issue, IBM made a significant re-investment in a number of administration and performance tools for DB2 for z/OS. As a result, the expanded product portfolio now provides a number of customers with tools solutions that deliver exceptional functionality while at the same time allow for reductions in their z/OS one time charge software costs.

2.4 DB2 Performance Expert

At this point in the product evolution, customers were very comfortable with the features provided by the batch reporting component of DB2 Performance Monitor, but wanted enhanced functionality to be delivered to the online monitoring component. At the same time, a number of customers made the point that in presenting certain types of DB2 performance data, a graphical user interface made more sense. Also, a number of customers who were running DB2 on distributed environments wanted performance information similar to what has been provided for z/OS for a number of years. Finally, a requirement was put forward to make the creation and management of historical data in the DB2 Performance Monitor performance database easier.

In order to satisfy these requirements, the DB2 Performance Expert was developed. As described in 1.5, “Which one? Performance Monitor or Performance Expert?” on page 9, the basic distinction between DB2 Performance Monitor and DB2 Performance Expert is the addition of the IBM DB2 Bufferpool Analyzer. However, significant additional functionality was included in the first version of DB2 Performance Expert. Some of these enhancements included the performance warehouse, sample rule of thumb analysis, performance queries, system health display, lock owner and waiter analysis from a sysplex perspective, and many other incremental improvements. Of course, as mentioned before, these enhancements were also made available to the Performance Monitor customers, with the obvious exception of the Bufferpool Analysis feature.

Continuing the long line of *day one* exploitation of new releases of DB2, when DB2 UDB for z/OS V8 became generally available, IBM also announced V2 of the DB2 Performance Expert offering. Among the new functionality provided by V2 was complete exploitation of new features of DB2 V8, including a Unicode catalog, long name support, and exploitation of incremental improvement in the V8 instrumentation facility data. In addition, significant enhancements in both the batch and online component for IFCID 225 reporting. This is of immediate interest to assist the V7 customer in their migration to V8.

2.5 IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS: Road map and convergence

After the IBM acquisition of Candle corporation, the first public statement of direction regarding the eventual technology convergence of the OMEGAMON XE for DB2 and DB2 Performance Expert product came on December 7, 2004 in the USA announcement letter 204-297. In part, the announcement contained the statement that IBM plans to develop new products for performance and storage management for DB2 UDB for z/OS. The technology base for these products will merge some of the recently acquired technology from Candle Corporation with existing technology from the IBM DB2 tools. The performance management products will, over time, converge and expand the capabilities of OMEGAMON XE for DB2, Performance Expert and Performance Monitor for DB2.

The idea behind the convergence strategy was to take the best of both products, with a heavy reliance on customer feedback, and then build on that merging of technology. The overall objectives of the convergence was to achieve the following goals:

- Provide a cross zSeries and integrated monitoring solution
- Satisfy current customer requirements for both product lines
- Combine the best-of-breed functions from both products

It was hoped that out of this strategy, the overall customer satisfaction for those customers having one or both product offerings would be enriched.

In order to understand how to best combine the individual product functionality, several different elements were considered, including these:

- An analysis of product portfolio, strategy, and market penetration was done.
- A detailed analysis, function by function, was conducted by both development teams.
- Experiences were collected from both development groups about strong and weak functions.
- Internal and external users of both products were interviewed to gather user experiences.
- Interviews were conducted and early assumptions validated with customers and the DB2 Tools Customer Advisory Council.
- Known customer requirements from both product lines were reviewed.

As a result of this analysis, a convergence proposal was developed. More on the details of actual migration strategy can be found in Chapter 3, “Architecture” on page 17. Many dozens of customer requirements, both OMEGAMON and DB2 PM/PE have been satisfied by this approach.

2.6 Converged product and entitlement

As part of the new product architecture, customers who owned either the OMEGAMON or DB2 Performance Monitor or DB2 Performance Expert would be subject to an increased or priced upgrade. There are several different combinations that depend on the current product mix, maintenance level of the installed product, and the type of subscription and support entitlement. In general, the rule of thumb is that if a customer is current in support and owns both DB2 OMEGAMON XE for DB2 V300 and either DB2 Performance Expert for z/OS V2 or DB2 Performance Monitor for z/OS V8, there would be a no-charge upgrade available. All other combinations would involve some additional priced upgrade. Table 2-1 further explains the different options.

Table 2-1 Available upgrade and trade-up options

Customer has installed	Wants converged IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS	Wants converged IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS
OMEGAMON XE V300	- Priced Upgrade -	- Priced Upgrade -
DB2 Performance Monitor V8	- Priced Upgrade -	- Priced Upgrade -
DB2 Performance Expert V2	- n/a -	- Priced Upgrade -
DB2 Bufferpool Analyzer V2	- Priced Upgrade -	- Priced Upgrade -
OM XE V300 and DB2 PM V8	- No Charge Upgrade-	- Priced Upgrade -
OMXE V300 and DB2 PE V2	- n/a-	- No Charge Upgrade -
OMXE V300 and BPA V2	- Priced Upgrade -	- Priced Upgrade -

For the latest information on the options and various upgrade offerings, please contact your local IBM sales professional. There is also some information about these offerings in the announcement letter 205-237, which can be obtained via document search from:

http://www.ibm.com/software/data/db2imstools/

3

Architecture

In this chapter we describe the architecture of the IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS product.

We discuss the following topics:

- Terminology and naming convention (Candle versus Tivoli)
- Architecture: Diagrams and data flow
- OMEGAMON XE Performance Expert components

3.1 Terminology and naming convention (Candle versus Tivoli)

As part of the technology convergence, there was a rebranding of the product from the legacy Candle product names, to the new IBM Tivoli brand. In particular, this manifests itself in the names of the different legacy Candle components that require configuration and customization via the customization tool.

Table 3-1 introduces some of the key terms and the appropriate translation into the Tivoli branded equivalent.

Table 3-1 Terminology matrix Candle versus Tivolii

Legacy terms used	Replaced by
Candle Data Warehouse (CDW)	Tivoli Data Warehouse (TDW)
Candle Management Agent (CMA) OMEGAMON Monitoring Agent (OMA) Intelligent Remote Agent (IRA)	Tivoli Enterprise™ Monitoring Agent (TEMA)
Candle Management Server® (CMS)	Tivoli Enterprise Monitoring Server (TEMS)
CandleNet® Portal® (CNP)	Tivoli Enterprise Portal (TEP)
CandleNet Portal Server (CNPS)	Tivoli Enterprise Portal Server (TEPS)
Candle Installation and Configuration Assistance Tool (CICAT)	Installation and Configuration Assistance Tool (ICAT)
Performance Expert Agent (PE Agent)	Performance Expert Agent for DB2 Connect Monitoring
ISPF Online Monitor	PE ISPF Online Monitor
OMEGAMON VTAM® Realtime Monitor	OMEGAMON Classic Interface

Table 3-2 introduces the abbreviations for the names of the products often used in this and other documents.

Table 3-2 Abbreviations of product names

Full name	Replaced by
IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS	OMEGAMON XE for DB2 PE
IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS	OMEGAMON XE for DB2 PM
IBM DB2 Buffer Pool Analyzer for z/OS	Buffer Pool Analyzer

When we read through the product documentation, install panels, and install shield, we observed a somewhat inconsistent application of terminology. We found that this sometimes contributed to confusion during our initial usage of the product, but we expect improvements and more consistency in terminology as time moves on.

Tip: When reading the *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Configuration and Customization Guide,* GC18-9637 and during the execution of the ICAT tool, it is recommended that you keep a copy of the above table readily available for reference.

3.2 Architecture: Diagrams and data flow

To gain an appreciation of the improvements of the new architecture, it helps to understand the current architecture and some of the operational complexities that challenge the existing DB2 Performance Monitor and DB2 Performance Expert customer.

3.2.1 Legacy Performance Expert architecture

The biggest change from an architecture perspective will be for the DB2 PE customer. Currently there is a requirement that for each DB2 subsystem (stand alone) or DB2 data sharing group in an LPAR, there needs to be a corresponding set of DB2 PE subsystem address spaces. This has been a big issue for many customers, resulting in additional resources, operational complexity, and increased maintenance overhead. See Figure 3-1.

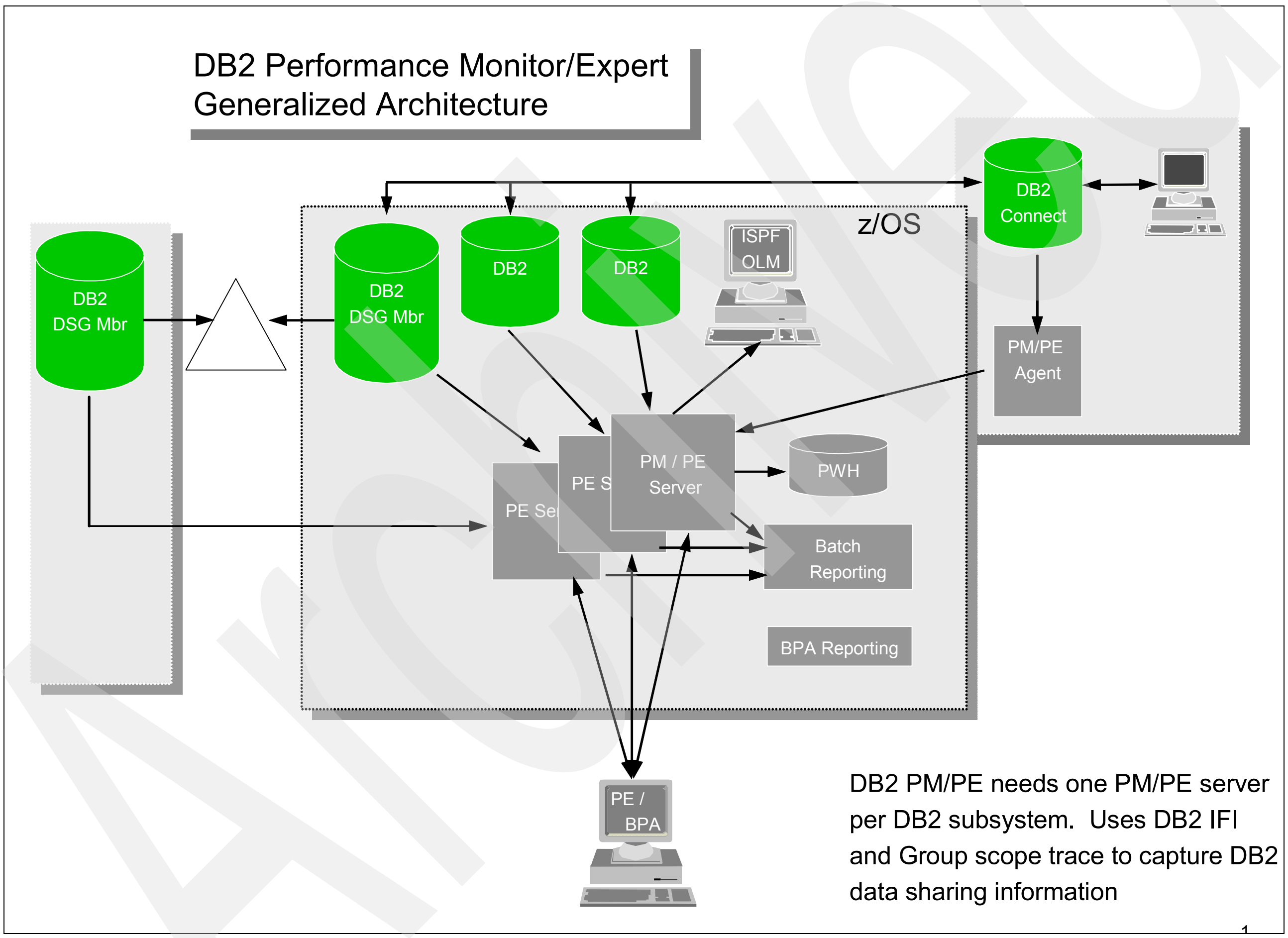


Figure 3-1 DB2 Performance Expert architecture

For each DB2 subsystem active in a particular z/OS LPAR that requires online monitoring, we need to have one DB2 Performance Expert Server active. What is not shown on this diagram is that for each DB2 Performance Expert Server instance, there can be as many as three z/OS started tasks active, depending on which server options were selected at configuration time. We can see that in the case of a DB2 data sharing group, we need only to connect one member of the group to a DB2 Performance Expert Server, the traces needed by the server are started with a group scope context, and the necessary information about the other members is then combined via the IFI trace consolidation mechanism.

3.2.2 Legacy OMEGAMON XE for DB2 architecture

In contrast to the DB2 Performance Expert architecture, the legacy OMEGAMON XE for DB2 product relies on a common server for each LPAR and instrumentation being collected by the different collection subtasks running in each monitored DB2 subsystem. There is no architectural limit to the number of DB2 subsystems that can be monitored from a single OMEGAMON server, but rather a practical limitation due to performance.

Figure 3-2 shows the OMEGAMON XE architecture.

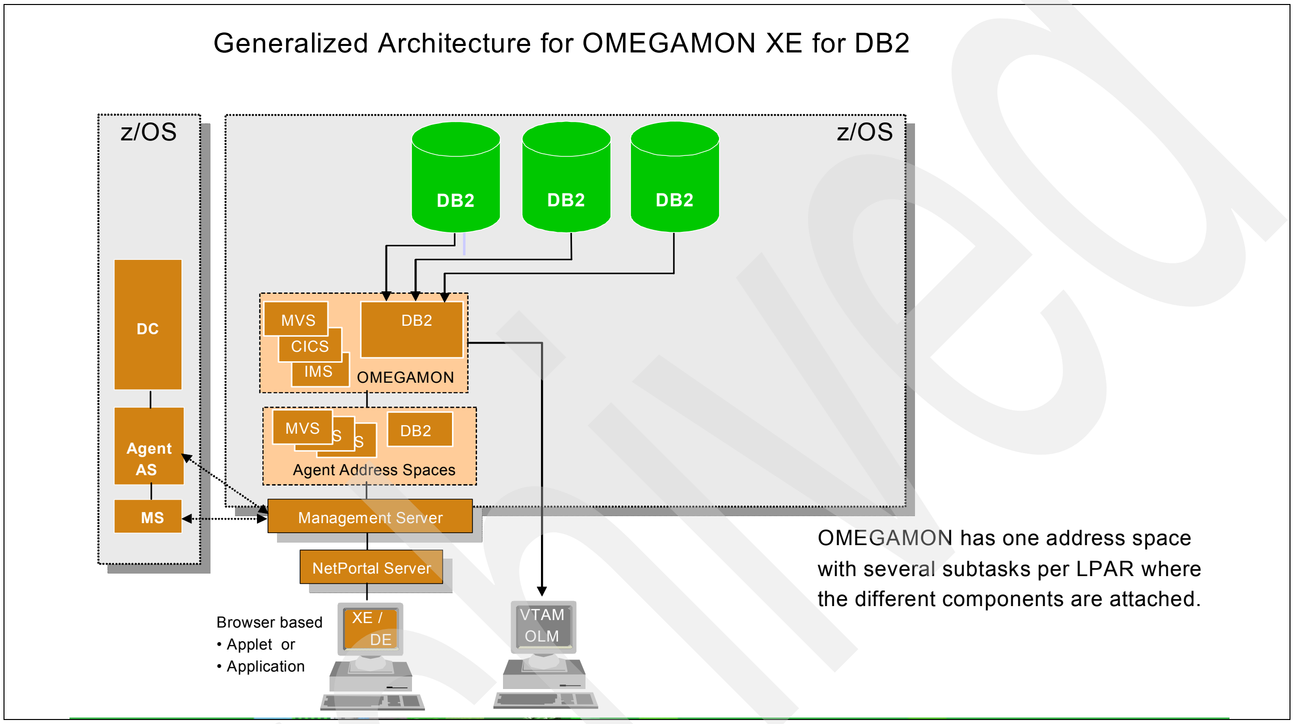


Figure 3-2 OMEGAMON XE for DB2 architecture

Notice that there are a number of DB2 subsystems monitored by a single OMEGAMON started task. Also notice that the OMEGAMON VTAM Realtime monitor connects directly to the started task. So, if the only online monitoring interface needed is the VTAM monitor, there is no need for the other components to be active. If there is a desire to use the browser based applet or thin client application interface, the other parts are needed.

3.2.3 Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS architecture: Phase One

IBM has elected to implement the product convergence in several different stages. At this point, it appears there will be at least two stages to this approach. The first stage, which is what is currently available with Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS, introduces changes in the server back end, while keeping all of the existing user interfaces intact. The main reason for this is to limit the scope of the change on the existing user install base. The biggest impact of the new architecture is the absorption of the DB2 Performance Expert server into the OMEGAMON XE started task. This will allow for a dramatic reduction in the number of monitoring address spaces needed to be resident in any one z/OS LPAR, of particular interest to the existing DB2 Performance Expert customer.

Figure 3-3 describes the first phase of the converged product.

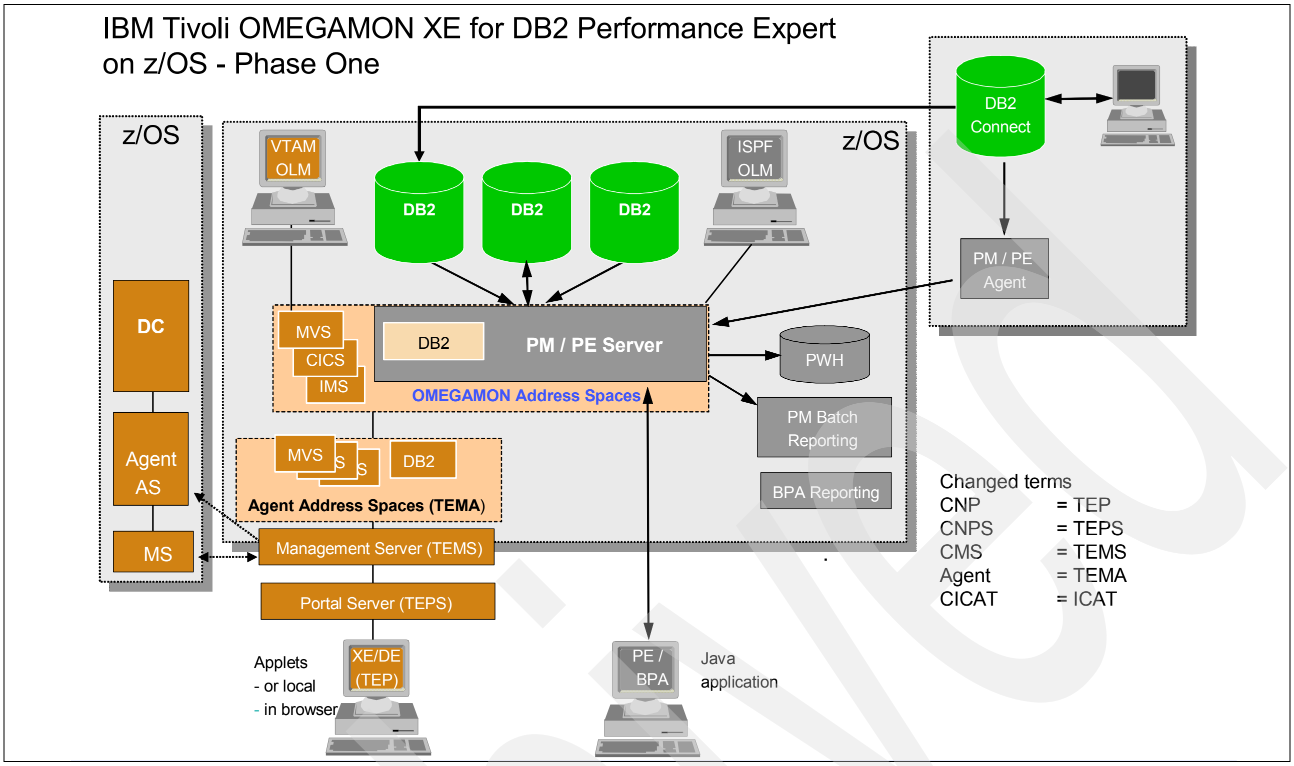


Figure 3-3 OMEGAMON XE for DB2 Performance Expert - Existing architecture

Note that we show a separate address space, the Tivoli Agent Address Space (TEMA) for the different agents. We can also elect to run the different agents as a subtask of the Tivoli Enterprise Monitoring Server (TEMS). This choice would reduce the number of address spaces, but our recommendation is to run a stand alone TEMA for performance and better workload separation. Also, as we can notice from the above diagram, the Tivoli Enterprise Monitoring Server (TEMS) is shown straddling the z/OS box. This is an allusion to acknowledge the fact that we can choose to run the TEMS as either a z/OS address space, or we can elect to place the TEMS on a distributed box running Windows XP. In our configuration, we chose to place the TEMS on z/OS.

Also, notice that the Tivoli Enterprise Portal Server (TEPS) box is placed completely outside of z/OS. This means that the only place TEPS can run is on a distributed machine running Windows XP or other supported distributed operating systems. Finally, notice that in order for the second z/OS LPAR to be monitored, there needs to be a second set of address spaces active in that environment. More details are given on this scenario in subsequent sections.

While most of the work in the current release focused on the server consolidation, there was one important change to existing user interfaces made. Early in the development process, it was decided that in order to demonstrate how the two different sets of user interfaces were to be consolidated, a set of instrumentation data from the DB2 Performance Expert server component was materialized in the OMEGAMON Classic Interface and the Tivoli Enterprise Portal. So, starting with Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS, DB2 Connect monitoring instrumentation can be seen in the DB2 Performance Expert classic client, the OMEGAMON Classic Interface and the Tivoli Enterprise Portal. We do not expect to see this information placed in the PE ISPF Online monitor, which brings us to the next future phase for Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS.

3.2.4 Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS architecture: Phase Two

While the following description has been officially announced as the future statement of direction, at this particular time, the details of the implementation have not been determined. However, understanding the overall long-term direction is important, since this information should guide the Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS customers as to what their user interface of choice should be. Significant focus should not be spent on using the DB2 Performance Expert ISPF Online monitor, as we expect this user interface to be removed in a future phase of the tool's evolution.

As depicted in Figure 3-4, the VTAM Classic OLM interface and the Tivoli Enterprise Portal are the primary choices for online monitoring. The PE workstation client will be gradually merged in the portal solution once all functional and performance-related considerations are taken into account. For instance, currently only the PE client provides a common interface to z/OS and multiplatform DB2s.

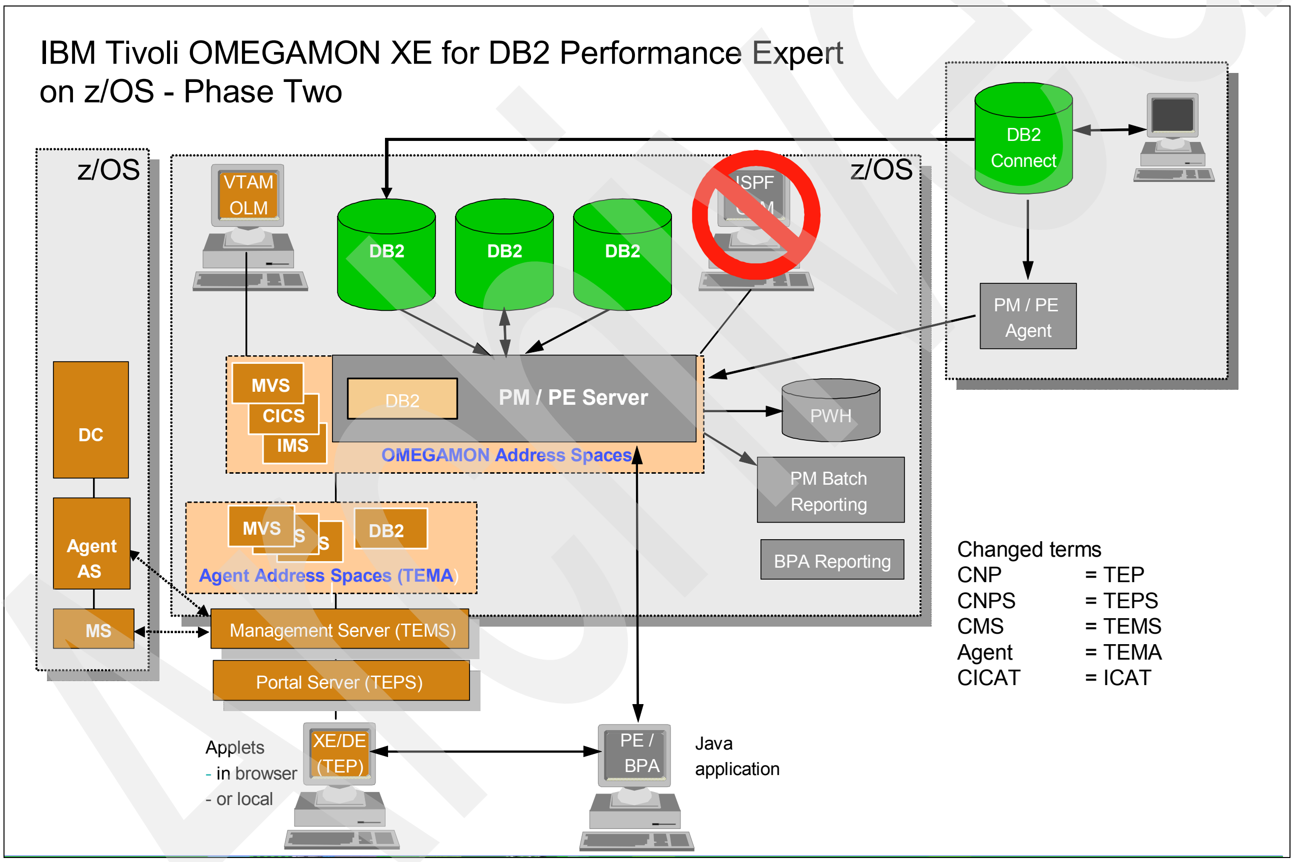


Figure 3-4 OMEGAMON XE for DB2 Performance Expert - Future architecture

> **Tip:** As much as possible, try to use the strategic interfaces for your work. This will reduce any requirement to rework training, documentation, or operational procedures as the product evolves.

3.2.5 Data collection and flow

The following program interfaces can be used to access the OMEGAMON Server:

- MVS Subsystem Interface:

  This interface runs in its own address space providing dynamic I/O device information to OMEGAMON XE for DB2 PE, running in other address spaces. It allows PE ISPF Online Monitor to access OMEGAMON Server functions. It helps you monitor dynamic device activity in z/OS. This subsystem interface allows the PE ISPF Online Monitor to invoke the same functions as the Performance Expert Client, however, the Performance Expert Client accesses those functions via a TCP/IP server.

- TCP/IP Interface:

  This interface is used by the Online Monitor component of the Performance Expert Client. An internal interface is used by the agent to retrieve required data from several collector subtasks.

- Performance Warehouse SQL interface:

  This interface is used by the Performance Warehouse client.

3.2.6 Failover

For purposes of our residency, we focused on building a simple environment with two Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS servers within a two-way data sharing configuration. We connected a hub and remote TEMS to these servers. In a product environment, more complex configuration decisions would be necessary. For example, we would expect that some redundancy and fail-over capability would be architected into the server and TEMS configuration created in the production environment. The details of these decisions are documented in the *IBM Tivoli OMEGAMON V3.1 Deep Dive on z/OS*, SG24-7155 redbook

3.2.7 Historical data collection

So, a brief discussion of the different types of historical data is now in order. Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS provides four different types of historical data:

- Near-term history
- Long-term history
- Snapshot history
- Short-term history

The various types of history data collected by Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS are shown with the diagram in Figure 3-5.

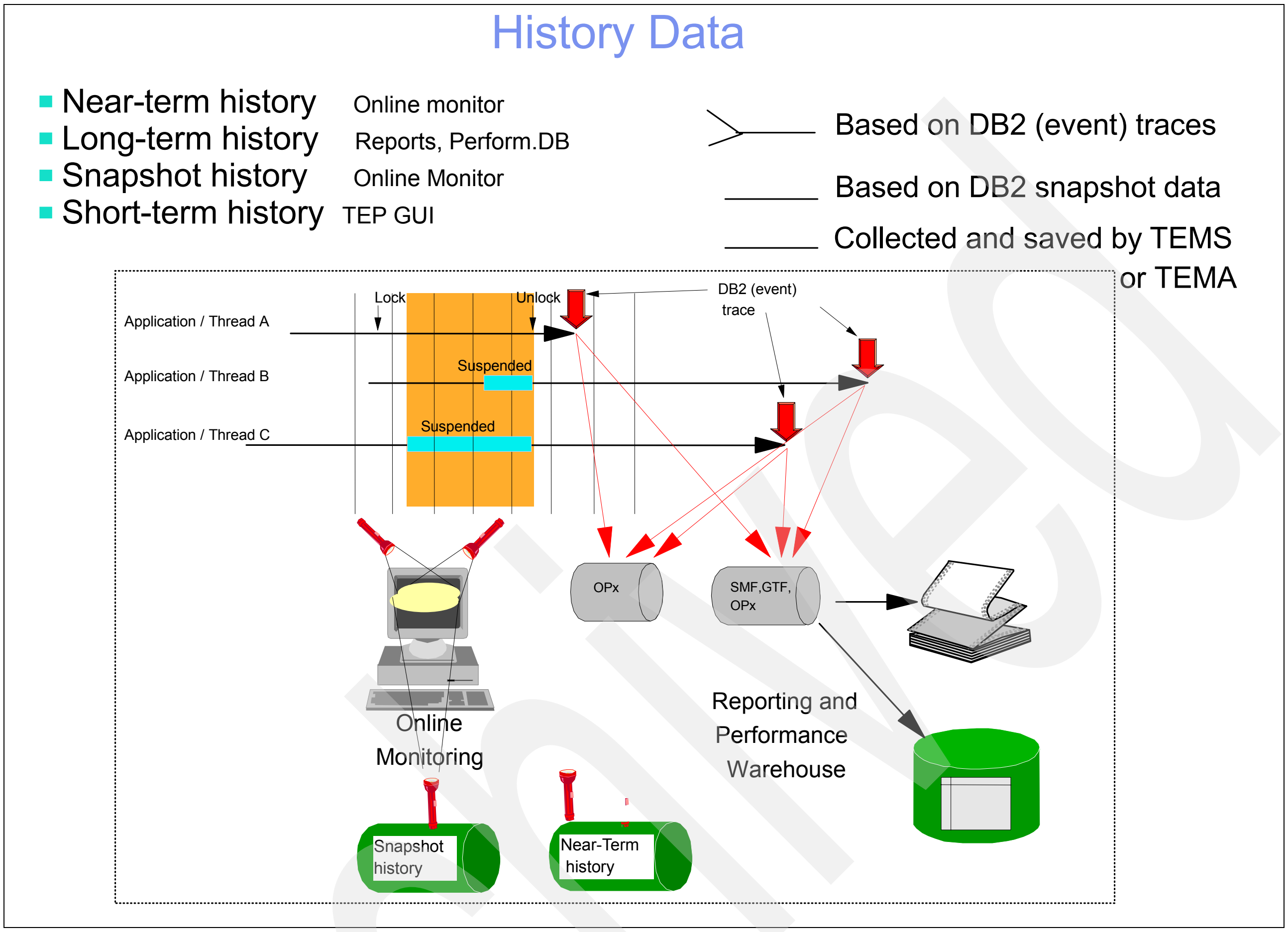


Figure 3-5 Historical data types

Near-term history, which is summarized historical information stored as each individual thread event completes, although always available with OMEGAMON XE, has been a very high requirement with DB2 PE. Some OMEGAMON customers struggle with the size of the backstore needed for this history collection. Much more information on the near-term history feature is available in Chapter 9, “Near-term history using the Classic Interface” on page 137,

Both DB2 PE and OMEGAMON have supported collection and storing of long-term history in DB2 tables (DB2 Performance Warehouse) with input usually coming from DB2 accounting and statistics traces directed to SMF. Management and reporting from a long-term history perspective is covered in Chapter 14, “Performance Database and Performance Warehouse” on page 247.

Snapshot history

Snapshot history has been a DB2 PE feature that is built on the DB2 Snapshot trace data. This PE user interface allows a better user view of concurrent activity over time. Figure 3-6 illustrates the value of snapshot history.

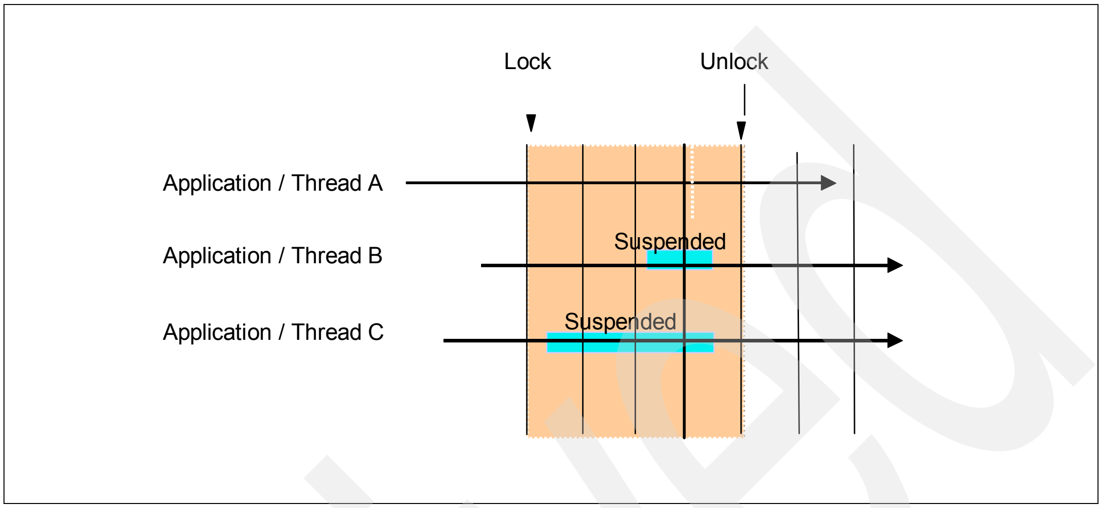


Figure 3-6 Lifeline for three different threads

The example above depicts the situation where three threads are running at the same point in time. When these threads complete, if you look at the accounting detail for these events, threads B and C would show elongated elapsed times and some class 3 wait times associated with lock suspension. There would be no direct correlation between what is seen in the accounting data and the preceptor of the event. Only with snapshot monitoring can we determine the cause for the delay. If there was a deadlock or timeout, then you could also view them from snapshot history.

You can view snapshot history data for threads, locking conflicts, statistics, and system parameters. Snapshot data is periodically stored by the Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS Server in a wrap-around managed snapshot history dataset. Snapshot history data is useful, for example, if you want to examine activities leading to, and following, an exception without recreating the situation. In the Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS server you can define an interval to specify how often and how many snapshots are to be stored. When the defined maximum number of snapshots is exceeded, the oldest snapshot is deleted and the latest snapshot is added. The snapshot history mode allows you to display recently stored snapshots at a specified point-in-time. You can then scroll forward and backward through the history of snapshot data to get a better understanding of what happened and to identify what caused the problem (for example, detected situations, bottlenecks, deadlocks, timeouts)

View current performance data using snapshots

You can display an active view of DB2 subsystem activity (system status and active threads) and identify performance problems, such as bottlenecks and concurrency problems, in online (real-time) mode. This view shows what is currently going on using snapshots. Snapshot data provides real-time information on the current status of a DB2 system or application. A snapshot value is a current status value, and is updated each time the Statistics or Accounting values are displayed. If a snapshot value is shown in a Statistics or Accounting delta mode, it represents the activities of the DB2 system or application between two refresh timestamps. You can also view a snapshot in interval mode, which represents the DB2 activities within the intervals that you have specified.

Examine the evolution of snapshot data in graphs

In short-term history mode you can examine the activity between the saved snapshot data. This information can be displayed in graphs of data over a defined period of time. This TEP user interface provides a global overview of the activities but you cannot identify individual snapshots.

Short-term history or persistent data store can be viewed as a high level “snapshot” implementation. It is stored in the TEMS persistent data store or TEMA. Persistent data store records and stores 24 hours worth of monitoring data on the z/OS system. We can access this data from Tivoli Enterprise Portal interface or the Candle Management Workstation® interface. The data being written to the persistent data store is organized by tables, groups, and datasets. Tivoli Enterprise Monitoring Server provides automatic maintenance for the datasets in the PDS.

Review the recently completed activities using near-term history

The near-term history function allows you to collect finished activities, such as finished threads, including SQL, Scan, and Sort activities in a dataset. You can now display these activities in intervals and zoom into a single thread execution with all its activities. This function is very useful if you want to have a complete statistical view about what happened in the recent past. The difference between the near-term history and the snapshot history, as described above, is that near-term history provides information at the end of a thread, while snapshot history provides periodically collected (down to a one-second interval) snapshot information that allows you to trace back activities, even if the thread is still running.

For more information about snapshot history monitoring and reporting, including how to view this information with both the PE ISPF Online monitor and the PE workstation client, please refer to the redbook, *IBM DB2 Performance Expert for z/OS Version 2*, SG24-6867.

3.3 OMEGAMON XE Performance Expert components

In this section we discuss highlights of the different components of OMEGAMON XE Performance Expert.

3.3.1 OMEGAMON XE Performance Expert Server

The OMEGAMON Server accesses the database engine address space on DB2 Universal Database for z/OS to obtain information about performance data. The Installation and Configuration Assistance Tool (ICAT) is used to customize the OMEGAMON Server and tailors the necessary startup parameters according to your requirements.

The OMEGAMON Server receives commands from its parameter library (as specified by the RKANPAR DD statement of the started task) or the MVS operator’s console. You can issue MVS MODIFY commands to the OMEGAMON Server to start and stop various OMEGAMON XE for DB2 PE components, such as the online data collector or Object Analysis. The OMEGAMON Server provides data and services as follows:

- Performance data for the OMEGAMON Classic Interface
- Tivoli Enterprise Portal user interface components
- Performance Expert Client
- PE ISPF Online Monitor
- Performance Warehouse
- Exception processing
- Event Collection Manager, to analyze object and volume data

3.3.2 Tivoli Enterprise Monitoring Server (TEMS)

Tivoli Enterprise Monitoring Server, or TEMS, acts as the central collector for IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS information. Tivoli Enterprise Monitoring Server (TEMS) provides the following functions:

- Consolidates the data collected by the agent and distributes the data either to the Tivoli Enterprise Portal Server or to the Hub Tivoli Enterprise Monitoring Server if it is a remote Tivoli Enterprise Monitoring Server.
- Evaluates situations and sends alerts to the Tivoli Enterprise Portal Server, when specified conditions are met. It also receives commands from the user interface and distributes them to the appropriate monitoring agents.
- Can optionally store historical data. Stores prototypes (templates) for configuration in the form of seed data, which consists of product-specific situations, templates, and other sample data added to the Tivoli Enterprise Monitoring Server Base (EIB) tables when a monitoring agent is installed. This seeding is initiated from the Tivoli Enterprise Portal Server (TEPS).

Depending on the complexity of your environment and the number of agents you install, Tivoli Enterprise Monitoring Server might need to be deployed in a hierarchical configuration, where one Tivoli Enterprise Monitoring Server is designated as the HUB TEMS. A hub Tivoli Enterprise Monitoring Server is the focal point for managing all of the DB2 for z/OS subsystems running in the z/OS environment. There can be only one hub Tivoli Enterprise Monitoring Server. It communicates with the Tivoli Enterprise Portal Server, with monitoring agents, and optionally with other Tivoli Enterprise Monitoring Servers running remotely (referred to as remote TEMS).

Each remote Tivoli Enterprise Monitoring Server should reside on its own system and have a unique Tivoli Enterprise Monitoring Server name (node). A remote Tivoli Enterprise Monitoring Server communicates with the hub Tivoli Enterprise Monitoring Server and with monitoring agents running on the same or different systems.

A remote Tivoli Enterprise Monitoring Server is deployed to distribute the workload of the hub Tivoli Enterprise Monitoring Server, but it is not required. A good example of where this configuration might be needed is when there multiple DB2 subsystems each running in separate LPARs. One of the configurations documented in Chapter 4, “New installation” on page 35 shows this type of TEMS implementation.

3.3.3 Tivoli Enterprise Monitoring Agent (TEMA)

Monitoring agents monitor and collect performance data from a managed system. The agents are installed on the systems or subsystems you want to monitor and communicate with a single Tivoli Enterprise Monitoring Server (remote or hub). They provide data and performance information to the Tivoli Enterprise Monitoring Server and receive instructions from the Tivoli Enterprise Monitoring Server. They are also able to issue commands to the system or application you are monitoring, either on request or as the result of automation triggered by a situation.

Monitoring agents can be installed and run as subtasks in the Tivoli Enterprise Monitoring Server TEMS, or in a separate z/OS started task. For the configuration used by our study, we elected to install the agent into its own started task.

3.3.4 Tivoli Enterprise Portal Server (TEPS)

Tivoli Enterprise Portal Server is a Java™ application server that enables retrieval, manipulation, and analysis of data from agents. In our configuration the TEPS was installed on a Windows XP platform. The Tivoli Enterprise Portal Server holds all the information needed to format the workspaces viewed in the Tivoli Enterprise Portal clients. The Tivoli Enterprise Portal Server connects Tivoli Enterprise Portal client to Tivoli Enterprise Monitoring Server.

The Tivoli Enterprise Portal Server consists of a collection of services for the Tivoli Enterprise Portal client that enables retrieval, manipulation and analysis of data from the IBM Tivoli Monitoring agents on your enterprise. The Tivoli Enterprise Portal Server connects to the Tivoli Enterprise Monitoring Server or if there is a multiple TEMS environment, connects to the hub TEMS. The Tivoli Enterprise Monitoring Server acts as a collection and control point for alerts received from the agents, and collects performance and availability data from the agents. The main, or hub, Tivoli Enterprise Monitoring Server correlates the monitoring data collected by agents and remote servers (remote TEMS) and passes it to the Tivoli Enterprise Portal Server for presentation and evaluation

Tivoli Enterprise Portal Server maps the request from Tivoli Enterprise Portal as queries and send them to Tivoli Enterprise Monitoring Server. It receives the data returned from Tivoli Enterprise Monitoring Server and process it to build the view on styles (workspaces) saved in each view's properties and returns it to Tivoli Enterprise Portal.

3.3.5 User interfaces for the host

OMEGAMON XE for DB2 PE provides user interfaces for DB2 performance analysis on the host. To monitor and analyze DB2 performance on the host, you can use:

- OMEGAMON Classic Interface
- PE ISPF Online Monitor
- CUA interface (recently re-introduced by PTF UK14018 for APAR PK18535)

OMEGAMON Classic Interface

The OMEGAMON Classic Interface provides real-time information about a DB2 subsystem. It is a Virtual Telecommunications Access Method (VTAM) application, which is configured during customization and configuration using the ICAT dialog. It is a stand-alone started task. We can use the OMEGAMON Classic Interface to:

- Review the current DB2 activity, such as threads or statistics
- Monitor DB2 Connect data received from the OMEGAMON Server and the Performance Expert Agent for DB2 Connect Monitoring
- Observe counter thresholds
- Collect and show near-term history data
- Trace the application or DB2 plan activity
- Analyze objects and drill down into object allocation and usage of thread activities, as well as volume allocation and activities
- Receive detailed runtime performance information at the thread, unit of work, program, and SQL statement levels using ATF
- View enclave information from the Workload Manager (WLM)

For more information on how to display information and navigate through the OMEGAMON Classic Interface, please refer to Chapter 6, "VTAM interface" on page 87.

DB2 PE ISPF Online Monitor

The ISPF Monitoring Dialogs provide access to the main functions or components of OMEGAMON XE for DB2 PE when running under TSO/ISPF. Using the PE ISPF Online monitor allows us to be able to:

- Create and run reporting commands:

  We can select the report specification and input data we want to run. Using this facility, we can create and run reporting commands to generate and run batch report command streams.

- Display performance information:

  It provides different views of the DB2 activity, such as system statistics, DB2 thread statistics, or system parameter settings. It notifies you about event exceptions and allows you to define thresholds on OMEGAMON XE for DB2 PE counters and to observe them. Current or snapshot history activities and information can be displayed and reviewed.

- Maintain parameter datasets:

  This selection is used to maintain the exception threshold dataset, the correlation translation member, the timezone information member, and the MAINPACK definition member. These datasets contain parameters that are used by the User-Tailored Reporting component.

- Customize report and trace layouts:

  Customized batch report and trace layouts according to user requirements can be generated using User-Tailored Reporting (UTR). This function gives full control over the volume, contents, and layout of traces and reports.

- Specify exception profiling:

  We can specify the required datasets and profiling criteria required to calculate and propose the thresholds in an exception threshold dataset. The dataset is used for periodic exception processing.

> **Note:** It is recommended that you familiarize yourself with the OMEGAMON Classic Interface, instead of using the PE ISPF Online Monitor. As previously discussed, the strategic direction for 3270 based OLM user interface is the OMEGAMON Classic VTAM interface.

3.3.6 User interfaces for the workstation

OMEGAMON XE for DB2 PE provides user interfaces for DB2 performance analysis on the workstation. It offers the following workstation user interfaces to monitor and analyze DB2 performance:

- Tivoli Enterprise Portal (TEP)
- Performance Expert Client

Tivoli Enterprise Portal

The Tivoli Enterprise Portal is a browser interface that offers a single point of view for online monitoring and analysis.The Tivoli Enterprise Portal can run as a browser applet or it can be installed as a Java application. The monitoring data is received via the Tivoli Enterprise Portal Server (TEPS) and the Tivoli Enterprise Monitoring Server from the Tivoli Enterprise Monitoring Agents (TEMA). Information is displayed in named workspaces.

The workspace is the working area in which the Tivoli Enterprise Portal presents information. Within a given workspace, information may appear in tabular form. The Tivoli Enterprise Portal refers to this tabular format for information as a table view. Information may also be displayed in the workspace as charts, graphs, or other formats we can specify.

As we select items, the workspace presents views pertinent to our selection. Each workspace has at least one view. Every workspace has a set of properties associated with it. We can customize the workspace by working in the Properties editor to change the style and content of each view. We can also change the type of view or add views to the workspace. Among the types of information that can be shown are:

- Status of the DB2 system
- Buffer pool activities
- Detailed information about the activity in a DB2 group buffer pool
- DB2 log manager active logging and archiving activities
- Workload-related information
- DB2 thread activity originating from connected CICS subsystems
- Information about the connection between two transaction programs
- Distributed Database Facility statistics, including send and receive counts
- Detailed information about threads and thread exceptions connected to DB2
- Environmental Descriptor Manager (EDM) pool activity connected with DB2
- Information about the performance of volumes that contain DB2 objects
- Information on a connection level about all IMS subsystems identified to DB2
- Database lock activities and detailed lock conflict information about we database locks
- Summary information about a data sharing group
- Object analysis information about DB2 databases
- Detailed thread activity information about our DB2 threads
- Performance of volumes that contain DB2 objects
- Active utility jobs

In addition to the display of metrics of performance, we can also use the Tivoli Enterprise Portal to respond to certain events, also known as situation events. A situation contains thresholds that can cause an event to be fired when a threshold is exceed. We can either build custom situations that monitor the performance of our DB2 environment using our attributes (thresholds), or we can use z/OS situations that are shipped with the product. These situations are referred to as *predefined situations*. Every Tivoli Enterprise Monitoring Agent comes with a set of predefined situations, loaded as part of the seeding operation, so we can begin monitoring as soon as we start using OMEGAMON XE for DB2 PE.

Situations allow us to perform the following operations:

- Acknowledge an event to indicate that it has been seen and the problem is being addressed.
- Define alerts and reaction (Take Action) on alerts. Such an alert can be a Boolean expression of all the monitored counters.
- For each alert, an automated action, such as sending e-mails or forwarding alerts can be defined. These alert and action definitions are then sent to the Tivoli Enterprise Monitoring Server and processed, even if the user logs off from the Tivoli Enterprise Portal.

More information on the use of the Tivoli Enterprise Portal can be found in Chapter 7, “Portal interface” on page 101.

Performance Expert Client

Performance Expert Client supports online monitoring and reporting, Performance Warehouse management, and buffer pool analysis on the workstation. The Performance Expert Client can view the activities of:

- Threads
- Database statistics
- System parameters
- DB2 Connect gateways v DB2 Connect specific data for a selected DBAT thread
- DB2 host correlated with the DB2 Connect information
- All connections to the selected DB2 subsystem from different DB2 Connect gateways

In addition, the Performance Expert Client provides access to the following components:

- Exception notification handling.
- The Performance Warehouse component, which generates reports, loads data into the Performance Warehouse, and allows you to define and run SQL queries and Rules-of-Thumbs to analyze and evaluate the data in the Performance Warehouse database. You can create and schedule a Performance Warehouse process and monitor its progress.
- Buffer Pool Analysis.

Using the Performance Warehouse component, you can define and schedule Performance Warehouse processes, run processes that automate the following tasks, and monitor their progress.These processes can include the following activities:

- Collect report data on z/OS
- Create reports based on collected DB2 trace data
- Load the preprocessed data into the Performance Warehouse database

More information on the use of the Performance Warehouse component and the Performance Expert client can be found in Chapter 14, “Performance Database and Performance Warehouse” on page 247.

The Buffer Pool Analysis component helps you manage buffer pools more efficiently by providing specific recommendations to adjust buffer pool sizes and threshold values. If you are interested in background information about current buffer pool behavior, you can get in-depth statistical data, comprehensive reports, and easy-to-read graphic information for all buffer pools and their objects. For more information about the Buffer Pool Analysis component, please refer to the redbook *IBM DB2 Performance Expert for z/OS Version 2,* SG24-6867-01.

> **Note:** If OMEGAMON XE for DB2 PE is installed, the client comprises Online monitoring, Performance Warehouse with extended Rules-of-Thumb, and buffer pool analysis. If OMEGAMON XE for DB2 PM is installed, the client comprises Online monitoring and Performance Warehouse with limited Rules-of-Thumb

Part 2

Installation

In this part of the book, we discuss the steps necessary for a new installation of Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS in a new environment and introduce the concepts of internal and external security.

This part of the book contains the following chapters:

For considerations related to installing the product in environments where the legacy OMEGAMON products are present, please refer to the specific documentation.

4

New installation

In this chapter we cover a new installation of Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS, with no existing legacy OMEGAMON products and associated SMP/E Global and target libraries. This installation generates a new set of SMP/E targets. We also explain the new installation function (ICAT) from the Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS point of view.

We discuss the following topics:

- Introduction
- Our scenario
- Program Directory, PSP, and maintenance
- RTE considerations
- Installation and Configuration Assistance Tool
- Summary of ICAT installation
- Portal Installation
- Installation of z/OS data files
- Seeding the TEMS

4.1 Introduction

Existing Candle customers have used the Candle Installation and Configuration Assistance Tool (CICAT) during the maintenance and configuration process for all of the z/OS OMEGAMON products for a number of years. With the introduction of Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS, some changes surrounding this process have been introduced. The biggest change is that the new dialog, Installation and Configuration Assistance Tool (ICAT), is no longer used for the SMP/E administration process. Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS customers will use the standard SMP/E ISPF dialog or run the GIMSMP batch utility to perform the necessary SMP Receive, Apply, and Accept process. For additional information and specific steps associated with this procedure, please refer to the *Program Directory for IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS,* GI10-8698.

Once the prerequisite SMP/E work has been completed, the Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS customer will use ICAT to perform product customization and configuration. ICAT is a sophisticated ISPF dialog, which will step the installer through various components that require the specification of user or site-specific values. These values are retained, and when subsequent maintenance is applied to the Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS product, ICAT can be run to ensure that a consistent re-application of these stored values are performed thereby eliminating a very time consuming manual re-configuration.

For the existing Candle OMEGAMON customer, ICAT should look very familiar, indeed the basic product navigation and many of the parameters were retained from the legacy OMEGAMON XE product. However, there have been additional parameters and sections added to the dialog to allow for configuration and specification of information to customize the new Performance Expert components.

For the existing Performance Expert customer, ICAT can appear somewhat intimidating, but we have attempted to encapsulate the necessary introductory information in a meaningful manner, and with some good initial planning, using ICAT should not be difficult. In particular, by using the planning worksheet, described in Table 4-8, customers should be able to anticipate any required system or security definitions, and will be able to ensure a consistent choice of ICAT variables throughout the configuration process.

4.2 Our scenario

We are using a two-way data sharing environment composed by one member DB2 (D8F1) in the SC63 LPAR and a second member DB2 (D8F2) in the SC64 LPAR. In addition, we also have a single stand-alone DB2 subsystem (DB8A) in the SC63 LPAR. All of these subsystems are running in V8 NFM. See Figure 4-1.

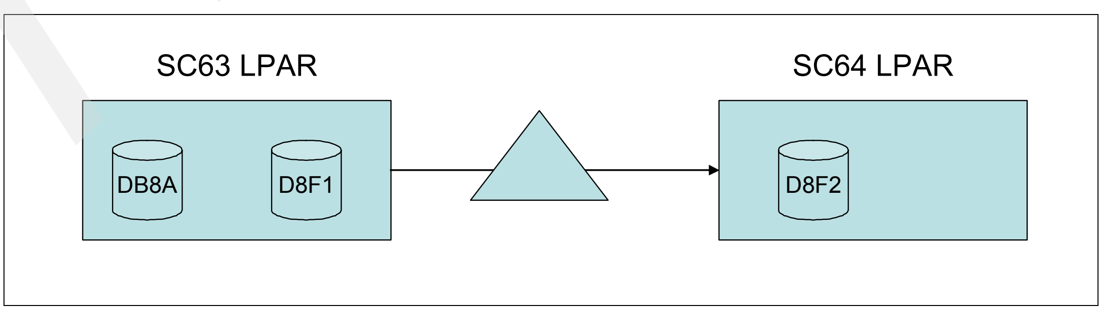


Figure 4-1 Our DB2 scenario

4.3 Program Directory, PSP, and maintenance

Before installing Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS, you should review the Program Directory and the current Preventive Service Planning (PSP) information.You can contact IBM Software Support to obtain the current PSP Bucket or connect to the PSP Bucket link available at:

`http://www14.software.ibm.com/webapp/set2/psp/srchBroker`

PSP Buckets are identified by UPGRADEs, which specify product levels, and SUBSETs, which specify the FMIDs for a product level. The UPGRADE and SUBSET values for Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS are shown in Table 4-1.

Table 4-1 PSP Upgrade and Subset IDs

Upgrade	Subset	Description
5655P07	HKDB31X	OMEGAMON XE for DB2 PE License Key
	HKDB310	OMEGAMON XE for DB2 PE BASE
	HKCI310	Configuration Assistance Tool
	HKDS360	Candle Management Server on z/OS
	HKLV190	CT/Engine
	HKOB550	OMNIMON BASE

The APARs applicable to a new installation of Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS are listed in Table 4-2. They are provided here as the most current maintenance known at the time of writing.

Make sure to contact your IBM Service Representative for the most current maintenance at the time of your installation.

Table 4-2 Current recommended maintenance

APAR #	Notes/Area	PTF #
PK22696	Hiper	UK13407
PK22907	Hiper	UK13417
PK20339	Hiper	UK13248
PK19430	Hiper	UK13215
PK19334	Hiper	UK12397
PK19460	Hiper	UK12444
PK19925	Hiper	UK12258
PK19499	Hiper	UK11617
PK18753	Hiper	UK11475
PK16563	Hiper	UK10007
PK16411	Hiper	UK09946
PK20819	Display active threads	UK12892
PK20650	Buffer pool statistics data	UK12583

APAR #	Notes/Area	PTF #
PK19458	Classic Interface	UK13219
PK19459	Default limit for trace data	UK13407
PK19472	EDM pool percentages	UK13001
PK19473	Command ATVP	UK13001
PK19480	Command EDSN	UK13001
PK19445	Classic STC	UK13407
PK19437	Exception processing	UK13449
PK19428	Exception processing	UK13449
PK19455	Classic STC	UK13407
PK18729	High CPU usage at DB2 classic started task shutdown	UK11475

4.4 RTE considerations

Runtime environment (RTE) is a new concept that IBM is implementing for your installation products. You need to plan your runtime environment before adding and configuring one or more RTEs. The way you set up your RTE depends on your site requirements, maintenance procedures, and DASD sharing capabilities.

This process starts after you have installed the product and applied the maintenance packages through SMP/E.

In the residency, we chose a phased approach. First, we configured a monitoring solution for DB8A demonstrating the usage of a full RTE (without TEMA and TEMS) to monitor the stand alone DB28A subsystem.

We then configured a monitoring solution for a data sharing group demonstrating the usage of a sharing RTE (with TEMA and TEMS configuration). Normally, only one SERVER per LPAR is configured to monitor the DB2 subsystems on that same LPAR, but we had two OMEGAMON servers running on LPAR SC63. In order to avoid conflicts in the server, the option AUTODETECT=NO was set in the run time data set *hlq.RKD2PAR(OMPEMSTR)*. This is a hidden parameter, since the architecture suggests only to have one server on an LPAR.

More on the concepts of RTE can be found in 4.4.1, “What is RTE?” on page 39. Figure 4-2 illustrates our RTE environment.

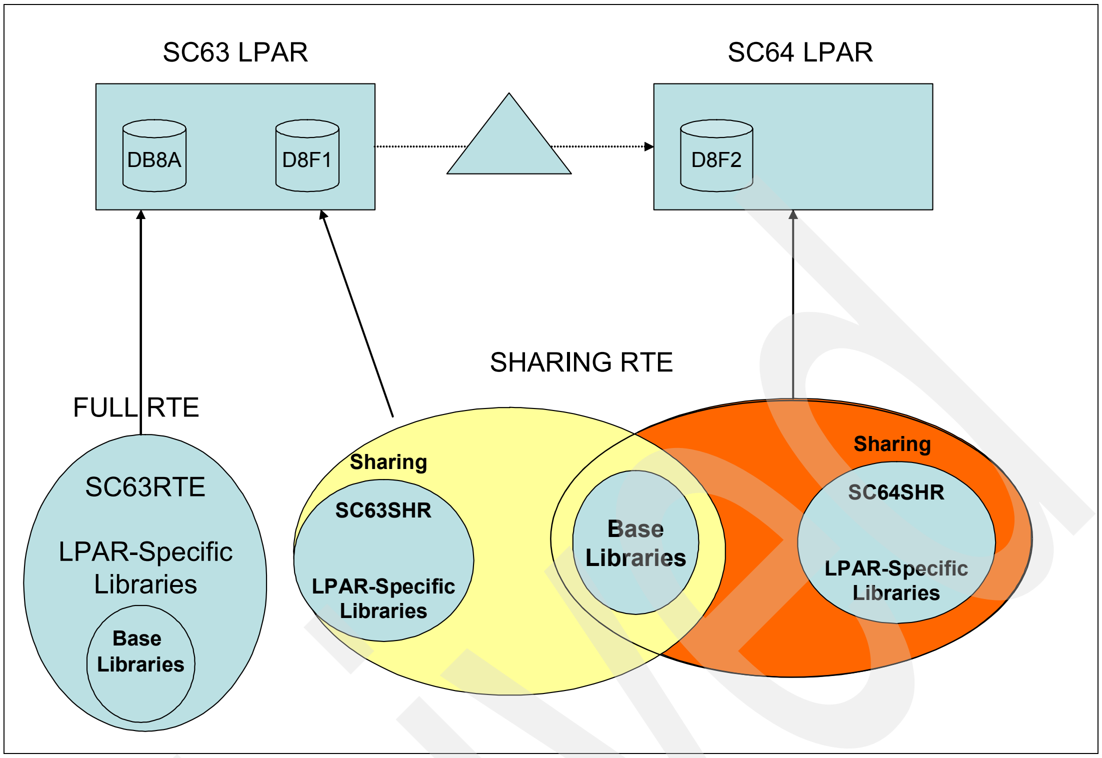


Figure 4-2 Our RTE environment

4.4.1 What is RTE?

RTE is a logical grouping of runtime libraries that are referenced by tasks as they execute on a z/OS image. These datasets belong only to the OMEGAMON set of products.

The different types of runtime libraries used by OMEGAMON are shown in Table 4-3.

Table 4-3 Different types of libraries

TERM	Explanation
Base libraries	Libraries that the configuration process does not change. They can be shared between systems.
LPAR-specific libraries	Libraries that are built during configuration process to run on specific LPAR. The libraries contain the unique elements that are required for a particular LPAR.
Runtime libraries	Libraries that tasks reference during execution.
SMP/E target libraries	SMP/E maintained target libraries.

Table 4-4 shows the different types of RTEs.

Table 4-4 Different types of RTEs

TERM	Explanation
FULL RTE	Configuration containing a full set of dedicated libraries that consist of BASE libraries and LPAR-specific libraries.
BASE RTE	Configuration containing only base libraries. They are a subset of the libraries needed to run your products.Therefore, they must be shared by another RTE
SHARING BASE RTE	Configuration containing LPAR-specific libraries. It references libraries that are configured in a BASE RTE
SHARING SMP/E TARGET RTE	Configuration containing LPAR-specific libraries. It references libraries that are managed by SMP/E

The distinction between types of libraries allows you to optimize your product environment. For example, by allocating common base libraries to a single RTE that can be shared by other RTEs, you can reduce the amount of disk space required. The FULL RTE configuration allows isolation of each OMEGAMON product environment often required when separate technical support organizations are present.

4.4.2 Scenarios of RTE

There are several options when defining the RTE organization depending on your scenario:

- Scenario One: Only one DB2 in one LPAR. In this case the best choice is to create one FULL RTE; it is easier.
- Scenario Two: Two LPARs with one or more DB2s in each one.

  In Table 4-5 you can see three options to consider when creating your RTE environment.

Table 4-5 RTE Environment

Option	LPAR A - Development	LPAR B - Production
	DB2A and DB2B	**DB2C and DB2D**
1	LPAR-specific + BASE sharing	LPAR-specific
2	FULL RTE + 1 server for DB2s	FULL RTE + 1 server for DB2s
3	FULL RTE + 1 server to DB2A and 1 server to DB2B	FULL RTE + 1 server to DB2C and 1 server to DB2D

4.4.3 RTE actions

There are four actions available in the runtime environment:

- Add RTE: When you installed new products or new versions of existing products and you want to create a new RTE
- Build libraries: When you want to use an existing RTE
- Configure: When you have to change the configuration values for the product
- Load libraries: When you installed maintenance and you do not have to change the configuration values

4.5 Installation and Configuration Assistance Tool

The Installation and Configuration Assistance Tool (ICAT) is used to execute mandatory steps to configure the product. You can use Interactive mode by the ISPF-driven facility or use Batch mode if you want to replicate an RTE to another z/OS system. The Batch mode is described in 4.5.3, “Batch mode” on page 58.

Default values are provided by ICAT, and an extensive online Help is provided on each panel. To display help from any ICAT panel, press Help key (F1) or Enter HELP on the command line.

The components of Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS you can see on Chapter 3, “Architecture” on page 17.

In all new installations, you cannot do it alone. You need help from VTAM, MVS, DB2, RACF® or ACF2 or e Trust® CA-Top Secret Security administrators.

> **Important:** If you want to change any parameter, do it using ICAT, instead of updating a dataset member directly. Otherwise, modifications will be lost by subsequent changes in ICAT.

4.5.1 Interactive mode

The first step is to copy the INSTLIB library from SMP/E target dataset to the work library. The values are: RECFM = FB, LRECL = 80, BLKSIZE = 8880, SPACE UNITS = CYL, PRIMARY 12, SECONDARY = 2, DIRECTORIES = 100. Use the IEBCOPY utility to do it.

For our environment, we choose OMEGASYS.SC64 as high level qualifier, then the first step is to execute the following command by option **6** on ISPF:

```
EXEC ‘OMEGASYS.SC64.INSTLIB’
```

The Main Menu panel is displayed in Figure 4-3.

```
---------------------------------- MAIN MENU ----------------------------------
OPTION ===>

 Enter the number to select an option:

    1   Set up work environment

    2   Install products or maintenance (for traditional Candle products only)

    3   Configure products

    I   Installation information                <=== Revised
    S   Services and utilities



        Installation and Configuration Assistance Tool Version 310.04
                       (C) Copyright IBM Corp. 1992-2006
                  Licensed Material - Program Property of IBM

 F1=Help  F3=Back
```

Figure 4-3 Main Menu panel

On the Main Menu, choose option **1** (Set Up Work Environment).

Set up work environment

Once in the Setup panel, you have two options:

- Option **1** (Specify options)

  To define the job card, unit/volser for work datasets, and the region value.

- Option **2** (Allocate work dataset)

  To define the following additional work libraries by submitting a job:

  – HLQ.INSTJOBS
  – HLQ.INSTDATA
  – HLQ.INSTDATW
  – HLQ.INSTLIBW
  – HLQ.INSTSTAT
  – HLQ.INSTQLCK

Attention: After you submit this job, you must exit the ICAT and allow the job to complete successfully before starting the installation tool.

Install products or maintenance

Option **2** (Install products or maintenance) of the Main Menu panel in Figure 4-3 on page 42 may be used from sites that have earlier versions of OMEGAMON.

Configure products

The Configure Products panel offers two options:

- With option **1** you choose the High-Level Qualifier for VSAM and non VSAM datasets as well as the unit name. You also provide the HLQ for SMP/E targets.

- With option **2** you select the product to configure, see Figure 4-4. Possibly, in your installation, you have only the second product presented in the list (Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS).

```
---------------------------- PRODUCT SELECTION MENU ---------------------------
COMMAND ===>

 Actions: S Select product

    IBM Tivoli OMEGAMON XE for CICS on z/OS V3.1.0
    IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS V3.1.0
    IBM Tivoli OMEGAMON XE for IMS on z/OS V3.1.0
    IBM Tivoli OMEGAMON XE for Mainframe Networks V3.1.0
    IBM Tivoli OMEGAMON XE for Storage on z/OS V3.1.0
    IBM Tivoli OMEGAMON XE on z/OS V3.1.0
F1=Help   F3=Back   F5=Refresh   F7=Up   F8=Down
```

Figure 4-4 Product Selection Menu panel

When you select the option for Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS, you see one of the most important panels, because here you define the RTE type of your installation. In our scenario, because it is a data sharing environment, we define one Base RTE and two Base Sharing RTEs — one for each LPAR. See the definition of a Base RTE in Figure 4-5.

```
------------------------- RUNTIME ENVIRONMENTS (RTEs) -------------------------
COMMAND ===>

Actions: A Add RTE, B Build libraries, C Configure,
         L Load all product libraries after SMP/E,
         D Delete, U Update, V View values, Z Utilities

Action Name     Type    Sharing  Description
  A    BASERTE  BASE             Base RTE for Both LPARS (SC63 and SC64)
```

Figure 4-5 Runtime Environment panel definition

The panel shows several actions:

- Action A: To add an new RTE. The BASE RTE requires only actions B and L.
- Action B: Generates a batch job to allocate the required runtime libraries.
- Action C: Through a series of panels, you specify product parameters. Must be performed after option B.
- Action L: Generates a batch job to load the runtime libraries from the SMP/E target libraries.
- Action D: Deletes an existing RTE with all its associated libraries.
- Action U: Permits you to update existing RTE definitions. To apply changes, you must use actions B, C, and L in sequence for the affected products.
- Action Z: Shows available utilities to work with RTEs (such as CREATE BATCH MODE PARAMETERS).

Table 4-6 summarizes the actions required to install the products in all the possible different RTE configurations.

Table 4-6 Actions required to install products

RTE type	Sharing with	Actions required	Actions required in shared / sharing RTE
FULL	N/A	B,C,L	
SHARING	SMP TLIBS	B,C,L	
SHARING	A FULL RTE	B,C,L	B,C,L in full RTE being shared
SHARING	A BASE RTE	B,C,L	B,L in base RTE being shared
BASE	N/A	B,L	B,C,L in all RTEs that share this base

By choosing option **A**, you start to create one new RTE. See Figure 4-6.

```
--------------------------- ADD RUNTIME ENVIRONMENT ---------------------------
COMMAND ===>

RTE: BASERTE  Type: BASE    Desc: BASE RTE for both LPARs ( SC63, SC64)

  Libraries  High-level Qualifier       Volser Unit     Storclas Mgmtclas PDSE
  Non-VSAM   OMEGASYS.SC64              SBOX1P 3390                        N
  VSAM       OMEGASYS.SC64              SBOX1P

  Mid-level qualifier ==> BASE        (Optional for a base RTE)

  JCL suffix               ==> B
  SYSOUT class             ==> X
  Diagnostic SYSOUT class  ==> X
  Load optimization        ==> N   (Y, N)
```

Figure 4-6 Add Runtime Environment panel

We choose JCL suffix = “B“ to reflect RTE type BASE, but this can be any choice. Also, the mid-level qualifier = “BASE” for the same reason. For the Load optimization field, you have two options:

- **Y** – The job Load copies only modified modules from target to runtime libraries and requires access to IBM's SuperC (ISRSUPC) utility, because this utility is responsible to compare modules from target against runtime library to create one IEBCOPY JCL with changed modules. ISRSUPC is a Super compare utility for C, the same that you use when you choose option 3.14 in ISPF.
- **N** – The job Load copies all members from target to runtime libraries whether or not they were modified.

After you press Enter, the ICAT returns to the RunTime Environments panel, where you need to choose the next option **B**. This option builds all necessary target libraries by generating the batch job as shown in Figure 4-7. Submit the job manually.

```
//PAOLOR4A JOB (999,POK),'DB8X INSTALL',CLASS=A,
// MSGCLASS=T,NOTIFY=PAOLOR4,TIME=1440,REGION=0M
//****************************************************************
//* DATE: 06/02/01
//* TIME: 13:05
//****************************************************************
//*
//* JOB: OMEGASYS.SC64.INSTJOBS(D5#1B)
//*
//****************************************************************
//* DATE: 06/02/01
//* TIME: 13:05
//* PURPOSE: ALLOCATE A RUNTIME ENVIRONMENT FOR THIS PRODUCT.
//*
//* NOTES:   THIS JOB CONTAINS 4 PARTS:
//*           1. NEW NON-VSAM LIBRARIES
//*           2. NEW VSAM LIBRARIES
//*           3. EXISTING NON-VSAM LIBRARIES THAT WERE FOUND IN
```

Figure 4-7 Batch job to allocate libraries

After you submit this job, you need to choose option **L** to load all target libraries. The option **C** is not necessary here because we are dealing with a BASE RTE. In the following pages, you will see how to configure the product for a Sharing RTE. For option **L**, one batch job is created to copy a dataset from SMP/E to target libraries by IEBCOPY utility. See Figure 4-8.

```
//****************************************************************
//* JOB: OMEGASYS.SC64.INSTJOBS(D5#2B)
//*******************************************************************
//* DATE: 06/02/01
//* TIME: 13:17
//* PURPOSE:  THIS JOB LOADS ALL PRODUCT LIBRARIES IN THIS RUNTIME
//*           ENVIRONMENT (RTE).
//*
//* NOTES: 1. RUNTIME LIBRARIES ARE LOADED WITH IEBCOPY.
//*           EACH LIBRARY IS LOADED WITH A 2 STEP PROCESS:
//*           A. COPY ALL MEMBERS FROM THE SMP/E TARGET TO THE
//*              RUNTIME LIBRARY.
//*           B. COMPRESS THE RUNTIME LIBRARY.
//*
//*        2. THE OUTPUT DATASETS ARE ALLOCATED USING DISP=OLD.
//*           IF THE DATASETS ARE CURRENTLY BEING USED, YOU MUST
//*           STOP THE USING TASKS BEFORE THE JOB CAN COMPLETE.
//KANCLI EXEC PGM=IEBCOPY,REGION=0M
//SYSPRINT DD SYSOUT=*
//TKANCLI DD DISP=SHR,
//        DSN=OMEGAXE3.TKANCLI
//RKANCLI DD DISP=OLD,
//        DSN=OMEGASYS.SC64.BASE.RKANCLI
```

Figure 4-8 Batch job to copy SMP/E libraries to target datasets

After the successful job run, the creation for RTE BASE is completed and you receive the message `ENTRY ADDED` on the right corner of the Runtime Environment panel.

Now it is time to create the other two LPAR-specific sharing RTEs. You need to add one sharing RTE and specify what base you are sharing. See the example in Figure 4-9.

```
------------------------- RUNTIME ENVIRONMENTS (RTEs) -------------------------
COMMAND ===>

Actions: A Add RTE, B Build libraries, C Configure,
         L Load all product libraries after SMP/E,
         D Delete, U Update, V View values, Z Utilities

Action Name     Type    Sharing  Description
  A    SC64SHR  SHARING BASERTE  Lpar-specific for SC64 LPAR
       BASERTE  BASE             BASE RTE for both LPARs ( SC63, SC64)
Enter=Next  F1=Help  F3=Back  F7=Up  F8=Down
```

Figure 4-9 Runtime Environment panel

When you start to create an LPAR-specific RTE (see Figure 4-10), you are presented with a different panel from the one you saw in Figure 4-6 on page 44. Notice that some of the Candle terminology still appears in this and other panels.

```
----------------------- ADD RUNTIME ENVIRONMENT (1 of 2) ----------------------
COMMAND ===>

RTE: SC64SHR  Type: SHARING Desc: LPAR-specifi for SC64 LPAR

  Libraries  High-level Qualifier       Volser Unit     Storclas Mgmtclas PDSE
  Non-VSAM   OMEGASYS.SC64              SBOX1P 3390                       N
  VSAM       OMEGASYS.SC64              SBOX1P
  Mid-level qualifier ==> SC64SHR

  JCL suffix          ==> 4
  STC prefix          ==> D8F2
  SYSOUT class        ==> T            Diagnostic SYSOUT class  ==> T
  Load optimization   ==> N     (Y, N)

  Will this RTE have a Candle Management Server                ==> Y  (Y, N)
     If Y, CMS name     ==> SC64SHR:CMS                      (Case sensitive)

  Copy configuration values from RTE ==>           (Optional)
```

Figure 4-10 Add Runtime Environment panel - Part 1

You can choose any *High-level* qualifier. The default is the RTE name. The *STC prefix* is the default prefix to be used when generating started task PROCs for products configured in this RTE. We chose the DB2 subsystem name.

If you plan to use only the Classic VTAM interface, you need to install only the Tivoli OMEGAMON XE for DB2 Performance Expert on the z/OS server.

If you plan to use the Browser, you need to install the Tivoli OMEGAMON XE for DB2 Performance Expert on the z/OS server, Tivoli Enterprise Monitoring Agent (TEMA) or Tivoli Enterprise Monitoring Server (TEMS), and Tivoli Enterprise Portal Server (TEPS).

The TEMS (or CMS in the panel) is one started task responsible to make communications between Agent Address Space and PORTAL. If you connect TEMS directly with the Portal, this is a *Hub* TEMS. A second option is that you can connect one TEMS to one Hub TEMS, then the name is *Remote* TEMS.

A Hub CMS receives data from agents and Remote CMSs in the environment and communicates data and to and receives command from an interface.

A Remote CMS receives data from agents and communicates data to the Hub CMS only. It does not communicate directly with an interface.

The default for Candle Management Server is “Y“. If you do not want to configure CMS, choose “N”.

This name is used by other non-local CMS and/or products running in their own address space, that need to communicate with this CMS. The CMS name must not contain character strings *SMFID, *HOST, or *NETID. This name represents that CMS_NODEID parameter value in the KDSENV member of the RKANPAR library for the CMS configured in this RTE.

After you press Enter, you are presented with a panel where you can configure the second part regarding options on VTAM, TCP/IP, and security. See Figure 4-11.

```
----------------------- ADD RUNTIME ENVIRONMENT (2 of 2) ----------------------
COMMAND ===>

Use z/OS system variables?     ==> N (Y, N)
  RTE name specification       ==> &SYSNAME
  RTE base alias specification ==>
  Applid prefix specification  ==> K&SYSCLONE.
  Use VTAM model applids?      ==> N (Y, N)

Security system       ==> NONE  (RACF, ACF2, TSS, NAM, None)
  ACF2 macro library  ==>

If you require VTAM communications for this RTE, complete these values:
  Applid prefix       ==> SC64       Network ID     ==> USIBMSC
  Logmode table       ==> KDSMTAB1   LU6.2 logmode  ==> CANCTDCS

If you require TCP/IP communications for this RTE, complete these values:
 *Hostname     ==> WTSC64.ITSO.IBM.COM               * (See F1=Help for
 *Address      ==> 9.12.6.9                             HOMETEST instructions)
  Started task ==> TCPIP
  Port number  ==> 1910
  Interlink su ==>        (if applicable)
 Enter=Next  F1=Help  F3=Back
```

Figure 4-11 Add Runtime Environment panel - Part 2

The Security system information is described in Chapter 5, “Security considerations” on page 77.

We chose `SC64` for the applid prefix. If you need more explanation on how the configuration tool processes VTAM applids, type `README APP` on the command line.

The Hostname is the domain name in the DB2 MSTR started task, and the Address can be obtained by doing a `PING` command of this hostname address. You can assign any Port number, as long as it matches the VTAM definition.

Press Enter and you receive an `ENTRY ADDED` message.

The next step is to Build, as done for the BASE RTE in Figure 4-7 on page 45. The new step here is to use option **C** (Configure the RTE). The components that you have to configure are shown in the Product Component Selection Menu in Figure 4-12.

```
COMPONENT TITLE

1   Candle Management Server
2   OMEGAMON XE for DB2 Performance Expert on z/OS
3   OMEGAMON XE for DB2 Agent
```

Figure 4-12 Product Component Selection Menu panel

Choosing option **1**, you start to define the CMS, as you can see in Figure 4-13. You do not need to configure CMS again if you already had configured a CMS component in this SC64SHR runtime environment (from a different product package).

```
------------- CONFIGURE THE CMS (V360) / RTE: SC64SHR -------------------------
OPTION ===>

 Each RTE can contain only one CMS.  To configure           Last selected
 the CMS for this RTE, perform these steps in order:        Date     Time

   1  Create LU6.2 logmode
   2  Specify configuration values
   3  Specify communication protocols
   4  Create runtime members
   5  Configure persistent datastore
   6  Complete the configuration

 Optional:

   7  View CMS list and registration status
```

Figure 4-13 Configure CMS panel

The Last Selected Date and Time is updated every time this option is exited. This allows you better control to view which items have already been specified.

Option 1 (Figure 4-13 above) defines a LU6.2 logmode by a batch job submission. See Figure 4-14. You need to have the authority to update SYS1.VTAMLIB.

```
----------------------------- CREATE LU6.2 LOGMODE ----------------------------
COMMAND ===>

 The CMS requires an LU6.2 logmode.  Complete the items on this panel and
 press Enter to create a job that will assemble and link the required logmode.

   LU6.2 logmode        ==> CANCTDCS
   Logmode table name   ==> KDSMTAB1

   VTAMLIB load library ==> SYS1.VTAMLIB
   VTAM macro library   ==> SYS1.SISTMAC1
```

Figure 4-14 LU6.2 Logmode panel

Option 2 (Figure 4-13 above) specifies the configuration values. There are two options to connect CMS to the Portal. If you connect CMS directly with the Portal, this is a Hub CMS. The second option is that you can connect one CMS to one Hub CMS, then the name is REMOTE CMS. We chose to connect this CMS directly with the Portal, so this is a Hub CMS. If you want to ensure that CMS security access validates the userid, select "Y", otherwise the application userid is authorized. This verification is made by RACF, TOP-SECRET, or ACF2. See Figure 4-15.

Note: Do not enable security validation until your security is set.

```
------------------------- SPECIFY CONFIGURATION VALUES ------------------------
COMMAND ===>

 To configure a CMS for your site, complete the items on this panel.

   CMS started task      ==> D8F2DSST
   Hub or Remote?        ==> HUB
   Security validation?  ==> N        (Y, N)
```

Figure 4-15 Configuration Values panel

Option 3 (Figure 4-13 on page 48) specifies communication protocol values. We chose the default values for this panel. The communication protocols are used in the priority sequence that you set. The Hostname information is obtained with the **TSO HOMETEST** command, and you have to use the first qualifier of the TCP hostname.

Option 4 (Figure 4-13 on page 48) provides a job that creates members required by CMS.These members are created in the runtime libraries for this RTE, like APPLIDs, Started task procedure to run Hub CMS. You need to submit the generated job.

Option 5 (Figure 4-13 on page 48) provides a batch job that configures a persistent datastore (PDS), which is added to the CMS address space for historical data collection. This option is required for the OMEGAMON XE applications configured to run in the CMS address space. You must also configure the PDS in the Allocate Persistent Detester Menu panel, as in Figure 4-16, if you want the CMS to collect historical data for the OMEGAMON XE product.

Tip: If SHORT TERM HISTORY mode is not available from the XE client, you do not need to select this option.

```
---------------------- ALLOCATE PERSISTENT DATASTORE MENU ---------------------
OPTION ===>
                                                            Last selected
 Perform these configuration steps in order:                Date     Time

   1  Modify and review datastore specifications           06/02/02  11:36
   2  Create or edit PDS maintenance jobcard
   3  Create runtime members
   4  Edit and submit datastore allocation job
   5  Complete persistent datastore configuration
```

Figure 4-16 Allocate Persistent Datastore Menu panel

Perform these configuration steps in order (as listed in Figure 4-16):

- **Option 1**: Define the allocation parameters.
- **Option 2**: Create the job card to be used for the Persistent Datastore KPDPROC1, KPDPROCC, and KPDPROC2 maintenance jobs.
- **Option 3**: This batch job creates Persistent Datastore Members.
- **Option 4**: This batch job allocates and initializes all datasets required for Persistent Datastore.
- **Option 5**: This provides instruction to complete the configuration.

Option 6 (Figure 4-13 on page 48) provides instructions to complete the configuration. Perform the Load all product libraries, and then you must execute a number of procedures outside of the Configuration tool to finalize installation and configuration of this product.

Option 7 (Figure 4-13 on page 48) allows you to view the CMS list and registration status.

After you press PF3, you return to the Product Component Selection Menu panel as shown in Figure 4-12 on page 48. Now you select **option 2** (Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS) to start to configure this product by the panel shown in Figure 4-17.

```
Configure OMEGAMON XE for DB2 PE on z/OS         RTE: SC64SHR -
Option ===>

 Perform the following configuration steps in numerical     Last selected
 order:                                                     Date     Time

   1  Specify configuration values
   2  Allocate additional runtime data sets
   3  Create runtime members
   4  Modify Classic Interface command security
   5  Complete the configuration

 Optional:

   6  Install OMEGAMON subsystem
   7  Run migration utility
```

Figure 4-17 Allocate Persistent Datastore Menu panel

Option 1 takes you to another panel where you need to select **option 1** to configure the Server values. See Figure 4-18.

```
 OMEGAMON XE for DB2 PE --- Server Information --- SSID:
 Command ===>

   Complete the following OMEGAMON Server information:
    Started task                             ==> D8F2O2S_
    Started task AUTHID                      ==> D8F2O2S_
    MVS system ID (SMF)                      ==> SC64
    Use DB2 authorization exit?              ==> N  (Y, N)
    Enable data server DB2 user exit?        ==> N  (Y, N)
    Autostart functions:
      Event Collection Manager               ==> Y  (Y, N)
      Near-Term History Data Collector       ==> Y  (Y, N)
      Object/Volume Analysis                 ==> N  (Y, N)
      DB2 event observation                  ==> N  (Y, N)
      CPU Parallelism                        ==> Y  (Y, N)
      Performance Expert Client support      ==> Y  (Y, N)

    Information for temporary data sets (if not SMS managed):
      Unit    ==> SYSDA___           Volume ==> 000000

    Dataserver Datasets High-Level Qualifier ==> OMEGASYS.D8F2__
  Enter=Next  F1=Help  F3=Back
```

Figure 4-18 Server Information panel

The started task name for the server is defined here, and its authid. This information needs to be agreed upon with your security administrator. The MVS system ID (SMF) is the LPAR name where the event collection manager is running. When Object/Volume Analysis is specified as **Y**, the Event collection manager option must also be specified as **Y**.

Tip: The Autostart function can be dynamically changed, but our recommendation is to start the desired functions from the beginning.

After you press Enter, the next panel shows you the Classic Interface Information. See Figure 4-19.

```
 OMEGAMON XE for DB2 PE - Classic Interface Information - SSID:
Command ===>

 Complete the following Classic Interface information:

  Classic Interface VTAM mode:
    Major node                            ==> SC64D2N2__________________
    Classic logon                         ==> SC64D2C___________________
  Maximum number of users (UMAX)          ==> 99     (1-99)
  Default DB2 ID for real-time VTAM mode  ==> D8F2
  Wait interval                           ==> _0     (0-99 seconds)



 Enter=Next  F1=Help  F3=Back
```

Figure 4-19 Classic Interface Information panel

The Classic Interface logon information is the name that you use to call the OMEGAMON Classic Interface in your emulation session. After you press Enter, the next panel allows you to specify the IP address for the OMEGAMON server. See Figure 4-20.

```
 OMEGAMON XE for DB2 PE --- PE Client support ---  SSID:
Command ===>

  Specify the following information for the graphical user interface (GUI)
  of Performance Expert Client:

   TCP/IP name             ==> TCPIP___
   IP address              ==> 9__ . 12_ . 6__ . 9__ (leave blank for default)
   Maximum number of sessions  ==> _99               (0-500)
```

Figure 4-20   PE Client support panel

Press Enter and a panel with ISPF language is displayed. Select your option and press Enter again to determine the size of your data space for CPU parallelism. Press Enter and in the Miscellaneous information panel you decide if you want all connected Performance Expert Agents for DB2 Connect Monitoring to search for program updates and download them automatically.

Press Enter and you return to the Tivoli OMEGAMON XE for DB2 Performance Expert on the z/OS Configuration Values pane. Here you select **option 2** to determine the subsystem name for DB2, as you can see in Figure 4-21.

```
------------------------ DB2ID SPECIFICATION/SELECTION ------------------------
Option ===>                                                  SCROLL ===> PAGE

  Select a DB2ID name with an action code, then press Enter.
  Actions: C Copy, D Delete, S Select for processing
  Note:  To add a new DB2ID, specify the DB2ID on the first blank line.
         To clone an existing DB2ID, specify the name of the clone on the
         first blank line and place C in front of the DB2ID you want to clone.

      DB2ID    Description                              Status Last Selected
      D8F2     Datasharing DB2 for LPAR SC64_________
            * * * End of DB2ID table  * * *
```

Figure 4-21   PE Client support panel

After you had added the new subsystem name, select it and the DB2 subsystem information panel is displayed. See Figure 4-22.

```
 Specify OMEGAMON XE for DB2 PE on z/OS Subsystem     RTE: SC64SHR -
Option ===>

                      DB2 Subsystem ID: D8F2

 Select one or more options to configure OMEGAMON XE for DB2 Performance
 Expert on z/OS. To update the entire configuration, type ALL. To update
 specific values, select the corresponding options from the list.
                                                               Last selected
                                                               Date     Time
  1  DB2 subsystem information                    (mandatory)
  2  Object/Volume Analysis configuration         (mandatory)
  3  Application Trace Facility configuration
  4  Automatic Exception Processing configuration
  5  Near-Term History Data Collector config.     (mandatory)
  6  Snapshot History configuration
  7  Performance Warehouse configuration
  8  DB2 EXPLAIN configuration

  Note: Options highlighted as mandatory require configuration.
```

Figure 4-22 PE Client support panel

Select **option 1** to update the Version of DB2, DSNLOAD, DSNEXIT, and RUNLIB.LOAD datasets on the DB2 subsystem information panel.

Press Enter and now provide the TCP/IP port for the PE Server communicating with the PE client. Press Enter, select another mandatory **option 2**, and provide values to Object and Volume analysis. Press Enter, and if you want to provide the dataset name for the Application Trace Facility, you have to select **option 3**. Otherwise, if you want to specify information to activate exception thresholds at start up, select **option 4**.

Press Enter and select the mandatory **option 5** to specify the first part of the information to create the Near-Term History VSAM log datasets and press Enter. You then see the last panel of Near-Term History configuration. See Figure 4-23.

```
 OMEGAMON XE for DB2 PE Near Term History configuration  SSID: D8F2
Command ===>

Specify how data should be collected.

 Collection interval...: 15     (RMF, 1-60 minutes)
 Buffer size...........: 2048   (50-2048 K - maximum for DB2 V7.1 is 1024 K)
  IFIREADTIME:           010000 (mmssth)
 Max hours.............: 24     (1-168 hours)
 Suspend collection....: Y      (Y, N)
 Threshold percentage..: 70     (1-99)

Thread specifications:
  Group by......:
  Select when...: AUTHID
                  ORIGAUTHID
                  PLANNAME
                  CONNID
                  CORRID
```

Figure 4-23 PE Client support panel

In this panel you provide the type of data to be collected, the time interval for data collection, and how data should be backed up. You can specify a buffer size 2048 also for DB2 V7, but you need to apply PTF UQ92335. If the historical VSAM file is full, it is considered unavailable until the end of a successful archive job execution. The **option Y** means that the collected historical data is discarded until a VAM file becomes available for use, and the **option N** means that the collected historical data maintains by collector in memory until a VSAM file becomes available. The Threshold percentage for historical collection is a percentage of the total number of bytes in the IFI buffer when this threshold is exceeded and DB2 posts the Historical Collector to drain the buffer. Press Enter and the panel where you can select the data to be collected is displayed. After Enter, the panel in Figure 4-22 is displayed again.

Option 6 (optional) determines the Snapshot History configuration and specific authid or plan name.

Option 7 (optional) determines if you are using Performance Warehouse or not. If **Y**, it needs information about the jobname and storage group because some tables need to be created.

Option 8 (optional) is to be configured if you want to activate DB2 Explain.The database name “DATBA8K” is the default, but you can change it. Pressing PF3, you return to the Configure OMEGAMON XE for DB2 PE on the z/OS panel to choose option **2** to allocate additional runtime datasets for batch job.

In **option 3** you create the runtime members. These members are divided in two distinct sets of elements. Product related members (generic) and DB2 subsystem selection/install DB2 related members where you select DB2ids (subsystem ids) to include them in the RTE.

Option 4 (Modify Classic Interface Command Security) is used when you want to protect commands in the Classic Interface. There is one internal security table with level authorization 0 to 3, where “0” indicates that the command is available for everyone, and the other levels need a password to access.

Options 6 and **7:** If your MVS operating system level is MVS/ESA™ Version 4.2 or above, the OMEGAMON subsystem is required. For prior releases of MVS, bypass this step. Press PF3 and now choose the configuration of the DB2 agent. See the options in Figure 4-24.

```
------------ CONFIGURE OMEGAMON XE FOR DB2 ON Z/OS / RTE: SC64SHR -------------
OPTION ===>
                                                            Last selected
 Perform the appropriate configuration steps in order:      Date     Time

 If you have defined a CMS in this RTE that this Agent
 will communicate with, select option 1.
   1  Register with local CMS                              06/02/03  16:37
   2  Configure persistent datastore    (Optional)         06/02/03  16:41

 Required if running in the Agent or CMS address space:
   3  Define DB2 subsystems to CMS                         06/02/03  16:42

 To run in an Agent address space:      (Recommended)
   4  Specify Agent address space parameters               06/02/03  16:44
   5  Create runtime members                               06/02/03  16:45
   6  Configure persistent datastore    (Optional)         06/02/03  17:32

 To run in the CMS address space:       (Alternative)
   7  Install Agent into local CMS

   8  Complete the configuration                           06/02/03  17:40
```

Figure 4-24 OMEGAMON XE for DB2 agent panel

This configuration is necessary because we choose to run the browser, than we need a definition of Portal, Agent, and CMS. To make this agent communicate with the CMS in this RTE, choose **option 1**, and one batch job needs to be submitted. The **option 2** (optional) provides a batch job that configures a persistent datastore (PDS) that is added to the CMS address space for historical data collection. You are taken to the next panel (Allocate Persistent Datastore Menu) with five new options. Two batch jobs need to be submitted to complete this configuration. To define DB2 to CMS, you need to choose **option 3** and add the new subsystem name and submit the batch job. Our recommendation is to run the agent in your own address space, in order not to cause stress on CMS. In Figure 4-25 you provide the agent started task name and the protocol communication.

```
-------------------- SPECIFY AGENT ADDRESS SPACE PARAMETERS -------------------
COMMAND ===>


 The following information is needed to define the Agent address space:

  Agent started task            ==> D8F2D5S
  Connect to CMS in this RTE    ==> Y         (Y, N)
     Name of Primary CMS        ==> SC64SHR:CMS

  Specify communication protocols in priority sequence:
     Protocol 1            ==> IPPIPE  (IPPIPE, IP, SNA)
     Protocol 2 (optional) ==> IP      (IPPIPE, IP, SNA)
     Protocol 3 (optional) ==> SNA     (IPPIPE, IP, SNA)
```

Figure 4-25 Agent address space panel

In **option 5** you create the Runtime members by batch job and **option 6** (optional) provides a batch job that configures a persistent datastore that is added to the Agent address space for historical data collection. You are taken to the next panel (Allocate Persistent Datastore Menu) with 5 new options. Two batch jobs need to be submitted to complete this configuration. **Option 7** is to be used if you want to run Agent into CMS, otherwise skip this option. **Option 8** is the step to complete the configuration out of ICAT. Press PF3 to return to the Runtime Environment and choose the next **option L** (Load all product libraries), as you can see in Figure 4-26.

```
KCIPRTE ----------------- RUNTIME ENVIRONMENTS (RTEs) -------------------------
COMMAND ===>

Actions: A Add RTE, B Build libraries, C Configure,
         L Load all product libraries after SMP/E,
         D Delete, U Update, V View values, Z Utilities

Action Name     Type    Sharing  Description

  L    SC64SHR  SHARING BASERTE  LPAR-specifi for SC64 LPAR
Enter=Next  F1=Help  F3=Back  F7=Up  F8=Down
```

Figure 4-26 Runtime Environment panel

This job provides a batch job to load the runtime libraries from the SMP/E target libraries for this RTE.

Now, after all outside configuration has done, you can start the server of Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS and the SDSF panel shows the Server, CMS and Agent running. See Figure 4-27.

```
JOBNAME  StepName ProcStep JobID    Owner    C Pos DP Real Paging    SIO
  D8F2IRLM D8F2IRLM         STC12499 D8F2IRLM   NS  FE 6019   0.00   0.00
  D8F2MSTR D8F2MSTR IEFPROC  STC12498 D8F2MSTR   NS  FE 2465   0.00   0.00
  D8F2DBM1 D8F2DBM1 IEFPROC  STC12500 D8F2DBM1   NS  FE  18T   0.00   0.00
  D8F2DIST D8F2DIST IEFPROC  STC12501 D8F2DIST   NS  FE 2797   0.00   0.00
  D8F2DSST D8F2DSST CMS      STC12810 D8F2DSST   NS  F9 8780   0.00   0.00
  D8F2D5S  D8F2D5S  AGENT    STC12811 D8F2D5S    NS  F9 4888   0.00   0.00
  D8F2O2S  D8F2O2S  O2CI     STC12815 D8F2O2S    NS  F9 8572   0.00   0.00
```

Figure 4-27 SDSF panel

If you choose to use Performance Warehouse, the first time that server comes up, you see the following messages:

- FPEV5001I D8F2 PWH IS STARTING
- FPEV5011I D8F2 CREATION OF PLAN DB2PM WAS SUCCESSFUL
- FPEV5029I D8F2 CREATION OF DATABASE DB2PM WAS SUCCESSFUL
- FPEV5017I D8F2 BINDING DBRM DGO@PRV1
- FPEV5017I D8F2 BINDING DBRM DGO@MT1, etc... and finally...
- FPEV5005I D8F2 DATABASE UPDATE COMPLETE

In the sysprint file you can see all tables that were created in this database to bind all these DBRMs. In Figure 4-28 you see part of those messages.

```
FPEV5009I  D8F2 Creating new table PROCESSGROUP ...
FPEV5009I  D8F2  ... creating primary index
FPEV5009I  D8F2  ... creating unique indexes
FPEV5009I  D8F2  ... setting comments
FPEV5009I  D8F2 Creating new table PROCESS ...
FPEV5009I  D8F2  ... creating primary index
FPEV5009I  D8F2  ... creating unique indexes
FPEV5009I  D8F2  ... setting comments
FPEV5009I  D8F2 Creating new table PROCESSEVEXCP ...
FPEV5009I  D8F2  ... creating primary index
FPEV5009I  D8F2  ... creating unique indexes
FPEV5009I  D8F2  ... setting comments
```

Figure 4-28 Server SYSPRINT dataset

4.5.2 Future installations

Our scenario until now is composed of two LPARs, SC63 and SC64, both with more than one DB2. Here is the scenario:

- One FULLRTE for SC63 for DB2 DB8A. Only the server was created.
- One BASERTE and SC64SHR(sharing) for SC64 for DB2 D8F2. Here we created:
  - Server
  - Hub CMS
  - Portal
  - Browser

In the next step we will create one SC63SHR(sharing) for DB2 D8F1 in SC63 LPAR and use the BASERTE for SC64, and we will create one Remote CMS to connect with the Hub. This way the Portal can manage both LPARs. To create the new one RTE, you have to execute **options A** and **B** from Runtime Environment panel, as shown in Figure 4-9 on page 46. When you define REMOTE CMS in panel KDS36PP1, ICAT takes you to another panel (KCIPCMSS) for you to specify the Hub CMS that you want to connect. See Figure 4-29.

```
KCIPCMSS -------- COMMUNICATION SELECTION PANEL FOR SC63SHR    Row 1 to 1 of 1
COMMAND ===>

 Select the hub (primary CMS) to connect to this remote CMS.


 The following lists eligible Candle Management Servers
 (CMS) that you have configured.  Enter S next to the CMS to connect
 to this remote.  To manually enter the CMS information, press F5.

    CMS     RTE name     RTE
    Type    (mid-level)   Description
------------------------------------------------------------------------------
 s  HUB      SC64SHR      LPAR-specifi for SC64 LPAR
 F1=Help  F3=Back  F5=Advanced  F7=Up  F8=Down
```

Figure 4-29 Communication selection panel - CMS

After you select the Hub CMS and press Enter, KDS36PP1 panel displays, showing with which Hub CMS your CMS is connected now. Then you have to complete the configuration for this new sharing RTE.

To check if your agent is talking with the remote CMS, you have to find the following message in the agent started task: `Connecting to CMS SC63SHR:CMS`.

Our Tivoli OMEGAMON XE for DB2 Performance Expert on the z/OS environment is shown in Figure 4-30. You can have more than one server per LPAR. To do so, you need to set the hidden parameter AUTODETECT to NO as explained before.

We configured a full RTE for DB8A without TEMA and TEMS and a sharing RTE with TEMA and TEMS for the data sharing group.

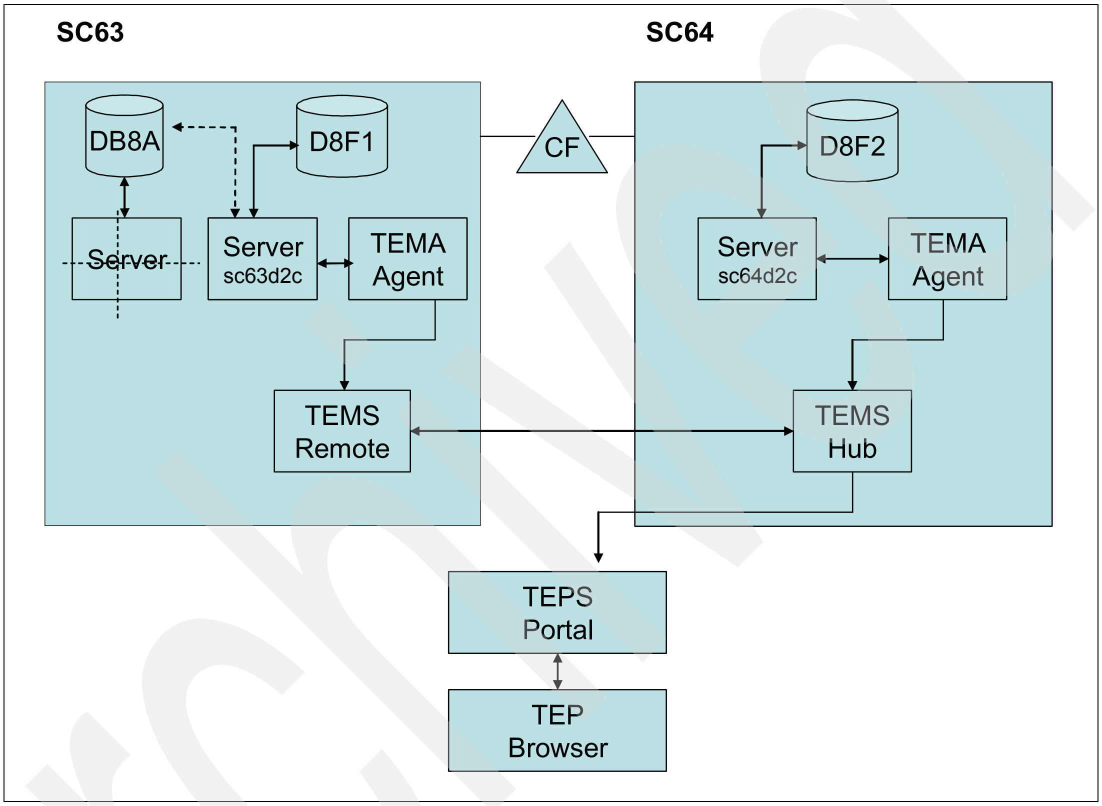


Figure 4-30 Our Tivoli OMEGAMON XE for DB2 Performance Expert on the z/OS environment

As shown by the dotted lines, you can use ICAT to add DB8A to the monitored subsystems of server sc63d2c. You could then remove the full RTE dedicated to DB8A. You would then have only one server for LPAR SC63, and all DB2 subsystems could be monitored via TEMA and TEMS.

4.5.3 Batch mode

The Batch processing mode is a simple way of replicating/cloning runtime environments (RTEs) to remote z/OS systems where there is no disk sharing with the environment in which the RTE was originally built. In this process you can use a single batch job to build and configure a new runtime environment. It uses the original parameter file to replicate an existing runtime environment. This function is recommended for usage for ICAT experts.

> **Attention:** Do not use batch processing to change a configuration, because a reload overrides any changes that have been made outside of ICAT.

To start this process, access ICAT interactive mode, executing the following command by **option 6** on ISPF:

EXEC ‘OMEGASYS.SC64.INSTLIB’

You need to use **option 3** (configure products) and then **option 2** (select product to configure) where you select your product. All RTEs defined are displayed as shown in Figure 4-31. Choose **option Z** to start the process.

```
------------------------- RUNTIME ENVIRONMENTS (RTEs) -------------------------
COMMAND ===>

Actions: A Add RTE, B Build libraries, C Configure,
         L Load all product libraries after SMP/E,
         D Delete, U Update, V View values, Z Utilities

Action Name     Type    Sharing  Description

       BASERTE  BASE             BASE RTE for both LPARs ( SC63, SC64)
  Z    SC64SHR  SHARING BASERTE  LPAR-specifi for SC64 LPAR
Enter=Next  F1=Help  F3=Back  F7=Up  F8=Down
```

Figure 4-31 Runtime Environments panel

As the first step, you need to alter the keys option of your session writing **KEYS** on your command line and changing the number of PF keys to 24, because batch mode can be built using PF keys or commands, and the customization batch job needs more than 12 PF keys.

In **option 1** you define the dataset and the member name, and by pressing Enter, this member is created with all definitions of source RTE. You only have to use **options 2** and **3** if you had defined z/OS system variable support in the original RTE, otherwise you receive the message on the right top of the screen: `SYSTEM VARIABLE ERROR`, and by pressing PF1, the message is: `z/OS System Variable support is not enabled for this RTE=SC64RTE`. In **option 5** you create some sample transport jobs that will be stored in the RKANSAM dataset and extra two members that contain the recommended scenarios for remote systems called XCNFG#01 (typical 1 system/1 RTE set up) and XCNFG#02 (remote Installation tool set-up). In Table 4-7 you see the name and explanation about each job generated in this option.

Table 4-7 Members in RKANSAM dataset generated by option 5 in KCIMRTU panel

MEMBER	Explanation	Scenario
XDFDMP01	DFDSS tape job will execute a dataset DUMP from an RTE at local to a remote site at initial build distribution.	XCNFG#01
XDFRST01	The DFDSS tape job will execute a dataset RESTORE from an RTE at local to a remote site at initial build distribution.	XCNFG#01
XDFDMP02	The DFDSS tape job will execute a dataset DUMP from an RTE at local to a remote site at maintenance distribution. Probably you need to exclude some datasets.	XCNFG#01
XDFRST02	The DFDSS tape job will execute a dataset RESTORE from an RTE at local to a remote site at maintenance distribution.	XCNFG#01
XDFDMP03	The DFDSS tape job will execute a dataset DUMP of the target libraries and installation tool dataset to a remote site at initial build distribution.	XCNFG#02

MEMBER	Explanation	Scenario
XDFRST03	The DFDSS tape job will execute a dataset RESTORE of the target libraries and installation tool dataset to a remote site at initial build distribution.	XCNFG#02

Option 6 creates KCISYPJP job in RKANSAM to copy procedures to SYS1.PROCLIB, and finally, **option 7** creates KCISYNJB job in RKANSAM to copy VTAM definitions to SYS1.VTAMLST. After using this option, you need to run jobs XDFDMP01 or XDFDMP02 or XDFDMP03 to generate the tape DUMP dataset.

4.6 Summary of ICAT installation

To summarize the installation options, we create an installation checklist containing started tasks needed, new userids, VTAM definitions, etc. See Table 4-8.

Table 4-8 ICAT installation checklist

Element	Value	Comments	Where obtained
Base RTE HLQ	OMEGASYS.SC64	The HLQ for all of the Base RTE datasets.	Picked during ICAT
Middle level qualifier	BASE	Middle node for the Base RTE datasets. Example: OMEGASYS.SC64.BASE.RKANMOD	Picked during ICAT
Sharing RTE HLQ	OMEGASYS		Picked during ICAT
Middle level qualifier	SC64SHR	Middle node for the Sharing RTE datasets. Example: OMEGASYS.SC64.SC64SHR.RKD2PAR	Picked during ICAT
STC prefix	D8F2	Determines the name of STCs generated. We chose D8F2 so the generated names would be the same as the DB2 member running on this LPAR. Example: D8F2MSTR,IRLM,DBM1	Picked during ICAT
VTAM Applid prefix	SC64	Prefix is your choice, suffix is generated by ICAT. Might present some naming standard issues at a customer site. For example, we are generating SC64D2C, SC64 is the LPAR SMF ID.	Picked during ICAT
TCP/IP Host name	WTSC64	Host name, used to define IP:PIPE in TEMS and TEPS - Hometest command.	From LPAR TCP/IP parms
Domain Name	WTSC64.ITSO.IBM.COM	This is the TCP/IP domain name for the MVS LPAR where the server runs. This can be found by looking at one of the DB2s running on the same LPAR.	From LPAR TCP/IP parms
TCP/IP MVS name	TCPIP	TCPIP MVS Started task name - one per LPAR.	TCPIP Sysprog
IP Address	9.12.6.9	This is the IP address of the TCPIP MVS STC running on the LPAR where the server runs.	From LPAR TCP/IP parms
Listener Ports		There are three listener ports needed. Some customers block unassigned ports, the TCP/IP sysprog needs to assign specific listener ports, others allow open port assignment. NETSTAT command shows current IP ports in use on any LPAR - RACF ID needs OMVS Segment to execute this command.	

Element	Value	Comments	Where obtained
Port for TEMS IP Port for TEMS IP:PIPE Port of OMEGAMON Server PE Listener	1995 1996 5100	Needs to be 1910 or greater Needs to be 1910 or greater Default	Obtain from TCPIP Sysprog
Started Tasks: OMEGAMON Server Tivoli Enterprise Management Server Tivoli Agent Address Space OMEGAMON Performance Warehouse Persistent Data Store Maintenance Procedure Persistent Data Store Maintenance Executing JCL	D8F2O2S D8F2DSST D8F2D5S D8F2PWH KPDPROC1 KPDPROC2	These all need copied from RKANSAM/RKD2SAM to SYS1.PROCLIB or equivalent We use the same RACF ID as the D8F2O2C stc	Generated from ICAT
USERIDs OMEGAMON Server Tivoli Enterprise Management Server Tivoli Agent Address Space	D8F2O2S D8F2DSST D8F2D5S	We created these ID's, connected the group DB2PM to then, and associated each to the above STCs via the RACF started Task Table. The DB2PM group needs to have UID(0) Open Edition Authority.	Created by RACF admin
APF Authorized libs Base RTE loadlibs	OMEGASYS.SC64.BASE.RKANMOD OMEGASYS.SC64.BASE.RKANMODL	Notice the names were constructed by our choice of RTE HLQ and middle nodes	Created by ICAT
VTAM Definition Logmode table entry MAJOR Node- TEMS MAJOR Node- Agent MAJOR Node - OMEGAMON Server	KDSMTABL SC64DSN SC64D5N SC64D2N2	Default first 4 characters defined by ICAT setting of VTAM Applid prefix, last characters are defined by product and should not be changed.	Created by ICAT

4.7 Portal installation

Once the Server, Agent (TEMA), and CMS (TEMS) are running, now it is time to install the Portal (TEPS) to control all your environment.

During our installation of the TEPS, we initially installed the server component first. When we subsequently attempted to rerun the InstallShield to install the workstation TEP client, it uninstalled the server component. To get around this — and we believe this is a more typical installation scenario — we installed the server on one machine, called the TEPS server machine, and then installed the client TEP code on our other workstations. We recommend a similar installation approach, but if both TEPS server and client are to be installed on the same machine, then both products should be installed together.

The TEPS installation is managed via a standard Windows InstallShield. This is located in the WINDOWS directory of the product files, which was obtained either via FTP download, or on the CD that was shipped as part of the Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS installation package, this CD is LCD7-0707-01. It is a file called SETUP.exe. You can either click this file to launch, or run it from the Windows RUN dialog. The first screen is shown in Figure 4-32.

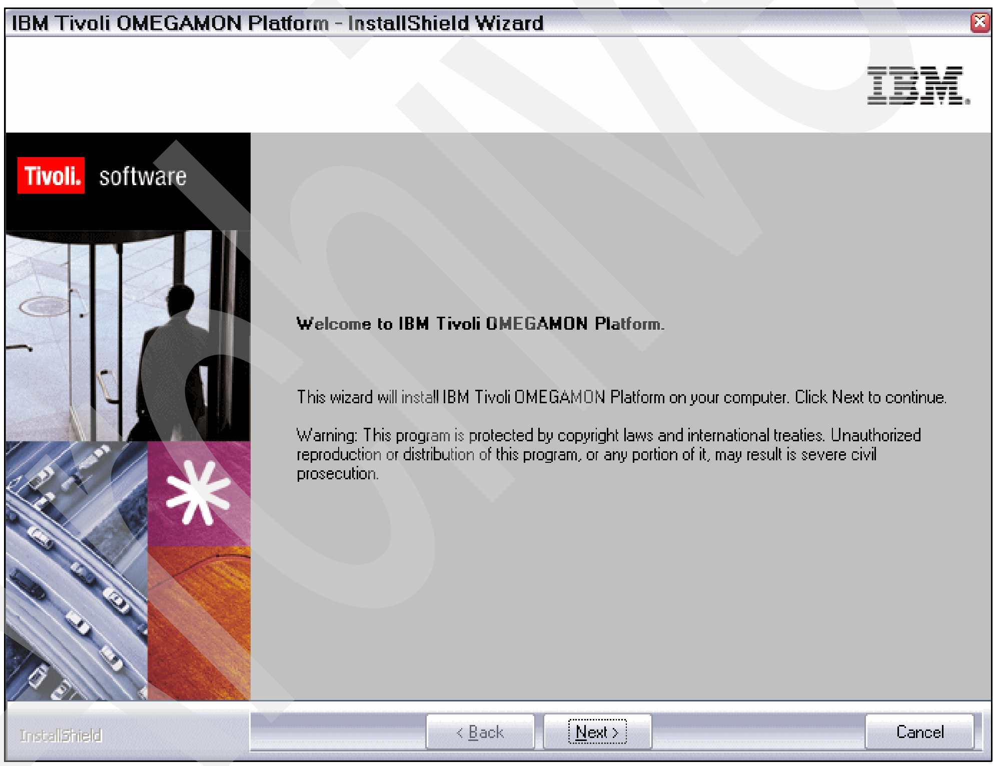


Figure 4-32 Installation of Portal panel

Click **Next** and you can set up the Portal destination/location, shown in Figure 4-33.

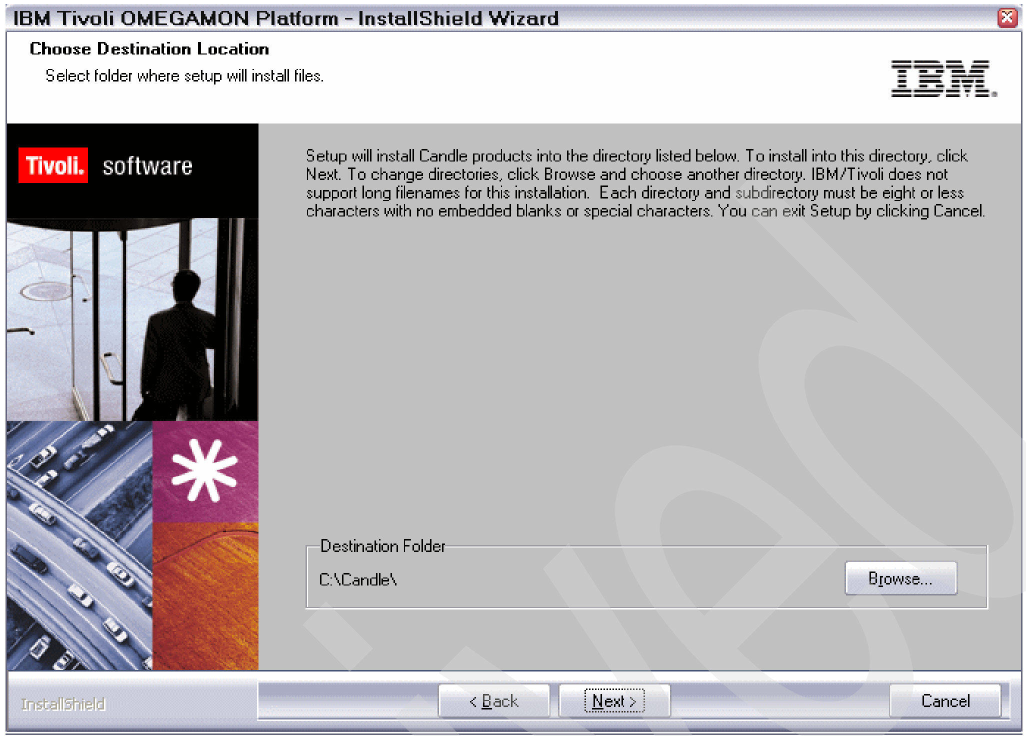


Figure 4-33 Destination / Location panel

Click **Next** and the feature CandleNetPortal Server box needs to be selected. At this moment the latest Sun™ Microsystems™ Java Runtime Environment is a prerequisite. The URL where the latest version of the JRE can be downloaded is shown on this panel. See Figure 4-34.

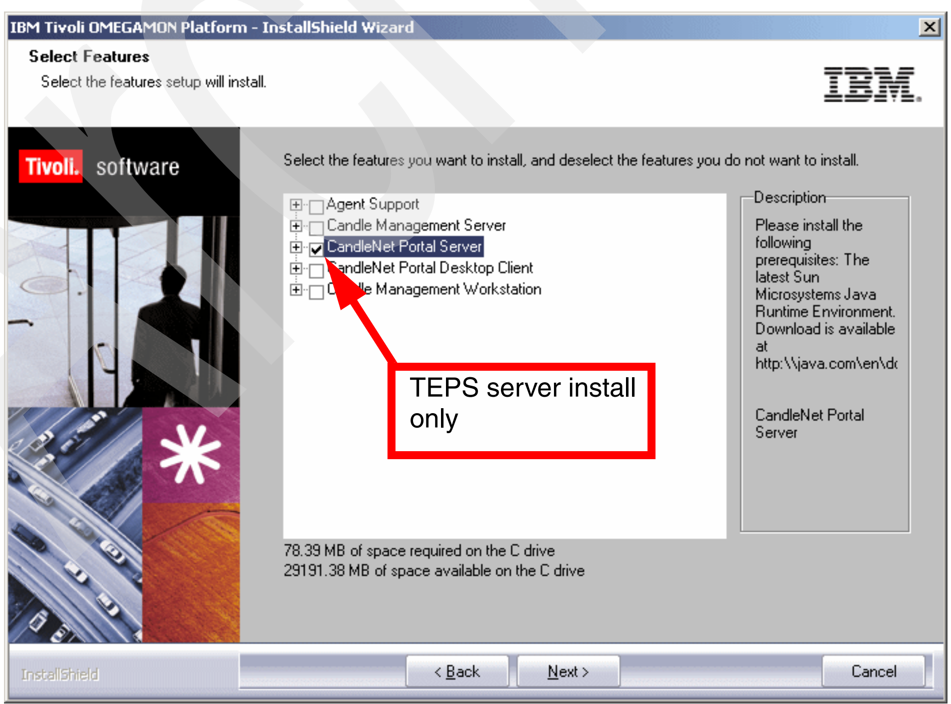


Figure 4-34 Features panel

Click **Next** and the setup process will add program icons to the program folder listed in the panel. Figure 4-35 shows an example of this information.

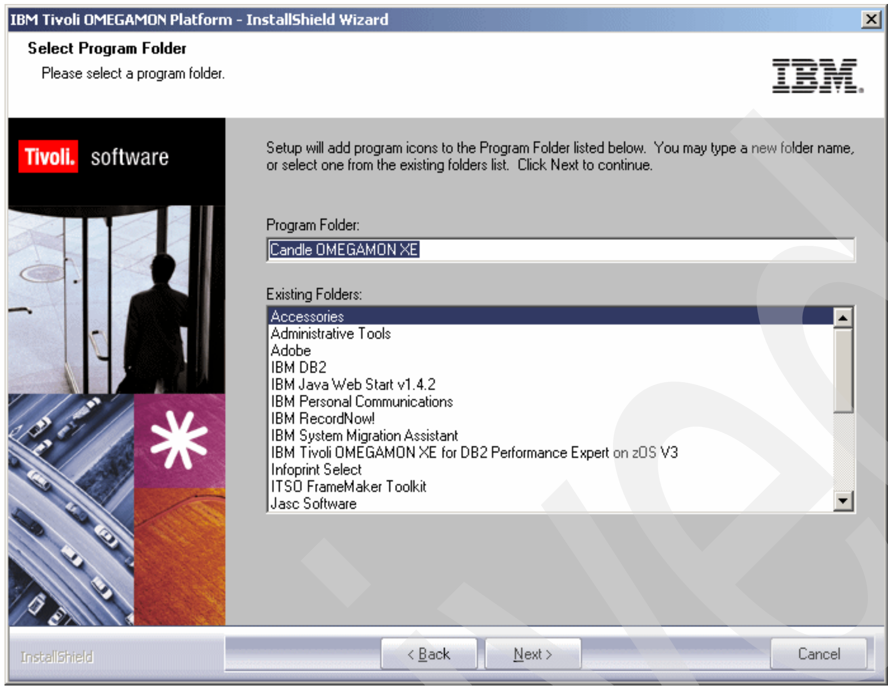


Figure 4-35 Select Program Folder panel

Clicking through the InstallShield will start the installation where the different TEPS files are copied into different installation folders. Once the files have been copied, the installation shield prompts for a list of products to be configured. In our example, we elect to both configure the TEPS as well as launch the Manage Candle Services panel for additional configuration options. This is shown in Figure 4-36.

Figure 4-36 CNP Configuration panel

Next, the TEPS host information needs to be entered. This consists of either the IP address of the Server machine or in the case where the InstallShield is being executed on the local server machine, which is the true for our configuration, the name of the PC or server machine. See Figure 4-37.

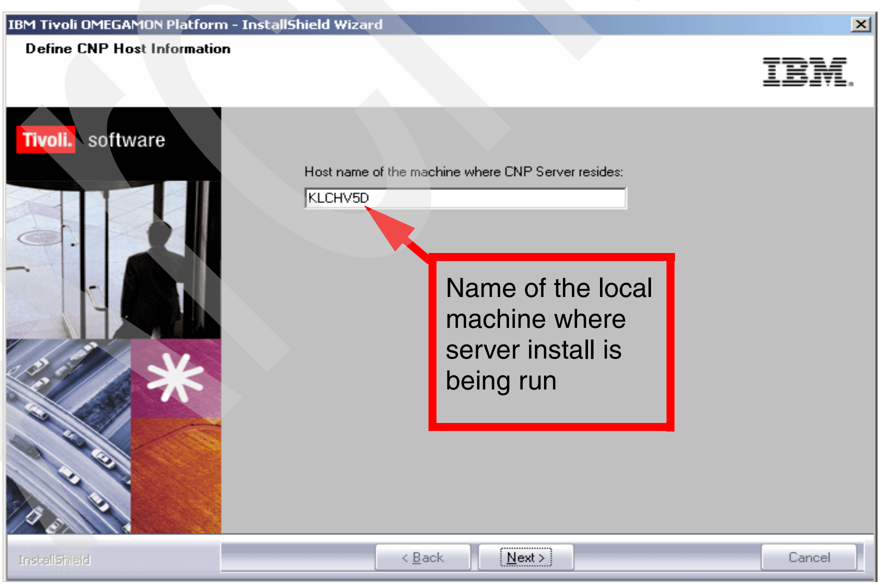


Figure 4-37 CNP Host Information panel

The TEPS requires that a local DB2 UDB instance be installed and active. The TEPS creates and catalogs tables into this instance used to configure and manage the TEPS. A valid DB2 UDB database administration ID and password combination need to be provided. This ID needs to have DBADMIN administration authorities to the local UDB instance. In addition, a TEPS (CNPS) userid and password are specified and registered in the TEPS control tables. We elected to take the default CNPS names and password. Figure 4-38 shows an example of this information.

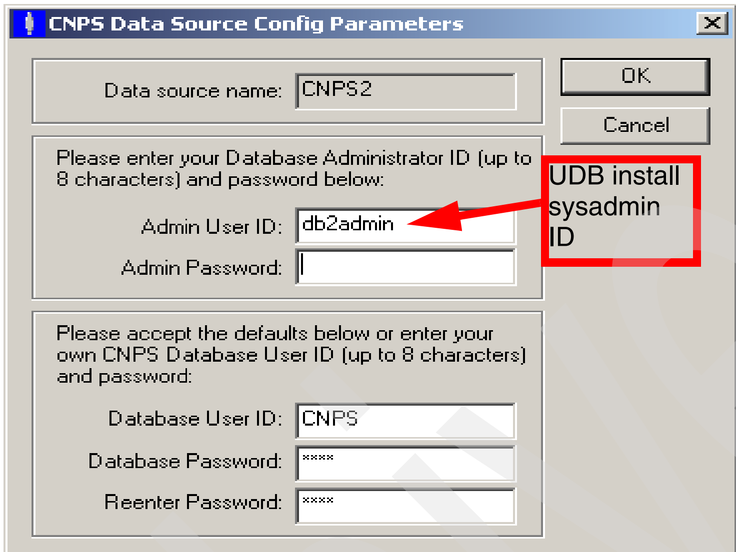


Figure 4-38 CNPS Data Source Configuration panel

Figure 4-39 shows the confirmation of the success of the CNPS configuration. In our installation, it took several minutes to define the necessary UDB tables and populate the database with data.

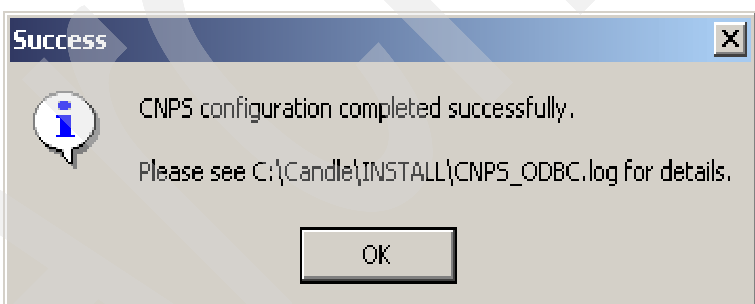


Figure 4-39 Configuration completed successfully panel

Once the TEPS has been configured, the connection information to be used for communication between the TEPS and the Hub TEMS needs to be defined. During ICAT configuration, we declared an IP port for use by the TEPS. The TEMS will open a listener on that port, and the TEPS needs to be configured to use the same port. There can be three different types of connections defined; the recommended implementation is to use the IP:PIPE connection for the best performance. In order to ensure that we used IP:PIPE, we elected not to configure the other two types of connections, SNA and IP. This choice was made on both the ICAT definitions as well as the TEPS configuration, which is shown in Figure 4-40.

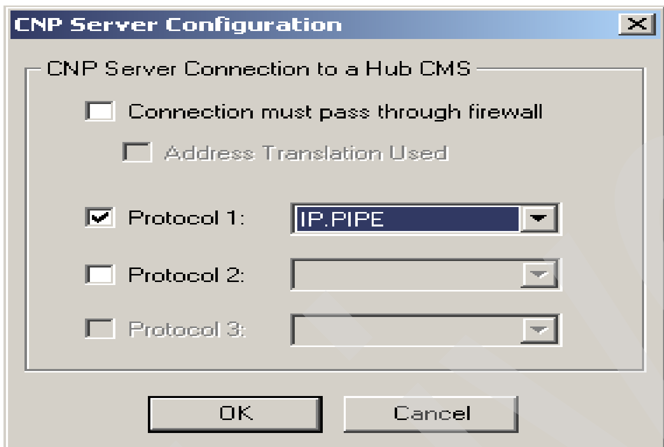


Figure 4-40 Protocol Definition panel

For the IP:PIPE definition, we need to specify the settings for IP address and IP Port Number that match our ICAT specifications for the TEMS. This is shown in Figure 4-41.

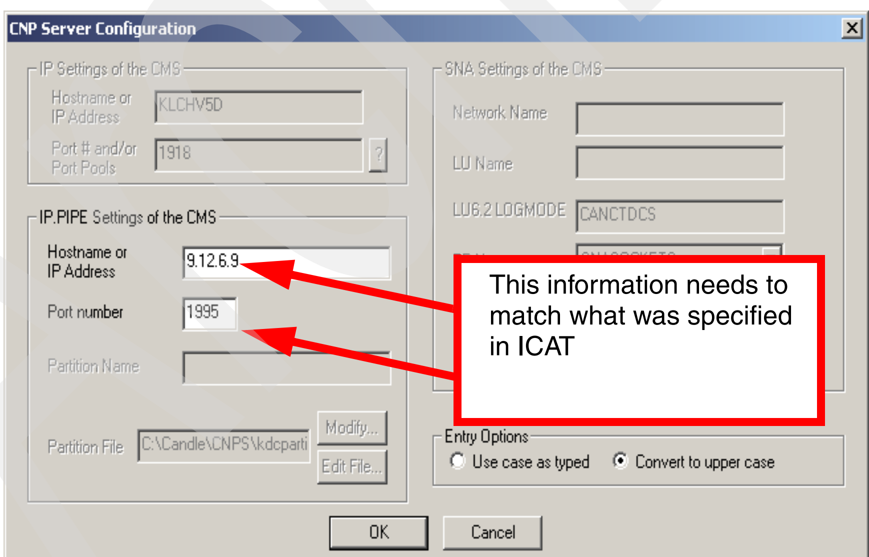


Figure 4-41 Host IP address and Port number panel

Click **OK** and the setup is completed. Click **Finish** on the next panel and the Manage Candle Services panel is displayed. See Figure 4-42.

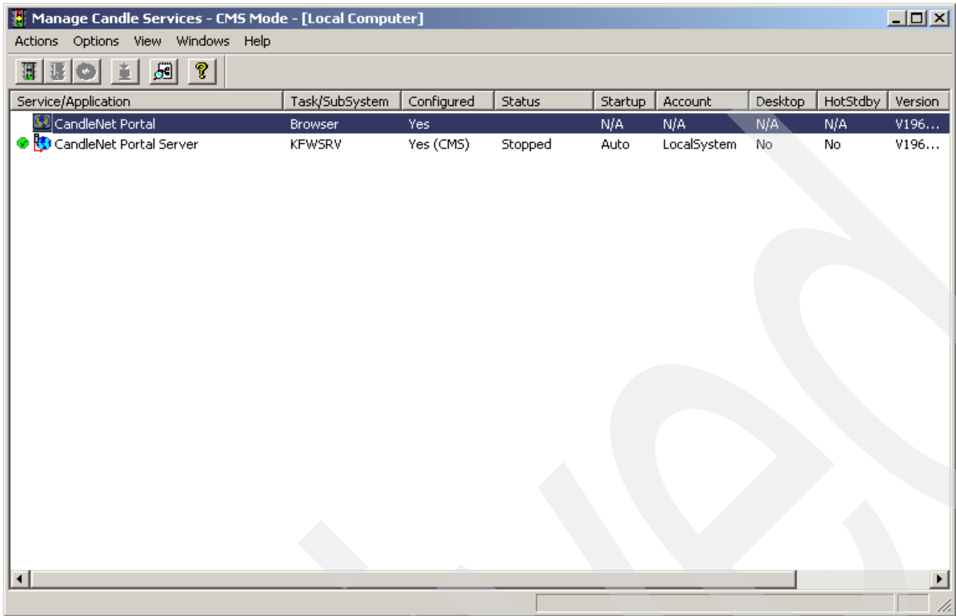


Figure 4-42 Manage Candle Services panel

4.8 Installation of z/OS data files

Once the TEPS has been installed, the z/OS data files component needs to be added. This contains information and situation data, which is sent ("seeded") from the TEPS to the TEMS address space.

As with the TEPS installation, the z/OS data files installation is managed via a standard InstallShield. This is located in the WINDOWS directory of the product files, obtained either via download or on the CD. It is a file called SETUP.exe. You can either click this file to launch, or run from the Windows RUN dialog, shown in Figure 4-43.

Figure 4-43 InstallShield - Welcome

Clicking through the first several panels of the InstallShield, you will be presented with a screen similar to that shown in Figure 4-44. The navigation tree will be collapsed, by clicking the **+** entry, the options will be exploded as shown in Figure 4-44.

> **Important:** Only select the data files for the DB2 for z/OS product. This is particularly important when installing into an existing Tivoli Enterprise Portal instance. If you select other products, you run the risk of inadvertently changing the seed data for the other components managed by your TEMS.

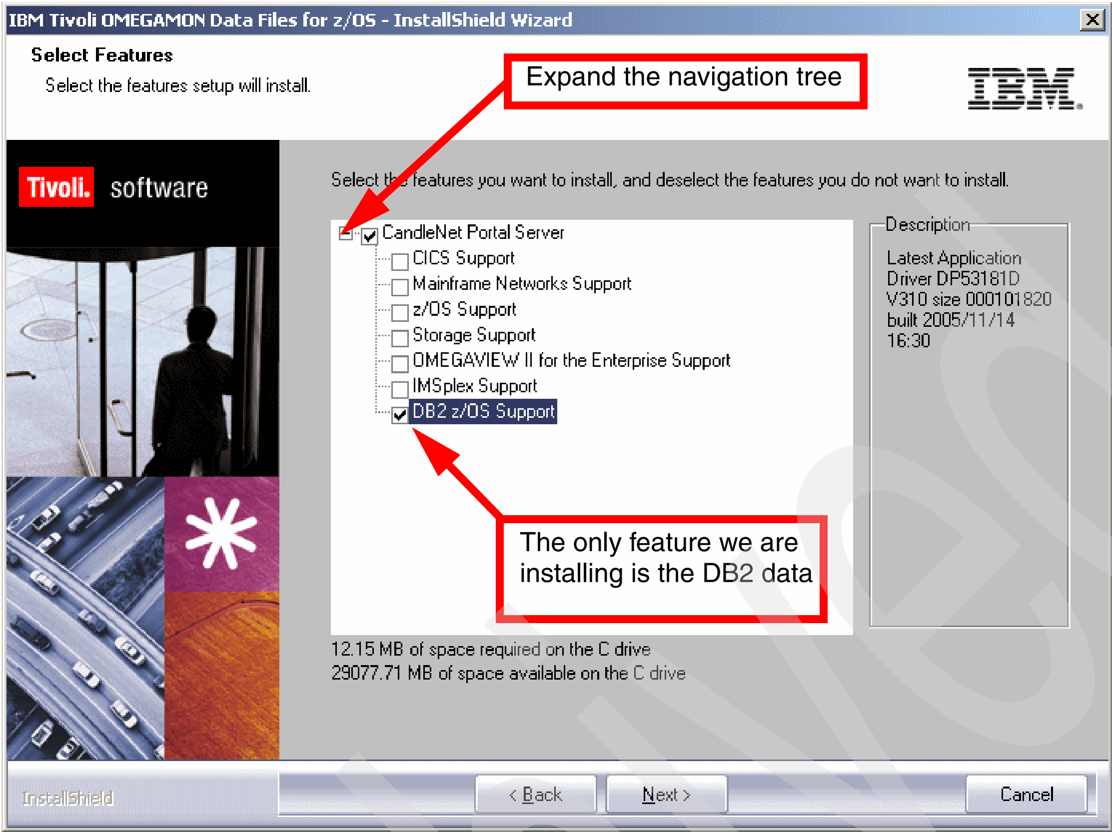


Figure 4-44 Data files - Feature selection

One we have selected the data files necessary, we then click through the confirmation panel, which just shows us the summary of what data is being installed. We then are shown the setup type panel. On this panel, the default settings have the check boxes both selected by default. Accept the defaults as shown in Figure 4-45 and click **Next**.

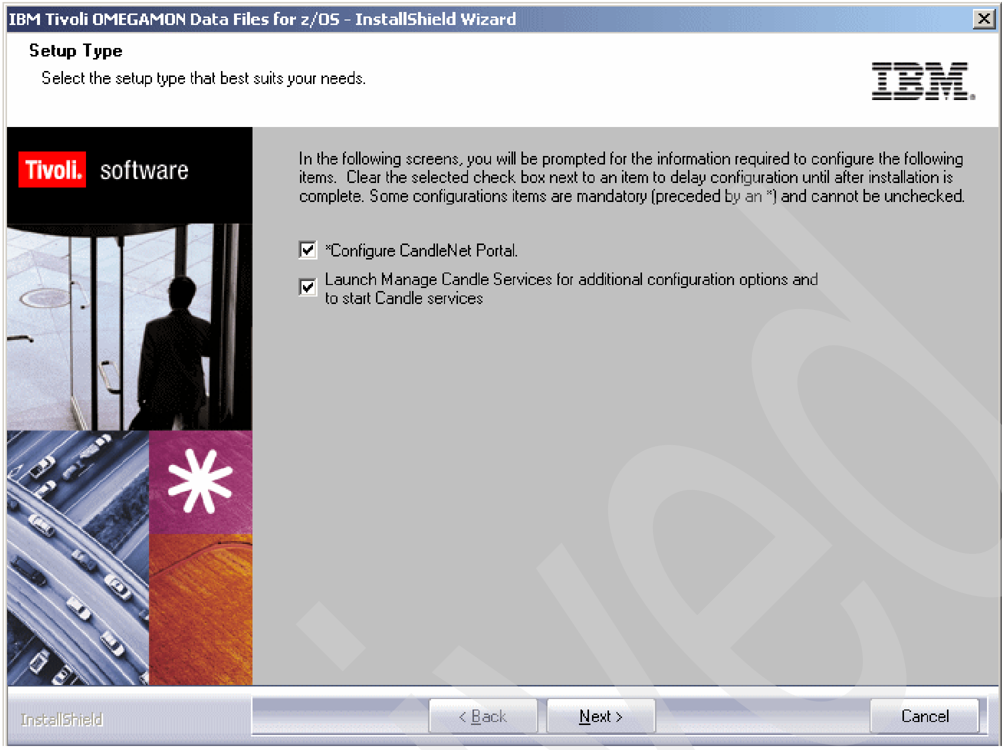


Figure 4-45 Setup type selection

The next panel shows where we specify the location of the TEPS server instance. In our scenario we are running the InstallShield on the same Windows machine as the TEPS server (which should be typical for most installations). The value in our example relates to the name of the workstation; for a remote installation; this value would be the IP address of the machine that is hosting the TEPS instance. This is shown in Figure 4-46.

Once specified, then click **Next**.

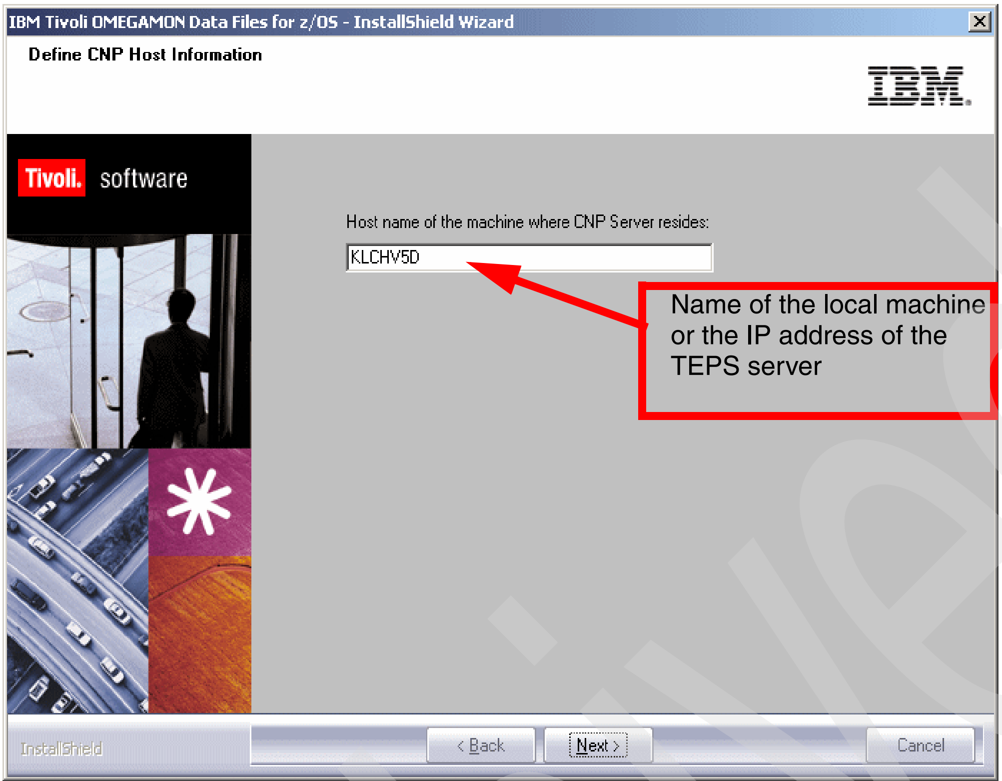


Figure 4-46 TEPS Host information

The InstallShield will then install the z/OS data onto the TEPS server. When finished, the InstallShield will show an installation complete panel, along with an invitation to review the README text document. Click through to complete the installation.

4.9 Seeding the TEMS

Once the z/OS data has been installed, this data is then used to *seed* the TEMS. Seeding is the process of transferring the z/OS data from the TEPS to the TEMS running on the z/OS LPAR. In our case there is both a *Hub* TEMS and a *Remote* TEMS in our configuration. Seeding is always performed with the Hub TEMS. Once updated, the Hub TEMS then coordinates and updates any necessary seed data with any connected Remote TEMS as well as with the associated agents running in the TEMA.

To start the seed process, verify that the TEPS is active. Next, open the Candle Management Services - CMS mode window on the TEPS server machine. This is shown in Figure 4-47.

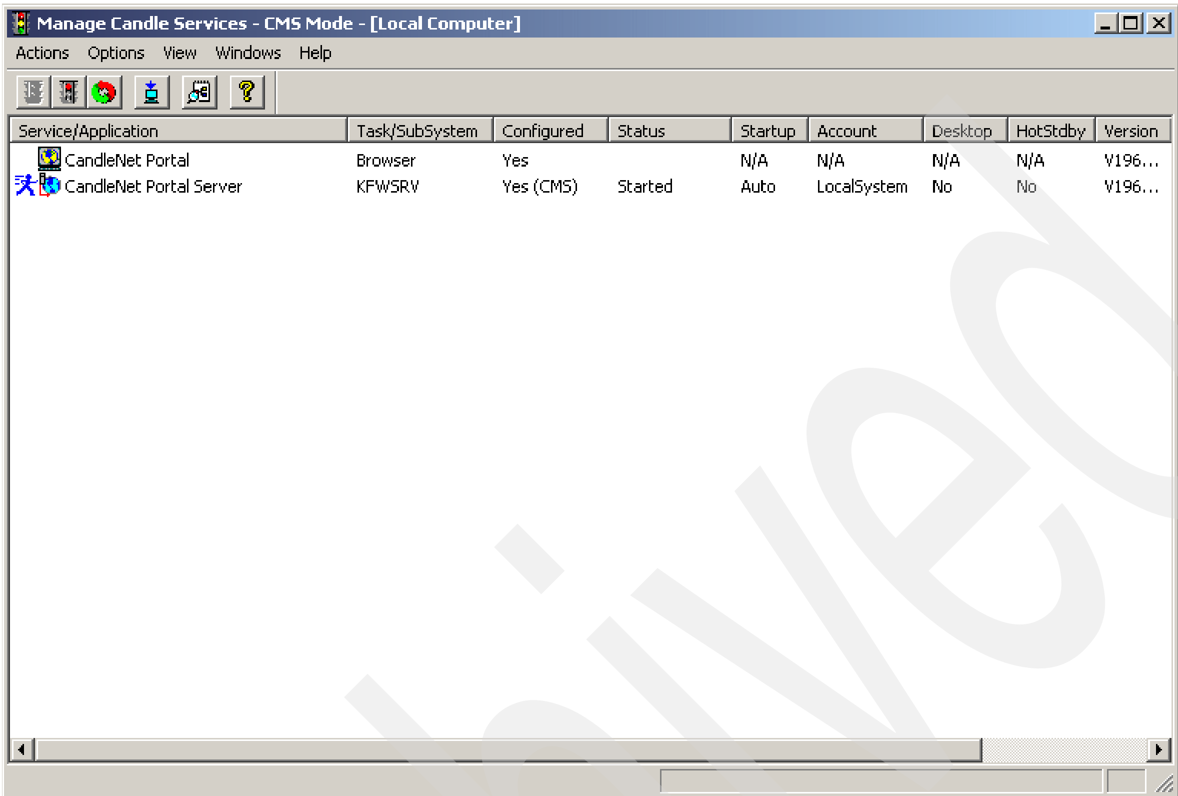


Figure 4-47 Manage Candle Services - CMS

Navigating using the action bar, we then select the seed CMS option shown as follows in Figure 4-48. Remember, the use of the term CMS should be viewed as TEMS instead.

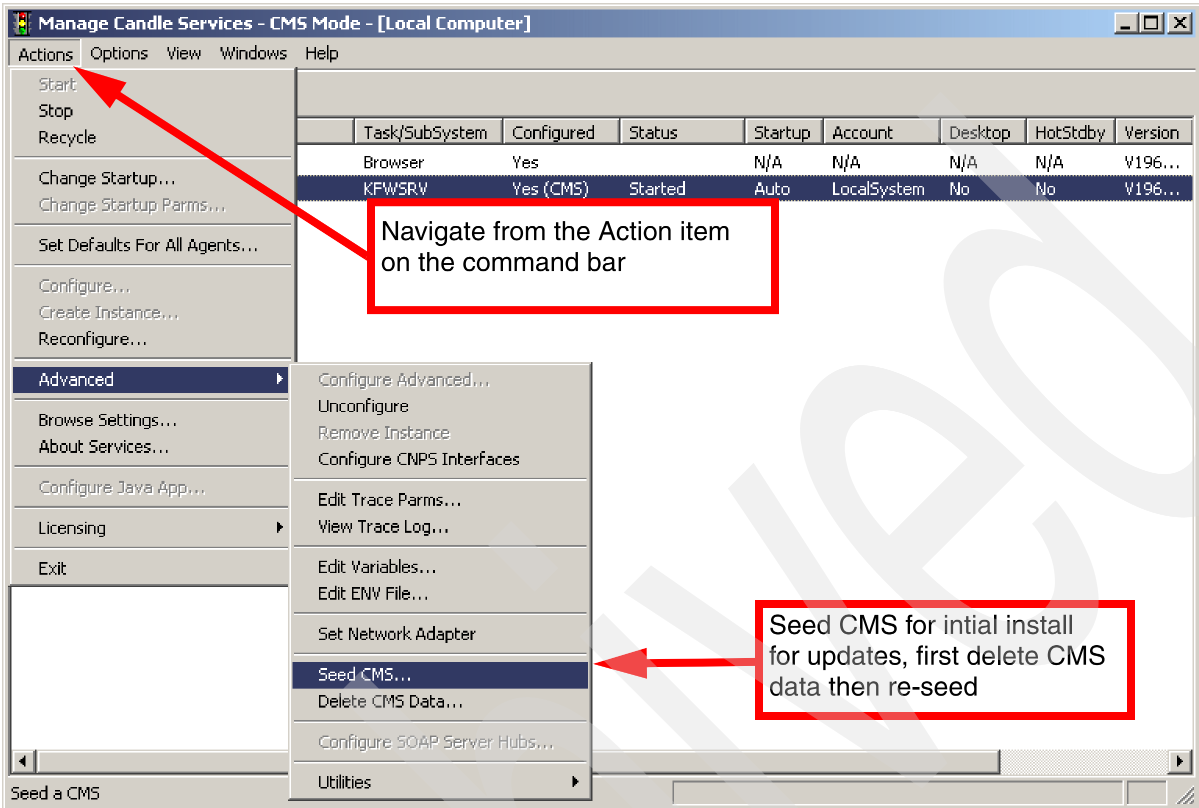


Figure 4-48 CMS Mode - Select Seed option

Once we select the Seed CMS option, we need to verify the TEMS location. In our configuration, we are running the TEMS on z/OS, so the TEMS is located on a different computer than the TEPS, as indicated in the answer noted in Figure 4-49.

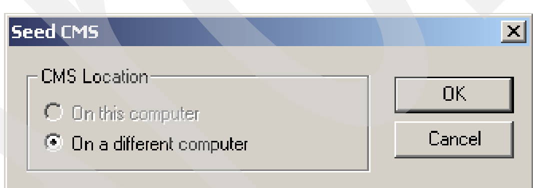


Figure 4-49 Seed CMS - Location of TEMS

Click through this panel and the next one, which is just a reminder to make sure that the TEMS is active and running. The next panel describes the seeding process, and how to connect to the TEMS. This is the same configuration performed earlier in this chapter, but it needs to be specified a second time. The CMS (TEMS) Node ID needs to match the value of the CMS (TEMS) that was defined during the ICAT configuration process. In our scenario, we elected to configure only TCP/IP PIPE, which is the recommended mechanism of connecting the TEMS and the TEPS, performance being best when using this approach. Figure 4-50 shows this panel and the values for our TEMS node ID.

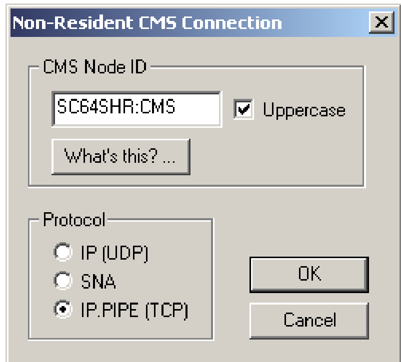


Figure 4-50 CMS (TEMS) Node specification

The IP address and the port number chosen during the ICAT configuration process must also be specified. These values must match, as the TEPS uses these to send the data to the TEMS via the declared IP listener port. Figure 4-51 shows our values, which match the values specified for our z/OS based TEMS.

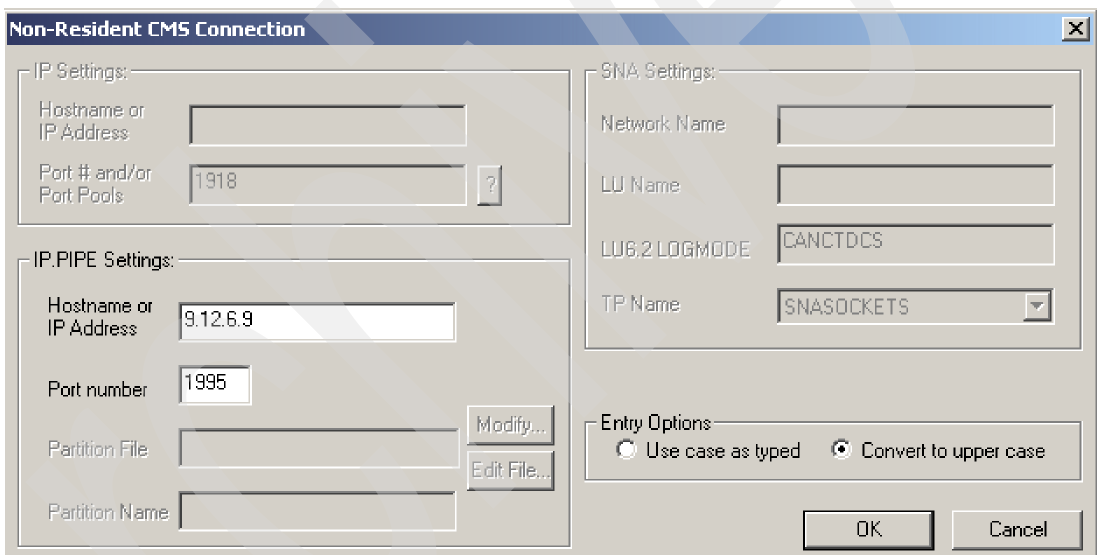


Figure 4-51 IP Configuration for TEMS

The next step is to identify what data needs to be seeded to the TEMS. In our case, the only data we choose to seed is the DB2 for z/OS data. If you have installed into an existing TEPS (CNPS), there would be other types of data presented in the pick list. Again, make sure you only select the data type for the request seeding operation. This is shown in Figure 4-52.

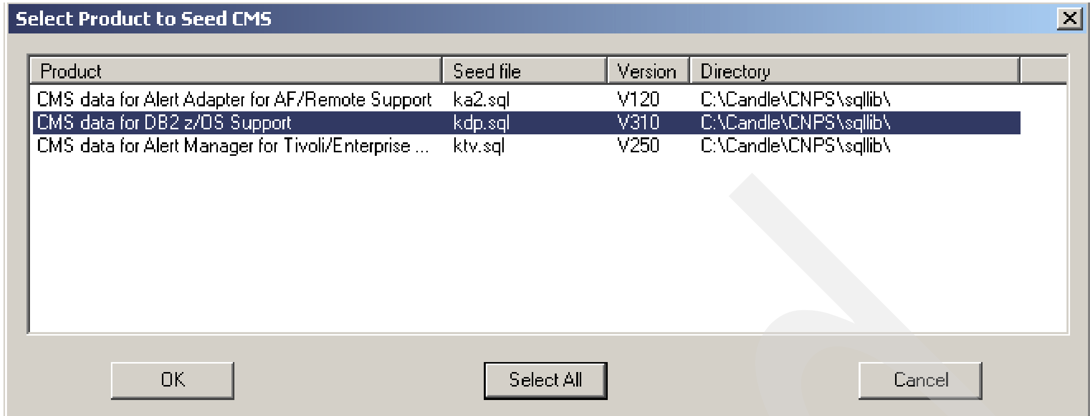


Figure 4-52 Select product data for seed

Once the seed operation is completed, there will be a status window opened up. In our testing we saw either return code of 0, which indicated a successful seeding, or a return code of 172, which indicated a problem with the connection from TEPS to TEMS. In all cases, we discovered that this was symptomatic of a definition error, either on the TEPS definition, or an incorrect specification of IP address in ICAT on the TEMS configuration.

Figure 4-53 shows an example of a successful seed request. Also note that the location of the TEPS log file is shown, in the event that further diagnosis of a failed seeding operation is needed.

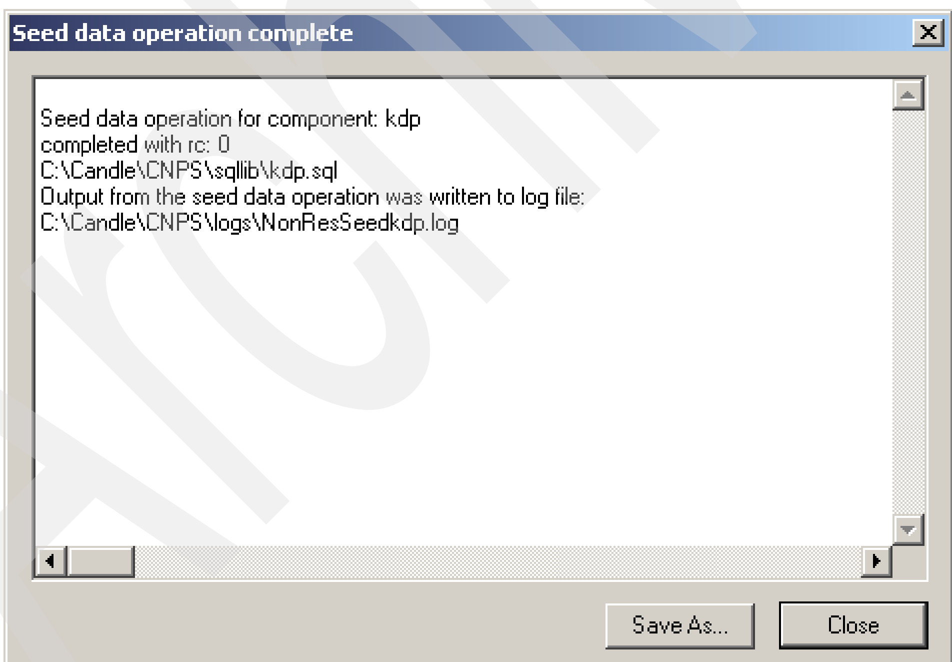


Figure 4-53 TEMS seed completed

Click through this panel and the reminder to recycle the remote TEMS will appear. Click through this to complete the installation. You should recycle both the TEMS and the TEMA address spaces on both the Hub and Remote side to ensure that the updated z/OS data is distributed.

For more details about the Portal, please refer to Chapter 7, “Portal interface” on page 101.

5

Security considerations

In this chapter we introduce the concepts of internal and external security for the Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS Classic Interface.

OMEGAMON XE for DB2 PE provides an Internal Security system for the Classic Interface. You can also set up an interface between the Classic Interface and an External Security package, such as RACF or ACF2. By default, the product is shipped with the External Security feature disabled.

We discuss the following topics:

- Relogon feature
- Why it is important to configure External Security
- Customizing External Security for the Classic Interface
- Relogon feature

5.1 Internal Security

The Internal Security feature that comes with Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS is, in our opinion, inadequate for production use, since it is unable to distinguish between specific users' authority. There are four levels of security, LEVEL0, LEVEL 1, LEVEL 2, and LEVEL 3. The LEVEL 3 is associated to authorized commands, which are the most sensitive and powerful commands. IBM ships all commands with a default security level of 0, except for authorized commands, which have a security level of 3. By supplying a default password, you can execute all commands for LEVEL3, independently of your userid.

To get access to the authorized commands, use the `/PWD` command in the INFO-line on the top of left side. Press Enter, and after that, you need to enter the password. The message `PASSWORD ACCEPTED` is displayed.

> **Attention:** We do not discuss the actual values of the passwords used for Internal Security in this book. If needed, the default values can be found in RKD2SAM.

5.2 Why it is important to configure External Security

When implementing the standard default security, anyone who knows the LEVEL 3 password can execute authorized commands. There is no association between the user executing the command and the command execution, so little or no auditing is possible. Of greater concern is the fact that the values of the LEVEL 3 password are well known among the OMEGAMON community and running with Internal Security could constitute a security exposure. With External Security enabled, each user of Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS is required to have a valid RACF userid on the system where the Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS server is executing and must provide a valid RACF authenticated userid and password combination.

Once validated, with the authorized Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS commands also protected by RACF, the degree of control on these commands can be very granular and unique for each Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS user. In addition, with the commands secured by RACF, the administration of the authorized command security can be performed by the RACF security administration group, removing the DBA from the security administration process, and therefore reducing one potential area of security exposure. This can be very important from a security audit perspective.

> **Important: We strongly recommend the use of External Security.**

5.3 Customizing External Security for the Classic Interface

By default, the product is shipped without the External Security feature. To enable this feature is a relatively simple process. The following components make up the External Security feature:

- One sample security table
- One sample exit routine
- One sample class definition for RACF or MUSASS attribute for started task for ACF2 or TSS Permit for specific user for e Trust® CA-Top Secret Security

In our scenario we use the RACF security package, and the following steps show what was necessary to implement the External Security feature.

5.3.1 RACF class definition

The RACF class definition is coded in the sample exit routine as described in "Exit routine" on page 79. The default RACF class name is OMCANDLE. We will choose another class definition for our scenario.

In order to show how to define a new definition using the ICHERCDE macro definition, we define a new class called O2OMPE, as shown in Example 5-1. It must be 8 characters, padded with blanks if necessary.

Example 5-1 IECHERCDE definition

```
ICHERCDE CLASS=O2OMPE,
               ID=200,
               MAXLNTH=8,
               FIRST=ALPHANUM,
               OTHER=ANY,
               POSIT=20,
               DFTUACC=NONE
```

We can check if the new class definition has been activated by using the command:

```
TSO SETR LIST
```

Look in the ACTIVE CLASSES section of the command output for new class. For this and other RACF commands discussed in this section, please refer to *z/OS V1R7.0 Security Server RACF Command Language Reference,* SA22-7687.

To permit all users to sign on to the Classic Interface, we have to define the INITIAL resource with the following RACF command:

```
TSO RDEFINE O2OMPE INITIAL UACC(read)
```

If we wanted to restrict access to the product to specific RACF ids, we would change the UACC from read to none, and then PERMIT specific group IDs to the INITIAL with UACC(read). In our environment, we chose to allow everyone access to INITIAL.

Important: All RACF changes, in order to be activated, need to be refreshed by the RACF command:

```
SETR RACL(O2OMPE) REFRESH
```

5.3.2 Exit routine

The hlq.RKD2SAM contains sample members for the exit routines: KO2RACFX for RACF and KO2ACF2X for ACF2. Many installations use these models without modification. In KO2RACFX, the name of RACF class is shown in Example 5-2. You have to change the resource class name that you defined in Example 5-1.

Example 5-2 RACF class change in exit

```
*        MVC   U#CHCLSD,=CL8'OMCANDLE'    RESOURCE CLASS NAME
         MVC   U#CHCLSD,=CL8'O2OMPE  '    RESOURCE CLASS NAME
```

In our scenario, we changed OMCANDLE to O2OMPE, which is our RACF resource class name, so we also changed our exit as shown above. Even if the default resource class name is used, the exit still needs to be assembled and link-edited, there is no default module in hlq.RKD2MOD.

After the sample has been updated, it needs to be assembled and link-edited. The member KO2RACFA in hlq.RKD2SAM contains the JCL needed to perform this task. The MACLIB and MODGEN datasets names should to be modified as shown in Example 5-3 to reflect the HLQ selected during configuration and customization. The Exit routine will be link-edited into the hlq.RKANMOD APF authorized dataset.

Example 5-3 Assemble and link-edit JCL

```
//ASMLKED  EXEC HLASMCL,PARM.L='RENT,NOXREF,LET,NCAL,MAP'
//*
//C.SYSLIB  DD  DISP=SHR,DSN=OMEGAXE3.TKANMAC
//          DD  DISP=SHR,DSN=SYS1.MACLIB
//          DD  DISP=SHR,DSN=SYS1.MODGEN
//C.SYSPRINT DD SYSOUT=*
//C.SYSIN   DD  DISP=SHR,
//          DSN=OMEGASYS.DB8A.SC63RTE.RKD2SAM(KO2RACFX)
//L.SYSLMOD DD  DISP=SHR,
//          DSN=OMEGASYS.DB8A.SC63RTE.RKANMOD(KO2RACFX)
//L.SYSPRINT DD SYSOUT=*
```

We strongly suggest that the two datasets hlq.RKD2SAM and hlq.RKANMOD be protected with sufficient RACF authority to prevent inadvertent or unauthorized updates.

5.3.3 Security table

The last step consists of modifying the sample security table. This is located in the hlq.KO2SUPDI member contained in the hlq.RKD2SAM dataset. In our case we modified only a few sample commands as shown in Example 5-4. Additional changes include:

1. The parameter External=YES needs to be inserted into this member, this enables External Security.
2. The module name is required and is the module name that was assembled and link-edited in Example 5-3.
3. Change the dataset name specified in the AUTHLIB parameter to reflect the correct HLQ for the RTE.

Example 5-4 Changes in security table

```
EXTERNAL=YES
MODULE=KO2RACFX
AUTHLIB=OMEGASYS.DB8A.SC63RTE.RKO2PROC,VOL=SBOX02
COMMAND=PEEK,EXTERNAL=YES,AUDIT=NONE
COMMAND=XMLS,EXTERNAL=YES,AUDIT=NONE
COMMAND=XMZP,LEVEL=DISABLE
COMMAND=DCMD,EXTERNAL=YES,AUDIT=NONE
COMMAND=DCNS,EXTERNAL=YES,AUDIT=NONE
```

Attention: IBM ships all commands with a default security level of 0 except for commands that were previously defined in the security table as authorized (LEVEL 3). If you specify EXTERNAL=YES and do not protect the command in RACF with **`RDEFINE O2OMPE <cmnd> UACC(NONE),`** any user can execute that command, regardless of what is coded in the security table. Our recommendation is to define to RACF, via RDEFINE, all commands that need protection.

If a specific command is to be made unavailable to everyone, the LEVEL=DISABLE can be specified. For example, as shown in Example 5-4, we elected to disable the XMZP command that allows modifications of operating system storage. Once the necessary modifications to the security table source have been made, the new definitions need to be generated using the security update program. The JCL necessary to run the security update program can be found in member KO2SUPD of the hlq.RKD2SAM dataset. Current Classic Interface sessions must be restarted in order to see the effect a security table update.

Important: *Every time* that a parameter in the exit routine is changed, a recycle of the Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS started task is needed to see the effect of the changes.

After identifying the list of authorized commands in the security table, each Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS command needs to be protected within the assigned RACF class. The following sample RACF command in Example 5-5 shows an example of this protection.

Example 5-5 Protecting commands to be executed

```
RDEFINE O2OMPE PEEK UACC(NONE) and
PERMIT PEEK CLASS(O2OMPE) ID(SYSPROG) ACCESS(READ)
```

One other point, where possible, try to associate command security to a group ID instead of individual RACF userids. In the above example, SYSPROG is a RACF group, which has been permitted to the PEEK command. We recommend the creation of groups such as SYSPROG and SYSDBA, which can have different lists of authorized commands permitted to them. Individual RACF userids would then be connected with the specific group as required.

5.4 Relogon feature

The relogon feature is a function of the /PWD command.

This feature allows us to change the userid and password for the External Security package from an active Classic Interface session, without having to close a current VTAM session. In the security table, do not mark the /PWD command as EXTERNAL=YES. To use the relogon feature, type /PWD and the new userid on the INFO-line as shown in Example 5-6.

Example 5-6 Relogon feature

```
/PWD PAOLOR4____ ZMENU    VTM     O2       V310./C DB8A 02/21/06 12:52:43   2
>  Help/News/Index PF1             Exit PF3                  PF Keys PF5
>    Type a selection letter at the left end of the top line and press ENTER.
===============================================================================
>         OMEGAMON II FOR DB2 CLASSIC INTERFACE -- REALTIME MAIN MENU
_ S  SUMMARY ............. Summary of DB2 activity
```

Tip: If the password is expired, the /PWD feature cannot be used.

To force External Security checking on all commands, regardless of how the EXTERNAL command was coded on the individual security table entry, the following RACF commands need to be defined. See Example 5-7.

Example 5-7 More definitions in RACF

```
RDEFINE O2OMPE INITIAL1 UACC(NONE)
RDEFINE O2OMPE INITIAL2 UACC(NONE)
RDEFINE O2OMPE INITIAL3 UACC(NONE)
```

With this definition we can guarantee that commands defined in the security table using EXTERNAL=NO cannot be executed.

5.4.1 Summary of steps to enable External Security

Table 5-1 summarizes the different actions needed to implement External Security checking for Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS authorized commands.

Table 5-1 How to configure External Security

ACTIONS	OBJECTIVE	CHANGES	WHERE	EFFECTS
Define new RACF class - Required	Build RACF defined resource	See "RACF class definition" on page 79	RACF Database	
Verify / Change Resource class name in sample exit routine	To be recognized by exit	MVC U#CHCLSD,=CL8'xxxxxxxx' where xxxxxxx is the RACF class name	Dataset HLQ.RKD2SAM(ko2racfx)	
Assemble / link edit sample exit routine	Enable exit routine in load module data set	Modify JCL and submit	Dataset HLQ.RKD2SAM(ko2racfa)	/PWD is locked Used only for Relogon feature
Alter security table	Identify protected commands	Add: EXTERNAL=YES Modify: MODULE=KO2RACFX) Include all commands: COMMAND=PEEK,EXTERNAL=YES COMMAND=DCMD,EXTERNAL=YES	Dataset HLQ.RKD2SAM(ko2supdi)	
Update Security program	Start using security commands	Modify JCL and submit	Dataset HLQ.RKD2SAM(ko2supd)	Commands is being controlled by RACF now
Default RACF definition	To force RACF signon authentication	RDEFINE O2OMPE INITIAL uacc(read)	RACF database	Permitting access on Classic Interface

ACTIONS	OBJECTIVE	CHANGES	WHERE	EFFECTS
RACF definition	To force RACF checking on all Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS commands	RDEFINE O2OMPE INITIAL1 uacc(none) RDEFINE O2OMPE INITIAL2 uacc(none) RDEFINE O2OMPE INITIAL3 uacc(none)	RACF database	Restricting access to others levels
Miscellaneous RACF definition	To protect commands and permit commands to specified groups	RDEFINE O2OMPE PEEK uacc(none) Permit PEEK class(o2ompe) id(sysprog) acc(read) Connect SYSPROG ID(PAOLOR4) SETR RACL(O2OMPE) REFRESH	RACF database	Protecting commands Authorizing specific group Connecting users to the group Refreshing RACF command

Part 3

What's new for DB2 PE users

In this part of the book, we describe some of the most important functions of OMEGAMON for DB2, now included in the converged Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS, which could be of great interest for a former DB2 PE user.

This part of the book contains the following chapters:

The choice between the interfaces for using the tool — the VTAM Classic Interface and the Tivoli Enterprise Portal (TEP) — depends on the type of functions to be executed. The VTAM interface is suitable for debugging and problem resolution, while the portal is needed for high level monitoring and alerting.

6

VTAM interface

If you are a previous user of DB2 Performance Expert (DB2 PE) with its ISPF interface, you will notice a number of significant differences when you start using the VTAM Classic Interface of OMEGAMON XE for DB2 PE. In this chapter you are introduced to these differences.

We discuss the following topics:

- Why use the VTAM Classic Interface?
- General features of the VTAM interface
- Using menus and commands
- Using the zoom and sort features
- Using online help
- Using user profiles

6.1 Why use the VTAM Classic Interface?

Users of DB2 PE who have used the DB2 PE ISPF interface and the PE client might well ask the following questions:

- Why am I now being advised to use the VTAM Classic Interface instead of the DB2 PE ISPF interface?
- Why should I not use the Tivoli Enterprise Portal (TEP) for all my access to OMEGAMON XE for DB2 PE?

There are two main reasons why the VTAM Classic Interface continues to play an important role: speed and reliability, which can be useful for critical problem resolution.

In many day-to-day production situations, the speed and reliability of the DB2 monitor still play a crucial role in achieving a solution. Where speed and reliability are important factors, the VTAM Classic Interface offers the following advantages:

- Once you have mastered navigating through the VTAM interface using menu options and zooming (PF11), it is extremely fast to use.
- Because of the direct VTAM connection to the OMEGAMON XE for DB2 PE server, response times are almost instantaneous.
- This speed can be critical when trying to solve online problems, which come and go very quickly. It also gives you a very good feel in real time for how the DB2 subsystem is performing.
- The VTAM interface is very reliable because you are only dependent on a direct connection to the OMEGAMON XE for DB2 PE server (one address space).
- The Tivoli Enterprise Portal (TEP) is somewhat slower because you must go through a number of layers of software for each interaction:
  - OMEGAMON XE for DB2 PE Server
  - Tivoli Enterprise Monitoring Agent (TEMA)
  - Tivoli Enterprise Monitoring Server (TEMS) Tivoli Enterprise Portal Server (TEPS)
  - Tivoli Enterprise Portal (TEP)
- Even when the VTAM interface experiences a problem with a particular function (such as a program check) it usually continues to operate for other functions.

Using the VTAM interface, you have the full range of online functions available including:

- Object analysis
- Near-term history
- Application trace facility
- DB2 Connect monitoring
- Monitoring of locking contention
- IFCID tracing

Using the Tivoli Enterprise Portal (TEP), not all of these functions are currently available. The functions available, such as thread activity, often do not show the full range of information that can be obtained using the VTAM Classic Interface.

6.2 General features of the VTAM interface

Probably the most obvious difference compared to ISPF, which you notice when you start using the VTAM Classic Interface, is that the panels are not formatted. This leads to the following differences in behavior.

No protected panel areas

As the panels are not formatted, there are no protected areas (apart from the product details on line one). This means that you can accidently type over information contained on the panel. Depending what text you type over, you might receive a variety of error messages when you press Enter.

Figure 6-1 shows a correctly displayed panel where the LOGM command has been executed.

```
>                            DB2 LOG MANAGER INFORMATION

 LOGM
+ Collection Interval:  REALTIME                         Start:  02/07 20:14:33
+ Report Interval:       7 sec                             End:  02/07 20:14:40
+
+ Logging Mode           =     DUAL     Archiving Mode          =         DUAL
```

Figure 6-1 DB2 Log Manager Information

In Figure 6-2, the letter L of the LOGM command has been accidently overtyped with a blank. As a result, the command can no longer be understood and an error message is produced. The normal output produced by the LOGM command is also missing, as the command cannot be executed.

```
>                            DB2 LOG MANAGER INFORMATION

  OGM
+       >> OB0900: Command OGM is not a valid command <<
```

Figure 6-2 DB2 Log Manager Information with error message

The key to understanding the behavior of the panel under these circumstances is the character displayed in column one. Table 6-1 describes the effect on the panel display of using various characters in column one of the panel.

Table 6-1 Behavior of characters in column one of a panel

Character in column one	Effect on panel display
`blank`	The first word following the blank executes as a command. If it is not a valid OMEGAMON XE for DB2 PE command the following error message shows: `>> OB0900: Command` *`xxxx`* `is not a valid command <<`
`>`	Lines preceded by > are regarded as text. You can overtype such lines without causing any problems. They are not refreshed with the correct text until you leave the panel and re-enter it. OMEGAMON XE for DB2 PE often puts > in front of a command to prevent it from being accidently executed (for example Start Object Analysis). You must then manually remove the > and press Enter to execute the command.
-	Some commands, where automatic execution when Enter is pressed is not desirable, must be preceded by - in order for the command to be executed. An example of such a command is PEEK.

Character in column one	Effect on panel display
+	Lines preceded by + are output lines generated by OMEGAMON XE for DB2 PE after it has executed a command. These lines are refreshed each time you press Enter.
`..`	All lines on the panel from the line starting with `..` to the end of the panel are cleared
`:`	Lines preceded by `:` allow you to enter a new value for a panel option after the = sign. Press Enter to make the new value active.
?	If you precede a command by ? and press Enter, you receive a single line help message describing the command.
/	If you precede a command by / and press Enter, you receive a long help message describing the command.

Restricted ability to tab between fields

A further consequence of the unformatted panels used by OMEGAMON XE for DB2 PE is that it is generally not possible to tab between input fields in the way you would in ISPF. You can tab vertically on menu panels to select a menu option, but no other kind of tabbing between input fields is possible.

Positioning in the input field

The standard input field on OMEGAMON XE for DB2 PE panels (it is usually the only input field, except on menu panels) is at the top left corner of the panel. It is 16 characters long and is highlighted with green underscore to make it more visible. You must type all commands starting at the left-hand end of this input field. If you do not type your command starting at the left-hand end, nothing will happen when you press Enter.

6.3 Using menus and commands

In this section we describe how to use menus and commands within the VTAM Classic Interface.

Using horizontally listed menu options

As shown in Figure 6-3, each available menu option is marked with a selection letter. To select an option, enter its selection letter on the top line of the panel. That is:

1. Place the cursor at the left-hand end of the top line.
2. Type the selection letter.
3. Press ENTER.
4. To return to the previous panel, press PF3.

The currently selected menu option has its option letter replaced by an asterisk (*).

```
 _______________ ZALLT    VTM     O2       V310./C D8F2 02/16/06 14:42:04   2
> Help PF1      Back PF3       Up PF7      Down PF8     Sort PF10    Zoom PF11
> T.A
>        THREAD ACTIVITY:  Enter a selection letter on the top line.

> *-ALL         B-TSO      C-CICS      D-IMS     E-BACKGROUND   F-DIST ALLIED
> G-DIST DBAC   H-UTIL     I-INACT     J-FILTER  K-FUNCTIONS    L-STORED PROC
> M-TRIGGERS    N-SYSPLEX  O-ENCLAVES  P-WORKSTA
===============================================================================
>                       ALL THREADS CONNECTED TO DB2
```

Figure 6-3 Selecting horizontally listed menu options

Using vertically listed menu options

On a vertically aligned menu panel as shown in Figure 6-4, you can also select an option as follows:

1. Place the cursor on the underline to the left of the selection.
2. Type the letter S (or s).
3. Press ENTER.
4. To return to the previous panel, press PF3.

```
>          RESOURCE MANAGERS AND OTHER DB2 SUBSYSTEM INFORMATION

_ A  BUFFER MANAGER ......... Buffer Manager Information
_ B  LOG MANAGER ............ DB2 Log Manager Information
_ C  EDM POOL ............... EDM Pool Information
_ D  BIND STATISTICS ........ Bind Statistics
_ E  SUBSYSTEM MANAGER ...... DB2 Subsystem Support Manager Statistics
_ F  ACTIVE TRACES .......... Current Trace Activity
_ G  START-UP OPTIONS........ IRLM and Stored Procedures Start-Up Options
_ H  DSNZPARM ............... DB2 Installation Parameters
_ I  LOCK/CLAIM/DRAIN........ Lock Manager/Claim/Drain Statistics
_ J  SQL/RID POOL/PARALLEL... SQL/RID Pool/Parallelism/Stored Proc. Information
_ K  OPEN/CLOSE STATISTICS... Dataset Open and Close Statistics
_ L  DB2 COMMANDS ........... DB2 Command Statistics
```

Figure 6-4 Selecting vertically listed menu options

Quick menu navigation

Every panel reached by selecting menus has a path - the sequence of letters you enter to reach it; for example R, C, A to reach the EDM pool snapshot summary. To reach such a panel directly, enter the letters of its path separated by periods (for example `R.C.A`) on the top line and press Enter. To reach a Main Menu selection, type its selection letter followed by a period (for example `R.`) and press Enter.

Do not enter panel names (such as ZEXSY) on the top line of the panel. This deactivates the PF key settings. If you do, press PF4 to return to the Main Menu and reactivate the PF key settings.

Major and minor commands

The output displayed on OMEGAMON XE for DB2 PE panels is generated by the use of major and minor commands. An example of the use of major and minor commands to display thread package summary information is shown in Figure 6-5. This panel is reached by typing `T.A` from the Main Menu to get to the Thread activity panel, followed by PF11 (zoom) to select a particular thread, and then option K to see the Thread package summary.

```
> A-THREAD DETAIL B-LOCK COUNTS C-LOCK WAITS     D-LOCKS OWNED  E-GLOBAL LOCKS
> F-CURRENT SQL   G-SQL COUNTS  H-DISTRIBUTED    I-BUFFER POOL  J-GROUP BP
> *-PACKAGES      L-RES LIMIT   M-PARALLEL TASKS N-UTILITY      O-OBJECTS
> P-CANCEL THREAD Q-DB2 CONSOLE R-DSN ACTIVITY   S-APPL TRACE   T-ENCLAVE
> U-LONG NAMES
===============================================================================
>                            THREAD PACKAGE SUMMARY
 PLAN
+ Thread:  Plan=KO2PLAN   Connid=DB2CALL  Corrid=D8F102S       Authid=D8F102S
+ Attach:  BATCH          JOB Name=D8F102S                     JOB Asid= 158
+ Package: DGO§WR2C       Collection=KO2OM310
 pkl
+ Package List: Location            Collection           Package
+               ----------------    ------------------   --------
+               DB8F                KO2OM310             DGO§WR2C
 pkg
+ Package/           SQL        In-DB2         In-DB2                 In-DB2
+     DBRM       Requests  Elapsed Time      CPU Time     Waits     Wait Time
+ --------     ----------  ------------  ------------  --------  ------------
+ DGO§WR2C  *        4186  00:00:14.597  00:00:01.083      3386  00:00:11.231
```

Figure 6-5   Examples of major and minor commands

The major command **PLAN** generates basic DB2 plan level information. A major command preceded by a blank can be used by itself on any panel to return information related to the command. Some major commands must be followed by a parameter. For example the major command BP 2 returns information about buffer pool BP2.

In Figure 6-5, the major command **PLAN** is followed by two minor commands:

- **pkl** to provide package list information.
- **pkg** to provide individual package summary information.

Minor commands cannot be used alone. They can only be used on a panel following the major command to which they belong. So **pkl** and **pkg** can only be used after the **PLAN** command to which they relate.

Using commands to build your own panel

Generally panels already have the correct major and minor commands present to display the correct information. However, it can sometimes be useful to blank out a minor command on a panel to avoid displaying unnecessary information, or to add a minor command to a panel where it does not normally occur to group related information for easier viewing. Remember that the related major command must be present on a panel for the minor command to work. You can also put two or more major commands on the same panel to display in one place information you are interested in.

To build your own panel from scratch, select option **M.F** (OMEGAMON Commands) from the Main Menu. This is a blank panel that allows you to enter your own commands. For example, Figure 6-6 shows the use of the major commands **XLOK**,**THDA** and **PTHDA** together to produce a single panel showing:

- Locking conflicts
- Thread summary at plan level
- Thread summary at package level

Using option M.F you can also enter special session and maintenance commands that do not occur on any of the panels. A description of these commands can be found in Chapter 13, “OMEGAMON Commands”, in *Monitoring Performance from the OMEGAMON Classic Interface*, SC18-9659.

```
>                        OMEGAMON COMMANDS
 XLOK
+      No Locking Conflicts Detected
 THDA
+ *
+ Elapsed      Planname  CPU    Status       GetPg   Update  Commit  Jobname
+ ----------   --------  -----  ------------ ------  ------  ------  --------
+ 20:54:37.4   KO2PLAN   00.0%  NOT-IN-DB2     3592    1296     325  D8F1O2S
+ 01:14:23.9   KO2PLAN   00.0%  NOT-IN-DB2        0       0       0  D8F1O2S
PTHDA
+ *
+ Elapsed      Package   CPU    Status       GetPg   Update  Commit  Jobname
+ ----------   --------  -----  ------------ ------  ------  ------  --------
+ 20:54:37.4   DGO§WR2C  00.0%  NOT-IN-DB2     3592    1296     325  D8F1O2S
+ 20:54:37.4   DGO§SDOB  00.0%  NOT-IN-DB2     2456      14       3  D8F1O2S
```

Figure 6-6 Using commands to build your own panel

Reusing panels you have built

If you have built a panel using commands that you want to use regularly, you can save the panel definition as a member in the RKO2PCSV library.

To save a new panel in the RKO2PCSV library, use the command:

`/SAVE cccccccc` where `cccccccc` is the name you want to use for the panel.

To replace a previously defined panel, use the command:

`/REP cccccccc` where `cccccccc` is the name of the existing panel.

To display a panel you have defined, type its name in the top left input field of any panel.

OMEGAMON XE for DB2 PE panels are delivered in the RKO2PROC library. If you define your own panel with the same name as an existing base panel, you can do this as your panel will be saved to the RKO2PCSV library. Your panel will be displayed instead of the base panel for as long as your panel exists in the RKO2PCSV library.

To delete a panel from the RKO2PCSV library, use the command:

`DELT cccccccc` where `cccccccc` is the name of the panel you want to delete.

This command will not delete a member of the base library RKO2PROC, even if this library has a member with the same name.

6.4 Using the zoom and sort features

In this section we show the zoom (PF11) and sort features (PF10).

Using PF11 (zoom)

Once you have selected a summary screen using menu option letters, you can generally select one of the summary lines and use PF11 (zoom) to drill down for further detail. This drilling down ability using PF11 is sometimes possible over several levels of detail. To zoom in from a summary line:

1. Place the cursor anywhere on the summary line for which you want to see more detail.
2. Press PF11 to zoom to the corresponding detail panel.
3. To return to the higher level, press PF3.

The zoom feature is available on all panels with Zoom PF11 displayed at the right-hand end of line two of the panel, as shown in Figure 6-7.

```
 ________________ ZALLT    VTM     02       V310./C D8F1 02/16/06 19:36:03   2
> Help PF1      Back PF3       Up PF7      Down PF8     Sort PF10    Zoom PF11
> T.A
>        THREAD ACTIVITY:  Enter a selection letter on the top line.

> *-ALL         B-TSO      C-CICS      D-IMS     E-BACKGROUND   F-DIST ALLIED
> G-DIST DBAC   H-UTIL     I-INACT     J-FILTER  K-FUNCTIONS    L-STORED PROC
> M-TRIGGERS    N-SYSPLEX  O-ENCLAVES  P-WORKSTA
```

Figure 6-7 Using the PF11 zoom feature

Using PF10 (sort)

On panels where Sort PF10 appears towards the end of line two, it is possible to sort by any of the columns in the summary display using PF10. This sorting ability is only possible on those panels where it is specifically indicated. To sort on a column:

1. Place the cursor anywhere within the column (except the column heading) that you want to sort.
2. Press PF10 to sort on the column.
3. Ascending or descending sort order is decided for you depending on which column you are sorting.

Figure 6-8 shows an example of sorting on the GetPg column (get pages). The rows are ordered in descending sequence on get pages and an asterisk (*) above the column heading indicates the column that has been sorted.

```
 ________________ ZALLT    VTM     02       V310./C D8F1 02/16/06 19:47:41   2
> Help PF1      Back PF3       Up PF7      Down PF8     Sort PF10    Zoom PF11
> T.A
===============================================================================
>                       ALL THREADS CONNECTED TO DB2
 THDA
+                                              *
+ Elapsed      Planname  CPU    Status       GetPg   Update  Commit  Jobname
+ ----------   --------  -----  ------------ ------  ------  ------  --------
+ 01:17:59.3   KO2PLAN   00.0%  NOT-IN-DB2     2456      14       3  D8F102S
+ 01:17:59.3   KO2PLAN   00.0%  NOT-IN-DB2     1318     468     118  D8F102S
+ 00:25:16.6   KO2PLAN   00.0%  NOT-IN-DB2        0       0       0  D8F102S
```

Figure 6-8 Using PF10 to Sort

6.5 Using online help

In this section we outline the significant amount of online help available within OMEGAMON XE for DB2 PE.

General help

If you press PF1 from the Main Menu or from any of the help panels, you see the General help menu shown in Figure 6-9. This gives general help about navigating through the panels using menus, zooming and PF keys.

```
>          OMEGAMON XE FOR DB2 PE GENERAL HELP MENU


_ A  MENUS ................ Explanation of menu facility
_ B  ZOOMING .............. How to get more detail
_ C  PF KEYS .............. Listing of PF Key settings
_ D  COMMANDS ............. How to get help for commands
_ E  SUPPORT .............. IBM Software Support Information

_ F  INDEX ................ Alphabetical index of DB2 performance topics
```

Figure 6-9 General Help Menu

The General help menu also provides an index to extensive online help on DB2 performance topics. Part of this index is shown in Figure 6-10.

```
 ==============================================================================
>           TOPIC                         ! MAIN PATH    ! NAVIGATION ! HELP+PF4
 ==============================================================================
> CICS exception messages               ! Exceptions   ! E.C.       ! ZXGRPCH
> CICS exception status                 ! Exceptions   ! E.F.C      ! ZXSUMCH
> CICS RTC detail                       ! Thread       ! T.C +2 Zms ! ZCICTRDH
> CICS RTC summary                      ! Thread       ! T.C+Zm/CICA! ZCICSTRH
> CICS threads                          ! Thread       ! T.C        ! ZCICSTH
> CICT exception, set options           ! Profile      ! P.B.A/P.C.B! ZXACBH
> CICT exception recommendation         !              !            ! ZCICT
```

Figure 6-10 Part of the index of DB2 Performance Topics

You can go directly to the help panel for the indexed item by typing the help panel name given in the right-hand column of Figure 6-10 in the entry field at the top left of the panel you are on.

Context sensitive help

In addition to general help there is extensive online, context sensitive help available to you. If you press PF1 from any OMEGAMON XE for DB2 PE panel, you receive detailed help relevant to the panel you are looking at. Figure 6-11 shows part of the Help panel displayed when you press PF1 from the Thread detail panel. This help includes:

- Detailed descriptions of each of the fields on the panel.
- For Status fields you receive a listing of the possible statuses and what they mean.
- Indications of possible problem causes.

```
> Total Elapsed Time
>     The total elapsed time for the thread in the format
>     HH:MM:SS.THT.  Elapsed time is the time which has elapsed
>     since thread creation or DB2 signon.

> CPU Utilization
>    The CPU rate (percent) associated with the thread.

> Total CPU Time
>     The total amount of CPU time accumulated for the thread.  This
>     value includes only MVS TCB time.

> Total Parallel Tasks
>     The total number of parallel tasks that were created on behalf
>     of this thread.  Any activity performed on behalf of this thread
>     by parallel tasks is not included in this thread.

> Total Stored Proc CPU
>     The total amount of CPU time (TCB time) spent processing SQL CALL
>     statements in the DB2 stored procedures address space.
>     For DB2 6.1 and above this is shown under Stored Procedures
>     Total CPU.
```

Figure 6-11 Example of context sensitive help for the Thread Detail Panel

Help for commands

Figure 6-12 shows the commands that you can type in column one of any panel to receive help on particular major and minor commands.

```
?cccc       One-line help description for command cccc.
/cccc       Extended help for command cccc.
;cccc       Extended help for command cccc deleted after one cycle.
 .ILC /cccc  Help for INFO-line command cccc.

H.MIN       One-line helps for all the minors of the major executed above.
H.MJI       One-line helps for all immediate commands.
H.ILC       One-line helps for all INFO-line commands.
H.MJC       One-line helps for all major commands.
```

Figure 6-12 Help for commands

Function key help

Figure 6-13 shows the use of function keys within OMEGAMON XE for DB2 PE. You can see this help panel at any time by pressing PF5.

```
>                        PROGRAM FUNCTION KEY DIRECTORY

> Press PF5 to return to this display at any time.

>    1- Help                              13- Help
>    2- Undefined                         14- Undefined
>    3- Back (to previous screen)         15- Back (to previous screen)
>    4- Back to Main Menu                 16- Back to Main Menu
>    5- PF keys list (this screen)        17- PF keys list (this screen)
>    6- Print screen                      18- Print screen
>    7- Scroll up                         19- Scroll up
>    8- Scroll down                       20- Scroll down
>    9- Undefined                         21- Undefined
>   10- Recommendation                    22- Undefined
>   11- Zoom                              23- Undefined
>   12- Undefined                         24- Undefined
```

Figure 6-13 Help for function keys

6.6 Using user profiles

Option P from the Main Menu, or typing P in the top left input field of any panel, brings you to the Profile Maintenance Menu shown in Figure 6-14.

```
>                        PROFILE MAINTENANCE MENU

_ A  SESSION OPTIONS ..... Set session options

_ B  EXCEPTION ANALYSIS... Exception analysis options

_ C  LIST PROFILES ....... List profiles
_ D  DESCRIBE PROFILE .... Describe a profile

_ E  SAVE USER ........... Save new/altered user profile
_ F  DELETE USER ......... Delete user profile
_ G  SAVE INSTALL ........ Save new/altered installation profile
_ H  DELETE INSTALL ...... Delete installation profile

_ I  LOGGING ............. Exception and automatic screen logging
_ J  MESSAGES ............ Set DB2/IRLM messages that MSGE exception monitors
```

Figure 6-14 Profile Maintenance Menu

From this menu you have a wide range of options that allow you customize your session. Changes you make here will only be visible to your session and, unless you save the changes in a named profile, will not be retained when you close the session.

Logging on with a user profile

Profiles are identified by means of a two character suffix as shown in Figure 6-15. The suffix is found on the top line of every panel following the product version number and separated by a dot.

```
 ________________ ZPROF    VTM     O2       V310./I DB8A 02/26/06 13:53:03   2
> Help PF1                                                          Back PF3
> P.
```

Figure 6-15 Profile suffix

In Figure 6-15 the profile suffix is `/I`, which represents the installation profile. Table 6-2 shows the options available for a profile suffix and how the suffix is used.

Table 6-2 Profile suffix options

Suffix	Purpose
/C	This is the default configuration profile supplied with OMEGAMON XE for DB2 PE and cannot be changed. It will be used if no other profile has been specified.
/I	This is the installation profile for your site.It must be created using Option G Save Install on the Profile Maintenance Menu and typically holds installation wide default values for your site. If it exists, it will automatically be used for all users who do not specify a user profile. Changes made to this profile will affect all users who do not specify their own profile.
cc	This is any two character suffix that you have used with Option E Save User on the Profile Maintenance Menu to save your customized session under a user profile. For this user profile to take effect, it must be explicitly specified when you logon to OMEGAMON XE for DB2 PE.

When you logon to OMEGAMON XE for DB2 PE, the `/C` profile (or `/I` if it exists) will be used by default.

To logon with a user profile suffix `cc`, use one of the following methods:

- If you are logging on to OMEGAMON XE for DB2 PE using the ISPF Main Menu option two (2. View online DB2 activity - Classic Interface), type `cc` in the User Profile field on the logon panel (Invoke OMEGAMON XE for DB2 PE Classic Interface), which appears after you select menu option two from the Main Menu.
- If you are logging directly on to OMEGAMON XE for DB2 PE as a native VTAM session outside of TSO/ISPF, use the VTAM logon parameter `USER=cc` (exactly how you pass this parameter will depend on the options available to you for starting native VTAM applications at your site).

Listing and saving user profiles

To list all existing profiles use option C from the Profile Maintenance Menu. Figure 6-16 shows a typical listing. As well as the names of all existing user profiles in the form `O2USERcc` where `cc` is the profile suffix, you can also see the names of the libraries where the profiles are saved.

To save a new or existing profile you have two options:

- Use option E to save the customized session as a user profile.
- Use option G to save the customized session as an installation profile. This will automatically be picked up by all users who do not use their own user profile.

```
>                            LIST PROFILES
>PPRF LIST
>   RKO2PFSV           DSN=OMEGASYS.DB8A.SC63RTE.RKO2PFSV
>      ____________________________________________________
>      Profile ID     Description            Date Modified
>
>      O2INSTAL    INSTALLATION PROFILE         02/24/06
>      O2USERJJ    USER PROFILE                 02/26/06
>      O2USERJ2    USER PROFILE                 02/26/06
>
>   RKO2PROF     (Dup.) DSN=OMEGASYS.DB8A.SC63RTE.RKO2PFSV
>      ____________________________________________________
>
>   RKO2PROF +01       DSN=OMEGASYS.DB8A.SC63RTE.RKO2PROF
>      ____________________________________________________
>      Dataset empty
```

Figure 6-16 List Profiles

Exploiting user profiles

Tip: Remember that any customizing changes you make to OMEGAMON XE for DB2 PE (including all the individual exception values you may want to set for exception analysis) will not be saved beyond the end of the session unless you save them in a user or installation profile.

The three areas where user profiles can be exploited to best effect are:

- Exception analysis (option B)
- Logging (option I)
- Messages (option J)

Practical examples of how logging can be used in conjunction with exception analysis to help solve difficult DB2 operational problems are shown in Chapter 16, “Solving common DB2 operational problems” on page 285.

7

Portal interface

This chapter provides an introduction to the Tivoli Enterprise Portal (TEP).

We discuss the following topics:

- Overview and terminology
- Customized workspaces
- Situations and exception processing

7.1 Overview and terminology

Tivoli Enterprise Portal provides a Navigator Tree view of the DB2 environment from which we can drill down to more closely examine individual performance components of the monitored DB2 subsystems. It consists of a window application that includes a Navigator that shows all the DB2 for z/OS systems in the environments where Tivoli agents are installed, and one or more workspaces that includes views of selected application system conditions in either table and chart presentation formats. Situations run at the TEMA regular intervals to verify that critical applications and system resources are running according to predefined threshold parameters. A true situation causes event indicators to appear in the Navigator.

Tivoli Enterprise Portal offers two modes of operation: desktop and browser.

- In desktop mode, the application software is installed on your system. In browser mode, the application is installed by the administrator on a Web server, and Tivoli Enterprise Portal is started from the browser.
- In browser mode, the JAR files are downloaded to the workstation the first time a logon to Tivoli Enterprise Portal is made, and thereafter only when there are software updates. Figure 7-1 shows the layout of a typical Tivoli Enterprise Portal screen.

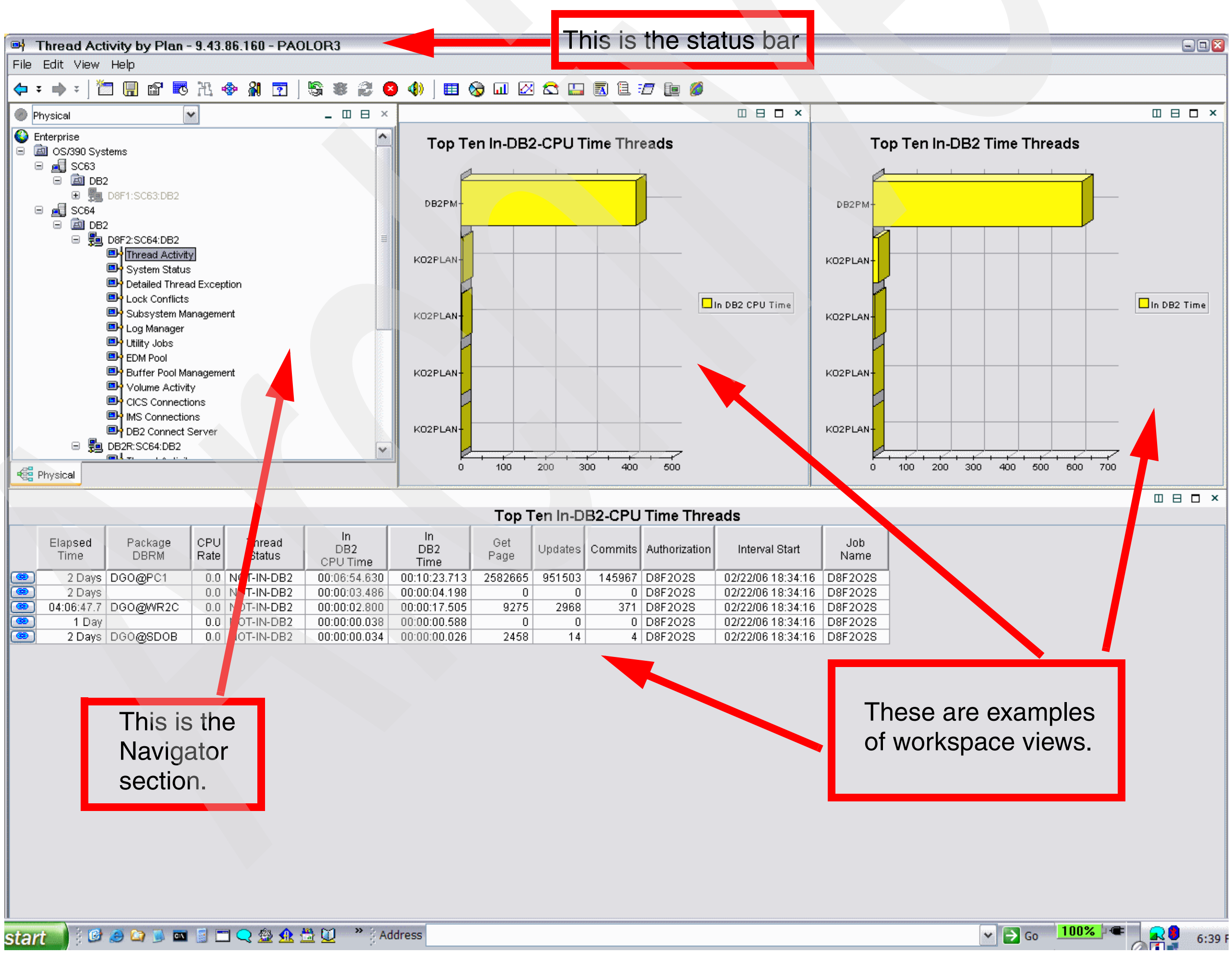


Figure 7-1 Net Portal - typical workspace

A workspace is the working area of the Tivoli Enterprise Portal application window. At the left of the workspace is a Navigator that permits you to select the workspace you want to display.

As part of the application window, the right side of the status bar shows the Tivoli Enterprise Portal server name and port number to which the displayed information applies, as well as the ID of the current user. As you select items in the Navigator, the workspace presents views pertinent to your selection. Each workspace has at least one view. Every workspace view has a set of properties associated with it. Within each property, there are queries, which defines what formula is used to return the agent data, the number of rows returned from the query, filters on selected columns that can be used to limit the amount of data shown, styles that select the type of font and font size, and the view type.

Tivoli Enterprise Portal information can be presented into any of the following types of views:

- Table view
- Pie chart view
- Bar chart view
- Plot chart view
- Needle gauge view
- Notepad view
- Event console view, which shows the status of the situations associated with the system.
- Take Action view, which is used to send a command to the system.
- Terminal view, which enables you to start a 3270 or 5250 work session.
- Browser view, which permits you to open a browser to see HTML pages and Web sites.

Figure 7-2 is an example of the same type of data, in this case the Top 10 In-DB2-CPU Time threads, shown in several different types of view.

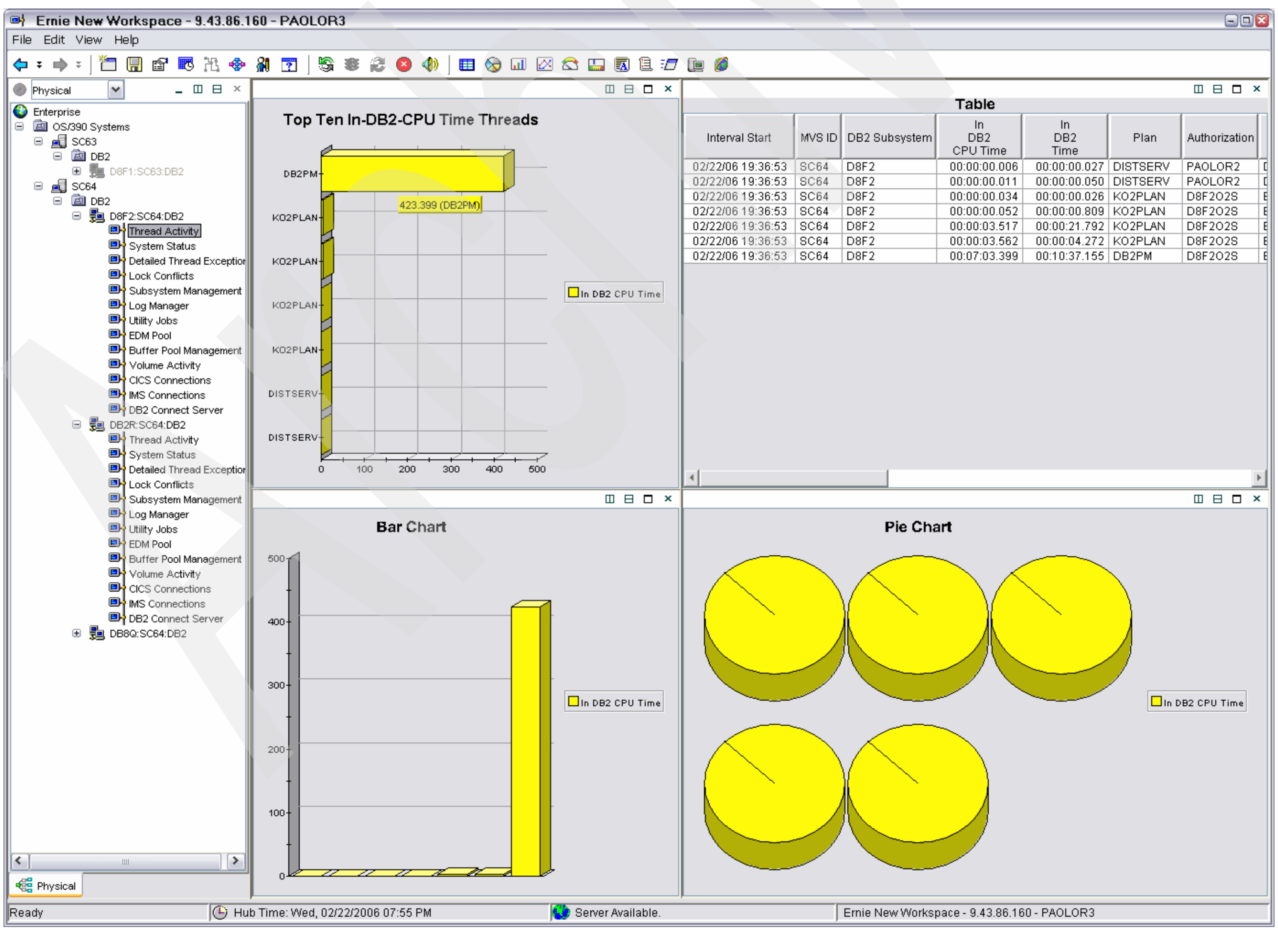


Figure 7-2 Different types of views

Tivoli Enterprise Portal ships with a number of pre-defined workspaces that can be used to investigate and possibly help to solve various performance related problems. Each workspace contains one or more views that have predefined properties.

In alphabetical order, these workspaces are:

- Buffer Pool Management
- CICS Connections
- Detailed Thread Exception
- DB2 Connect Server
- EDM Pool
- IMS Connections
- Lock Conflicts
- Log Manager
- Subsystem Management
- System Status
- Thread Activity
- Utility Jobs
- Volume Activity

More information on these workspaces, including the views contained within each workspace, can be found in the product online help.

Data flow: Opening a workspace

Each data view in the workspace is assigned a query. When the user opens a workspace, by clicking one listed in the navigator, the query is sent immediately to the Tivoli Enterprise Portal server (TEPS). The queries will request certain data from the various monitoring agents.

Next, the TEPS performs consolidation and sends the request to the Tivoli Enterprise Monitoring Server (TEMS). The TEMS then passes the request to the appropriate Tivoli Enterprise Monitoring Agent (TEMA). If this agent is located on a remote TEMA or as a subtask of a remote TEMS, the hub TEMS passes the request forward to be managed by the remote TEMS.

Based on the attributes of the data needed by the query, the TEMA receives the request, takes the necessary samples, and returns them to the TEMS. In the case of the remote agent, the remote TEMS sends the result back to the hub TEMS.

The TEMS then ships the result down to the TEPS, which then passes the data back to the requesting Tivoli Enterprise Portal workspace.

7.2 Customized workspaces

For most purposes, we should be able to make effective use of the Tivoli Enterprise Portal with the default workspaces and associated views. We are going to show a simple example of how one might modify a workspace for their own use, and at the same time illustrate the TN3270 terminal view. One possible scenario involves using the Tivoli Enterprise Portal workspace to detect a suspicious event or performance trend, and then from the workspace be able to launch a Classic VTAM session.

In order to save customized workspaces, the user must be defined in the TEPS with administrator authorities. Customized screenspaces are saved at the Tivoli Enterprise Portal Server and can be shared between different TEPS users.

For this example, we will first open the default Thread Activity workspace, shown in Figure 7-3.

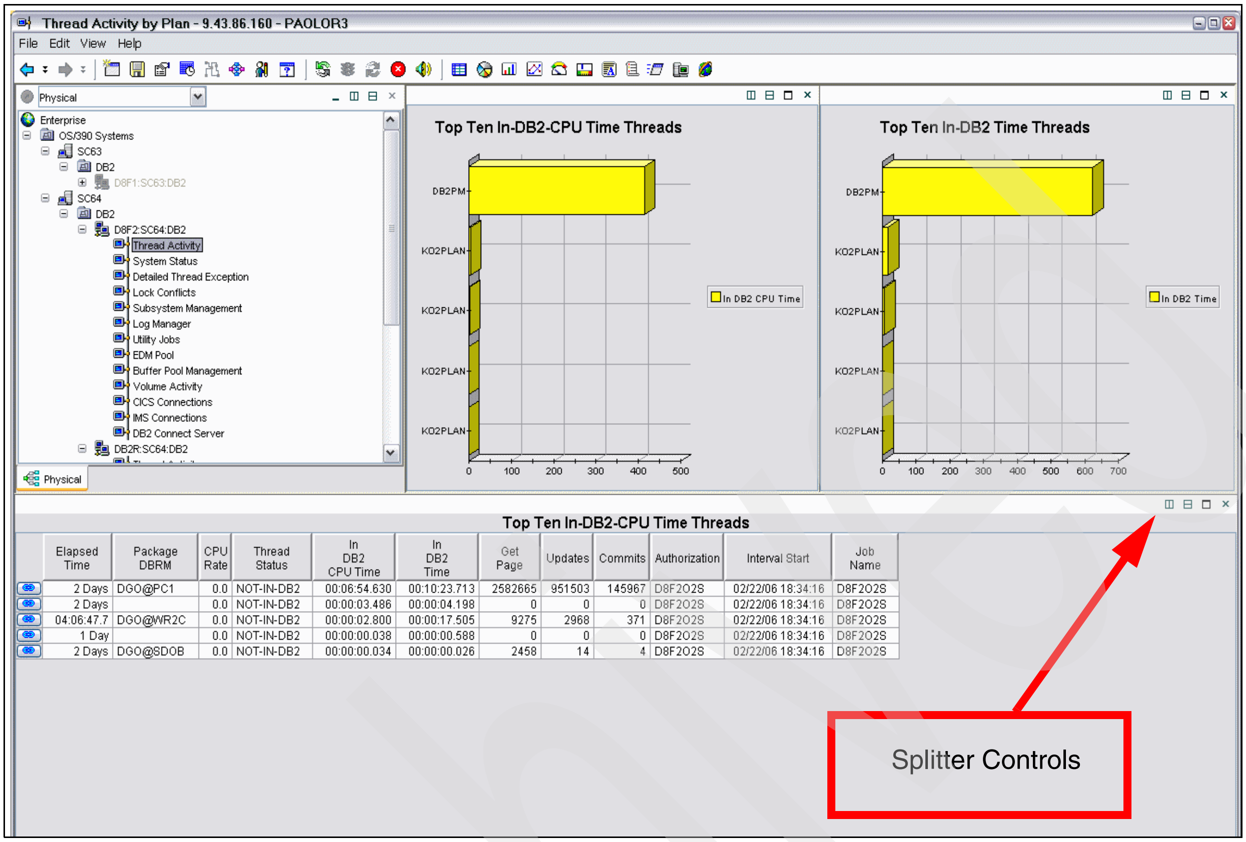


Figure 7-3 Default Workspace

Notice that, with the splitter controls, we can split horizontally, vertically, or zoom to full screen. In this case we chose to split vertically as shown in Figure 7-3, and this gives us two duplicate views.

Next we replace the duplicate In-DB2-CPU view with the TN3270 terminal view. Figure 7-4 shows an example of this.

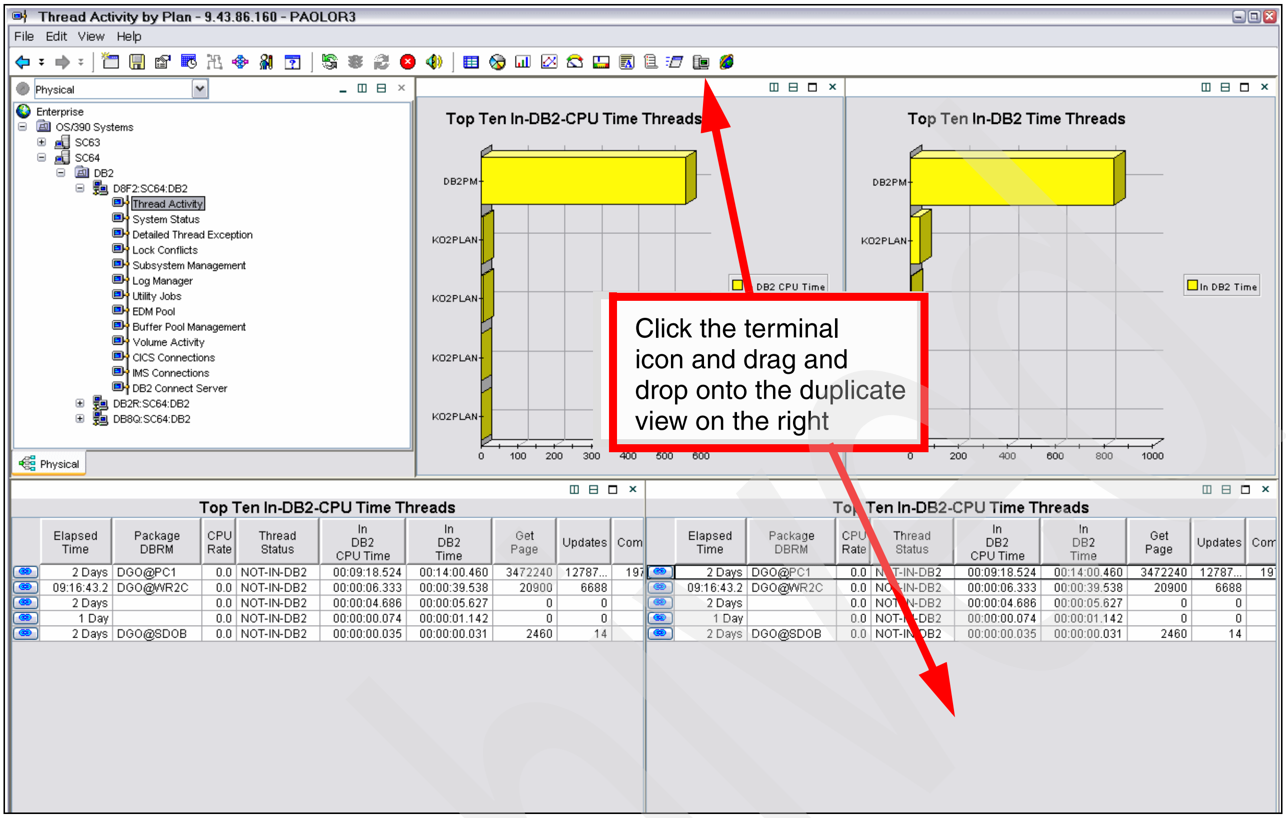


Figure 7-4 Drag terminal view to overlay split view

After the duplicate view — in our example, the Top Ten In-DB2-CPU Time Thread view — has been replaced by the terminal view, then the TN3270 configuration box needs to be completed. This is shown in Figure 7-5. Remember to confirm by entering **OK**.

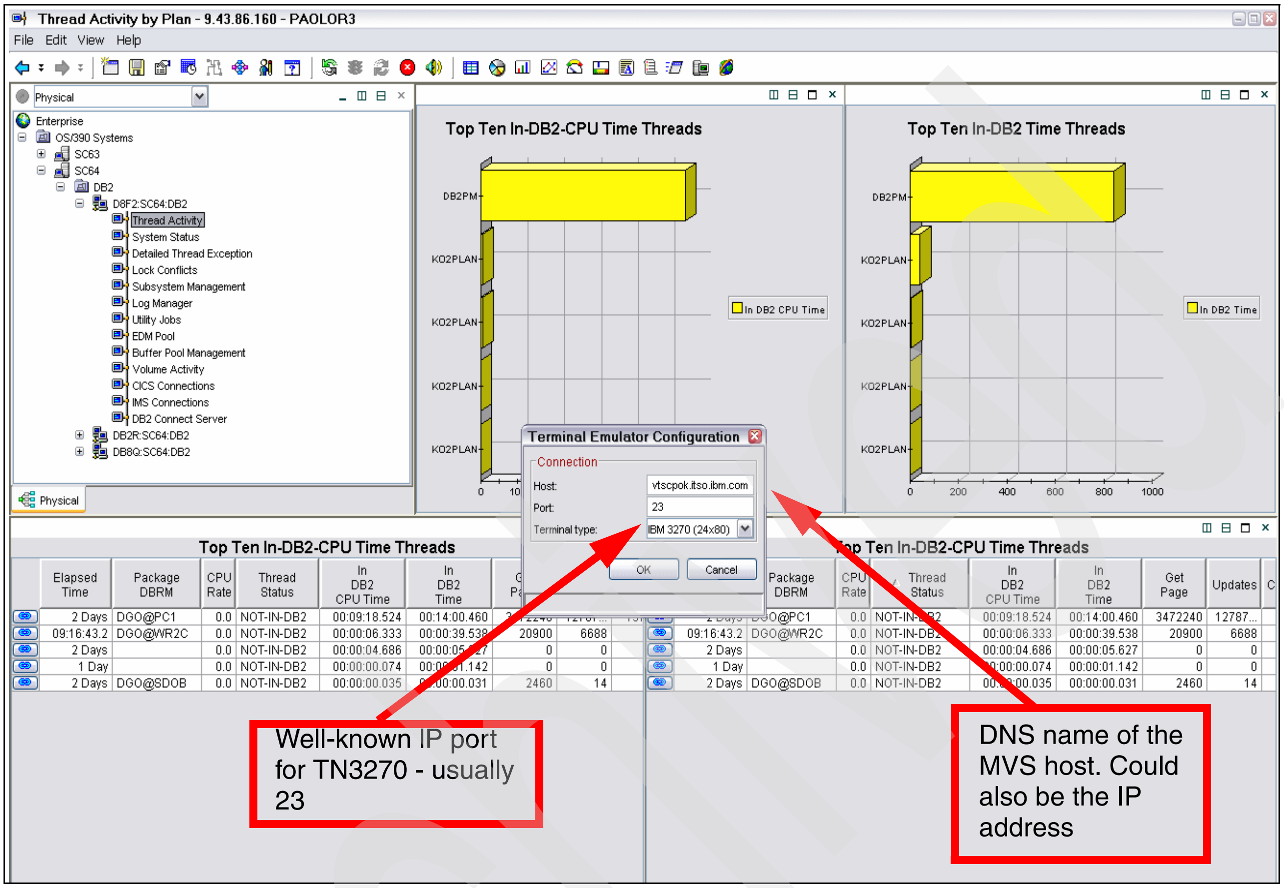


Figure 7-5 TN3270 Configuration

Once the configuration has been confirmed, and the configuration values are correct, the new view should contain the VTAM LOGMSG10 screen. Your screen could be different, and you need to ensure that your installation supports connection via TN3270. With a successful connection, a screen view like the one in Figure 7-6 should appear in the workspace.

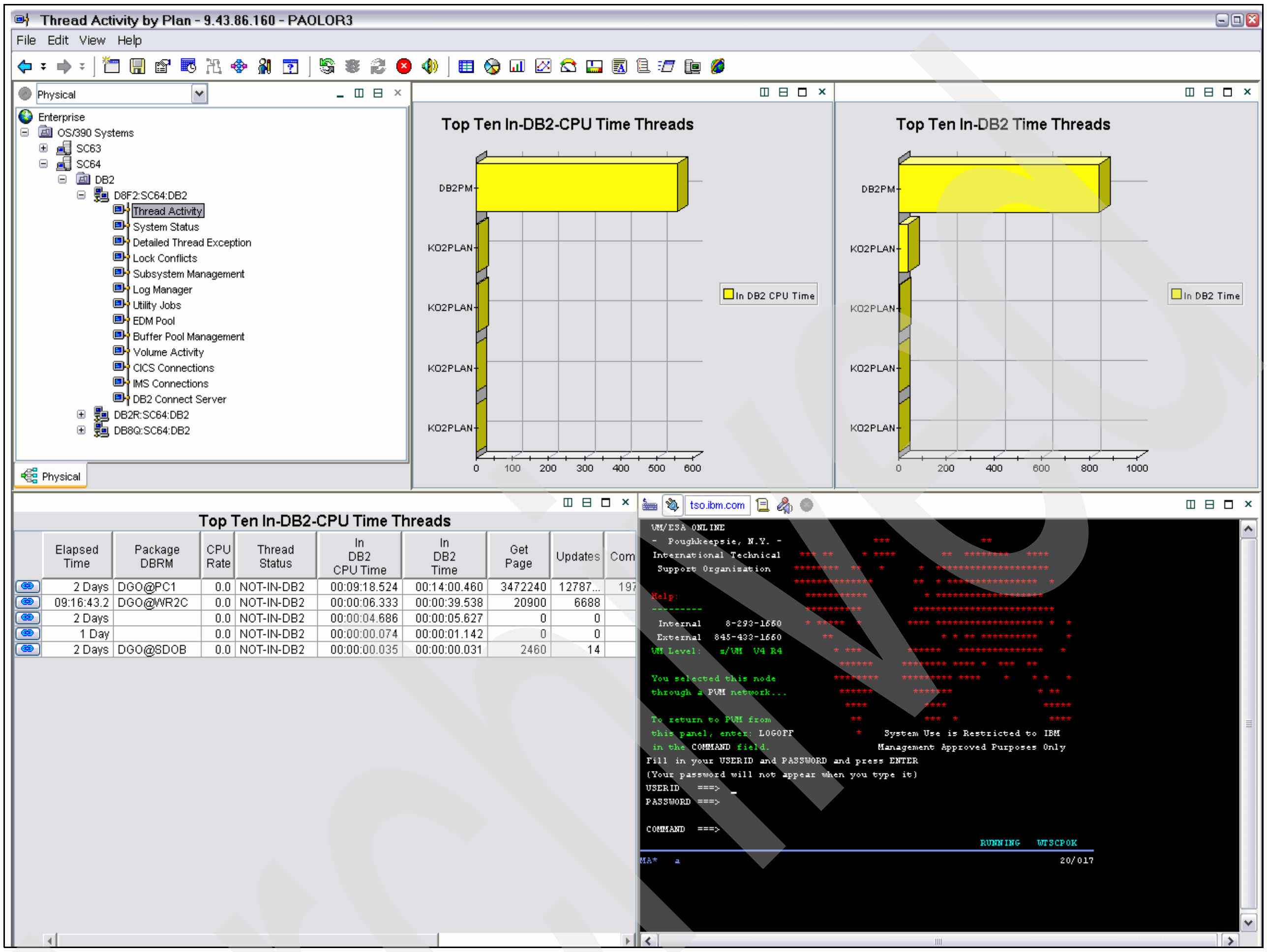


Figure 7-6 TN3270 Terminal view in the workspace

Once we have successfully configured the view, if we close the workspace, we are then prompted to save or discard our modifications. When we reply *YES,* we then create a new workspace name, select the workspace options, and then exit. This is shown in Figure 7-7.

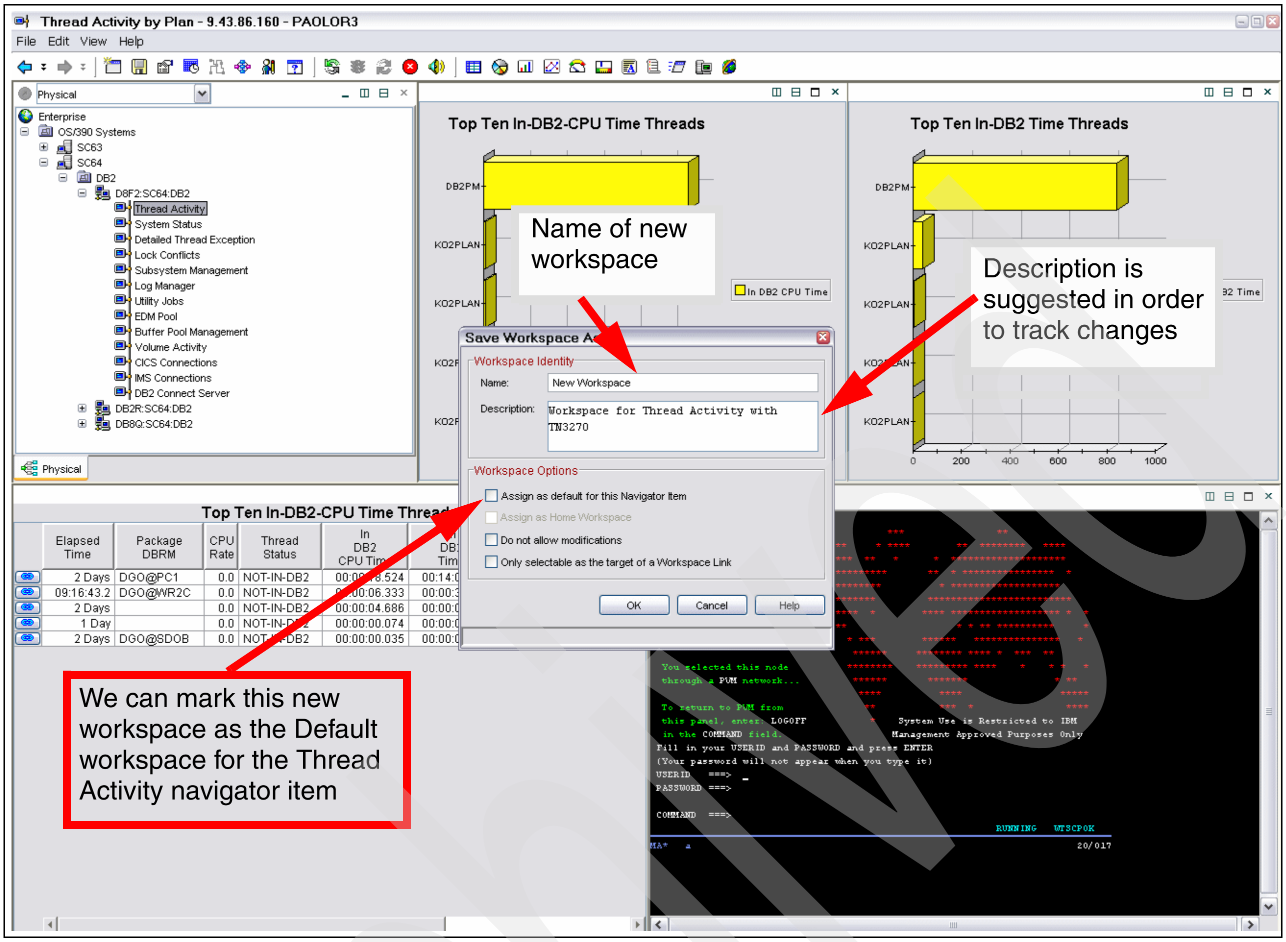


Figure 7-7 New workspace save options and exit

There are many more types of customizations that can be performed; please refer to the online help panels for more information. For additional information on the workspace support within the Tivoli Enterprise Portal, please refer to *IBM Tivoli OMEGAMON V3.1 Deep Dive on z/OS*, SG24-7155.

7.3 Situations and exception processing

A situation is a logical expression involving one or more system conditions. Situations are used to monitor the condition of systems in your environment. Situations can trigger notifications and schedule actions when an event occurs on a managed system. Situations can be managed from Tivoli Enterprise Portal using the situation editor.

The Tivoli Management Agents used to monitor the system environment are shipped with a set of predefined situations that can be used as-is or modified to meet unique requirements. Predefined situations contain attributes that check for system conditions common to many enterprises. Using predefined situations can improve the speed with which Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS can be deployed and used. Conditions or values being monitored by a predefined situation can be examined and, if necessary, changed to those best suited to the enterprise.

The Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS Tivoli Management Portal predefined situations are associated with the following workspaces:

- Buffer Pool Management
- CICS Connections
- Detailed Thread Exception
- IMS Connection
- Log Manager
- Subsystem Management
- System Status
- Utility Jobs
- Volume Activity

Situation scenario

Situations are created and managed using the Situation Editor. Situations can be displayed, new situations can be created, existing situations can be copied to create new situations, and existing situations can be deleted. In the following scenario, we will show how to use the Situation Editor to create a situation to monitor when threads from a specific combination of plan and authorization ID exceeds a predefined elapsed time threshold. We will also define an automated action based on this exception to cancel the associated DB2 thread.

From the Tivoli Enterprise Portal, we can review the navigator and select the workspace to which we will attach the situation. Positioning the mouse pointer and right-clicking will materialize a drop-down as shown in Figure 7-8.

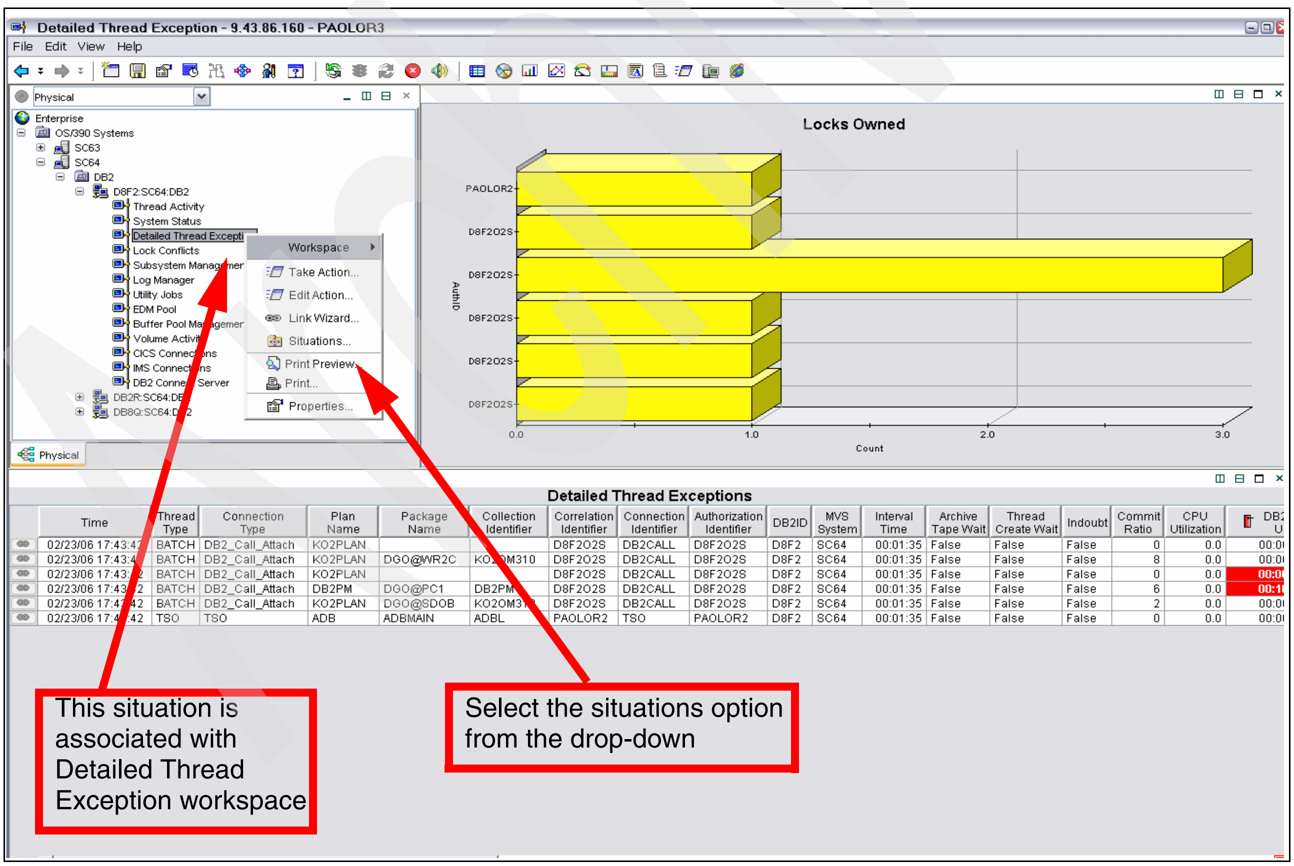


Figure 7-8 Workspace drop-down - situations option

Once in the Situation Editor, we can create new situations, copy existing situations, modify situations, or delete situations. The icons in the Situation Editor task bar control these choices as shown in Figure 7-9.

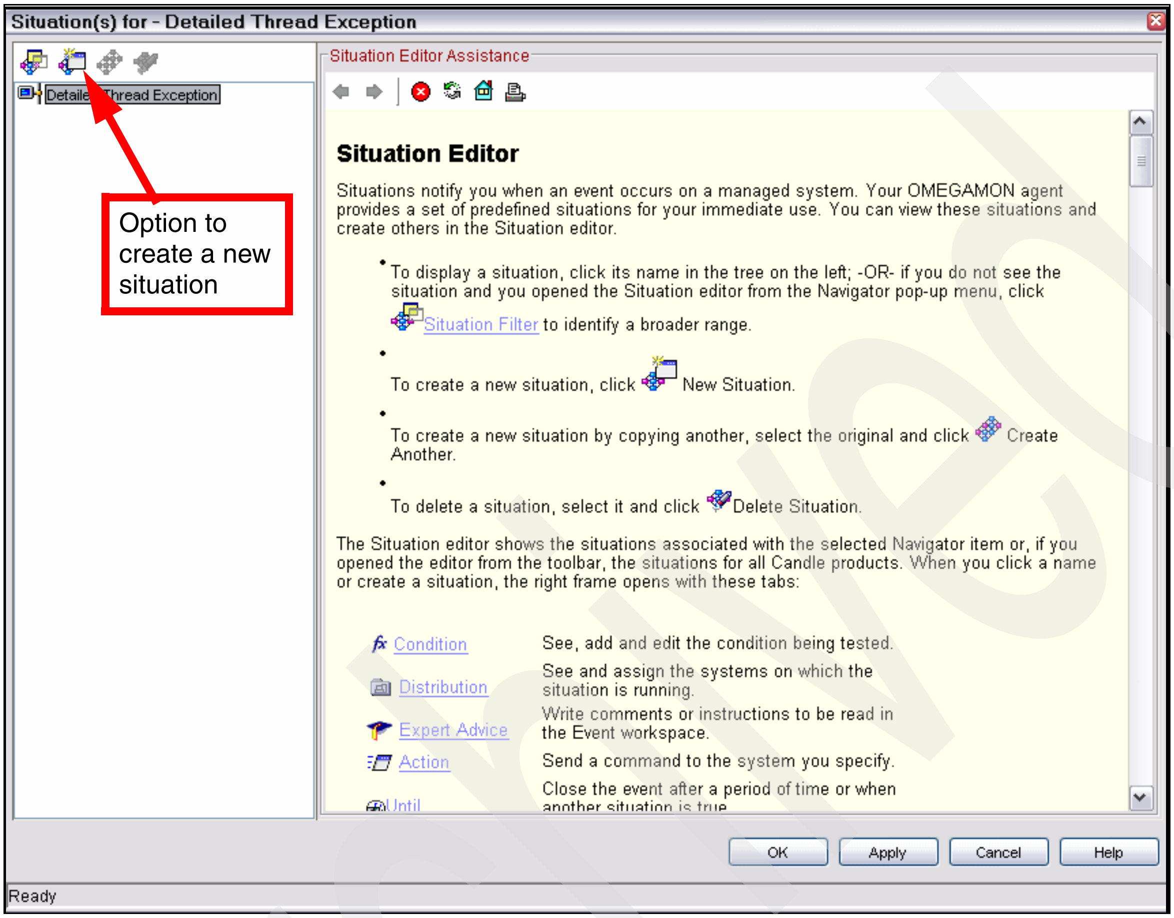


Figure 7-9 Situation Editor

In order to create a situation, you first choose a name and optional comments. The field Monitored Applications will contain the folder under which the agents associated with your chosen workspace execute. Figure 7-10 shows this screen.

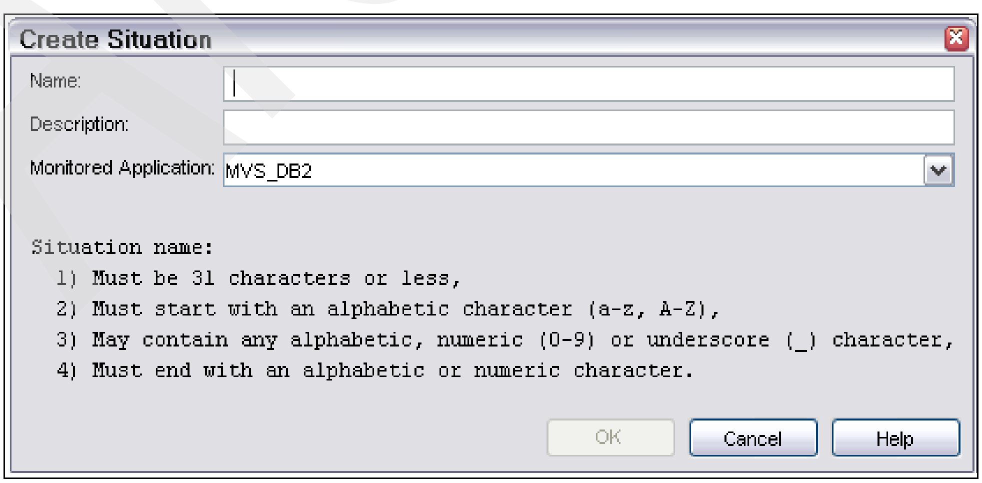


Figure 7-10 Create situation

Next in the definition process is selecting the attribute(s) associated with the situation. There can be up to ten attributes with any one situation row, and situations can consist of multiple rows. If attributes need to be evaluated together and all met (Boolean AND logic), as in our example of applying an evaluation of authid, plan, and elapsed time, the attributes should be in the same row. If the evaluation dictates that if any set of attributes are met the situation is met (Boolean OR logic) these can be placed on separate rows. So again, in our example if we wanted to test TRUE if either authid and elapsed time or plan name and elapsed time exceeded our threshold, we would built a two row situation with two attributes.

In our scenario, we have one row with three attributes. Figure 7-11 shows how we would associate the elapsed time attribute to our situation. We would use this select attribute dialog to also select the Plan Name and Authorization ID attributes.

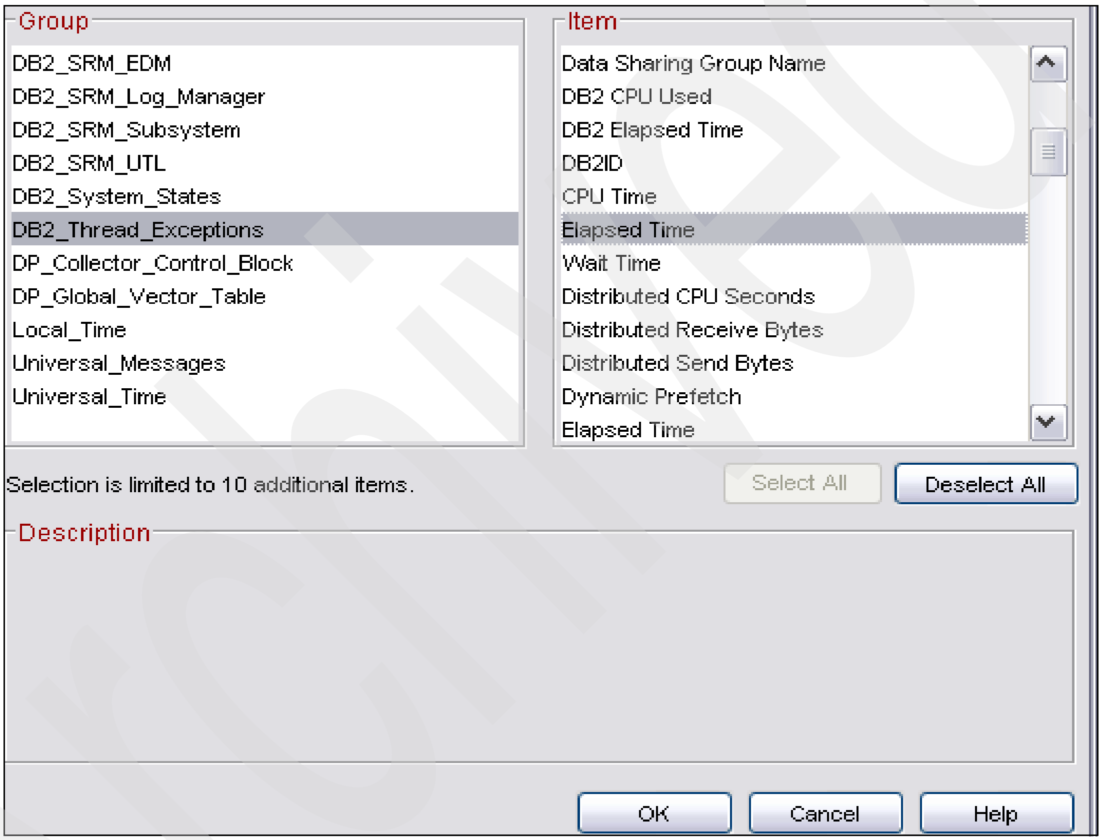


Figure 7-11 Situation attribute selection

Once we have selected the desired attributes, we can next specify conditions that can be applied to each of the attributes in our situation row. The condition consists of a function, an operator, and the value to be tested against the data values arriving from the agent. Figure 7-12 shows how to set conditions for the values used in our situation.

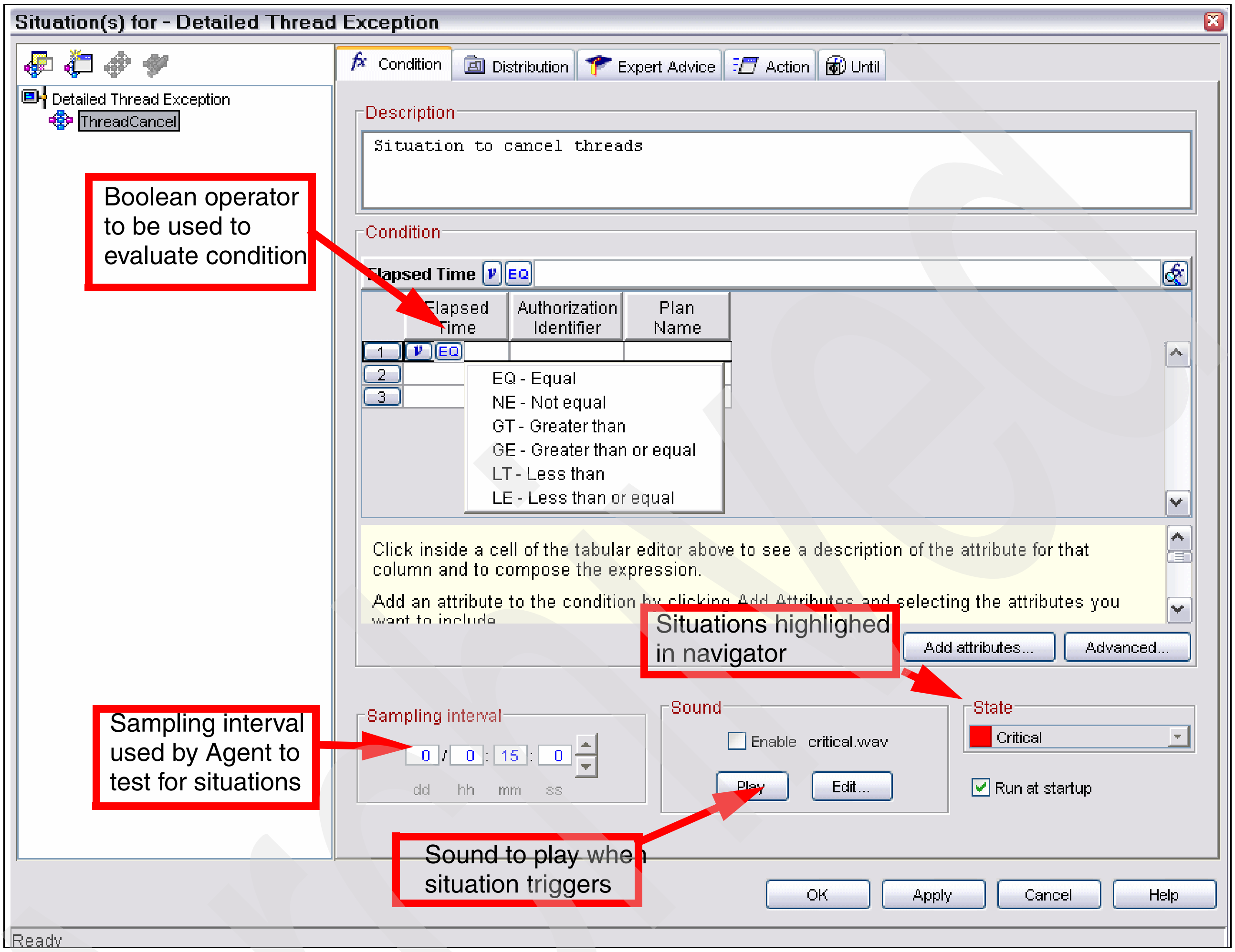


Figure 7-12 Situation condition attribute assignment

Among the formula functions supported by the Situation Editor are:

- AVG
- BETWEEN
- COUNT
- DATE
- MAX
- MIN

Operators supported include:

- EQUAL
- GREATER THAN
- LESS THAN
- NOT, OR

Once the values have been established for all of our conditions defined in our situation, we can see how these will be evaluated in Figure 7-13.

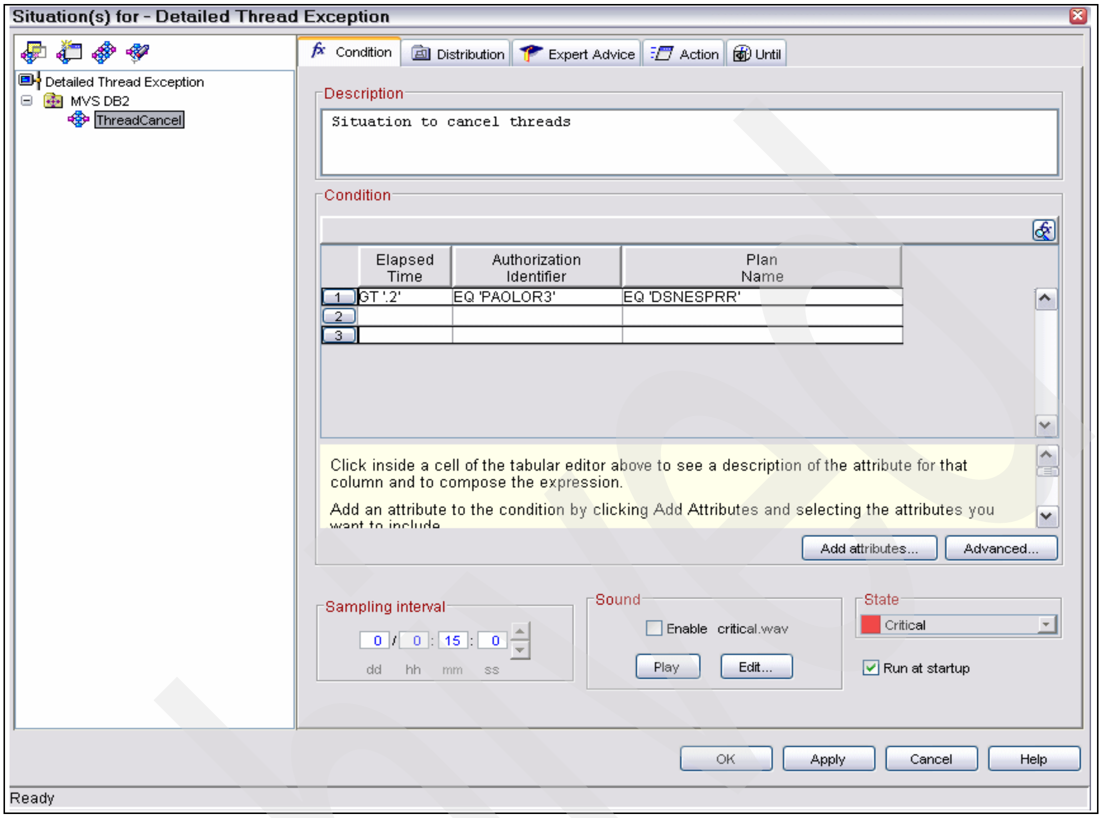


Figure 7-13 Completed conditions for situation definition

When the conditions have been established, next we need to associate this situation. This involves assigning it to one or more systems as shown in the navigator using the Distribution tab.

These assignments, as seen in Figure 7-14, can be DB2 subsystems, LPARs, or TEMS. As shown above, we are going to associate our situation only with D8F2, which is our DB2 V8 subsystem.

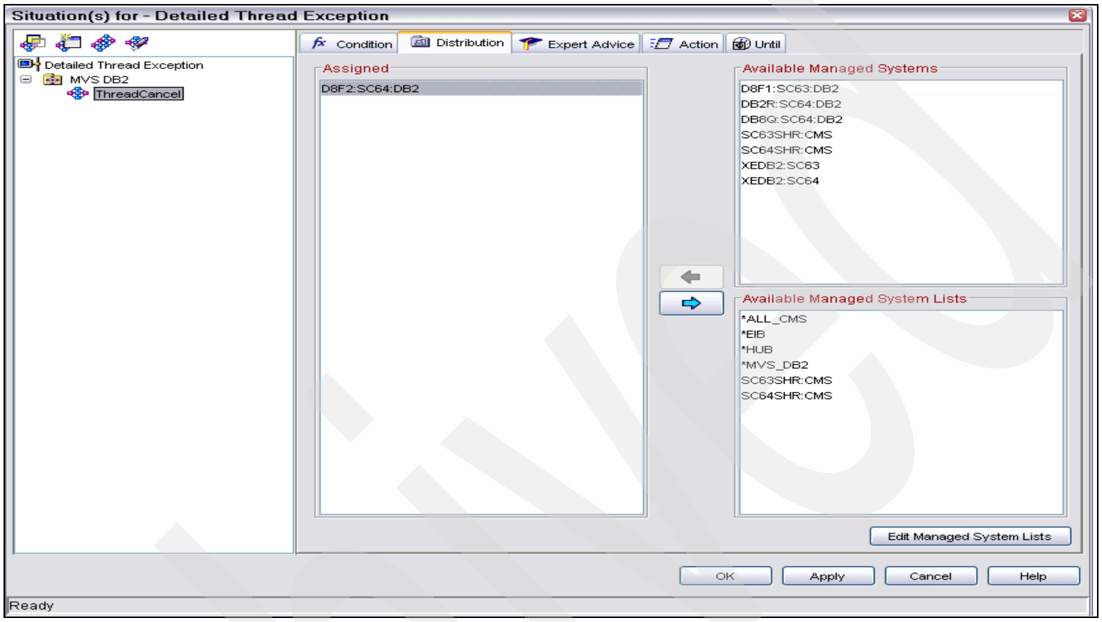


Figure 7-14 Assign Managed Systems

Situations can also have help or expert advice associated with them. This could be application specific documentation, contact and problem escalation information, references to relevant IBM documentation, etc.

There is also an HTML editor feature that allows us to build expert advise text with HTML, edit using HTLM codes, and insert HTML tags to add external or internal Internet links. We can also link to any Web page by entering in the URL without any text. Finally, there is a text wizard that helps us generate text and perform variable substitution as part of the expert advise generation function.

In our example, we will just put in a generic comment to demonstrate this feature; we are actually going to see how to build an automated action instead. Expert advice definition is performed as shown in Figure 7-15.

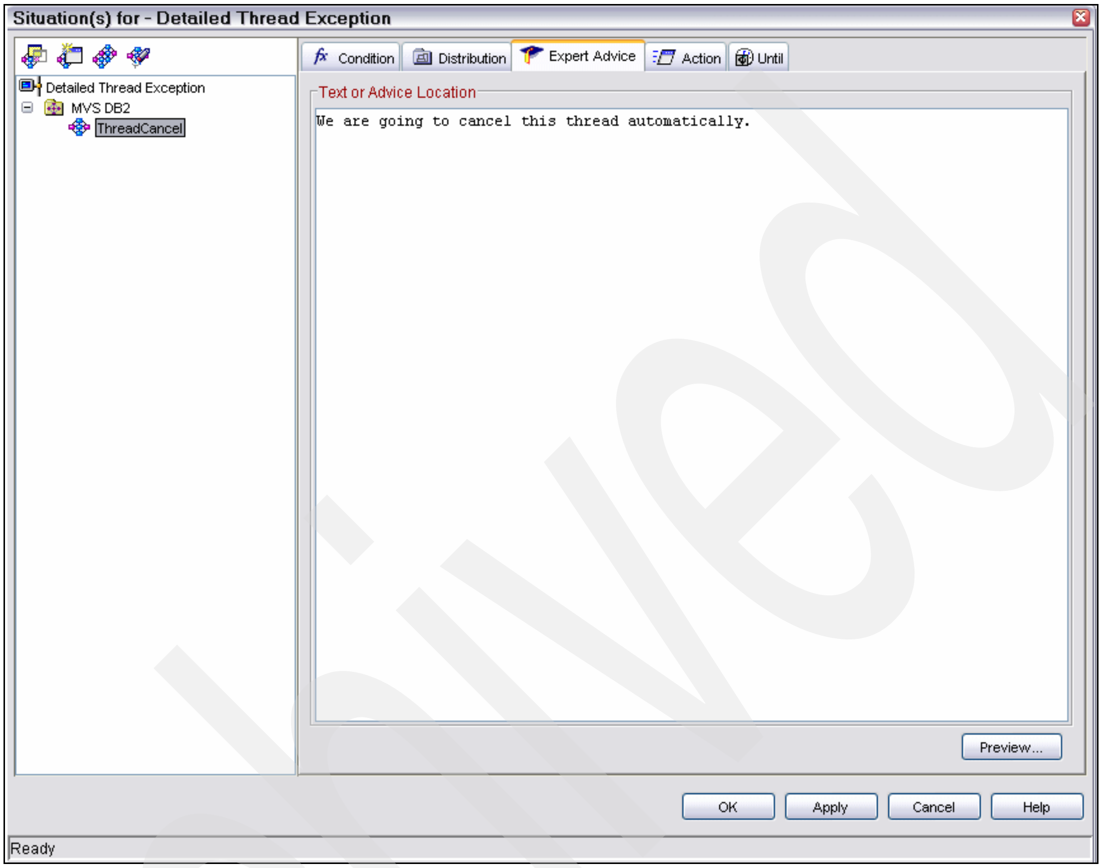


Figure 7-15 Situation definition - Expert Advice

In order to associate a command or message to a situation the action tab is used. The action tab consists of several sections. The action selection contains two radio buttons, which control the type of action that is generated. When the system command button is selected, the system command selection box is displayed, and the attribute substitute tab is shown to assist in the formulation of the command. If the universal message radio button is selected, a different selection box is displayed, this being used to format the message to be generated in association with the situation.

Additional radio buttons control the application of the action when multiple situations are detected in any one interval, or if a single situation stays true across multiple intervals, these are shown in Figure 7-16.

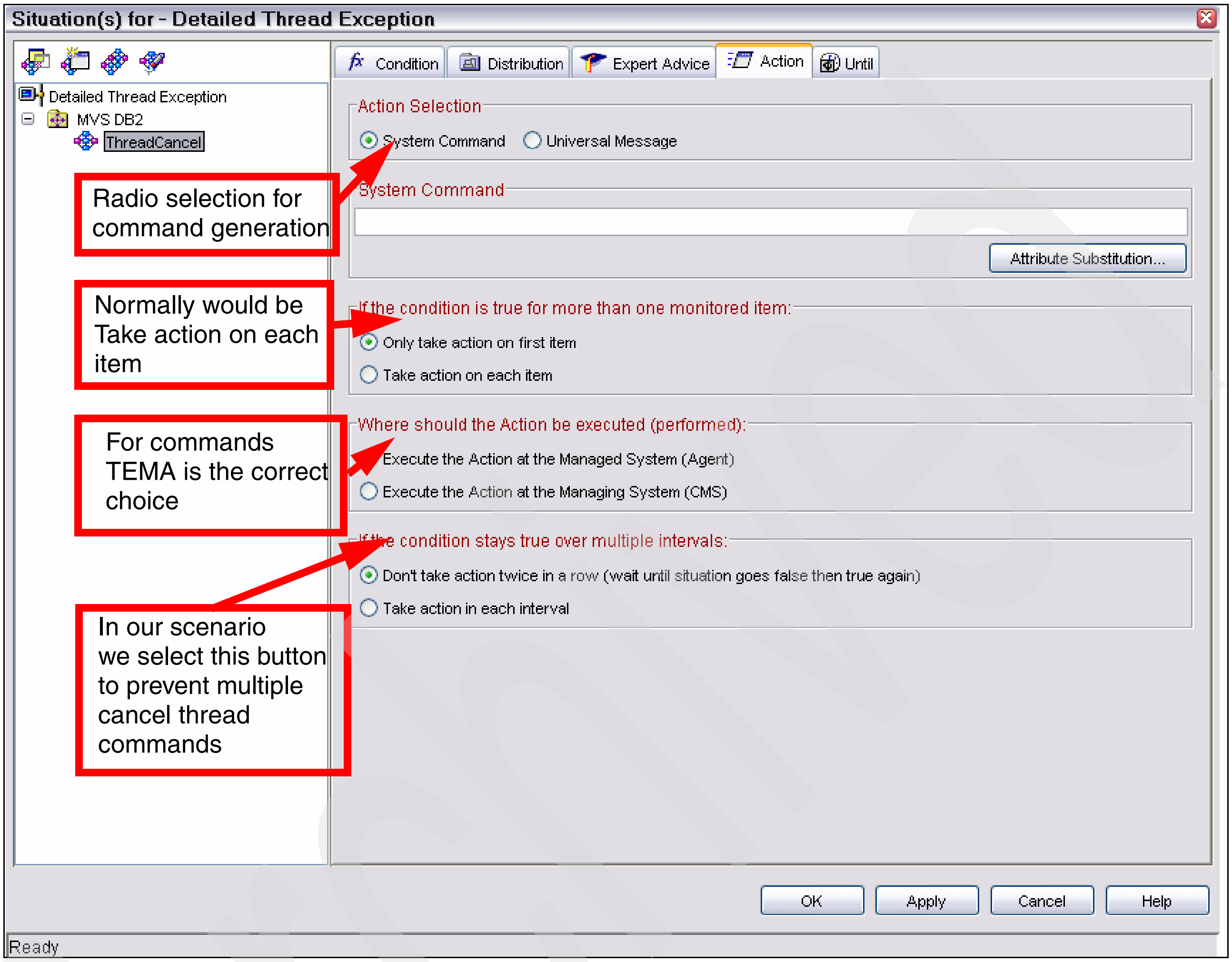


Figure 7-16 Action assignment for situation

The attribute substitution box will open a dialog that helps in choosing an attribute to associate with a specific pre-defined command. The attribute name is replaced by the actual value during execution of the action command.

In our scenario, we need to identify the thread by the thread token attribute in order to generate the CANCEL THREAD command with the correct form. The attribute association dialog is shown in Figure 7-17.

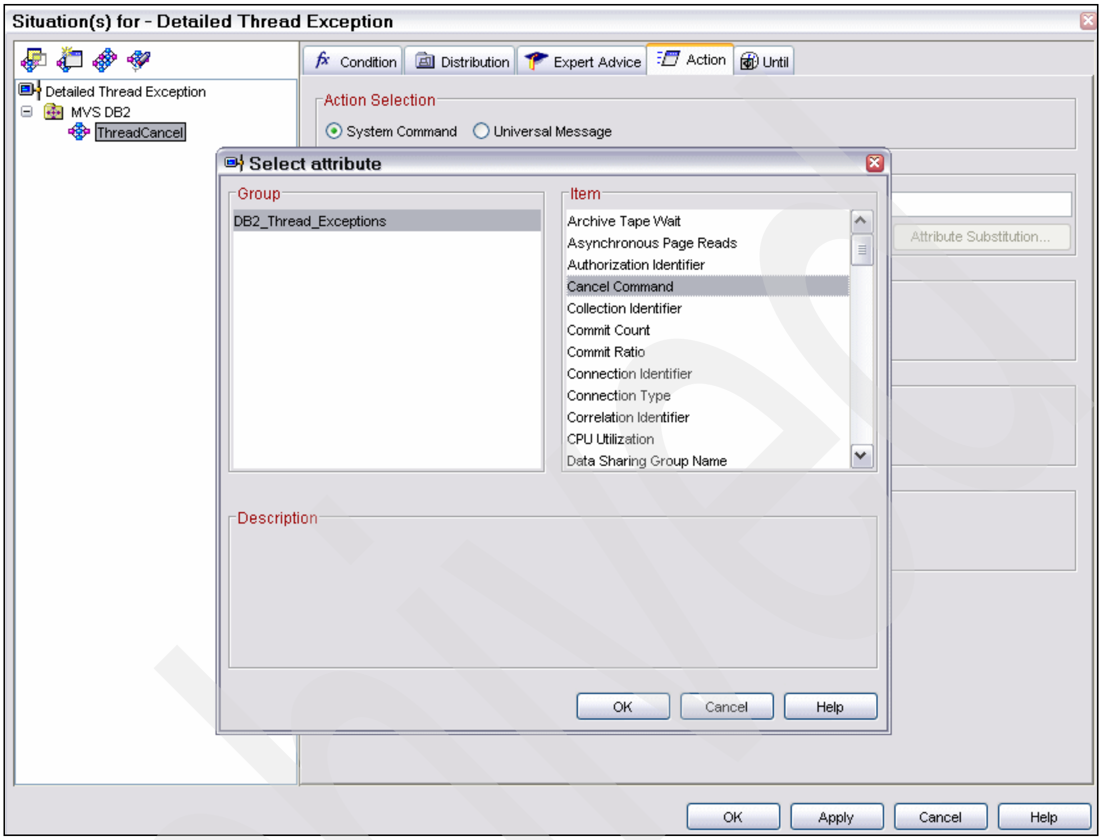


Figure 7-17 Attribute selection for action based situation

When we defined the conditions on which our situation will be based, we also could choose to start the situation checking immediately. This is done by selecting the **Run at Startup** option on the condition definition panel, this is shown in Figure 7-12. When selected, this option initiates the situation monitoring as soon as the definition is saved with either **OK** or **Apply**. When the situation has been saved, it will appear as a property when a workspace in the navigator is highlighted and right-clicked.

We also could defer the start of the situation until manually started. To manually start, disable, or delete situations, we use the situation editor, right-click to get the options displayed, and select the appropriate action. This is illustrated in Figure 7-18.

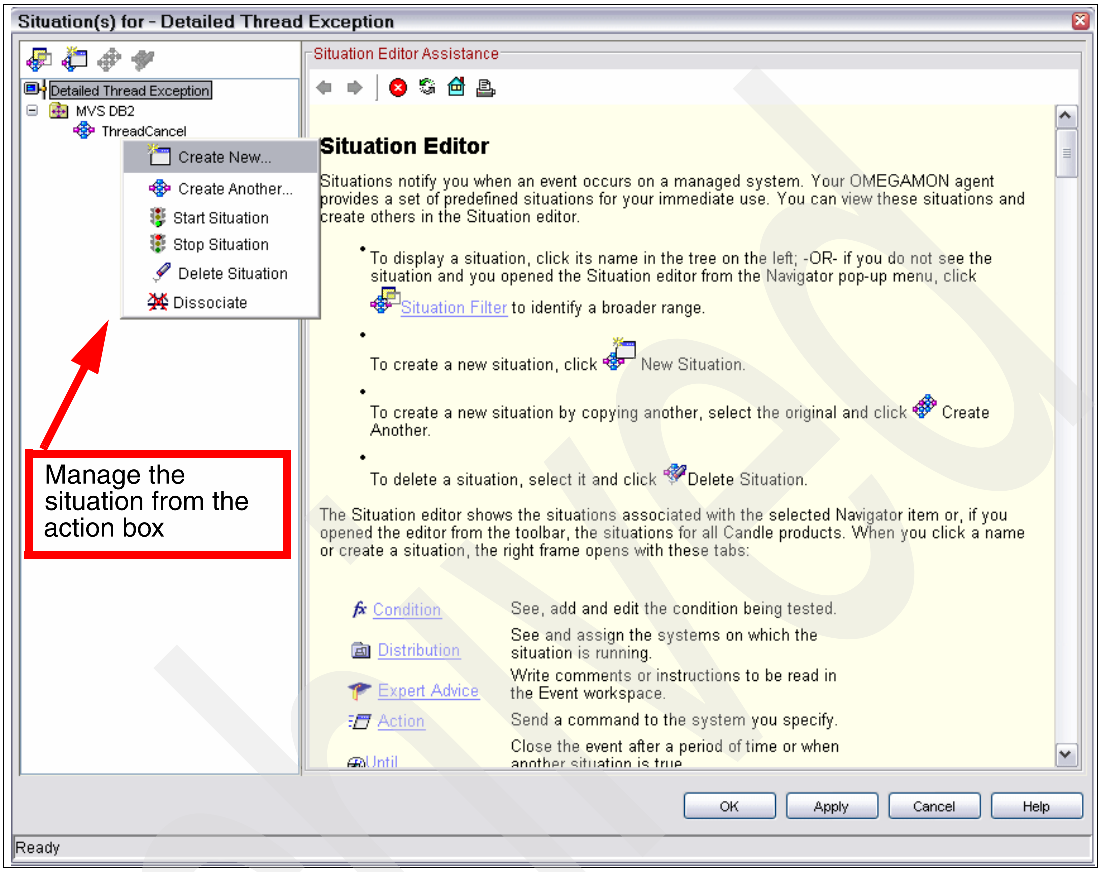


Figure 7-18 Situation editor to manage situations

Once the situation has been activated, we can get a high level view of the number of whenever we first start up the Tivoli Enterprise Portal and select the enterprise view. This workspace will open a view that shows the number of open situations that have occurred in the previous twenty-four hour period. We see an example of this in Figure 7-19. When the mouse is positioned over one of the bars represented in the chart, the number of situations and the timestamp of the last situation are shown.

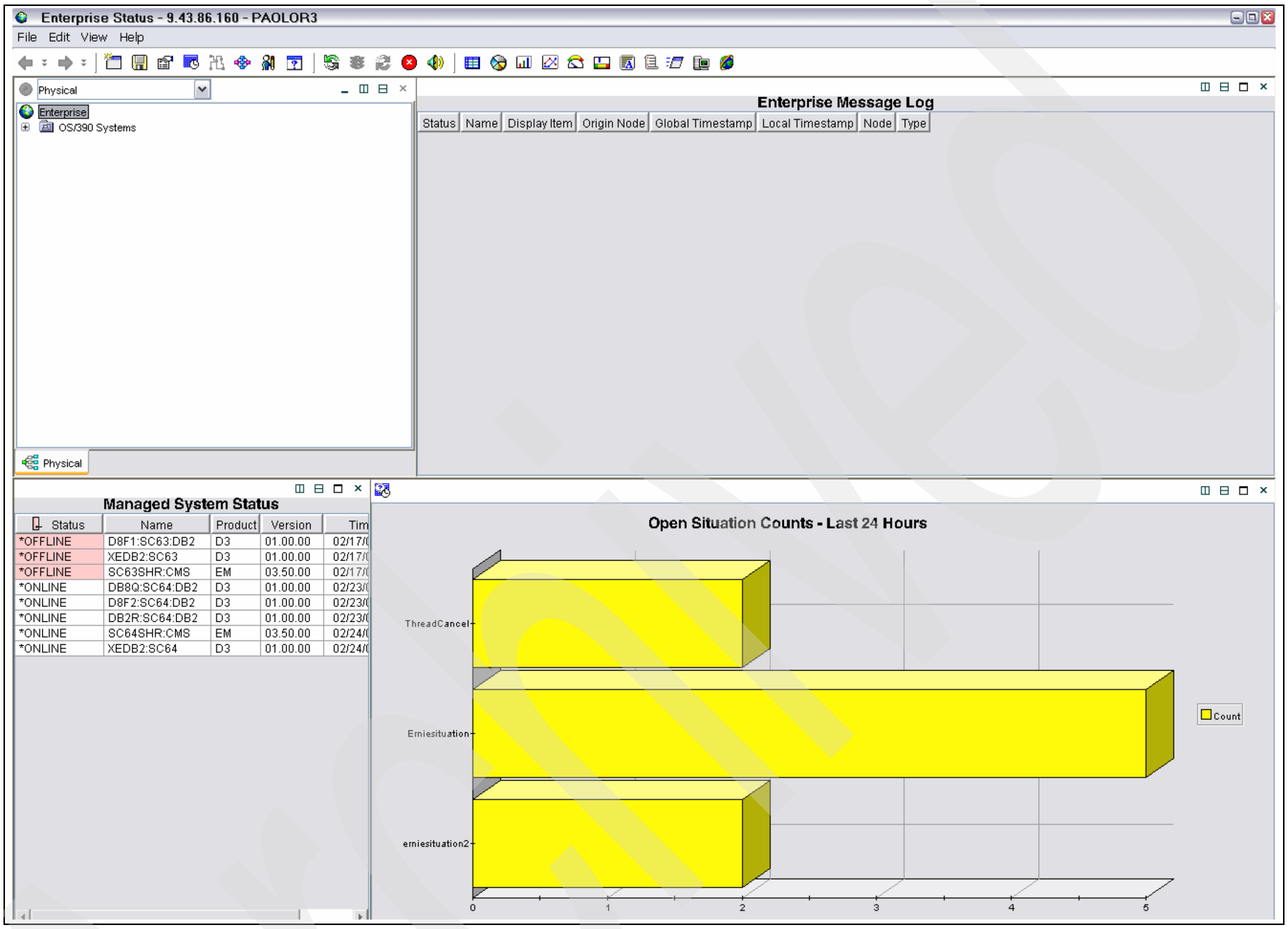


Figure 7-19 Enterprise workspace - Open Situation Counts

Using the navigator, we can next see which monitored systems show situations captured against activity by their agents. Figure 7-20 shows how the navigator displays the active situations in effect for the different managed systems.

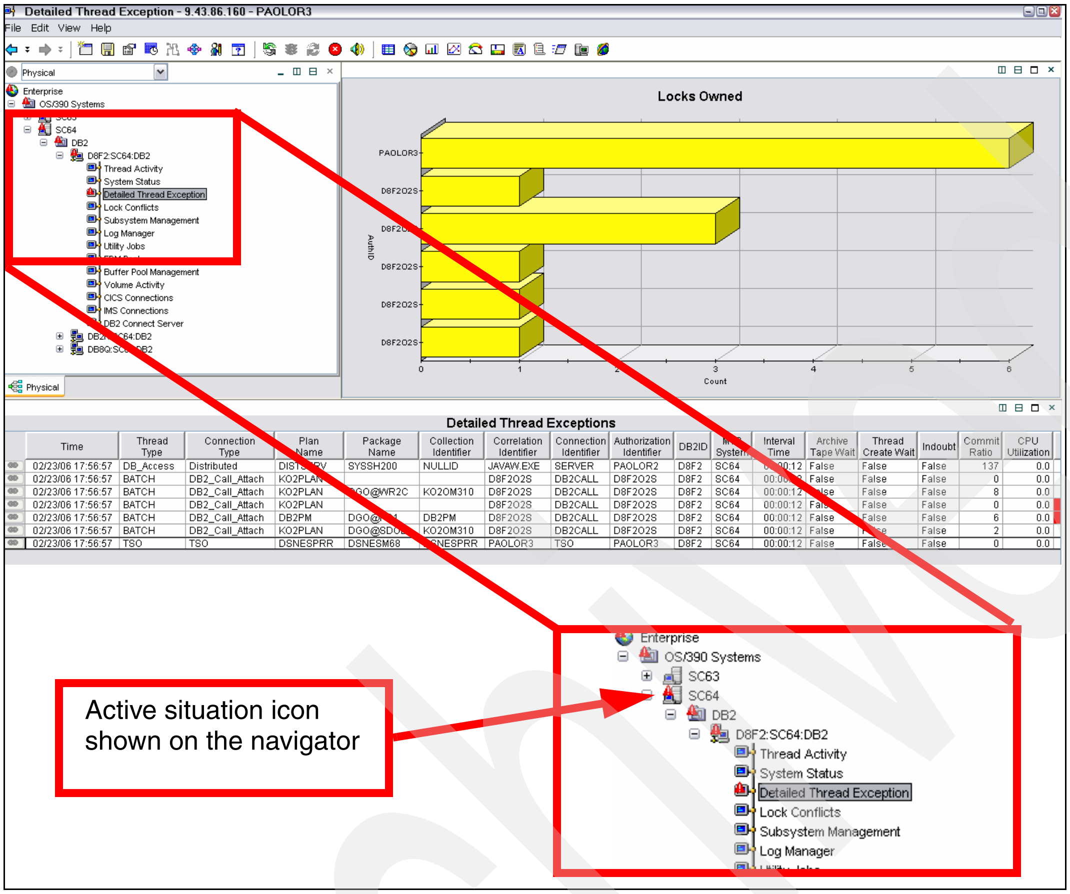


Figure 7-20 Situations viewed from the navigator

Clicking on one of the managed systems will then show the situation details workspace. There are several views with this workspace. If the situation is still active, in the current interval, there will be an entry in a table view showing the current values. In our scenario, we elected to cancel the offending thread immediately upon exceeding our threshold, so by the time we looked at the situation details, the thread had already been canceled and the situation was removed from the current situation value view.

We also are shown a table view with initial situation values, so when the situation was first detected, the row in this table view contains the information associated with the thread that triggered this event. Because not all situations are associated with an action that immediately takes effect, the situation workspace will also contain the expert advice and the take action views.

From a situation without a predefined action, we could refer to the expert advice view for additional information, possibly a problem escalation or problem determination script for example, as well as taking some manual action from the take action view.

Figure 7-21 shows this workspace and the views.

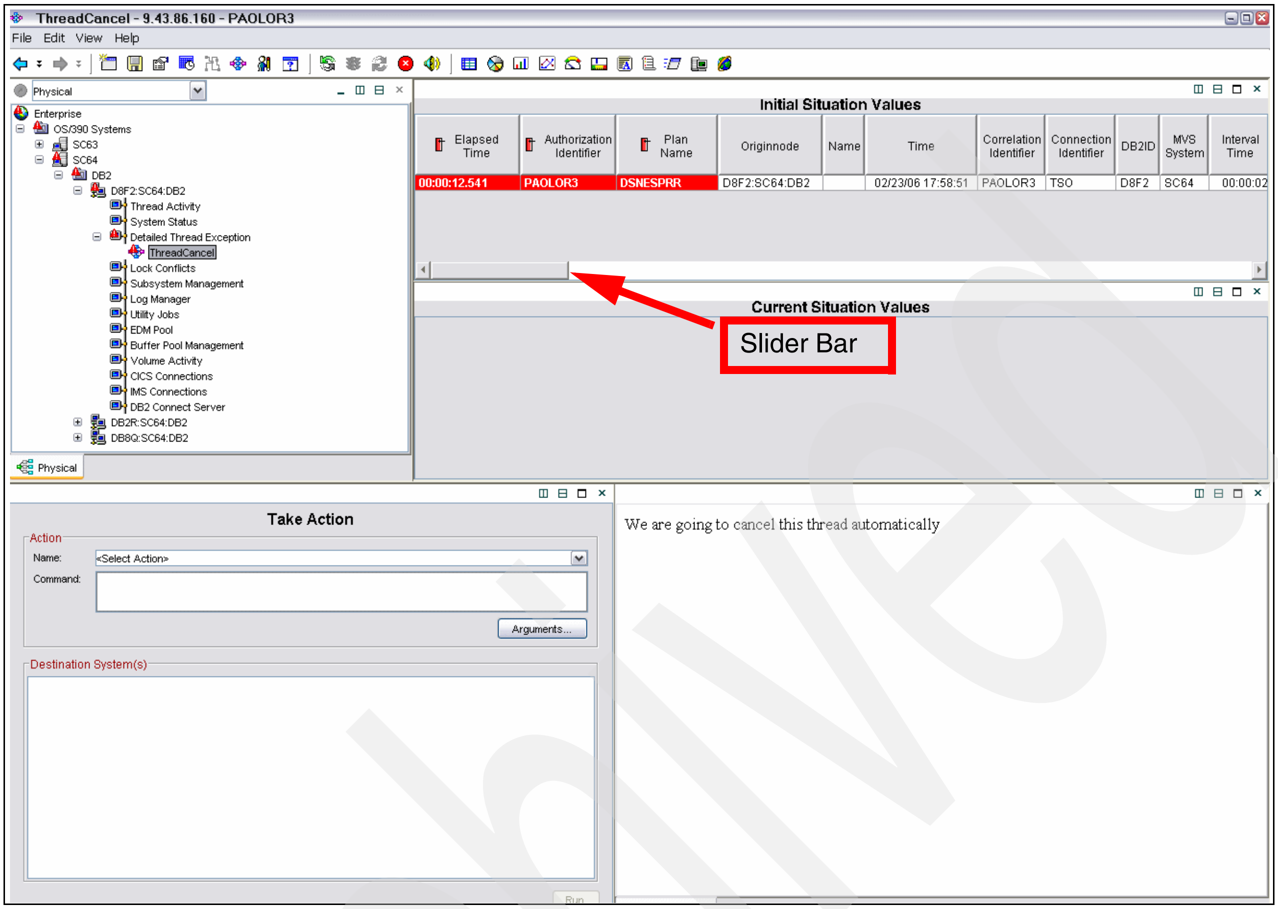


Figure 7-21 Situation values workspace

If we move the slider bar to the right, we can see additional thread related details, including the thread token. Then we need to get the thread token to verify that the cancel thread command was executed successfully. Figure 7-22 shows this information.

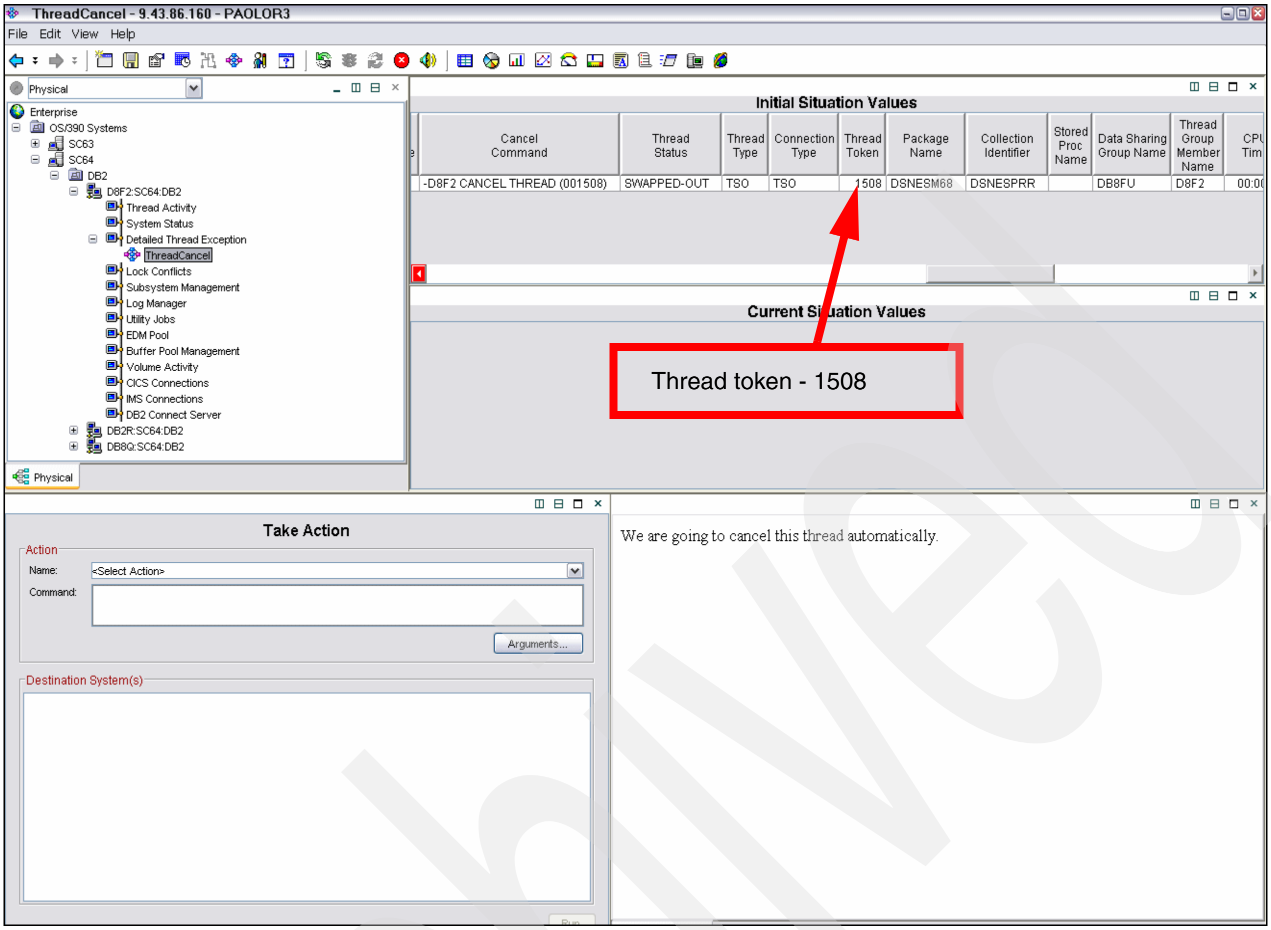


Figure 7-22 Situation values workspace - additional columns

Finally, taking the thread token, we then look into the DB2 MSTR address space log to find the CANCEL THREAD command and verify that the correct token was specified in the command generation. This is shown in Figure 7-23.

```
 Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
SDSF OUTPUT DISPLAY D8F2MSTR STC12498  DSID     2 LINE 2,813   COLUMNS 02- 81
COMMAND INPUT ===> .                                     SCROLL ===> CSR
  303             ASID=00A4 TCB=007AF138
7.59.12 STC12498  DSNV426I  -D8F2 DSNVCT THREAD '001508' HAS BEEN CANCELED
7.59.12 STC12498  DSN3201I  -D8F2 ABNORMAL EOT IN PROGRESS FOR  307
  307             USER=PAOLOR3 CONNECTION-ID=TSO CORRELATION-ID=PAOLOR3 JOBNAME
  307             ASID=00A4 TCB=007E24C8
8.03.29 STC12498  DSNL027I  -D8F2 SERVER DISTRIBUTED AGENT WITH  331
  331                        LUWID=G92B5268.0405.060223230002=1509
  331                        THREAD-INFO=PAOLOR2:BL3TFTRL:PAOLOR2:javaw.exe
  331                        RECEIVED ABEND=04E
  331                        FOR REASON=00D3003B
8.03.29 STC12498  DSNL028I  -D8F2 G92B5268.0405.060223230002=1509  332
  332                        ACCESSING DATA FOR
```

Figure 7-23 Cancel thread verification

8

Object Analysis

In this chapter we describe the Object Analysis functionality in the VTAM classic component of the Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS. This functionality provides information about DB2 related activity from a dataset perspective.

We discuss the following topics:

- Object Analysis functional overview
- Managing the Object Analysis function
- Object Analysis panels and key indicators
- Scenarios and usage examples
- Overhead and performance considerations

8.1 Object Analysis functional overview

Often the first indicator of a DB2 performance problem is manifested in the I/O activity being performed on a particular DASD volume or DB2 table space or index space. Being able to isolate this activity down to a granular level can often lead to an understanding of a specific workload or external event that might be contributing to an I/O related DB2 problem.

The Object Analysis function of Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS helps provide you information to aid in the analysis of DB2 object allocations, linear VSAM dataset extend activity, I/O activity at the DASD volume level, and object activity from a DB2 perspective.

Note: Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS only provides Object Analysis information for objects currently allocated and active to the DB2 subsystem. Information for objects that are closed is not available.

8.2 Managing the Object Analysis function

Object Analysis can be started in several ways. During the ICAT configuration process, the KD260OS panel provides for a specification of the AUTOSTART function for Object/Volume Analysis. When this function is selected, with a value of “Y”, the automatic startup of Object Analysis is then enabled when the Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS server activates.

Optionally, the Object Analysis component can be started manually via the START OBJECT ANALYSIS COLLECTORS panel.

Figure 8-1 shows panel ZOJIN, used to start Object Analysis Collection.

```
________________ ZOJIN    VTM     O2       V310./C DB8A 01/31/06 12:34:43   2
>  Help PF1                                                        Back PF3
>  O.F
>       OBJECT ANALYSIS:  Enter a selection letter on the top line.
>
>  *-START                                                           B-STOP
===============================================================================
>                      START OBJECT ANALYSIS COLLECTORS
>
> To START Object Analysis and Volume Analysis Collectors, remove the '>'
> preceeding the following command and press the ENTER key.
>
> To modify the collection interval time used by Object Analysis and Volume
> Analysis collectors change the default collection interval of 5 minutes to
> the value desired.  The interval selected must be from 1 to 1440.
>
>OJIN INTERVAL=5
```

Figure 8-1 Start Object Analysis Collection panel

By default, this function is shipped with a security level of 3, and requires that you enter a level 3 password in order to successfully complete the startup. For the installation that chooses to secure Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS functions with external security, you must have the appropriate resource class definition attached to your Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS logon identifier.

When starting the Object Analysis function, you must also select a collection interval time, expressed in minutes. The interval value selected must be between 1 and 1440. In order to start Object Analysis, you must also ensure that Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS Event Collection Manager (EVENTMGR) is activated. To determine if EVENTMGR is active, you can issue the following MVS modify command from the SDSF Console Interface, replacing `DB8A02S` with the name of your Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS server address space:

`F DB8A02S, DISPLAY`

Once this command executes, review the command output shown in the SDSF OPERLOG display. The output from this command is shown in Figure 8-2.

```
 SDSF OPERLOG  DATE 01/31/2006     1 WTOR                       COLUMNS 31- 110
 COMMAND INPUT ===>                                            SCROLL ===> CSR
:41:45.59 PAOLOR3  00000290  F DB8A02S,DISPLAY
:41:45.61 STC12532 00000090  CI0760: PROCESSING COMMAND
:41:45.61 STC12532 00000090  CI0720: PROCESS MESSAGES FOLLOW
:41:45.61 STC12532 00000090  CI0543: THE FOLLOWING TASK IDS ARE ACTIVE:
:41:45.61 STC12532 00000090  CI0594:  ID=OBVTAM             PROGRAM=KOBVTAM
:41:45.61 STC12532 00000090  CI0594:  ID=H2DB8A             PROGRAM=KO2HWLMB
:41:45.61 STC12532 00000090  CI0594:  ID=EVENTMGR           PROGRAM=KO2EINTB
:41:45.61 STC12532 00000090  CI0594:  ID=PESERVER           PROGRAM=DGOVMSTR
:41:45.61 STC12532 00000090  CI0594:  ID=COMMCOLL           PROGRAM=KO2AINIB
:41:45.61 STC12532 00000090  CI0700: CANDLE COMMON INTERFACE READY FOR COMMANDS
11714 *088 DFS996I *IMS READY*  IMSG
```

Figure 8-2 EVENTMGR Display results

Ensure that the EVENTMGR task ID is shown as active. If it is not active, you must first start the Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS Event Collection Manager before activating the Object Analysis function. From the SDSF Console Interface you can issue the following command to start EVENTMGR, substituting `DB8A02S` with the name of your Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS server address space:

`F DB8A02S, S EVENTMGR`

Review the output of the command in the SDSF OPERLOG display, and in particular ensure that the message `KO2E3001I EVENTMGR INITIALIZATION SUCCESSFUL` is shown. Verify your expected results against the sample command output shown in Figure 8-3.

```
 --------------------------------------------------------------------------------
 SDSF OPERLOG  DATE 01/26/2006     1 WTOR              CHARS 'S EVENTMGR' FOUN
 COMMAND INPUT ===>                                            SCROLL ===> CSR
:03:12.07 PAOLOR3  00000290  F DB8A02S,S EVENTMGR
:03:12.09 STC12275 00000090  CI0760: PROCESSING COMMAND
:03:12.09 STC12275 00000090  CI0720: PROCESS MESSAGES FOLLOW
:03:12.09 STC12275 00000090  CI0510: ATTACH PROCESSING - TASK ID=KO2EINTB
:03:12.09 STC12275 00000090  CI0700: CANDLE COMMON INTERFACE READY FOR COMMANDS
:03:12.10 STC12275 00000090  KO2E3000I EVENTMGR INITIALIZATION IN PROGRESS
:03:12.10 STC12275 00000090  KO2E3001I EVENTMGR INITIALIZATION SUCCESSFUL
```

Figure 8-3 Event Manager startup

Once you have determined that the Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS Event Collection Manager subtask is running, you can then start the Object Analysis function by issuing one the following commands.

To start Object Analysis using the default collection interval established during the ICAT customization process, use this command:

F DB8A02S,F EVENTMGR,START DB2=DB8A

In this example, substitute DB8A02S with the name of your Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS server address space, and DB8A with the identifier of a DB2 subsystem monitored by your server address space.

To specify a different collection interval for Object Analysis, use a command in this format:

F DB8A02S, F EVENTMGR,START DB2=DB8A,INTERVAL=15,THREAD=YES

Figure 8-4 illustrates the resulting messages from a successful EVENTMGR subtask startup.

```
 --------------------------------------------------------------------------------
 SDSF OPERLOG  DATE 01/26/2006     1 WTOR                          COLUMNS 52- 131
 COMMAND INPUT ===>                                               SCROLL ===> CSR
000290  F DB8A02S,F EVENTMGR,START DB2=DB8A
000090  CI0760: PROCESSING COMMAND
000090  CI0720: PROCESS MESSAGES FOLLOW
000090  KO2E3002I EVENTMGR MODIFY PROCESSING IN PROGRESS
000090  CI0551: MODIFY MESSAGE SENT TO TASK
000090  CI0700: CANDLE COMMON INTERFACE READY FOR COMMANDS
000090  KO2E3040I VOLUME ANALYSIS INITIALIZATION COMPLETE
000090  KO2E3070I OBJECT ANALYSIS - INITIALIZATION IN PROGRESS FOR DB2=DB8A
000090  KO2E3050I OBJECT ANALYSIS - PHASE1 INITIALIZATION COMPLETE FOR
        DB2=DB8A
000090  KO2E3051I OBJECT ANALYSIS - PHASE2 INITIALIZATION COMPLETE FOR
        DB2=DB8A
000090  KO2E3071I OBJECT ANALYSIS - INITIALIZATION SUCCESSFUL FOR DB2=DB8A
```

Figure 8-4 EVENTMGR subtask initialization

The message: KO2E3071I OBJECT ANALYSIS - INITIALIZATION SUCCESSFUL FOR DB2=DB8A verifies that the Object Analysis subtask is active and that I/O activity is now being collected.

Tip: In an environment where there are significant levels of I/O activity on monitored DASD volumes, evaluate the cost of activating the Object Analysis function versus the benefits you can obtain. It is our recommendation that Object Analysis not be specified to automatically start in the AUTOSTART configuration, but be started as needed to measure specific workloads or help manage isolated performance situations. Object Analysis can either be started manually as shown, or the previously referenced commands can be executed through your installation automated operations product.

8.3 Object Analysis panels and key indicators

After Object Analysis has started, data collection occurs for the specified period of time (or interval), at the end of which, the data contained in the Object Analysis counter is reset (data wraps). If you wish to determine where you stand in any single active collection interval, the field *Interval Time* contains the length of the collection interval; the field *Interval Elapsed* denotes the time into the current interval. Once the collection interval is reached, the counters collected by Object Analysis are reset, and a new collection interval is started. Because of this behavior, Object Analysis is best used as a tool for reactive problem determination, given that the data is not stored across collection intervals.

On entry to the Object Analysis function, the Object Allocation Summary panel is displayed. This contains a list of Databases that are allocated to this particular DB2 subsystem and some summary information as shown in Figure 8-5.

```
 ________________ ZOJAS    VTM     O2       V310./C DB8A 01/31/06 19:31:45   2
> Help PF1      Back PF3      Up PF7      Down PF8     Sort PF10     Zoom PF11
> O.A
>       OBJECT ANALYSIS:  Enter a selection letter on the top line.
>
> *-OBJECT ALLOCATIONS        B-OBJECT ACTIVITY            C-VOLUME ACTIVITY
> D-EXTEND ACTIVITY           E-DISPLAY OPTIONS            F-COLLECTOR CONTROL
===============================================================================
>                           OBJECT ALLOCATION SUMMARY
 OJAS
+ Total Databases = 8        Total Spaces = 173        Total Datasets = 190
+ *
+ Database   Spaces   Tblsp    Indxs    DSNs    Extents   Exts/DSN    Max Exts
+ --------   ------   ------   ------   ------  -------   --------    --------
+ ADBDCH          1        1        0        1        0         .0           0
+ DSNATPDB        4        4        0        4        0         .0           0
+ DSNDB01        12        5        7       12        0         .0           0
+ DSNDB06       102       16       86      102        0         .0           0
+ DSNDB07         2        2        0        2        0         .0           0
+ DSN8D81A        7        3        4       12        0         .0           0
+ GLWSAMP        43       10       33       55        0         .0           0
+ TEMPDB          2        2        0        2        0         .0           0
===============================================================================
```

Figure 8-5 Object Analysis Object Allocation Summary

This object list is sorted by the default sort column of Database column. To resort the list in a different order, position the cursor on any value under the desired sort column and press PF10 (SORT).

If more detailed information about a specific database is desired, position the cursor under the selected database name and press PF11(ZOOM).

On all Object Activity and Volume Activity panels of the Object Analysis component, you can filter the presented list of items by using the Display Options panel shown in Figure 8-6.

```
 ________________ ZOJOO    VTM     O2       V310./C DB8A 01/31/06 21:20:22   2
>  Help PF1                                                          Back PF3
>  O.E
===============================================================================
>                            DISPLAY OPTIONS FOR
>        OBJECT ACTIVITY SUMMARY AND OBJECT ALLOCATION SUMMARY DISPLAYS
 OJOO
+  Following options applied to both displays ;
:  DATABASE=________      Filter display output by database name
+
+  Following options applied to OBJECT ALLOCATION SUMMARY display
:  EXTSDSN>___            Filter display output by extents/dataset ratio
:  MAXEXTS>___            "                   " by max extents/dataset
+
+  Following options applied to OBJECT ACTIVITY SUMMARY display
:  PERCGETP>___           Filter display output by % of getpage activity
:  PERCIO>___             "                   " by % of I/O activity
:  RATE=___               Display activity rates or counts (yes/no)
===============================================================================
```

Figure 8-6 Object Analysis - Display Options

Particularly useful in an environment where there is a diverse workload performing I/O activity on a number of different page sets, the Display Options Specifications panel allows you to filter on a number of different indicators. Note that this option only filters the display results and does not apply any filtering during the Object Analysis collection process.

If you refer back to Figure 8-5 on page 129, you notice a number of options displayed on the top section of the screen. As we have seen earlier in the book, the option specific to the current selected panel is shown with an “*”, and other available options are selected by entering the appropriate letter on the command line at the top of the screen.

Object Analysis provides a view of I/O activity by database. When sorted by `% of I/O`, it provides you a quick picture of which database is incurring the highest level of DB2 initiated activity. See Figure 8-7.

```
 ________________ ZOJOS    VTM     O2       V310./C DB8A 01/31/06 20:59:13   2
> Help PF1      Back PF3      Up PF7      Down PF8     Sort PF10     Zoom PF11
> O.B
>       OBJECT ANALYSIS:  Enter a selection letter on the top line.
>
> A-OBJECT ALLOCATIONS        *-OBJECT ACTIVITY           C-VOLUME ACTIVITY
> D-EXTEND ACTIVITY           E-DISPLAY OPTIONS           F-COLLECTOR CONTROL
===============================================================================
>                           OBJECT ACTIVITY SUMMARY
 OJOS
+ Interval Time   = 00:05:00                        Interval Elapsed = 00:02:54
+ Total Getpage   =      286                        Total I/O        =        0
+ *
+          % of    % of   Getp               Sync     Pre      Async    Other
+ Database  Getp    I/O    per RIO   Getpage  Read     Fetch    Write    Write
+ --------  ------  ------ --------- -------  -------  -------  -------  -----
+ DSNDB06   100.0%     .0%     286.0     286        0        0        0      0
===============================================================================
```

Figure 8-7 Object Analysis Object Activity

Notice in the preceding figure that the ratio of I/O activity versus getpage activity. In this example, 100% of the getpage requests are being satisfied with no I/O, in other words, all of the necessary pages were already resident in the bufferpool.

In order to further isolate the activity to a particular table space, select the ZOOM option by pressing the PF11 key, then drill down to a list of table spaces in the database that had activity during the collection interval. See Figure 8-8.

```
>
>       OBJECT ANALYSIS:  Enter a selection letter on the top line.
>
> A-DATABASE ALLOCS  *-DATABASE ACTIVITY   C-THREAD ACTIVITY  D-DISPLAY OPTIONS
===============================================================================
>                              DATABASE ACTIVITY
 OJ01
+ Interval Time   = 00:05:00                      Interval Elapsed = 00:03:49
+ Total Getpage   =      378                      Total I/O        =        1
+
+ Database = DSNDB06
+ *
+          % of    % of   Getp                Sync     Pre      Async    Other
+ Spacenam  Getp    I/O    per RIO    Getpage  Read     Fetch    Write    Write
+ --------  ------  ------ ---------  -------  -------  -------  -------  -----
+ DSNADH01    1.5%     .0%       6.0        6        0        0        0      0
+ DSNATX02    3.1%     .0%      12.0       12        0        0        0      0
+ DSNAUH01    2.1%     .0%       8.0        8        0        0        0      0
+ DSNDCX01     .5%  100.0%       2.0        2        1        0        0      0
+ DSNDSX01    1.0%     .0%       4.0        4        0        0        0      0
+ DSNDTX01    2.1%     .0%       8.0        8        0        0        0      0
+ DSNDTX02    1.0%     .0%       4.0        4        0        0        0      0
```

Figure 8-8 Database Activity - Table Space Activity Summary

The example displays the active table spaces for the selected database. In this example we can see the percentage of getpage activity expressed not as a number within the selected database, but instead as a reflection of the getpage activity occurring within the entire subsystem for the current Object Analysis interval.

For the situation where the spacename refers to a segmented table space, with multiple tables, we might need to look at the activity from a different perspective, possibly starting at Thread Activity, looking at a single active SQL statement (which gives us the table name in the SQL predicate), and then observing object activity by launching Object Analysis from within the Thread Detail menu. We will discuss this scenario in more detail later.

In the case of a partitioned table space, we might be interested to see if I/O activity is skewed to a particular partition. If we elect to ZOOM from the table space activity display, we see the activity spread across each dataset piece. Figure 8-9 shows an example of this display.

```
 ________________ ZOJ02    VTM     02       V310./C DB8A 01/31/06 21:09:58   2
> Help PF1        Back PF3         Up PF7         Down PF8         Sort PF10
>
>       OBJECT ANALYSIS:  Enter a selection letter on the top line.
>
> A-SPACENAME ALLOCS          *-SPACENAME ACTIVITY          C-THREAD ACTIVITY
===============================================================================
>                             SPACENAME ACTIVITY
 OJO2
+ Interval Time   = 00:05:00                      Interval Elapsed = 00:03:39
+ Total Getpage   =       12                      Total I/O        =       0
+
+ Database = DSNDB06         Spacename = DSNATX02
+ *
+       % of     % of   Getp                 Sync      Pre       Async    Other
+ DSN   Getp     I/O    per RIO    Getpage   Read      Fetch     Write    Write
+ ---   ------   ------ ---------  --------  --------  --------  -------  -----
+ 001   100.0%      .0%      12.0       12         0         0         0      0
===============================================================================
```

Figure 8-9 Spacename Activity

Shifting focus from the physical object perspective, we can also use Object Analysis to help understand from a thread perspective what currently active threads are involved in an I/O related issue. This information would then allow us to possibly locate the thread as it is currently executing from the Thread Activity screen, and in certain situations might lead us to being able to attribute the I/O activity to a specific SQL statement. We can see an example of this in Figure 8-10.

One additional observation, you can navigate to the Thread Activity screen via ZOOM from both Activity By Database as well as Activity By Spacename, although it would appear that the table space point of reference would be more valuable in most problem analysis situations.

```
 ________________ ZOJT7    VTM     02       V310./C DB8A 01/31/06 21:14:54   2
> Help PF1        Back PF3         Up PF7         Down PF8         Sort PF10
>
===============================================================================
>                          THREAD ACTIVITY BY SPACENAME
 OJT7
+ Interval Time   = 00:05:00                      Interval Elapsed = 00:03:35
+ Total Getpage   =        6                      Total I/O        =       0
+
+ Database = DSNDB06        Spacename = DSNADH01
+ *
+                                            Sync     ------ Prefetch I/O -----
+ Planname  Authid    Correlation   Getpage  Read       Seq      List   Dynamic
+ --------  --------  ------------  -------  -------  -------  -------  -------
+ ADB       PAOLOR3   PAOLOR3             6        0        0        0        0
===============================================================================
```

Figure 8-10 Thread Activity by Spacename

This example of correlating thread activity against the Object Analysis data requires you to start from a volume or object level and drill down. A different frame of reference might be one where we start from the list of active threads and then drill down to see what objects are being referenced by this particular thread. There are several points to keep in mind. The first consideration is that Object Analysis must be active in order for this option to be valid. The second consideration is that only objects referenced by that thread during the current Object Analysis collection period are displayed. From the Thread Details screen, if we choose option "O" we then launch the *Objects Used by Thread* screen. We can see what this information view looks like in Figure 8-11.

```
 ________________ ZTOBJ    VTM     O2       V310./C DB8A 02/02/06 12:28:35   2
> Help PF1                                                         Back PF3

>       THREAD INFORMATION:  Enter a selection letter on the top line.

===============================================================================
> A-THREAD DETAIL B-LOCK COUNTS C-LOCK WAITS     D-LOCKS OWNED  E-GLOBAL LOCKS
> F-CURRENT SQL   G-SQL COUNTS  H-DISTRIBUTED    I-BUFFER POOL  J-GROUP BP
> K-PACKAGES      L-RES LIMIT   M-PARALLEL TASKS N-UTILITY      *-OBJECTS
> P-CANCEL THREAD Q-DB2 CONSOLE R-DSN ACTIVITY   S-APPL TRACE   T-ENCLAVE
> U-LONG NAMES
>                            OBJECTS USED BY THREAD
 PLAN
+ Thread:  Plan=ADB       Connid=TSO      Corrid=PAOLOR3      Authid=PAOLOR3
+ Attach:  TSO            Userid=PAOLOR3                      Region asid= 16K
+ Package: ADBMAIN        Collection=ADBL
 tobj
+                                            Sync     ------ Prefetch I/O -----
+ Database  Spacenam  Dsn  Volume   Getpage  Read       Seq      List   Dynamic
+ --------  --------  ---  ------   -------  -------  -------  -------  -------
+ DSNDB06   DSNADH01  001                 9        0        0        0        0
+ DSNDB06   DSNATX02  001                18        0        0        0        0
+ DSNDB06   DSNAUH01  001                12        0        0        0        0
+ DSNDB06   DSNDCX01  001  SBOX1P         4        2        0        0
```

Figure 8-11 Thread display - Objects Used by Thread

8.4 Scenarios and usage examples

As mentioned previously, it is our recommendation that for most production workloads, you should not autostart the Object Analysis component, but rather start it up manually on an as-needed basis. So, let us consider some performance scenarios where you might need an understanding of the I/O activity.

8.4.1 Object placement

Many customers have implemented DB2 managed objects and let SMS determine the placing of the physical datasets based on the defined constructs. Most of the time, SMS makes good choices in object placement, but occasionally poor object placement choices can occur. One example of this might be where an highly active index and a table space part are placed on the same physical device. What can occur is that the workload mix drives I/O up for both pagesets and they start to experience response time degradation.

A more likely scenario might be when there is heavy I/O activity against a dataset that is not owned by DB2. Looking at this situation strictly from a DB2 perspective, we might expect to notice symptoms of I/O delay from the DB2 data, for example, high synchronous or asynchronous I/O times. If we use Object Analysis to determine which volume is involved with the suspected I/O issue, for example, from the volume column in the *Objects Used by Thread* screen, we could then take a look at the activity from the volume perspective by selecting the Volume Activity summary screen by choosing option “C”. This then presents a list of volumes as we see in Figure 8-12.

```
 ________________ ZOJVS    VTM     O2      V310./C DB8A 02/02/06 13:44:54   2
> Help PF1      Back PF3      Up PF7      Down PF8     Sort PF10     Zoom PF11
> O.C
>       OBJECT ANALYSIS:  Enter a selection letter on the top line.
>
> A-OBJECT ALLOCATIONS         B-OBJECT ACTIVITY           *-VOLUME ACTIVITY
> D-EXTEND ACTIVITY            E-DISPLAY OPTIONS           F-COLLECTOR CONTROL
===============================================================================
>                            VOLUME ACTIVITY SUMMARY
 OJVS
+ Interval Time   = 00:05:00                      Interval Elapsed   = 00:00:43
+ *
+            Vol     Vol     Total    DB2       % DB2   Alloc   Alloc   Exts/
+ Volume Unit Util%  Serv    I/O      I/O       I/O     DSNs    Exts    DSN
+ ------ ---- ------ ------  -------  -------   ------  ------  ------  ------
+ SBOX1A 3390   1.0%      .0        2        2    33.3%       0       0      .0
+ SBOX1P 3390   3.0%      .0        4        4    66.6%       0       0      .0
===============================================================================
```

Figure 8-12 Volume Activity Summary

Of particular interest would be the columns showing Volume Utilization % and % DB2 I/O. In the case of excessive I/O occurring outside of DB2, we would possibly expect to see a High number in the Volume Utilization % and a low number in the % DB2 I/O. This information might then lead us to investigate the activity from a z/OS perspective, possibly using RMF™ or the IBM Tivoli OMEGAMON for z/OS V3.1 product. If the two percentages were close to equivalent, we might conclude that our problem is isolated to DB2 workload and might concentrate on looking at the Object Activity by thread screen to isolate the demand to a specific application thread.

8.5 Overhead and performance considerations

Object Analysis is designed to run as a sampling collector. As mentioned in Figure 8.1 on page 126, take into account the extra CPU cost for continually running this component under heavy workloads. The other characteristic of Object Analysis, when running in a continuous mode, is that the data buckets are being reset at each interval expiration and reused. In other words, we are constantly gathering data into the collection interval, and in all probability having it thrown away without review.

Our recommendation is that you autostart the EVENTMGR subtask, this needing to be active before Object Analysis can be activated. If you need to monitor I/O activity, manually start Object Analysis as described in 8.2, “Managing the Object Analysis function” on page 126. One other consideration when starting Object Analysis is the choice to gather Object Analysis information at the thread detail level.

By default, when Object Analysis is started, thread details are also collected. As we previously discussed, this information correlates thread activity to DB2 getpage and I/O, but is a little more costly to collect. In other words, it is nice data to have, but as a general rule, more threads = more volumes = more overhead. Our second recommendation is to start Object Analysis without thread level detail collection. The form of the command needed to do this is as follows:

```
F DB8A02S, F EVENTMGR,START DB2=DB8A,INTERVAL=15,THREAD=No
```

If your initial investigation requires you to investigate the I/O activity from a thread level, then start Object Analysis with thread level details as follows:

```
F DB8A02S, F EVENTMGR,START DB2=DB8A,INTERVAL=15,THREAD=YES
```

In either scenario, once you have completed your investigation, we would recommend that you terminate Object Analysis as described in 8.2, “Managing the Object Analysis function” on page 126.

Once Object Analysis has started, any overhead associated with its collection is reflected in the OMEGAMON server started task. Object Analysis does not use traces, but given the potential for resource consumption, we would also suggest that automation might be created to notify the support personnel responsible for the overall health of DB2. This automation might consist of using your incumbent automated operations product to catch whenever the message `K02E3071I OBJECT ANALYSIS - INITIALIZATION SUCCESSFUL FOR DB2=DB8A` is generated. This automation could then cause other notification to take place, possibly the generation of an e-mail alert, for example.

9

Near-term history using the Classic Interface

In this chapter we look at the options available for near-term history collection using the OMEGAMON XE for DB2 PE Classic Interface. By near-term history we mean the storing of all statistics and accounting records written by DB2 over a recent period of time. In addition, we may also be collecting certain performance data at the thread level, such as dynamic SQL or sorting, locking, and scanning information.

Near-term history should be distinguished from snapshot history. With snapshot history, a real-time snapshot of the system is taken at regular intervals (say once a minute) and stored. Thus a snapshot of a long-running thread would be available at one minute intervals, as if you were watching the thread on the real-time monitor. However short-running threads might not show up at all using snapshot history, whereas near-term history records all threads that complete in the recording interval. The near-term history availability depends on the number of VSAM log data sets, their size, and the number of traces turned on. It will range from few hours to many hours depending on the amount of activity in the DB2 subsystem.

We discuss the following topics:

- Benefits of collecting near-term history
- How to access and control near-term history collection
- Performance issues with near-term history collection
- Near-term thread history
- Near-term statistics history

9.1 Benefits of collecting near-term history

In a busy environment many events are too short lived to be viewed in real time. One hundred transactions per second or more may be being processed by an online system. Deadlocks and timeouts may be resolved before there is time to look at them in detail. Batch jobs may be executing thousands of SQL calls per second. In all of these situations the ability to look at thread details over the previous few hours can be of enormous benefit.

The information recorded at thread level by near-term history is the accounting record written by DB2 when the thread terminates. Until the thread terminates, no information at all is available about the thread in the near-term history display. In addition to the accounting information, there may also be details about dynamic SQL and locking, sorting, and scanning occurring in the thread, provided that the appropriate traces have been switched on within the near-term history collection options. Details of static SQL calls (other than counts of calls executed as held in the accounting record) cannot be recorded by near-term history.

Probably the biggest benefit of near-term history collection is the identification of threads that have experienced problems in the past few hours. Threads experiencing problems can be quickly identified on the Thread History Summary panel. Using PF11 the problem thread can then be examined in more detail to help identify the cause of the problem. Some of the types of thread problems that can be identified in this way are:

- Excessive CPU or elapsed time
- Excessive in-DB2 CPU or elapsed time
- Threads that have had timeouts or deadlocks
- Threads committing too infrequently
- Threads that have aborted
- Threads with excessive lock waiting time
- Threads with excessive DB2 wait time for I/O

In busy environments with large numbers of threads completing every minute, you can use the filter options to identify these types of problem threads more quickly.

The Application Trace Facility can also be used to collect information similar to near-term history. However, collecting this information through an application trace has a number of disadvantages compared to near-term history collection:

- The application trace has a very high overhead cost and can usually only be run for a few minutes in a production environment.
- Because of the high overhead cost it is not practical to have an application trace running continuously to provide a history of problems.
- The volume of data produced from an application trace can be extremely high and therefore the cost of processing the collected data to produce meaningful results is correspondingly high.

9.2 How to access and control near-term history collection

Unlike options such as the Application Trace Facility and Object Analysis, you cannot control the operation of near-term history from within the OMEGAMON Classic Interface real-time panels. This section tells you how you can access and control near-term history.

9.2.1 Accessing near-term history

You can access the Near-Term History Menu shown in Figure 9-1 by selecting option H from the OMEGAMON XE for DB2 PE Classic Interface - Main Menu. You then have the following options available:

- Display statistics near-term history (option A)
- Display thread near-term history (option B)
- Display collector parameters (option C)

> **Note:** You cannot start and stop near-term history collection from this panel or change near-term history collector options. See 9.2.2, "Starting and stopping near-term history collection" on page 139 and 9.2.3, "Changing the collection options" on page 141 for how to do this.

```
 ________________ ZHIST    VTM     O2       V310./C DB8A 01/29/06 23:21:34   2
>       Help PF1                                              Back PF3
> H.

>                Enter a selection letter on the top line.
===============================================================================
>                    NEAR-TERM HISTORY INFORMATION MENU


_ A  STATISTICS ............. Near-Term History Statistics Information

_ B  THREAD HISTORY.......... Near-Term Thread History Information

_ C  COLLECTOR INFORMATION .. Near-Term History Data Collector Information
```

Figure 9-1 Near-Term History Information Menu

If near-term history collection has not been started, you receive the following message when trying to access Option C (Collector Information):

```
Collection Options Not Found
```

In this case, follow the instructions in 9.2.2, "Starting and stopping near-term history collection" on page 139 to start near-term history collection.

9.2.2 Starting and stopping near-term history collection

Whether near-term history collection is active by default when you start OMEGAMON XE for DB2 PE depends on the configuration options you choose in the Installation and Configuration Assistance Tool (ICAT) at installation time.

However, whatever option you choose at installation time, you have the ability to start and stop the near-term history collection subtask using operator commands. All the following examples of operator commands are shown as you would execute them from SDSF (that is, they include the character / at the beginning of the command). You must of course have the necessary security authority as defined at your installation to execute these commands.

Checking to see if near-term history collection is active

To see if near-term history collection is active, you issue this command:

```
/F cccccccc,DISPLAY
```

Where:

- *cccccccc* is the name of the OMEGAMON XE for DB2 PE started task.

This command displays details of all the active subtasks of the OMEGAMON XE started task, and you see an output similar to that shown in Example 9-1.

Example 9-1 Output after issuing display command

```
THE FOLLOWING TASK IDS ARE ACTIVE:
 ID=H2DB8A              PROGRAM=KO2HWLMB
 ID=OBVTAM              PROGRAM=KOBVTAM
 ID=EVENTMGR            PROGRAM=KO2EINTB
 ID=PESERVER            PROGRAM=DGOVMSTR
 ID=COMMCOLL            PROGRAM=KO2AINIB
```

The presence of the subtask-id `H2DB8A` (`H2` followed by the DB2 subsystem name `DB8A`) indicates that near-term history collection is active.

Starting near-term history collection

If the display command shows that near-term history collection is not active, it can be started with the following command:

```
/F cccccccc,S H2WLMGR,OPTION=COPTxxxx
```

Where:

- *cccccccc* is the name of the OMEGAMON XE for DB2 PE started task.
- *xxxx* is the DB2 subsystem name for which near-term history collection is to be started, and the member name `COPT`*xxxx* is the member in the `RKD2PAR` library holding the near-term history collection options.

If the command successfully starts the near-term history collection subtask, you see the messages shown in Example 9-2.

Example 9-2 Output after starting near-term history collection

```
NEAR-TERM HISTORY DATA COLLECTOR - USED DEFAULT INTERVAL(15) TO EVALUATE MAXHOUR
NEAR-TERM HISTORY DATA COLLECTOR OPTION PARAMETERS PROCESSED SUCCESSFULLY FROM MEMBER COPTDB8A
NEAR-TERM HISTORY DATA COLLECTOR - NEW OPTIONS IN EFFECT FOR CURRENT INTERVAL
NEAR-TERM HISTORY DATA COLLECTOR SERVER KO2DMGRB IS ACTIVE (SERVINIE)
DRIVER INITIALIZED (CPINIT  )
CAPTURING OUTPUT ON DATA SET OMEGASYS.DB8A.SC63RTE.DB8A.RKD2VS03
SET STARTED - CONTSET1 DB8A
NEAR-TERM HISTORY DATA COLLECTOR SERVER KO2SSRVB IS ACTIVE (SERVINIE)
NEAR-TERM HISTORY DATA COLLECTOR ACTIVELY MONITORING DB2 DB8A (H2WLMGRE)
DB8A> -START TRACE (MON) IFCID(318)  DEST(OPX) CLASS( 1) BUFSIZE(1024)
DB8A> INSTRUMENTATION FACILITY BUFFER INFORMATION BLOCK+
DB8A> ADDRESS   LEN      EYE       ECB      BCLIMIT   TEXT
DB8A> 252527C4  00100000 E6C2E4C6  24537A28 000B3333 *....WBUF. :.....*
DB8A> DSNW135I  -DB8A MON TRACE ALREADY ACTIVE, TRACE NUMBER 04
DB8A> -START TRACE(ACCTG) TDATA(CORRELATION,CPU,DIST) DEST(OPX) CLASS( 1 2 3 7 8
DB8A> UFSIZE(1024)
DB8A> -START TRACE(PERFM) TDATA(CORRELATION,CPU,DIST) CLASS(30) IFCID(105,107, 1
DB8A> 196  63,22 ) DEST(OP2) BUFSIZE(1024)
NEAR-TERM HISTORY DATA COLLECTOR SERVER KO2ISRVB IS ACTIVE (SERVINIE)
```

Stopping near-term history collection

The near-term history collection subtask can be stopped with the following command:

```
/F cccccccc,P H2xxxx
```

Where:

- *cccccccc* is the name of the OMEGAMON XE for DB2 PE started task.
- *xxxx* is the DB2 subsystem name for which near-term history collection was started.

If the command successfully stops the near-term history collection subtask, you see messages similar to those shown in Example 9-3.

Example 9-3 Output after stopping near-term history collection

```
TASK ID=H2DB8A   HAS BEEN STOPPED VIA POST
NEAR-TERM HISTORY DATA COLLECTOR IS TERMINATING  (H2INITE )
CANDLE COMMON INTERFACE READY FOR COMMANDS
DB8A> -STOP TRACE(PERFM)  TNO(6   )
DB8A> -STOP TRACE(ACCTG)  TNO(5   )
DB8A> -STOP TRACE(MON)  TNO(    )
DB8A> STOP TRACE NOT ISSUED. TRACE NUMBER IS NOT AVAILABLE (FORMATRS)
NEAR-TERM HISTORY DATA COLLECTOR SERVER KO2ISR HAS TERMINATED (SVESTAE )
SET STOPPED - CONTSET1 DB8A
DRIVER TERMINATED (CPTERM  )
NEAR-TERM HISTORY DATA COLLECTOR SERVER KO2SSR HAS TERMINATED (SVESTAE )
```

Archiving near term history data sets

Member `COPTxxxx` of the `RKD2PAR` library also holds the options about archiving the near-term history. You must archive in order to empty the log VSAM data sets.

Once you have a collection of archival data sets, you can run the report functions on accounting, statistics, etc., against them, just like you can do for trace records. The OMEGAMON Historical Reporter is no longer supported. APAR PK23439 states that the ICAT process for archive files should be set to *dummy* since the archive files are not used for reporting.

For how to migrate to the OMEGAMON XE for DB2 PE Reporter, please refer to the *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Getting Started,* GC18-9634.

Currently ICAT only allows three near term VSAM data sets to be defined in Member `COPTxxxx.` You can keep more data in the near term log data sets (more than 24 hours) by defining up to 10 VSAM data sets and allocating them in `COPTxxxx`. This is particularly useful in a busy system where the VSAM data sets fill up and wrap quickly, subsequently being overwritten and losing the near term history.

9.2.3 Changing the collection options

The collection options for near-term history are held in member `COPTxxxx` of the `RKD2PAR` library where `xxxx` is the DB2 subsystem name for which near-term history is being collected. The full collection options and syntax details are described in Chapter 21, “Specification of near-term history data collection options” in *Monitoring Performance from the OMEGAMON Classic Interface*, SC18-9659.

Looking at the existing data collection options

To look at the existing near-term history data collection options, select option C from the near-term history information menu to see the panel shown in Figure 9-2.

```
> *-COLLECTION OPTIONS       B-RECORD INFORMATION          C-DATASET STATUS
===============================================================================
>                 NEAR-TERM HISTORY DATA COLLECTION OPTIONS
 COPT
+                            H2 Collection Options
+
+ DB2sys       = DB8A        Writeoption   = VSAM       Interval    =   15
+ Archivejcl   = ARCVDB8A    Tracebufsz    = 1024K      Ifireadtime = 010000
+ Maxhours     = 24          Suspcoll      = Yes        PostPCT     = 70
+ Destination  = None
+
+ Statistics   = Yes         Dsnzparm      = Yes
+ Auditing     = none
+ Accounting   = (1 2 3 7 8 ) Sort         = No         Lock Contention = Yes
+                            Scan          = No         Lock Suspension = No
+                            Dynamic SQL   = Yes        Negative SQL    = Yes
+
+ H2 Data Sets :
+                OMEGASYS.DB8A.SC63RTE.DB8A.RKD2VS01
+                OMEGASYS.DB8A.SC63RTE.DB8A.RKD2VS02
```

Figure 9-2 Near-term History Data Collection Options

The panel displays a summary of the options stored in the `COPT`*xxxx* member of the `RKD2PAR` library. If you select option B from this panel, you can see a summary of which near-term history records have been collected since near-term history was last started. This is shown in Figure 9-3.

```
 >             NEAR-TERM HISTORY DATA COLLECTION RECORD INFORMATION
 CREC
+                            H2 Record Information
+
+ Record Type   Count    Timestamp of First Record  Timestamp of Last Record
+ ------------  -------- -------------------------- --------------------------
+ Accounting          31 2006-01-26-20.25.24.591459 2006-01-29-21.30.54.630120
+  DDF Section
+  Pkg Section
+  BP Section         87 2006-01-26-20.25.24.591459 2006-01-29-20.06.31.332880
+ Acctg Sum
+  DDF Section
+  BP Section
+ Statistics         307 2006-01-26-20.45.00.168782 2006-01-30-01.15.00.222564
+  DDF Section
+  BP Section        357 2006-01-26-20.45.00.168782 2006-01-30-01.15.00.222564
+ Perf-Lock Cont      45 2006-01-26-20.27.53.206636 2006-01-26-20.42.18.359096
+ Perf-Lock Susp
+ Perf-Dynam SQL   31458 2006-01-26-20.18.03.387769 2006-01-30-01.15.54.081594
+ Perf-Neg SQL     13850 2006-01-26-20.18.03.388531 2006-01-30-01.15.54.081937
+ Dsnzparm             1 2006-01-26-20.17.18.644680 2006-01-26-20.17.18.644680
```

Figure 9-3 Near-Term History Data Collection Record Information

If you select option C, you can see the status of the data sets used to store the near-term history as shown in Figure 9-4.

```
>                 NEAR-TERM HISTORY DATA COLLECTOR DATASET STATUS
 H2DS
+  H2 Dataset                                        Status        %Full
+  --------------------------------------------      --------      ------
+  OMEGASYS.DB8A.SC63RTE.DB8A.RKD2VS01               AVAIL           4.0
+  OMEGASYS.DB8A.SC63RTE.DB8A.RKD2VS02               ACTIVE         46.3
+  OMEGASYS.DB8A.SC63RTE.DB8A.RKD2VS03               UNAVAIL       100.0
```

Figure 9-4 Near-Term History Data Collector Dataset Status

Typical data collection option changes

To make changes to the existing data collection options, you need to edit member `COPTxxxx` of the `RKD2PAR` library where *xxxx* is the DB2 subsystem name. Look at Table 9-1 to see some of the more common data collection options that you might want to change.

Table 9-1 Near-term history collection options

Option	Effect
SCAN(NO)	turn off recording of SQL scan information
DYNAMICSQL(YES)	turn on recording of the text of dynamic SQL statements
NEGSQL(NO)	turn off recording of negative SQL codes
LOCKCOUNT(YES)	turn on recording of deadlock and timeout information
LOCKSUSP(NO)	turn off recording off lock suspension information

Making the changes active

To make the changes in the collection option member active, use the following command:

```
/F cccccccc,F H2xxxx,VARY OPTION=COPTxxxx
```

Where:

- *cccccccc* is the name of the OMEGAMON XE for DB2 PE started task.
- *xxxx* is the DB2 subsystem name for which near-term history collection options are to be changed, and the member name `COPTxxxx` is the member in the `RKD2PAR` library holding the changed near-term history collection options.

If the near-term collection options are successfully changed, you see messages similar to those shown in Example 9-4 where changes have been made active from member `COPTDB8A`:

Example 9-4 Output after refreshing collection options

```
DATA COLLECTOR - USED DEFAULT INTERVAL(15) TO EVALUATE MAXHOURS PARAMETER
DATA COLLECTOR OPTION PARAMETERS PROCESSED SUCCESSFULLY FROM MEMBER COPTDB8A
DATA COLLECTOR - NEW OPTIONS WILL TAKE EFFECT WHEN THE CURRENT INTERVAL EXPIRES
```

The changes will take effect when the current collection interval expires.

9.3 Performance issues with near-term history collection

The performance overhead for near-term history collection was measured using the DB2 Workload Generator as described in Appendix A, "DB2 Workload Generator" on page 311.

The CPU usage in seconds was measured for both the OMEGAMON XE for DB2 PE address space (DB8AO2S) and the Workload Manager address space being used by the DB2 Workload Generator (DB8AWLM1).

The DB2 Workload Generator ran as a batch job (GLWRUNY) producing an SQL workload that was almost constant each time, consisting of 1.1 M static SQL calls and 162 dynamic SQL calls. Table 9-2 shows the results from the measurements.

Table 9-2 Results of performance measurements for near-term history collection

Workload GLWRUNY RUNTIME(1) RUNMODE(FIXED) RUNPROF(QUERY) OBJECT ANALYSIS(OFF)	Average DB8AO2S sec.	Average DB8AWLM1 sec.
NTH OFF	0.72	54.26
NTH ON DYNSQL(NO) SQLERR(NO)	0.76	54.56
NTH ON DYNSQL(NO) SQLERR(YES)	6.30	70.62
NTH ON DYNSQL(YES) SQLERR(YES)	6.30	70.59
NTH ON DYNSQL(YES) SQLERR(NO)	0.73	54.11

The main finding from these measurements is that the near-term history option `SQLERR(YES)`, which collects negative SQLCODE values for both dynamic and static SQL, creates a significant performance overhead in both address spaces.

As the SQLCODE values are not reported on the near-term history panels, we recommend that you always switch this option off by using `SQLERR(NO)` on the near-term history startup parameters.

9.4 Near-term thread history

The Main Menu options T (Thread activity at plan level) and U (Thread activity at package level) allow you to see extensive details of DB2 threads that are currently executing. The Near-Term Thread History shows most of the same information, but only for threads that have completed in the last few hours. Threads that are still running are not shown.

Exactly how much history is available depends on how active your system is and how large the near-term history data sets are. It should be possible even in a very active large production environment to store five to ten hours of history.

Note: Near-term history only stores the text of dynamics SQL calls (and then only if the collector option for dynamic SQL is switched on). This is in contrast to the active thread displays in Main Menu options T and U where both static and dynamic SQL are displayed for active threads.

9.4.1 Selecting the near-term threads you want to see

Figure 9-5 shows the first panel you see after selecting Near-Term Thread History Option B from the Near-Term History Information Menu.

```
 ________________ ZHAFL    VTM     02       V310./C DB8A 01/29/06 23:23:12   6
>                     NEAR-TERM THREAD HISTORY FILTER OPTIONS

:   START-DATE/TIME =  __________  _____     (mm/dd/yyyy or dd.mm.yyyy, hh:mm)
:   END-DATE/TIME   =  __________  _____     (mm/dd/yyyy or dd.mm.yyyy, hh:mm)
:   RELATIVE-START  =    6 HOURS  ago        (1-nn, MINS or HOURS)
:   RELATIVE-END    =  ___ _____  from start (1-nn, MINS or HOURS)
:   REPORT-INTERVAL =   15 minutes           ( 5-60 mins, in  5 min increments)
+
+ Specify the values to be used to filter the Thread History displays. Wildcard
+ values * (multiple characters) or ? (single character) can be specified.
:   PLAN            =  ________   ________   ________   ________   ________
:   AUTHID          =  ________   ________   ________   ________   ________
:   CONNID          =  ________   ________   ________   ________   ________
:   CONNTYPE        =  ________   ________   ________   ________   ________
:   PACKAGE         =  ________   ________   ________   ________   ________
:   CORRID          =  ____________  ____________  ____________  ____________
:   GETPAGES        =  __________
```

Figure 9-5 Near-Term Thread History Filter Options

This panel allows you to filter the threads displayed on the following panels by various criteria including:

- Start/end date and time. Remember that this is near-term history and in a busy production environment the stored data may only go back a few hours.
- Relative start time in minutes or hours.
- A variety of DB2 plan related options such as plan name, package name, authid and correlation-id. The wildcard characters * and ? can be used here.
- Performance related fields such as CPU time, elapsed time, I/O time and getpage requests. The normal operators =, >, <, >=, <= can be used here.
- Locking factors such as number of escalations, deadlocks and timeouts.

Depending on the filter options you have chosen, they may take effect on the Thread History by Report Interval panel that follows, or you may have to drill down to an individual thread level before they become effective.

> **Tip:** Many panels in the OMEGAMON XE for DB2 PE Classic Interface extend over more than 24 lines. Use PF8 to page down and ensure that you have not overlooked important options.

In our example all threads are displayed that have completed during the last six hours, provided that the near-term history data sets are large enough to hold six hours of history data. The threads are reported grouped into 15 minute intervals.

On pressing Enter you see the Near-Term Thread History Filter Options - Confirmation panel. If you are happy with the filter options selected, press Enter to see the Thread History By Report Interval panel as shown in Figure 9-6.

```
 ________________ ZHARP    VTM     O2       V310./C DB8A 01/29/06 23:25:00   4
>       Type a selection letter next to an Interval and press Enter.
>
> A-BY PLAN          B-BY AUTHID       C-BY PLAN,AUTHID     D-BY AUTHID,PLAN
> E-THREAD SUMMARY    F-BY SUBINTERVAL
===============================================================================
>                          THREAD HISTORY BY REPORT INTERVAL
 HARP
+ Report Interval:  15 mins                    Start:   01/29 17:25:00.000000
+ Report Filtered:       NO                      End:   01/29 23:19:59.999999
+
+                                Dlk/  In-DB2  In-DB2  In-DB2          GetP/
+   Time  Thrds Commit Abort DML   TOut  Elap Tm CPU Tm  Wait Tm  Getpage RIO
+   ----- ----- ------ ----- ----- ----  ------- ------- -------  ------- -----
+   23:15-21:45   No Thread Activity
: _ 21:30     1      1     0     0    0       .0     .01      .0        0    .0
+   21:15-20:15   No Thread Activity
: _ 20:00     1   3665     0 2036K    0     11.2    3.01    23.1  6662975 3331K
+   19:45-18:45   No Thread Activity
: _ 18:30     1      1     0     0    0      1.7     .58     1.1        0    .0
+   18:15         No Thread Activity
```

Figure 9-6 Thread History By Report Interval

When looking at this panel it is important to remember that threads are only listed in near-term history after the accounting record has been written by DB2. For batch jobs this is after the job has completed, and for online transactions generally after the transaction has terminated. This is different from the Main Menu Thread Activity options T and U, where only active threads are displayed.

Another reason why an expected thread might not be displayed is the settings you have used for filter options. Check in particular that there are no unexpected filter options when you use PF8 to page down on the Filter Options panel.

9.4.2 Identifying DB2 problems at the thread history summary level

The Thread History by Report Interval panel as shown in Figure 9-6 displays one line of information for each reporting interval and shows how many threads completed in the interval. The Time column shows the start time for the reporting interval. This is followed by summarized data for all qualifying threads that completed within the report interval.

To select the reporting interval you are interested in, type one of the option letters A to F in the left hand column next to the reporting interval (not in the normal position at the top left of the panel). Generally using option letter E gives you the most useful Thread History Summary List as shown in Figure 9-7.

```
 ________________ ZHATACT  VTM     02       V310./C DB8A 02/06/06 20:40:35   3
> Help PF1         Back PF3        Up PF7         Down PF8         Zoom PF11
>
> *-SUMMARY        B-BUFFER POOL       C-DB2 TIME        D-LOCK/SCAN/SORT
> O-OPTIONS
===============================================================================
>                             THREAD HISTORY SUMMARY
 HATH
+ Report Interval:   15 mins                    Start: 02/06 14:15:00.000000
+ Report Filtered:       NO                       End: 02/06 14:29:59.999999
 act
+
+                                  Elapsed  CPU                        Term
+ End Time       Plan     Authid   Time     Time   SQL   Commit Abrt Pkg Status
+ ------------   -------- -------- -------  ------ ----- ------ ---- --- ------
+ 14:27:03.708   KO2PLAN  DB8A02S    62.79    .071     0      0     0   0
+ 14:26:39.570   ADB      PAOLOR3   235.87    .136  1003     12     0   0
+ 14:23:38.219   KO2PLAN  DB8A02S   136.52    .123     0      0     0   0
```

Figure 9-7 Thread History Summary

The various views of the Thread History panel provide very useful information for identifying problem threads that you may need to look at in more detail. Because information is summarized as one line per thread, you can quickly scan through the list looking for problems. This is even more efficient if you have selected the appropriate filter options to reduce the number of threads to search.

Identifying problems from the thread history summary

The thread history summary view is shown in Figure 9-8.

```
>                             THREAD HISTORY SUMMARY
+
+                                  Elapsed  CPU                        Term
+ End Time       Plan     Authid   Time     Time   SQL   Commit Abrt Pkg Status
+ ------------   -------- -------- -------  ------ ----- ------ ---- --- ------
+ 14:27:03.708   KO2PLAN  DB8A02S    62.79    .071     0      0     0   0
+ 14:26:39.570   ADB      PAOLOR3   235.87    .136  1003     12     0   0
```

Figure 9-8 Thread History Summary

This panel can help you in identifying the following possible problems:

- Threads that have aborted with their termination status
- Threads that may have committed too infrequently
- Threads with excessive CPU time, especially if the SQL count is very low
- Threads with excessive elapsed time
- Threads with an unexpectedly high SQL count

Identifying problems from the buffer pool summary

From the buffer pool summary shown in Figure 9-9, the following possible problems can be identified:

- Threads with a poor buffer pool hit-ratio (getpages per read I/O is low).
- Threads with a very large number of getpages.

- Threads with a very high number of updates, especially if the commit count is low. However, it should be remembered that this column also contains updates to DB2 workfiles and might only indicate sort activity.
- Threads performing an unexpected number of prefetches.

```
>                    THREAD HISTORY BUFFER POOL SUMMARY

+                                            Read   GetP/               Immed
+ End Time       Plan     Authid   GetPage  I/O    RIO    Pfetch Update Write
+ ------------   -------- -------- -------  -----  -----  ------ ------ ------
+ 14:27:03.708   KO2PLAN  DB8A02S        0      0    .00       0      0      0
+ 14:26:39.570   ADB      PAOLOR3     1277      0   1277      62     28      0
```

Figure 9-9 Thread History Buffer Pool Summary

Identifying problems from the lock/scan/sort summary

From the lock/scan/sort summary shown in Figure 9-10, the following possible problems can be identified:

- Threads that have received deadlocks or timeouts
- Threads with excessive lock waiting time
- Threads performing unexpected or very large sorts that may be filling the available DB2 sort workspace and causing problems for other threads that need to sort

```
>                  THREAD HISTORY LOCK/SCAN/SORT SUMMARY

+                                          Dlk/  Lock   Lock     TS        Recs
+ End Time       Plan     Authid   Locks  Tout Waits  Wait Tm  Scan Sort Sorted
+ ------------   -------- -------- ------ ---- -----  -------  ---- ---- ------
+ 14:27:03.708   KO2PLAN  DB8A02S       0    0     0     .000   N/A  N/A    N/A
+ 14:26:39.570   ADB      PAOLOR3     711    0     0     .000   N/A  N/A    N/A
```

Figure 9-10 Thread History Lock/Scan/Sort Summary

9.4.3 Viewing near-term thread detail

When you have identified a thread that may have a problem from one of the summary views described in 9.4.2, "Identifying DB2 problems at the thread history summary level" on page 146, you can position the cursor on the thread you are interested in and press PF11 to zoom in to a detailed view of the thread.

The various options for Thread History Detail are shown in Figure 9-11. These can be selected in turn by typing the option letters A to M in the top left corner of the panel. They are very similar to those available under the Main Menu Thread Activity options T and U, although there are some differences that we will refer to in the following discussions.

```
 ________________ ZHTACT   VTM     O2      V310./C DB8A 01/29/06 23:31:39   4
>          THREAD HISTORY:  Enter a selection letter on the top line.

> *-THREAD DETAIL   B-LOCK COUNTS  C-LOCK WAITS     D-GLOBAL LOCKS  E-SORT/SCAN
> F-DYNAMIC SQL     G-SQL COUNTS   H-DISTRIBUTED    I-BUFFER POOL   J-GROUP BP
> K-PACKAGE SUMMARY L-RES LIMIT    M-PARALLEL TASKS
===============================================================================
>                            THREAD HISTORY DETAIL
 HPLN
+ Thread:  Plan=DSNREXX   Connid=DB2CALL  Corrid=GLWRUND      Authid=PAOLOR1
+ Attach:  CALLATCH       DB2=DB8A        MVS=SC63
+ Time  :  Start=01/29/2006 18:01:39.491672     End=01/29/2006 18:11:42.998036
 act
+ Termination Status    =         DEALLOC     Commits              =       400
+ Total Elapsed Time    =    00:10:03.506     Aborts               =         0
+ Total CPU Time        =    00:00:00.893     Parallel Tasks       =         0
+ Total Stored Proc CPU =    00:03:59.173
+ Stored Proc Wait      =    00:00:01.807     Stored Proc Wait Cnt =      1012
+
```

Figure 9-11 Thread History Detail

We now look at some of the more useful diagnostic information that you can find from the near-term thread detail panels A to M, which may help you in identifying why a thread is experiencing a problem.

Thread history detail (option A)

Figure 9-12 shows part of the Thread history detail available under option A.

```
>                            THREAD HISTORY DETAIL
+ In-DB2 Times                                    Total
+ ------------------------------                  ------------
+ Elapsed Time                                    00:00:03.785
+ CPU Time                                        00:00:00.542
+ Stored Procedure CPU Time                       00:03:48.973
+
+ Waits                                 Count     Total
+ ------------------------------    ----------    ------------
+ Synchronous I/O Wait                    425     00:00:01.077
+ Asynchronous Read I/O Wait                2     00:00:00.004
+ Asynchronous Write I/O Wait             772     00:00:02.487
+ Local Lock/Latch Wait                 13526     00:00:07.114
+ Page Latch Wait                           1     00:00:00.001
```

Figure 9-12 Thread History Detail

This panel provides a very detailed breakdown of the elapsed and CPU time for the thread. It can help you answer some of the following diagnostic questions to narrow down the cause of a thread problem:

- Is the thread spending most of its time in DB2 (in-DB2 elapsed time is close to total elapsed time)? You should be aware that time spent in WLM managed stored procedures (stored procedure CPU time + stored procedure wait time) is accounted for separately from in-DB2 elapsed time in this display.
- Is the thread CPU bound (total wait time is much smaller than total elapsed time)?

- If the thread has a large DB2 wait time, where is this occurring? Look here at the breakdown of db2 wait time into I/O waits, locking waits and other types of wait.

Thread history lock/claim/drain activity (option B)

Figure 9-13 shows part of the Thread history lock/claim/drain activity available under option B.

```
>                  THREAD HISTORY LOCK/CLAIM/DRAIN ACTIVITY
+ Lock Requests         =       9935   Deadlocks Detected         =          1
+ Unlock Requests       =       1466   Timeouts Detected          =          1
+ Query Requests        =          0   Suspends - Lock Only       =          3
+ Change Requests       =        234   Suspends - Latch Only      =          0
+ Other IRLM Requests   =          0   Suspends - Other           =          0
+ Escalations to Shared  =         0   Escalations to Exclusive   =          0
```

Figure 9-13 Thread History Lock/Claim/Drain Activity

This panel can provide answers to the following locking related problems:

- Are there a very large number of lock requests increasing the CPU usage? If so, could you use uncommitted read (UR) locking to reduce the number?
- Did the thread receive a deadlock or timeout condition? If so, you can use option C to find out which thread caused the deadlock/timeout and which DB2 resources were involved.
- Did this thread escalate any shared or exclusive locks to tablespace level? In an online environment using CICS or IMS this might result in many other threads receiving a timeout condition, unless the escalated lock was only held for a very short duration.
- Even if the thread did not deadlock or timeout, a large number of lock suspends might indicate a locking problem, especially if the lock wait time shown under option A (thread history detail) is high.

Thread history lock waits (option C)

Figure 9-14 shows the Thread history lock wait information available under option C.

If you identified deadlocks or timeouts as a problem on one of the thread history summary panels or under option B (thread history lock/claim/drain activity), you can use the information here to get a more detailed look at the deadlock or timeout.

The following details are particularly helpful:

- Whether it was a deadlock or a timeout that occurred.
- The exact time of the deadlock/timeout.
- The DB2 resource for which the locking problem occurred. This may be given in the form of internal DB2 object identifiers (OBIDs) such as DBID and PSID if the database name and tablespace name as recorded in the DB2 catalog were not available at the time the record was written by DB2.
- Details of the thread receiving the deadlock/timeout condition, including the type of lock it was waiting for.
- Details of the thread causing the deadlock/timeout condition, including the type of lock it was holding.

```
>                          THREAD HISTORY LOCK WAITS
+ Time of Deadlock      = 17:35:07.713       Deadlock Interval = 16229
+ Resource:   Type=DPAG  DBID=00266  PSID=00079   PAGE=00000E
+
+ Wait:  Plan=DSNREXX   Connid=DB2CALL  Corrid=GLWRUNY
+        Luwid=USIBMSC.SCPDB8A.BE5C4EE95359
+        Lock Level     = S                Duration        = MANUAL
+        IRLM Function  = LOCK             Request Type    = UNCONDITIONAL
+        DBMS Subsystem = DB8A
+
+ Own:   Plan=DSNREXX   Connid=DB2CALL  Corrid=GLWRUNX
+        Luwid=USIBMSC.SCPDB8A.BE5C4F4BDBA4
+        Lock Level     = X                Duration        = COMMIT
+        DBMS Subsystem = DB8A             Request Type    = UNCONDITIONAL
```

Figure 9-14 Thread History Lock Waits

Thread history sort and scan activity (option E)

Figure 9-15 shows part of the Thread history sort and scan activity available under option E. While the overhead of collecting sort information is relatively low, scan information can be very expensive to collect where large numbers of SQL calls are being executed, and many installations choose to switch off the collection of scan information. How to do this is described in 9.2.3, “Changing the collection options” on page 141.

```
>                    THREAD HISTORY SORT AND SCAN ACTIVITY
 HPLN
+ Thread:  Plan=DSNREXX   Connid=DB2CALL  Corrid=GLWRUNY      Authid=PAOLOR1
+ Attach:  CALLATCH       DB2=DB8A        MVS=SC63
+ Time  :  Start=02/13/2006 17:32:02.924242     End=02/13/2006 17:35:07.719547
 socs
+                               Sort Statistics
+
+ Number of Sorts      =           17     Number of Work Files   =          0
+ Records Sorted       =         1742     Work File Inserts      =          0
+ Sort Elapsed Time    = 00:00:00.118     Work File Reads        =          0
+ Sort CPU Time        = 00:00:00.028
```

Figure 9-15 Thread History Sort And Scan Activity

The sort and scan information provided by option E is one area where near-term history provides information not available under the active thread Main Menu options T and U. The sort information provided here can be particularly useful in identifying threads that are consuming large amounts of sort workspace (very high count of records sorted or number of work files), and that are possibly preventing other threads from completing their sorts successfully.

Thread history dynamic SQL calls (option F)

Figure 9-16 shows the display of a dynamic SQL call on the Thread history dynamic SQL calls available under option F. It is important to remember that only dynamic SQL calls are stored in near-term history. If the thread also executed static SQL calls, these cannot be displayed.

```
>                      THREAD HISTORY DYNAMIC SQL CALLS
+ Time  :  Start=01/29/2006 18:01:39.491672     End=01/29/2006 18:11:42.998036
 call
+
: Select Call=NEXT      (FIRST/LAST/NEXT/PREV/+nnnnn/-nnnnn/Snnnnn)
+
+                              SQL Statement      (      45 of     491)
+
+ SELECT EMP_NO, FIRSTNME, MIDINIT, LASTNAME, WORKDEPT, PHONENO,
+      HIREDATE, JOB, MANAGER, EDLEVEL, SEX, BIRTHDATE, SALARY, BONUS,
+      COMM, CREATED_TS, CREATED_BY, UPDATED_TS, UPDATED_BY
+    FROM GLWTEMP
+    WHERE JOB = ?
+    ORDER BY WORKDEPT, JOB
```

Figure 9-16 Thread History Dynamic SQL Calls

By default, the next dynamic SQL call executed in the thread is displayed each time Enter is pressed. The effect of pressing Enter can be controlled by over-typing the following field with `FIRST/LAST/NEXT/PREV` and so on:

```
: Select Call=NEXT
```

If a miniplan for the DB2 access path is generated by the execution of the SQL, this is displayed below the SQL statement as shown in Figure 9-17.

```
+                              Miniplan Generated
+
+ Plan #  1    (Estimated Cost:        4 )
+
+ Table: GLWTEPA_T              Access Type: SEQUENTIAL SCAN
+ Prefetch Activity: SEQUENTIAL
```

Figure 9-17 Miniplan for Dynamic SQL Call

There are a number of factors to consider relating to the usefulness of the display of dynamic SQL calls under option F:

- No static SQL calls in the thread are displayed.
- If a large amount of dynamic SQL is being executed in the environment, the cost of collecting the SQL calls can be very high.
- No information is provided about the CPU or elapsed time of the individual SQL calls. This means it can be difficult to identify which SQL calls are causing a problem from the near-term history display.
- The active thread display options T and U from the Main Menu can have an advantage when trying to identify poorly performing SQL calls. By watching a thread using these options while it is actually running, it is often possible to see which SQL calls are taking the most time to execute.
- Despite the restrictions, capturing dynamic SQL calls for further investigation can be extremely useful in some environments.

Thread history SQL counts (option G)

Figure 9-18 shows part of the Thread history SQL counts available under option G. Much useful information for problem determination can be derived from this panel.

```
 >                        THREAD HISTORY SQL COUNTS
 HPLN
++ Time  :  Start=01/29/2006 18:01:39.491672     End=01/29/2006 18:11:42.998036
 sqls
+ Commit         =     400  Abort          =       0  Select          =    3636
+ Open Cursor    =     646  Close Cursor   =     646  Fetch           = 3099322
+ Insert         =     798  Delete         =       1  Update          =       0
+ Describe       =      62  Lock Table     =       0  Prepare         =     490
+ Grant          =       0  Revoke         =       0  Set Rules       =       0
+ Increm Bind    =       0  Label/Comm On  =       0  Set SQLID       =       0
+ Set Host Var   =  138281  Set Connection =       0  Set Degree      =       0
+ Connect Type 1 =       0  Connect Type 2 =       0  Set Path        =       0
+ Rename Table   =       0  Hold Locator   =       0  Free Locator    =       0
+ Release        =       0  Assoc Locator  =       0  Allocate Cursor =       0
```

Figure 9-18 Thread History SQL Counts

Some of the more important possibilities are:

- How frequently the thread is committing in relation to update activity and elapsed time for the thread. This can have important consequences for the impact the thread may have running against other online threads. It also enables an estimate to be made of the cost involved in rollback activity if the thread abends.
- Whether the thread is executing far more calls than expected of a particular type, which may indicate a problem.
- Whether a large number of incremental binds are being executed.
- Whether LOCK TABLE statements are being executed.
- Use of parallelism by the thread.
- Reasons for parallelism failing.
- RID pool usage for list prefetch.
- RID pool failures resulting in access paths reverting to tablespace scan.
- Use of stored procedures and triggers.
- Use of cache by dynamic SQL.

Thread history buffer pool activity (option I)

Figure 9-19 shows part of the Thread history buffer pool activity available under option I.

```
 >                     THREAD HISTORY BUFFER POOL ACTIVITY
 HPLN
+ Time  :  Start=01/29/2006 18:01:39.491672     End=01/29/2006 18:11:42.998036
 buf
+ Buffer Pool: ALL
+
+ Getpage Requests       =    2256489   Failed Getpage Requests    =          0
+ Synchronous Read I/O   =          0   Getpage/Read I/O           = 2256489.00
+ Page Updates           =    2155377   Seq Prefetch Requests      =      15286
+ List Prefetch Requests =       1160   Dynamic Prefetch Requests  =        748
+ Prefetch Pages Read    =        116   Prefetch Pages In Hiperpool=          0
+ Hiperpool Reads        =          0   Failed Hiperpool Reads     =          0
+ Hiperpool Writes       =          0   Failed Hiperpool Writes    =          0
+ Immediate Writes       =          0   BP Hit Percentage          =      99.9%
```

Figure 9-19 Thread History Buffer Pool Activity

If you suspect that poorly performing SQL may be causing a problem, information from the Buffer pool activity panel may provide an indication of where the problem lies:

- A very large number of getpage requests together with high CPU usage and low I/O wait time may point to SQL with a poor access path where most of the pages are being retrieved from the buffer.
- A high rate of sequential prefetch requests might be caused by unwanted tablespace scans.
- A large number of synchronous read I/O requests might indicate lack of buffer pool space or that a tablespace or its related indexes are in need of reorganization.
- A large number of list prefetch requests might indicate that a tablespace needs to be reorganized (poor cluster ratio for a clustering index), or that unwanted hybrid join or multi-index access is taking place. It might also be possible that list prefetch is failing and reverting to a tablespace scan (which you can check under SQL counts - option G).

Other near-term thread detail options

We have indicated through the previous examples the kind of diagnostic information that is available through the near-term thread detail options to help you to identify thread problems and discover their causes. A number of other options are also available, which we have not looked at in detail, but which can also be very helpful in problem diagnosis in particular environments:

- If you are running in a data sharing environment then options D (Global locks) and J (Group BP) can provide helpful information at the data sharing level.
- For threads with a distributed component use option H (Distributed) for further information on distributed calls.
- For threads with a parallel component use option M (Parallel tasks).
- For threads running many packages within one plan use option K (Package summary) to get information at package level.

You should also be aware that to accurately diagnose problems, you need to look at the complete picture. Do not rely on information from one panel, but try to build up a picture from information provided from several thread detail panels, which all point in the same direction.

9.5 Near-term statistics history

While a thread history gives you an application view of what is running in a DB2 subsystem, showing details of threads that typically represent individual batch programs or online transactions, a statistics history gives you a system-wide view of a DB2 subsystem aggregated across all threads.

The Main Menu option R (Resource Managers And Other DB2 Subsystem Information) allows you to see the current values of the DB2 Statistics counters. Near-term statistics history provides the same information summarized into reporting intervals that are typically between 5 and 15 minutes.

Figure 9-20 shows you the first panel you see after selecting option A from the Near-term history information menu.

```
 ________________ ZHSMS    VTM     02      V310./C DB8A 01/30/06  0:49:00   4
>                  Enter a selection letter on the top line.
>
> *-SUBSYSTEM SUPPORT      B-BIND              C-BUFFER POOL    D-GROUP BP
> E-DISTRIBUTED DATABASE   F-EDM POOL          G-LOG MANAGER    H-OPEN/CLOSE
> I-SQL/RID/PARALLEL/PROC  J-LOCK/CLAIM/DRAIN  K-GLOBAL LOCK    L-DB2 COMMANDS
> O-OPTIONS
===============================================================================
>        SUBSYSTEM SUPPORT MANAGER STATISTICS SUMMARY BY REPORT INTERVAL
 HSMS
+ Collection Interval:  15 min                          Start:  01/29 00:30
+ Report Interval:      15 min   Combine Level:  NONE     End:  01/30 00:49
+
+                Create      Total                    Queued
+               Thread/    Commit/    Total   Abort       at  Indoubt  EOT+EOM
+ Interval       Minute     Minute  Commits    Reqs  Cthread  Threads   Abends
+ -----------  ---------  ---------  -------  ------  -------  -------  -------
+ 01/29 18:30        .00        .00        0       0        0        0        0
+ 01/29 18:15        .27     101.27     1519       0        0        0        0
+ 01/29 18:00        .00      14.73      221       0        0        0        0
```

Figure 9-20 Subsystem Support Manager Statistics Summary by Report Interval

9.5.1 Summary level statistics reporting

In Figure 9-20 you can see 12 summary options A to L listed at the top of the panel. These 12 summary options provide a very convenient way of quickly reviewing the most important system statistics over the previous 24 hours to check for system problems. We will briefly review some of the information that is available from these panels.

Subsystem support manager statistics summary (option A)

The Subsystem support manager statistics summary shown in Figure 9-21 can be used to provide the following information:

- An indication of the overall system throughput in terms of create threads/minute.
- The system commit rate/minute, which also gives an indication of system throughput.
- Abort requests during the reporting interval. A high value needs to be investigated. It may represent a program error or severe deadlocks and timeouts.
- A count of threads queued at create thread. A value greater than zero here probably indicates a system overload.
- A count of in doubt threads. A value greater than zero needs investigation using the DB2 command `-DISPLAY THREAD(*) TYPE(INDOUBT)`.

```
>        SUBSYSTEM SUPPORT MANAGER STATISTICS SUMMARY BY REPORT INTERVAL
 HSMS
+ Collection Interval:   5 min                           Start:  02/14 02:15
+ Report Interval:       5 min   Combine Level:  NONE      End:  02/14 18:52
+
+                 Create      Total                       Queued
+                Thread/    Commit/    Total   Abort        at  Indoubt  EOT+EOM
+ Interval        Minute     Minute  Commits    Reqs   Cthread  Threads   Abends
+ -----------  ---------  ---------  -------  ------  -------  -------  -------
+ 02/14 18:52        .00     269.50      539       0        0        0        0
+ 02/14 18:50        .00     109.80      549      12        0        0        2
+ 02/14 18:45        .80     158.60      793      12        0        0        0
```

Figure 9-21 Subsystem Support Manager Statistics Summary

Bind statistics summary (option B)

The Bind statistics summary shown in Figure 9-22 can be used to provide the following information:

- A count of the number of unsuccessful attempts by DB2 to automatically rebind a plan or package (auto bind attempts - auto bind successes). If this value is high, it might indicate that a number of invalid plans or packages exists because of DB2 table or index changes where a needed rebind has not been performed. This can be checked by running queries against the SYSIBM.SYSPLAN and SYSIBM.SYSPACKAGE catalog tables.

```
>                    BIND STATISTICS SUMMARY BY REPORT INTERVAL
 HBNS
+ Collection Interval:   5 min                           Start:  02/14 02:20
+ Report Interval:       5 min   Combine Level:  NONE      End:  02/14 18:56
+
+                Auto Bind      Static Bind       Rebind           Free
+ Interval    Attempt Success Attempt Success Attempt Success Attempt Success
+ ----------- ------- ------- ------- ------- ------- ------- ------- -------
+ 02/14 18:56       0       0       0       0       0       0       0       0
+ 02/14 18:55       0       0       0       0       0       0       0       0
+ 02/14 18:50       0       0       0       0       0       0       0       0
```

Figure 9-22 Bind Statistics Summary

Buffer pool statistics summary (option C)

The Buffer pool statistics summary in Figure 9-23 shows statistics for BP1. To show statistics for a different buffer pool, overtype the number with the required buffer pool number on the line starting with `HBPS` and press Enter.

```
>                BUFFER POOL STATISTICS SUMMARY BY REPORT INTERVAL
 HBPS  1
+ Collection Interval:   5 min                           Start:  02/14 02:20
+ Report Interval:       5 min   Combine Level:  NONE      End:  02/14 18:56
+
+               *
+                 Pages   Prefetch   Prefetch   Getpages/      Pages/
+ Interval       in Use   Requests   Failures    Sync I/O   Write I/O      DMTH
+ -----------   -------   --------   --------   ---------   ---------   -------
+ 02/14 18:50       785       5676          0    32535.95       14.23         0
+ 02/14 18:45       659       7236          0     7648.98        7.35         0
+ 02/14 18:55       447       7184          0    33175.66       15.71         0
```

Figure 9-23 Buffer Pool Statistics Summary

The summary can be used to provide basic information on how the buffer pool is performing across all threads:

- The Data manager threshold count (DMTH), which represents a critical threshold for the buffer pool above, for which performance severely deteriorates, should never be greater than zero.
- Prefetch failures should be less than 1% to 5% of prefetch requests.

Distributed data facility statistics summary (option E)

The Distributed data facility statistics summary shown in Figure 9-24 can be used to provide an overview of how well distributed SQL is performing and the size of the distributed workload.

```
>       DISTRIBUTED DATA FACILITY STATISTICS SUMMARY BY REPORT INTERVAL
 HDFS
+ Collection Interval:   5 min                           Start:  02/14 02:20
+ Report Interval:       5 min   Combine Level:  NONE      End:  02/14 18:57
+
+              # of  Total  Total  Total  Total  Total  Total Total  Total Total
+ Interval     Rmts  Trans    SQL   Rows   Msgs  Bytes Commit Abort   Conv Convq
+ ----------- ---- ------ ------ ------ ------ ------ ------ ----- ------ -----
+ 02/14 18:57    1      0      0      0      0      0      0     0      0     0
+ 02/14 18:55    1      0      0      0      0      0      0     0      0     0
+ 02/14 18:50    1      0      0      0      0      0      0     0      0     0
```

Figure 9-24 Distributed Data Facility Statistics Summary

EDM pool statistics summary (option F)

The EDM pool statistics summary shown in Figure 9-25 can be used to provide the following information:

- The percentage of non-stealable pages in the EDM pool in use by database descriptors (DBD), cursor tables (CT) and package tables (PT). If the EDM pool is adequately allocated this figure should not generally rise above 50%.
- The percentage of times a DBD is not found in the EDM pool and needs to be loaded from the DB2 directory (DSNDB01). This should generally be between 1% and 5%.
- The percentage of times a CT is not found in the EDM pool and needs to be loaded from the DB2 directory (DSNDB01). This should generally be between 1% and 5%.
- The percentage of times a PT is not found in the EDM pool and needs to be loaded from the DB2 directory (DSNDB01). This should generally be less than 5%.
- The number of stealable pages being used for the dynamic SQL cache (DSC).

```
>                  EDM POOL STATISTICS SUMMARY BY REPORT INTERVAL
 HEDS
+ Collection Interval:   5 min                            Start:  02/14 02:20
+ Report Interval:       5 min   Combine Level:  NONE       End:  02/14 18:57
+
+              Pages    DBD    DBD     CT     CT     PT     PT     DSC    DSC
+ Interval     in Use%  Pages  Load%  Pages  Load%  Pages  Load%  Pages  Load%
+ -----------  -------  -----  -----  -----  -----  -----  -----  -----  -----
+ 02/14 18:55      13%    494   .00%      4   .00%      0   .00%    130   .00%
+ 02/14 18:50      18%    494   .00%      7   .00%    165   .00%    130   .00%
+ 02/14 18:45      19%    494   .00%      8   .00%    213   .00%    130   .37%
```

Figure 9-25 EDM Pool Statistics Summary

Log manager statistics summary (option G)

The Log manager statistics summary shown in Figure 9-26 can be used to provide an overview of the performance of the DB2 logs. The number of read delays and write delays in an interval should not generally be greater than zero.

```
>                LOG MANAGER STATISTICS SUMMARY BY REPORT INTERVAL
 HLGS
+ Collection Interval:   5 min                            Start:  02/14 02:20
+ Report Interval:       5 min   Combine Level:  NONE       End:  02/14 18:58
+
+                                                            Archive  Archive
+                Total     Read    Total    Write     BSDS     Read    Write
+ Interval       Reads    Delay   Writes    Delay   Access   Allocs   Allocs
+ -----------  -------  -------  -------  -------  -------  -------  -------
+ 02/14 18:55     4540        0    53496        0        6        0        0
+ 02/14 18:50     4625        0    51061        0        3        0        0
+ 02/14 18:45     6801        0    58926        0        4        0        0
```

Figure 9-26 Log Manager Statistics Summary

Open/Close statistics summary (option H)

The Open/Close statistics summary shown in Figure 9-27 can be used to provide an overview of data set activity. The count of not-in-use data sets closed, which represents data sets that have been physically closed because the deferred threshold has been reached, should generally be low.

```
>                OPEN/CLOSE STATISTICS SUMMARY BY REPORT INTERVAL
 HOPS
+ Collection Interval:   5 min                          Start:  02/14 02:20
+ Report Interval:       5 min   Combine Level:  NONE      End:  02/14 18:58
+
+              High Water   Current   Not-in-use   Not-in-use   DS Conv to
+ Interval        Open DS   Open DS   DS Request    DS Closed    Read-only
+ -----------  ----------   -------   ----------   ----------   ----------
+ 02/14 18:55         254       253         6295            0            0
+ 02/14 18:50         254       253         4479            0            0
+ 02/14 18:45         254       253         6275            0            0
```

Figure 9-27 Open/Close Statistics Summary

SQL statistics summary (option I)

The SQL statistics summary shown in Figure 9-28 can be used to provide an overview of the rate at which SQL is being executed on the system. The manipulative DML quantity per minute (SELECT, INSERT, UPDATE, DELETE etc.) gives a good indication of the DB2 throughput and whether the DB2 subsystem is under stress. It is also a good indicator of when the peak DB2 loads occur during the day.

```
>                    SQL STATISTICS SUMMARY BY REPORT INTERVAL
 HSQS
+ Collection Interval:   5 min                          Start:  02/14 02:20
+ Report Interval:       5 min   Combine Level:  NONE      End:  02/14 18:59
+
+              Manipulative (DML)     Control (DCL)     Definitional (DDL)
+ Interval     Quantity   /Minute   Quantity   /Minute   Quantity   /Minute
+ -----------  --------  --------   --------  --------   --------  --------
+ 02/14 18:55   1461306  292261.2      78287  15657.40         66     13.20
+ 02/14 18:50   1091748  218349.6      68795  13759.00         40      8.00
+ 02/14 18:45   1500093  300018.6     104656  20931.20         50     10.00
```

Figure 9-28 SQL Statistics Summary

Lock manager statistics summary (option J)

The Lock manager statistics summary shown in Figure 9-29 can be used to provide the following information:

- A count of the number of deadlocks and timeouts occurring across the DB2 subsystem. If more than 5 to 10 deadlocks and timeouts per hour are occurring, the cause should be investigated.
- A count of locks that have been escalated to shared table level (SHR) or exclusive table level (EXC). Escalated locks should generally be investigated to ensure that they are not causing disruption to online applications.

```
>               LOCK MANAGER STATISTICS SUMMARY BY REPORT INTERVAL
 HLKS
+ Collection Interval:   5 min                           Start:  02/14 02:20
+ Report Interval:       5 min   Combine Level:  NONE      End:  02/14 18:59
+
+                                      Total      Total  Escalate  Escalate
+ Interval     Deadlocks  Timeouts  Suspends  Lock Reqs    to SHR    to EXC
+ -----------  ---------  --------  --------  ---------  --------  --------
+ 02/14 18:55          0         0        24     439756         0         0
+ 02/14 18:50         11         0       202     454029         0         0
+ 02/14 18:45         12         0       248     500548         0         0
```

Figure 9-29 Lock Manager Statistics Summary

DB2 command statistics summary (option L)

The DB2 command statistics summary shown in Figure 9-30 can be used to provide an overview of DB2 commands issued. These should be investigated if unexpected commands are being issued (for example, `-STOP DATABASE` or `-CANCEL THREAD`).

```
>              DB2 COMMAND STATISTICS SUMMARY BY REPORT INTERVAL
 HCMS
+ Collection Interval:   5 min                           Start:  02/14 02:25
+ Report Interval:       5 min   Combine Level:  NONE      End:  02/14 19:00
+
+              START/      START/STOP/           RECOVER/  CANCEL    ALTER
+              STOP        MODIFY       ARCHIVE  RESET     DDF       BUFFER
+ Interval     DATABASE    TRACE        LOG      INDOUBT   THREAD    POOL
+ -----------  ----------  -----------  -------- --------  --------  --------
+ 02/14 19:00           0            0         0        0         0         0
+ 02/14 18:55           0            0         0        0         0         0
+ 02/14 18:50           0            0         0        0         1         0
```

Figure 9-30 DB2 Command Statistics Summary

9.5.2 Detail level statistics reporting

For each of the summary level statistics options A to L described in 9.5.1, “Summary level statistics reporting” on page 155, there is a corresponding detail level option. The detail level option is reached by positioning the cursor on the interval you are interested in and pressing PF11 (Zoom).

Figure 9-31 shows part of a typical statistics detail level panel.

```
>                 SQL/RID POOL/I/O PARALLELISM STATISTICS DETAIL
 HSQD
+ Collection Interval:  15 min                          Start:  02/15 21:00
+ Report Interval:       2 min   Combine Level:  NONE      End:  02/15 21:01
+ Maximum Degree of Parallelism Executed:       0
+
+                                  INTERVAL  /MINUTE  /THREAD  /COMMIT    % OF
+ SQL Manipulative (DML)           QUANTITY  (    2)  (    1)  (  106)     DML
+ ----------------------           --------  -------  -------  -------  ------
+ SELECT                                 15     7.50    15.00      .14    5.62
+ INSERT                                  0      .00      .00      .00     .00
+ UPDATE                                 16     8.00    16.00      .15    5.99
+ DELETE                                  1      .50     1.00      .01     .37
+ OPEN CURSOR                            77    38.50    77.00      .73   28.84
+
+                                  INTERVAL  /MINUTE  /THREAD  /COMMIT
+ RID Pool Accesses                QUANTITY  (    2)  (    1)  (  106)
+ ----------------------           --------  -------  -------  -------
+ Successful                              0      .00      .00      .00
+ Not Used (No Storage)                   0      .00      .00      .00
+ Not Used (Max Limit)                    0      .00      .00      .00
+ RID Terminated (> RDS)                  0      .00      .00      .00
+ RID Terminated (> DM)                   0      .00      .00      .00
```

Figure 9-31 SQL/RID POOL/I/O Parallelism Statistics Detail

Whereas the summary panels highlight a few important statistics counters, the detail level panels give access to the full range of statistics counters provided by DB2.

For each statistics counter, four values are generally provided:

- The interval quantity (reflecting the amount of activity that occurred during the interval)
- The rate per minute
- The rate per thread
- The rate per commit

Note that the column headings for the three rate columns include a count in parentheses. The number under `/MINUTE` is the number of minutes in the interval; the number under `/THREAD` is the number of create threads during the interval; the number under `/COMMIT` is the number of commit requests (including abort requests) during the interval.

Full details of all the statistics counters available on the statistics detail panels can be found in Chapter 15, “Near-term history information”, in *Monitoring Performance from the OMEGAMON Classic Interface*, SC18-9659.

10

Application Trace Facility

In this chapter we introduce the Application Trace Facility, the function allowing selective tracing of thread events associated with DB2 applications.

We discuss the following topics:

- Application Trace Facility overview
- Application Trace Facility dataset management
- Trace qualification and management
- Trace output review
- Performance considerations and overhead

10.1 Application Trace Facility overview

The Application Trace Facility allows the selective tracing of thread events associated with DB2 applications. Some of the information that can be obtained from the Application Trace Facility includes SQL activity, Sort, DB2 pageset and buffer activity, Class 3 times including locking and I/O delay, and Class 1 and Class 2 elapsed and CPU times.

Trace data collected by the Application Trace Facility can be directed to either z/OS data spaces, or can be stored into pre-allocated VSAM linear datasets. During the ICAT configuration process, there was a “default” set of Application Trace Facility datasets created. In order to support concurrent tracing activity, we suggest that each Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS user allocate their own trace datasets using the trace dataset management facility described in the next section.

Once the Application Trace Facility starts, as qualified threads complete, they are available for viewing from the view trace dialog. More than one Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS use can view the trace output simultaneously. Again, keep in mind that only completed threads are displayed, so for example, if there are long running threads that appear in the active thread display, trace events for these threads are not reflected in the Application Trace Facility output.

Figure 10-1 shows the panel ZATMENU, used to manage the Application Trace Facility.

```
 ________________ ZATMENU  VTM     O2       V310./C DB8A 02/07/06 19:14:56   2
>       Help PF1                                              Back PF3
> A.
>                 Enter a selection letter on the top line.
===============================================================================
>                       APPLICATION TRACE FACILITY MENU

_ A  SPECIFY TRACE ...... Request and start an application trace
_ B  VIEW TRACE ......... View the active trace
_ C  STOP TRACE ......... Stop the active trace

_ D  SELECT DATASET ..... Specify a trace dataset to view
_ E  VIEW DATASET ....... View the selected trace dataset
_ F  STOP VIEW .......... Release the selected dataset

_ G  CREATE DSN ......... Create a new VSAM LDS for trace output

===============================================================================
```

Figure 10-1 Application Trace Facility Menu

In the following sections, we discuss how each of these options can be used to allocate, use, and display trace records collected by the Application Trace Facility.

For a detailed and accurate description of the trace records of the DB2 release you are using, check member DSNWMSGS of either SDSNSAMP or SDSNIVPD library.

10.2 Application Trace Facility dataset management

Prior to initiating tracing, we must create a trace dataset for the Application Trace Facility to store collected thread trace records if we wish to save these trace events for subsequent viewing. These datasets can be allocated by the Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS using the Create DSN option "G". The following Figure 10-2 shows panel ZATMENU, used to manage the Application Trace Facility.

```
 ________________ ZLDSC    VTM     O2      V310./C DB8A 02/07/06 19:19:01   2
>         Help PF1                                           Back PF3
> A.G
> A-SPECIFY TRACE    B-VIEW TRACE      C-STOP TRACE      D-SELECT DSN
> E-VIEW DATASET     F-STOP VIEW       *-CREATE VSAM LDS
===============================================================================
>                          CREATE APPLICATION TRACE DSN
 LDSC
+
:     DSN=____________________________________________      Data set name
:     SIZE=050           Size of dataset in megabytes (1-999)
:     VOLUME=______      Volume id for dataset creation
===============================================================================
```

Figure 10-2 Create Application Trace DSN

In our environment, we took the default setting of 50 MB, this resulted in an allocation of the trace dataset that was 72 cylinders in size. Note that the Application Trace Facility uses only the primary extent of a dataset; it does not use secondary extents. Keep this in mind when considering space requirements for the trace data. The allocation also requires a VOLSER to be specified, and the Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS user needs to have RACF create authority into the catalog alias that owns the dataset name selected. Application Trace Facility datasets can be preallocated and reused as needed.

In referencing the *Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS Configuration and Customization Guide,* GC18-9637, there are some suggested sizings based on an average accounting record size of 1880 bytes. This results in approximately 452 records (or threads) per 3390 cylinder. Given the 50 MB allocation selected above, we would expect to be able to trace approximately 32,500 threads before exhausting the Application Trace Facility dataset. Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS users should take the product of the desired trace interval duration and the number of anticipated completed thread events per minute to assist in the sizing of their Application Trace Facility dataset.

10.3 Trace qualification and management

In order to start the trace collection process, you must first configure the trace using the Specify Application Trace screen. This screen, like many of the VTAM classic screens, has additional options that require the user to press PF8 in order to be seen. Figure 10-3 shows panel ZATREQ, used to manage the Application Trace Facility.

```
 ________________ ZATRQ    VTM     02       V310./C DB8A 02/07/06 21:16:59   2
>        Help PF1                                      Back PF3
> A.A
> *-SPECIFY TRACE    B-VIEW TRACE       C-STOP TRACE       D-SELECT DSN
> E-VIEW DATASET     F-STOP VIEW        G-CREATE VSAM LDS
===============================================================================
>                          SPECIFY APPLICATION TRACE
 ATRQ
+ Type DB2 Plan name to be traced. Also, provide additional optional
+ selection information to limit trace output. To save trace records
+ for later viewing you must specify a data set name for DSN
+
:     DSN=____________________________________________      Data set name
:     TIME=005           Number of mins to trace (001-060)
:     PLANNAME=________  Plan name or ALL for all active threads
:     AUTHID=________    DB2 authorization identifier
:     TSOUSER=________   TSO USERID (TSO foreground app)
:     JOBNAME=________   Jobname (TSO batch app)
:     CICSTRAN=____      CICS trans id
:     CICSCONN=________  CICS connection id)
:     PSBNAME=________   IMS PSB name
:     IMSID=________     IMS ID of the IMS region
:     LOCKDATA=Y         Collect DB2 lock trace recs? (Y/N)
```

Figure 10-3 Specify Application Trace - Part 1

If the trace data is only needed for the current Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS session and is not intended to be saved for subsequent analysis, the DSN value is not required; this collection choice is also referred to as an *in-core* collection. If the collected trace records are to be stored for later review, the DSN field is required and then needs to reflect a valid linear VSAM dataset created as described in the previous section.

There are a number of optional selection criteria that can allow us to limit the number of trace records collected. Using one or more of these filters can provide for more efficient storage of trace data. The PLANNAME is required and should either specify a specific plan, or *ALL* if you wish to collect events for all plans. If you specify PLANNAME or AUTHID filters, the DB2 traces that are started by the Application Trace Facility are filtered and some reduction in trace overhead can also be achieved.

The trace interval is also required in order to control the duration of the trace activity. We need to weigh the choice for the value specified in this field against the size of the in-core storage allocation or the VSAM dataset allocated as described above. In selecting a value, ensure that sufficient collection allocation has been provided in order to support the number of committed traced thread events that would arrive over the chosen trace time interval.

Figure 10-4 shows the second half of panel ZATREQ, with some additional parameters used to manage the Application Trace Facility.

```
 ________________ ZATRQ    VTM     02       V310./C DB8A 02/08/06 11:56:46  24
 :     SCANDATA=Y         Collect DB2 scan trace recs? (Y/N)
 :     SQLDATA=Y          Collect DB2 sql  trace recs? (Y/N)
 :     THRDDATA=Y         Collect DB2 thread trace recs?  (Y/N)
 :     CONNDATA=Y         Collect DB2 connect trace recs? (Y/N)
 :     SMF=N              Write trace data to SMF? (Y/N)
 :     GTF=N              Write trace data to GTF? (Y/N)
 :     MEMSIZE=02         Collection workarea memory size (01-04 meg)
 ===============================================================================
```

Figure 10-4   Specify Application Trace - Part 2

Some additional controls can be applied as shown above to further reduce the level of detail collected by the Application Trace Facility.

Tip: While the default for these parameters is “Y”, we recommend that you try to collect the minimum data needed for your problem analysis scenario. So if we are suspecting a problem relating to concurrency, we might specify LOCKDATA = Y; with other situations, we might elect to specify LOCKDATA = N.

Optionally, in addition to directing the trace to a dataset or data space, the trace output can also be directed to SMF or GTF. These two parameters are mutually exclusive, only one can be specified as “Y”.

The memory size parameter MEMSIZE has a maximum value of 04 MB. If we apply the formula discussed in the above section, we would expect that with all of the data collection parameters specified as “Y”, we should be able to collect data for approximately 2600 threads with the maximum MEMSIZE setting of 04 MB. So without filtering, in an environment with a high transaction arrival rate, we would fill up the storage quickly. A better choice in this scenario might be to direct the trace to a VSAM trace dataset, or to apply filtering to limit the number of qualifying thread events that would be captured.

Once we complete the trace configuration, we can then start the trace by pressing Enter, we are then prompted as shown in Figure 10-5.

```
 ________________ ZATRQ    VTM     O2       V310./C DB8A 02/08/06 19:52:05   2
>         Help PF1                                            Back PF3
> A.A
> *-SPECIFY TRACE    B-VIEW TRACE       C-STOP TRACE       D-SELECT DSN
> E-VIEW DATASET     F-STOP VIEW        G-CREATE VSAM LDS
===============================================================================
>                          SPECIFY APPLICATION TRACE
 ATRQ
+    The following trace request has been selected, to activate the trace
+    request press the ENTER key, to CANCEL the trace request use PF3 :
+
:     DSN=____________________________________________       Data set name
:     TIME=015            Number of mins to trace (001-060)
:     PLANNAME=ALL        Plan name or ALL for all active threads
:     AUTHID=PAOLOR3      DB2 authorization identifier
:     TSOUSER=________    TSO USERID (TSO foreground app)
:     JOBNAME=________    Jobname (TSO batch app)
:     CICSTRAN=____       CICS trans id
:     CICSCONN=________   CICS connection id)
:     PSBNAME=________    IMS PSB name
:     IMSID=________      IMS ID of the IMS region
:     LOCKDATA=Y          Collect DB2 lock trace recs? (Y/N)
:     SCANDATA=Y          Collect DB2 scan trace recs? (Y/N)
```

Figure 10-5 Start Application Trace - Prompt

To confirm, press Enter a second time, and the trace confirmation prompt is generated as shown in Figure 10-6.

```
 ________________ ZATRQ    VTM     O2       V310./C DB8A 02/08/06 19:47:43   2
>         Help PF1                                            Back PF3
> A.A
> *-SPECIFY TRACE    B-VIEW TRACE       C-STOP TRACE       D-SELECT DSN
> E-VIEW DATASET     F-STOP VIEW        G-CREATE VSAM LDS
===============================================================================
>                          SPECIFY APPLICATION TRACE
 ATRQ
+      >> KO201101I APPLICATION TRACE REQUEST HAS BEEN STARTED. TRACE INITI <<
+      >> ALIZATION IS IN PROGRESS (ATRCATRQ)                               <<
===============================================================================
```

Figure 10-6 Start Application Trace - Confirmation

While the trace is executing, we can monitor the trace progress. Option B, entered on the command line, shows the status of the trace execution. An example of this is shown in Figure 10-7.

```
 ________________ ZATVW    VTM     O2       V310./C DB8A 02/08/06 20:22:13   2
>       Help PF1      Back PF3      Up PF7      Down PF8       Zoom PF11
> A.B
> A-SPECIFY TRACE    *-VIEW TRACE       C-STOP TRACE       D-SELECT DSN
> E-VIEW DATASET     F-STOP VIEW        G-CREATE VSAM LDS
===============================================================================
>                     APPLICATION TRACE THREAD SUMMARY
 ATVW
+ Trace Status          = ACTIVE       Trace Time Remaining      = 00:12:35
+ Trace Start Time      = 20:19:48     Trace End Time            = 00:00:00
+ Trace Time Limit      = 00:15:00     Trace Records Collected   =     2957
+
+ Trace Request Information :
+
+ PLANNAME   = ALL
+ AUTHID     = PAOLOR3
+
+ Planname Connid   Corrid       Authid   InDB2 CPU  SQL      Commits   Aborts
+ -------- -------- ------------ -------- ---------  -------  --------  ------
+ ADB      TSO      PAOLOR3      PAOLOR3     .02741      488         3       0
+ DSNESPCS TSO      PAOLOR3      PAOLOR3     .00015        1         0       0
```

Figure 10-7 View Application Trace

View trace shows us the status of the trace, requested time interval, trace time remaining, number of trace records collected, and other important status indicators. In this example, we elected to perform an in-core trace collection. If we had specified a dataset based collection, we would see some additional fields. Among these would be a count of the number of trace records and UOW records that were lost during the collection period. If these numbers are high, this is an indication that the transaction arrival rate exceeds the ability of the Application Trace Facility to externalize the data. In this event, we might elect to restart the trace with additional trace criteria specified to further filter the data collection.

Based on our qualification of all threads for the authid of PAOLOR3, we can see some summary details about the collected thread activity. For example, in the above figure, we see that there were three threads running in planname ADB that qualify to the authid filter of PAOLOR3. We also can see the Class 2 In-DB2 CPU as well as the number of SQL statements. In this case, we would probably not spend much time looking at this set of events; the average CPU per SQL statement is very modest. If, on the other hand, the In-DB2 CPU was excessive, we would then drill down by positioning the cursor on the ADB entry, and then press PF11 (ZOOM), this then allows us to begin navigation through the application trace details

10.4 Trace output review

In order to better understand the navigation within the Application Trace Facility, please refer to Figure 10-8.

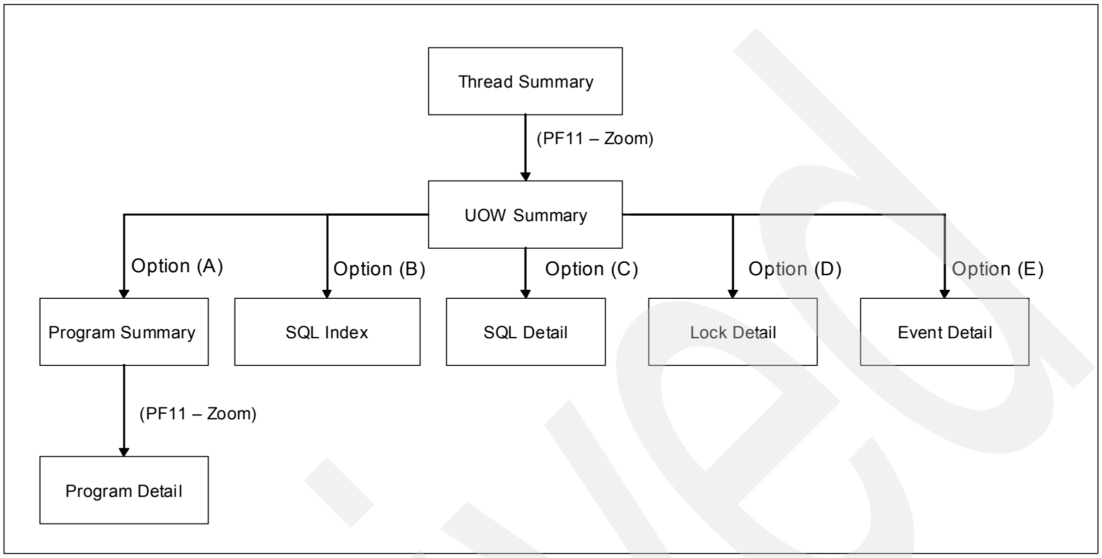


Figure 10-8 Application Trace Hierarchy

From each entry on the Thread Summary screen, we can press PF11 to ZOOM to the next level of detail, UOW Summary. The Unit of Work summary contains a single line of summary information defined by either a commit or a thread abort for the located thread. The Unit of Work entries are stored and presented in a LIFO or in last in, first out order, this can be verified by looking at the Start Time column of the display. When viewing a trace that has been captured to VSAM, the data may also be sorted by date of occurrence, assuming the trace collection crossed date boundaries. shows an example of the Unit of Work summary screen. Figure 10-9 shows a typical Unit of Work summary screen.

```
 ________________ ZATVC    VTM     O2       V310./C D8F2 02/09/06 17:06:04   2
> Help PF1        Back PF3         Up PF7         Down PF8         Zoom PF11
===============================================================================
>                      APPLICATION TRACE UNIT OF WORK SUMMARY
 ATVC
+ Planname=ADB        Connid=TSO        Corrid=PAOLOR3        Authid=PAOLOR3
+
+ Start Time   Progname  InDB2 Time   InDB2 CPU  SQL    Sorts   Locks   Rows
+ ------------ --------  ------------ ---------  ------ ------  ------  -------
+ 17:05:31.914 ADBMAIN   00:00.01962     .01540      72      0     445     8483
+ 17:05:29.622 ADBMAIN   00:00.00113     .00073      19      0       3       34
+ 17:05:27.294 ADBMAIN   00:00.02635     .02329     323      0     503     8480
===============================================================================
```

Figure 10-9 Application Unit of Work Summary

The values shown on this screen are total counts. To find out more detail about a specific unit of work, for example, the one that consumed the most CPU, position the cursor under the entry of interest and press PF11. This will launch the Program Summary screen, shown in Figure 10-10.

```
 ________________ ZATVS    VTM     02       V310./C D8F2 02/13/06 18:16:07   2
> Help PF1         Back PF3         Up PF7         Down PF8         Zoom PF11

>         APPLICATION TRACE:  Enter a selection letter on the top line.

> *-PROGRAM    B-SQL INDEX    C-SQL DETAIL    D-LOCK DETAIL    E-EVENT DETAIL
===============================================================================
>                      APPLICATION TRACE PROGRAM SUMMARY
 ATVS
+ Planname=ADB        Connid=TSO        Corrid=PAOLOR3        Authid=PAOLOR3
+
+ Progname  InDB2 Time    InDB2 CPU  SQL     Sorts   Locks   Pages   Rows
+ --------  ------------  ---------  ------  ------  ------  ------  -------
+ ADBMAIN   00:00.01319      .01227      72       0     440     606     8480
===============================================================================
```

Figure 10-10 Application Trace Program Summary

Notice that we launch into the Program section of the trace event in question. From the Program Summary screen, we can select different sections of detail for further analysis. SQL Index, for example, gives us a view of each SQL statement summarized by SQL statement type, and again the values are totaled by each statement call type. Figure 10-11 shows an example of this screen.

```
 ________________ ZATSI    VTM     02       V310./C D8F2 02/13/06 18:28:43   2
> Help PF1         Back PF3         Up PF7         Down PF8         Zoom PF11

>         APPLICATION TRACE:  Enter a selection letter on the top line.

> A-PROGRAM    *-SQL INDEX    C-SQL DETAIL    D-LOCK DETAIL    E-EVENT DETAIL
===============================================================================
>                      APPLICATION TRACE SQL INDEX
 ATSI
+ Planname=ADB        Connid=TSO        Corrid=PAOLOR3        Authid=PAOLOR3
+
+ Call Type        Stm#  Program  Count InDB2 Time   MRet Rws Pc Rws DM Rws RD
+ ---------------- ----- -------- ----- ------------ ---- ------ ------ ------
+ PREPARE           3740 ADBMAIN      1 00:00.00009     0      0      0      0
+ OPEN CURSOR          0 ADBMAIN      1 00:00.00002     0      0      0      0
+ FETCH                0 ADBMAIN     69 00:00.01316   100   8480     68     68
+ CLOSE CURSOR         0 ADBMAIN      1 00:00.00001     0      0      0      0
===============================================================================
```

Figure 10-11 Application Trace Program Summary

Positioning the cursor on a statement that has a valid statement number — in the above example, the prepare statement — if we press PF11, we then navigate to the SQL detail screen. We can also navigate to this screen by selecting option B on the command line. Selecting SQL Detail, we then can see each SQL statement execution as it occurs, in order of statement execution. Figure 10-12 shows an example of this screen.

```
 _______________ ZATD1    VTM     02      V310./C D8F2 02/13/06 18:33:16   2
> Help PF1              Back PF3              Up PF7                Down PF8

>        APPLICATION TRACE:  Enter a selection letter on the top line.

> A-PROGRAM    B-SQL INDEX    *-SQL DETAIL    D-LOCK DETAIL    E-EVENT DETAIL
===============================================================================
>                        APPLICATION TRACE SQL DETAIL
 ATD1
+ Planname=ADB        Connid=TSO        Corrid=PAOLOR3        Authid=PAOLOR3
+
: Control=NEXT    (Valid options are FIRST/LAST/NEXT/PREV/+nnnnn/-nnnnn/Snnnnn)
+ Current=000001  Total Number of SQL Calls=000072
+
+ Start Time   Progname  SQL Call         Stmt# Retcode  InDB2 Time   InDB2 CPU
+ ------------ --------  ---------------- ----- -------  ------------ ---------
+ 17:55:04.180 ADBMAIN   PREPARE          03740       0  00:00.00009     .00009
+
+ Data Rows   Rows   Rows    Rows    Rows   Rows   Rows   Rows   Pages  Pages
+ Type Proces Looked Qual/DM Qual/RD Update Insert Delete De/Ref Scand  Sc/Ref
+ ---- ------ ------ ------- ------- ------ ------ ------ ------ ------ ------
+              (No Data Activity Located For This Call)
+
```

Figure 10-12 Application Trace SQL detail

This screen shows each SQL statement in order of execution. The current field shows the relative number of the SQL statement being displayed. It also shows the total number of SQL statements captured by the trace. The control field allows us to page forward through the trace reviewing each SQL event as captured in the trace, we can also see the first or the last statement by using the appropriate control option. The information that is displayed in the SQL detail screen includes the time that the call was processed, the name of the program, the SQL call type, the SQL statement number, the SQL return code, the elapsed time spent executing the SQL call, and the CPU time spent executing the SQL call.

For dynamic SQL, the statement number has no meaning; therefore, the text of the SQL call is displayed. In addition, for each SQL call, for index pagesets, data pagesets, and data workfile pagesets, it shows the number of rows processed, the number of rows examined, the number of rows qualified by the data manager, the number of rows qualified by the relational manager, the number of rows inserted and deleted, the number of rows deleted due to referential integrity enforcement, the number of pages scanned, and the number of pages scanned to enforce referential integrity.

In order to see the SQL statement text (if available) you must press PF8 to page down in order to locate the SQL text. Figure 10-13 shows an example of this screen.

```
 ________________ ZATD1    VTM     02       V310./C D8F2 02/13/06 18:48:34  25
+ Dynamic SQL Call Text :
+ -----------------------
+ SELECT T.* FROM SYSIBM.SYSTABLESPACE T   FOR FETCH ONLY
+
+ Miniplan Generated by DB2 :
+ ---------------------------
+ Miniplan Not Available
================================================================================
```

Figure 10-13   Application Trace Program Summary

10.5 Performance considerations and overhead

As we discussed in 10.3, "Trace qualification and management" on page 166, the level of trace data collection can be controlled within the Application Trace Facility using the different trace data collection flags. We first took a look at the performance overhead from a DB2 tracing perspective. We recycled the server and observed what traces were active without the Application Trace Facility running. The following display trace (*) command output in Figure 10-14 shows the active traces.

```
DSNW127I  -D8F2 CURRENT TRACE ACTIVITY IS -
TNO TYPE   CLASS        DEST QUAL
01  STAT   01,03,04,05, SMF  NO
01         06
02  ACCTG  01           SMF  NO
03  ACCTG  01,02,03,07, OP1  NO
03         08
04  MON    01           OP2  NO
05  ACCTG  01,02,03,07, OP2  NO
05         08
06  PERFM  30           OP2  NO
*********END OF DISPLAY TRACE SUMMARY DATA*********
DSN9022I  -D8F2 DSNWVCM1 '-DISPLAY TRACE' NORMAL COMPLETION
***
```

Figure 10-14   Display trace command output - ATF inactive

Next, we started the Application Trace Facility with the default collection flags specified and observed the following trace activity, this is shown in Figure 10-15.

```
DSNW127I  -D8F2 CURRENT TRACE ACTIVITY IS -
TNO TYPE   CLASS        DEST QUAL
01  STAT   01,03,04,05, SMF  NO
01         06
02  ACCTG  01           SMF  NO
03  ACCTG  01,02,03,07, OP1  NO
03         08
04  MON    01           OP2  NO
05  ACCTG  01,02,03,07, OP2  NO
05         08
06  PERFM  30           OP2  NO
07  PERFM  31           OP3  YES
*********END OF DISPLAY TRACE SUMMARY DATA*********
DSN9022I  -D8F2 DSNWVCM1 '-DISPLAY TRACE' NORMAL COMPLETION
***
```

Figure 10-15 Display trace command output - ATF active

We see that there is an additional PERFM class 31 (user initiated) trace active. We then used the Current Trace Activity screen, which can be navigated to from the Miscellaneous Options selection on the main menu. Looking at the trace detail, we can see the individual IFCIDs started for each particular execution of the Application Trace Facility.

```
 ________________ ZTRACED  VTM     O2       V310./C D8F2 02/13/06 20:34:31   2
> Help PF1              Back PF3             Up PF7                  Down PF8
===============================================================================
>                            ACTIVE TRACE DETAIL
 DTRD
+    Trace Number (TNO)        = 007               Trace Planname = *
+    Trace Type                = PERFORMANCE       Trace Authid   = PAOLOR3
+    Trace Destination         = OP3
+    Trace Rmids               = *
+    Trace Class               = 31
+    Trace Location            = *
+    Trace Tdata               = CORRELATION,CPU
+
+    Misc Trace Information    = JOBNAME  = D8F202S  ASID    = 0095
+                                PLANNAME = KO2PLAN  AUTHID  = D8F202S
+                                CONNID   = D8F202S  CORRID  = D8F202S
+                                BUFSIZE  = 2048K
+
+    Total IFCIDS Active       = 34
+
+               IFCID    IFCID Description
+               _____    ________________________________________
+                015     START OF INDEX SCAN
```

Figure 10-16 Active Trace Detail

Figure 10-16 shows the trace qualification, the number of active IFCIDs, and each IFCID along with a short description. In our analysis, we elected to review three different scenarios. The first scenario was turning on the Application Trace Facility with all of the optional data collection flags set to “N”. For second case, we turned on all of the data collection flags, suppressing only Lock Data and Scan Data. For the final permutation, we started the Application Trace Facility with all data collection flags turned on. During this process, we observed which IFCIDs were active for each of the scenarios. Table 10-1 shows the IFCIDs associated with the different Application Trace Facility flag settings.

Table 10-1 IFCIDs used by Application Trace Facility

IFCIDS	None	No Scan or Sort	All	Description
15	N	N	Y	START OF INDEX SCAN
16	N	N	Y	START OF INSERT SCAN
17	N	N	Y	START OF SEQUENTIAL SCAN
18	N	N	Y	END OF HASH/INDX/SEQU/ISRT SCAN
21	N	N	Y	RECORD LOCK DETAIL EVENTS
22	Y	Y	Y	RECORD MINI-PLANS GENERATED
58	N	Y	Y	END OF SQL CALL
59	N	Y	Y	START SQL FETCH CALL
60	N	Y	Y	START SQL SELECT CALL
61	N	Y	Y	START SQL INSERT/UPDATE DELETE CALL
62	N	Y	Y	START SQL DDL CALL
63	N	Y	Y	RECORD BIND EVENT PROCESSING
64	N	Y	Y	START SQL PREPARE
65	N	Y	Y	START SQL OPEN CURSOR
66	N	Y	Y	START SQL CLOSE CURSOR
68	N	Y	Y	START ABORT REQUEST
69	N	Y	Y	END ABORT REQUEST
70	N	Y	Y	START COMMIT PHASE II
71	N	Y	Y	END COMMIT PHASE II
73	N	Y	Y	END CREATE THREAD
74	N	Y	Y	START TERMINATE THREAD
84	N	N	Y	START COMMIT PHASE I
85	N	N	Y	END COMMIT PHASE I
86	N	N	Y	START SIGNON REQUEST
87	N	N	Y	END SIGNON REQUEST
88	N	N	Y	START COMMIT (SYNC) REQUEST
89	N	N	Y	END COMMIT (SYNC) REQUEST

IFCIDS	None	No Scan or Sort	All	Description
95	Y	Y	Y	START SORT PROCESSING
96	Y	Y	Y	END SORT PROCESSING
177	Y	Y	Y	RECORD SUCCESSFUL PACKAGE ALLOC
221	Y	Y	Y	PARALLEL DEGREE FOR PARALLEL GROUP
222	Y	Y	Y	PARALLEL GROUP ELAPSED TIME
223	Y	Y	Y	PARALLEL GROUP TASK TIME TRACE
231	N	Y	Y	PARALLEL GROUP TASK TIME TRACE
233	Y	Y	Y	START/END STORED PROCEDURE CALL

In order to assess the individual overhead associated with the different IFCIDs started by the Application Trace Facility, we need to associate the Performance class with the IFCIDs shown above. This information is contained in Table 10-2.

Table 10-2 Performance Class and associated IFCIDs

Class	Description	Associated IFCIDs
2	Subsystem-related events	68, 69, 70, 71, 72, 73, 74, 75, 84, 85, 86, 87, 88, 89, 106, 174, 175
3	SQL-related events	22, 53, 55, 58, 59, 60, 61, 62, 63, 64, 65, 66, 92, 95, 96, 97, 177, 233, 237, 272, 273
4	Buffer I/O and EDM pool requests	6, 7, 8, 9, 226, 227
6	Locking information	20, 44, 45, 213, 214, 218
8	Data manager detail	15, 16, 17, 18, 106, 125, 221, 222, 231, 305, 325
9	Sort detail	28, 95, 96
10	Autobind	105, 106, 107, 108, 109
13	Edit and validation exits	11, 12, 19
16	Distributed activity	157, 159, 160, 162, 163, 183
17	Drain and claim detail	213, 214, 215, 216
31	Installation-defined classes	188, 324

Finally, based on entries in the first 2 tables, we can then make a determination which of the Application Trace Facility options cause the most overhead. This information is contained in Table 10-3 Performance class cost.

Table 10-3 Performance class cost

Performance class	Events	Overhead
1	Background events	Low
2	Subsystem events	Low
3	SQL events	High
4	Buffer manager and EDM I/O	High
5	Log manager I/O	Medium
6	Locking summary	Medium
7	Locking detail	Very high
8	Data scanning detail	Very high
9	Sort detail	High
10	Utilities, binds, commands	Low
11	Execution unit switch and latch contention	High
12	Storage manager	Very high
13	Edit validation	Low
14	In and out of DB2	High
16	Remote location events	Medium
17	Claim and drain	High

Now, to tie this all together. We see that when we set the Lock Data and Scan Data flags to “Y”, this results in IFCID collections for 15,16,17 and,18. IFCIDs 15 - 18 are located in Performance Class 8, and are in the very high overhead trace category. By selectively setting the different trace collection flags, you can determine from the combination of Performance trace class and IFCIDs started, what level of overhead might be expected for each combination of flags.

We observed little difference in the level of detail, with the exception of some locking and scan metrics, between suppressing Lock Data and Scan Data and running with all collection flags turned on. However, when we turned off all of the collection flags, we saw little usable data collected in the trace. So our recommendation is to suppress Lock Data and Scan Data in the trace collection, unless you need the additional scan and lock information as this combination results in expensive IFCIDs being collected.

Most of the filters used in the Application Trace Facility are intended to control the amount of data that is placed into the trace dataset or data space. The effect of setting most of these filters is to control the speed at which the trace collection occurs, but has little or no impact on the trace overhead. Two exceptions to this are when the trace is qualified by either PLANAME or AUTHID. With these trace parameters, the generated DB2 START TRACE command can be qualified and a resulting improvement in trace performance can be achieved.

11

CICS and IMS Attach

The VTAM Classic Interface of OMEGAMON XE for DB2 PE provides information about active CICS and IMS threads and regions. It is able to display statistics for the active CICS and IMS attach facilities at region level and at individual thread level.

We discuss the following topics:

- What are the CICS and IMS attach facilities?
- System level CICS and IMS information
- CICS attach at thread level
- CICS RCT monitoring
- IMS attach at thread level
- IMS connection monitoring

11.1 What are the CICS and IMS attach facilities?

We provide here a brief description of the CICS and IMS attach facilities.

CICS attach facility

The CICS DB2 attachment facility, which is included with CICS, provides CICS applications with access to DB2 data while operating in the CICS environment. The CICS applications, therefore, can access both DB2 data and CICS data. CICS coordinates recovery of both DB2 and CICS data if transaction or system failure occurs.

The CICS DB2 attachment facility creates an overall connection between CICS and DB2. CICS applications use this connection to issue commands and requests to DB2. The connection between CICS and DB2 can be created or terminated at any time, and CICS and DB2 can be started and stopped independently. You can name an individual DB2 subsystem to which CICS connects, or (if you have DB2 Version 7 or later) you can use the group attach facility to let DB2 choose any active member of a data-sharing group of DB2 subsystems for the connection. You also have the option of CICS automatically connecting and reconnecting to DB2. A DB2 system can be shared by several CICS systems, but each CICS system can be connected to only one DB2 subsystem at a time.

Within the overall connection between CICS and DB2, each CICS transaction that accesses DB2 needs a thread, an individual connection into DB2. Threads are created when they are needed by transactions, at the point when the application issues its first SQL or command request.

The transaction uses the thread to access resources managed by DB2. When a thread is no longer needed by the transaction, because the transaction has accessed all the resources it needs to use, the thread is released (typically after syncpoint completion). It takes processor resources to create a thread, so when a thread is released, the CICS DB2 attachment facility checks to see if another transaction needs a thread. If another transaction is waiting for a thread, the CICS DB2 attachment facility reuses the existing thread for that transaction to access DB2. If the thread is no longer needed by any transaction, it is terminated, unless you have requested that it should be protected (kept) for a period of time. A protected thread is reused if another transaction requests it within that period of time; if not, it is terminated when the protection time expires.

In releases of CICS earlier than CICS Transaction Server for OS/390, Version 1 Release 2, the connection between CICS and DB2 was defined in the resource control table (RCT). The RCT described the relationship between CICS transactions and DB2 resources (application plans and command processors) and was generated using the DSNCRCT macro provided by the CICS DB2 attachment facility. Versions and releases of CICS from CICS Transaction Server for OS/390, Version 1 Release 3 onwards do not support running the CICS DB2 attachment facility using the macro DSNCRCT. If you are migrating from a CICS release that defined the CICS DB2 connection using a resource control table, you now need to define DB2 resource definitions using CICS resource definition online (RDO).

Despite this change in the way in which the CICS attach facility is defined, the terminology of the resource control table (RCT) is still widely used when discussing the CICS attach facility, including in OMEGAMON XE for DB2 PE, and will also be used here.

IMS attach facility

The IMS attach facility provides IMS applications with access to DB2 data while operating in the IMS environment. IMS applications, therefore, can access both DB2 data and IMS data. IMS coordinates recovery of both DB2 and IMS data if transaction or system failure occurs.The IMS attach facility used in the control region is also loaded into dependent regions. A connection is made from each dependent region to DB2. This connection is used to pass SQL statements and to coordinate the commitment of DB2 and IMS work.

The IMS attach facility gives you options to:

- Control the number of IMS regions connected to DB2. For IMS, this is also the maximum number of concurrent threads.
- Optimize the number of concurrent threads used by IMS.

A dependent region with a subsystem member (SSM) that is not empty is connected to DB2 at start up time. Regions with a null SSM cannot create a thread to DB2. A thread to DB2 is created at the first execution of an SQL statement in an IMS application schedule. It is terminated when the application terminates. The maximum number of concurrent threads used by IMS can be controlled by the number of IMS regions that can connect to DB2 by transaction class assignments.

The IMS attach facility can also provide efficient thread reuse for high volume transactions. Thread creation and termination is a significant cost in IMS transactions. IMS transactions identified as wait for input (WFI) can reuse threads: they create a thread at the first execution of an SQL statement and reuse it until the region is terminated. Some degree of thread reuse can also be achieved with IMS class scheduling, queuing, and a PROCLIM count greater than one. IMS Fast Path (IFP) dependent regions always reuse the DB2 thread.

11.2 System level CICS and IMS information

The VTAM Classic Interface provides important information at the system level for monitoring CICS and IMS connections to DB2.

CICS and IMS connection level information

Figure 11-1 shows the Summary of DB2 activity panel, which is reached by typing option letter S from the Main Menu.

```
>                           SUMMARY OF DB2 ACTIVITY
 DSYS
+  SSAS+DBAS+IRLM+DIST CPU  =    02.4%       Thread Commit Rate  =    77.0/sec
+  Create Thread Rate       =      .3/sec    Thread Signon Rate  =    25.0/sec
+  Synch Read I/O Rate      =  2068.0/sec    Prefetch Req Rate   =   955.7/sec
+  Update Request Rate      =  4317.0/sec    Write I/O Rate      =   139.3/sec
+  Getpages/Read I/O        =    14.71       Pages/Write I/O     =     2.18
+  Current Lock Suspensions =       0        Locking Timeouts    =    1533
+  Locking Deadlocks        =     255        Locking Escalations =     129
+
+  Connection Type     Connections  Threads   CPU    Getpage Rate  Elapsed Time
+  ------------------  -----------  -------   -----  ------------  ------------
+  IMS                      42         11     00.0%        .0/sec   00:25:43.4
+  CICS                      7         25     00.0%        .0/sec   02:06:41.8
```

Figure 11-1 Summary of DB2 Activity

Under the column heading Connection Type, this provides a one line system level summary of CICS and IMS connections giving the following information:

Connection Type A summary of connection information by DB2 connection type (here we are interested in CICS and IMS connection types only).

Connections The number of active connections originating from CICS and IMS regions in total (one connection per region).

Threads The total number of active threads originating from CICS and IMS.

CPU The total CPU rate (percent) attributable to the CICS and IMS connections.

Getpage Rate The total get page rate per second for active threads originating from CICS and IMS connections.

Elapsed Time The average elapsed time for active threads within CICS and IMS.

Rates on this panel are calculated on a cycle-to-cycle basis (from the last OMEGAMON display to this one).

System level information

If CICS or IMS is the major workload generator on the DB2 subsystem, then the system level fields at the top of the Summary of DB2 activity panel are useful indicators of CICS or IMS activity levels. The following information is available:

SSAS+DBAS+IRLM+DIST CPU The CPU rate (percent) used by the four main DB2 address spaces (MSTR, DBM1, IRLM and DIST). It includes both TCB and SRB time.

Thread Commit Rate The number of commits per second.

Create Thread Rate The number of create thread requests per second.

Thread Signon Rate The number of thread signon requests per second. Thread signon processing is applicable only in the CICS and IMS DB2 attachment environments. A comparison of Create thread rate with Thread signon rate gives an indication of how much thread reuse is taking place.

Synch Read I/O Rate The number of synchronous read I/Os per second.

Prefetch Req Rate The number of sequential and list prefetch request per second.

Update Request Rate The number of update requests per second. The update count is incremented each time a row in a page is updated. This update activity may be in the work database for sorting and joining activity, and does not necessarily represent SQL update activity.

Write I/O Rate The number of write I/Os per second.

Getpages/Read I/O The getpage to read I/O ratio. This value assists in measuring read and buffer pool efficiency. The value is computed by dividing the total number of getpage requests by the number of synchronous read I/O requests since the last OMEGAMON cycle.

Pages/Write I/O The average number of pages written per write I/O. This value is computed by dividing the number of pages written by the number of write I/Os since the last OMEGAMON cycle.

Current Lock Suspensions The current number of threads that are waiting due to a lock request issued for a resource that is unavailable.

Locking Timeouts The number of locking timeouts that have occurred since DB2 was started. Timeouts occur because lock requests were suspended for an amount of time in excess of the locking timeout value.

Locking Deadlocks The number of locking deadlocks that have occurred since DB2 was started. Deadlocks are a result of locking contention.

Lock Escalations The number of lock escalations that have occurred since DB2 was started. This count includes the number of escalations to both shared and exclusive modes.

11.3 CICS attach at thread level

From the Summary of DB2 activity panel shown in Figure 11-1 on page 181, you can reach the CICS thread summary panel by placing the cursor on the CICS connection type summary line and pressing PF11 (zoom).

Figure 11-2 shows the CICS thread summary panel. The top part of the panel uses the **CICA** command to provide a summary of CICS connection information. It displays one summary line for each CICS region with an active connection to the DB2 subsystem.

```
>        THREAD ACTIVITY:  Enter a selection letter on the top line.

> A-ALL   B-TSO   *-CICS   D-IMS       E-BACKGROUND  F-DIST ALLIED  G-DIST DBAC
> H-UTIL  I-INACT J-FILTER K-FUNCTIONS L-STORED PROC M-TRIGGERS     N-SYSPLEX
> O-ENCLAVES
===============================================================================
>                          CICS THREAD SUMMARY
 CICA
+                 Total    DB2    Pct. of   Active     Commit    RO Commit
+       Jobname    CPU     CPU    THRDMAX   Threads   Rate/sec   Rate/sec
+       -------   -----   -----   -------   -------   --------   ---------
+       CICSA01   00.0%   00.0%      .0%       0          .5          .5
+       CICSA02   00.0%   00.0%      .0%       0          .0          .0
+       CICSA03   00.0%   00.0%      .0%       0          .0          .0
+       CICSA04   00.0%   00.0%     1.1%       1          .0          .0
 THDC
+ *
+ Elapsed      Planname Tran CPU    Status       GetPg  Update  Commit Jobname
+ ----------   -------- ---- -----  ------------ ------ ------  ------ --------
+ 15:16:37.0   AX000    AXVG 00.0%  NOT-IN-DB2       18      4       0 CICSA01
+ 13:31:26.4   AX010    AXVG 00.0%  NOT-IN-DB2     7194   3710     445 CICSA04
```

Figure 11-2 CICS Thread Summary

The following details are provided for each active CICS connection:

Jobname The jobname of the connected CICS region.

Total CPU The total CPU rate (percent) of the CICS region connected to DB2. This includes both TCB and SRB time.

DB2 CPU The total CPU rate (percent) that is attributable to active threads originating from the CICS connection. This value is a subset of the total CICS region CPU utilization.

Pct. of THRDMAX The percentage of THRDMAX (CICS maximum threads) that the current threads have reached.

Active Threads The number of active threads originating from the CICS connection. This value includes outstanding create thread requests not yet satisfied by DB2. This value excludes threads originating from the CICS connection that are waiting for reuse.

Commit Rate The number of DB2 commits per second originating from the CICS connection.

RO Commit Rate The number of DB2 read-only commits per second originating from the CICS connection.

Following the summary of CICS connections at CICS region level shown in Figure 11-2, there is a summary of active CICS threads ordered by thread elapsed time. This is produced using the `THDC` command and shows the following information:

Elapsed The elapsed time since thread creation, or DB2 signon if the thread is reused and signon is driven. If signon is not driven (no change in authorization id) then the elapsed time may represent many transactions. Usually the commit count will give a good indication of how many transaction have been processed since the thread was created.

Planname The DB2 plan name of the active thread.

Tran The CICS transaction identifier active in the thread. In many environments this is a better identifier than the plan name, since it is not uncommon for one plan name to be used throughout an application.

CPU The CPU rate (percent) attributable to the thread.

Status The current DB2 status of the thread. Table 11-1 shows some typical status values and their meaning.

Getpg The number of thread getpage requests. This logical read request may not actually result in physical I/O if the page requested is currently in the buffer pool.

Update The number of DB2 page updates made by the thread. This update activity may be in the work database for sorting and joining activity, and does not necessarily represent SQL update activity.

Commit The number of times the thread successfully completed commit processing.

Jobname The jobname of the connected CICS region.

Figure 11-2 shows some typical values for thread status.

Table 11-1 Typical DB2 thread status values

Reported Status	Description
IN-ABORT	The thread is in abort processing.
IN-COMT-PHS1	The thread is in commit phase 1 processing.

Reported Status	Description
IN-COMT-PHS2	The thread is in commit phase 2 processing.
IN-CRTE-THRD	The thread is in create thread processing.
INDOUBT	The thread is in doubt.
IN-SQL-CALL	The thread is processing an SQL call.
IN-STOR-PROC	The thread is currently running in a stored procedure.
IN-TERM-THRD	The thread is in termination as a result of allied task termination. This status corresponds to the DB2 DISPLAY THREAD status of D.
NOT-IN-DB2	The thread is not currently executing in DB2.
WAIT-ASYNCRD	The thread is currently waiting for completion of a read I/O that is being done under a thread other than this one (such as sequential or list prefetch).
WAIT-GLBLOCK	The thread is currently waiting for: – either a lock held by another subsystem in the data sharing group – or inter-system communication within the data sharing group to determine if there is lock contention.
WAIT-LOCK	The thread is waiting for a local lock.
WAIT-REMREQ	The database access thread is in a synchronous wait (waiting for a response or a request from the originating DB2 subsystem).
WAIT-REUSE	The thread is not currently in use and is waiting to be reused. This status applies only to CICS and IMS threads.
WAIT-SERVICE	The thread is currently waiting for completion of a DB2 service. Types of DB2 services include open/close of dataset, DFHSM recall of a dataset, SYSLGRNG update, or define/extend/delete of dataset, commit phase 2 for read only threads.
WAIT-SYNC-I/O	The thread is currently waiting for completion of a synchronous read or write I/O.
WAIT-TERM-TH	The thread is queued and waiting for thread termination as a result of allied task termination. This status corresponds to the DB2 DISPLAY THREAD status of QD.

11.4 CICS RCT monitoring

The CICS RCT summary panel shown in Figure 11-3 can be reached by placing the cursor on the required CICS region jobname on the top part of the CICS thread summary panel, and pressing PF11 (zoom).

Following CICS connection level information at the top of the panel, each row of the display relates to an individual RCT entry. Each RCT entry usually relates to many DB2 threads. The columns provide information about:

- SQL call activity.
- Maximum threads allowed for an individual entry (THRDM).
- Maximum active threads (THRDA).
- The number of MVS TCBs to be attached at connection time (THRDS).
- Total TCB time of all currently active TCBs for this RCT entry.

To limit the number of displayed RCT entries, enter a full or generic value for `DB2ENTRY` and/or `PLANNAME`. By default, all RCT entries for the CICS region are displayed.

```
>                           CICS RCT SUMMARY
 RCTS
:                        CICS=CICSA01    (required)
+
+   DB2CONN Name              =    RCT01  Statistics Dest (SHDDEST) =     DB00
+   Error MSG Dest 1 (ERRDEST) =     CSMT  SNAP SYSOUT Class  (SNAP) =
+   Error MSG Dest 2           =           CICS Auth Name   (SIGNID) = CICSCA01
+   Error MSG Dest 3           =           THRD Purge Time Interval  =       30
+   Traceid 1        (TRACEID) =        0  Maximum Threads (THRDMAX) =       85
+   Traceid 2                  =        0  Active MVS Subtasks       =       32
+   Traceid 3                  =        0
+
:                    DB2ENTRY=D*          PLANNAME=________
+
+   DB2ENTRY     Calls     Waits    Aborts   THRDM   THRDA   THRDS  TCB Time
+   --------  --------  --------  --------  ------  ------  ------  --------
+   DB2001    90097044         0       101     N/A   50, 3   20, 6   300.329
+   DB2003        3407         0         0     N/A   10, 1    5, 1    23.174
+   DB2020           0         0         0     N/A   10, 0    5, 0     0.000
+   DB2021         127         0         2     N/A   10, 1    5, 2    67.288
```

Figure 11-3 CICS RCT Summary

At the top of the panel are fields relating to CICS connection level information:

CICS The CICS jobname. If you enter a different jobname, the corresponding RCT summary information will appear.

DB2CONN The RCT name in use by the CICS jobname.

Statistics Dest The transient data destination used for the shutdown statistics when the attachment facility is stopped.

Error MSG Dest 1 The identifier of the DFHDCT destination for unsolicited error messages. As many as three destination IDs can be assigned.

SNAP SYSOUT Class The MVS dump output class used to take a SNAP dump if a thread subtask fails.

CICS Auth Name The authorization ID used when the attachment connects to DB2.

THRD Purge Time Interval Specifies the length of the protected thread purge cycle.

Traceid 1 The CICS user trace identifier used by the attachment facility when it traces calls to DB2.

Maximum Threads The maximum number of threads that can be created between CICS and DB2 within this RCT.

Traceid 2 The dynamic plan entry trace ID.

Active MVS Subtasks The number of MVS TCBs that will be attached when the attach facility is started.

Traceid 3 The dynamic plan exit trace ID.

To see the full RCT definition for an individual DB2ENTRY, place the cursor on the required RCT row and press PF11 (zoom). If necessary you can restrict the number of rows displayed by giving a full or generic value (for example D*) for DB2ENTRY or PLANNAME. In Figure 11-4 you can see the CICS RCT detail panel that is displayed.

```
>                               CICS RCT DETAIL
 RCTD
+                                 CICS=CICSA01
+
+   Transaction ID      (TXID) =     AX0* , AX1*, AX2*
+   DB2ENTRY Name              =   DB2001  Plan Name          (PLAN) =    AX001
+   Thread TCB Prty   (DPMODE) =    Equal  Plan Alloc Exit (PLNEXIT) =
+   Rollbck on Deadlok (ROLBE) =      Yes  Thread Wait       (TWAIT) =     Pool
+   THRDM              (THRDM) =      N/A  THRDA             (THRDA) =    50, 2
+   THRDS              (THRDS) =    20, 5
+   Authorization Opt.  (AUTH) = (AUTH01,*,*)
+
+                            Statistics Information
+   Count of Calls             = 90405016  Current Trans Active      =        2
+   Count of Commits           =  1689896  Current TCBs Active       =        2
+   Count of Authorizations    =   470529  Current Total TCB Time    =  676.253
+   Count of Aborts            =      101  Current Waits/Ovflw Pool  =        0
+   Count of RO Commits        =   246986  Highwater Trans Active    =       18
+   Count of Waits/Ovflw Pool  =        0  Current Free TCBs         =        0
```

Figure 11-4 CICS RCT Detail

The fields in the top half of the panel define the parameters for the DB2ENTRY:

CICS The CICS jobname.

Transaction ID The transaction ids of all CICS transactions that are defined as belonging to this DB2ENTRY definition. The transactions can be generically defined (for example AX0*).

DB2ENTRY Name The unique name for the DB2ENTRY definition.

Plan Name The unique name of the plan assigned to the transactions belonging to this DB2ENTRY definition.

Thread TCB Prty The MVS dispatching priority of thread TCBs relative to the CICS main TCB. Possible values are high, equal and low. It is generally recommended to set this to equal or low.

Plan Alloc Exit The name of the exit program that can dynamically allocate the planname for the transaction id when the first SQL call is issued. Because of the widespread use of DB2 packages, this optional field is now rarely used.

Rollbck on Deadlok Determines whether or not CICS will issue a syncpoint rollback when the transaction experiences an IRLM-detected deadlock or timeout. For correct application behavior after deadlocks and timeouts, it is essential that this parameter is set properly- that is in agreement with the intended application behavior. The possible values are:

- **Yes** CICS will roll back all DB2 work and all CICS-protected resources to the last syncpoint.
- **No** CICS will roll back only the incomplete SQL call that was involved in the deadlock/timeout.

Thread Wait The action to take if no thread is available to execute a DB2-CICS transaction. The usual recommendation is to send threads to the pool rather than to wait or abend. The possible values are:

- **Pool** If all threads are in use, DB2 will place this transaction in the pool. If all the pool threads are in use, DB2 will place this transaction in a queue.
- **Yes** If all threads are in use, the transaction will wait until a thread is available.
- **No** If all threads are in use, the transaction will abend.

THRDM The maximum number of active threads that can originate from this DB2ENTRY definition simultaneously.

THRDA The maximum number of active threads for this DB2ENTRY definition.

THRDS The number of MVS TCBs that will be defined to the transactions originating from this DB2ENTRY definition. These represent the protected threads that remain on average 45 seconds after a transaction has completed and encourage thread reuse.

Authorization Opt The type of authorization id required to execute the DB2 plan for transactions originating from this DB2ENTRY definition.

Following the fields that define the DB2ENTRY are a number of statistical counters. You should note the following points when looking at these statistics:

- Current TCBs Active and Current Total TCB Time show values for those threads that currently have at least one TCB active.
- If THRDS is specified, the thread will not be terminated until an average of 45 seconds after a transaction has used it (protected entry).
- If the thread is non-protected, it terminates immediately after the end of the transaction, unless another transaction is queued for the thread (TWAIT=YES).
- Current Waits/Ovflw Pool and Count of Waits/Ovflw Pool display according to the value specified for Thread Wait (TWAIT).
- If Thread Wait = Pool then overflow into pool counts will be displayed.
- If Thread Wait = YES then the counts will reflect transactions waiting for a thread to be freed.

The statistical counters are defined as follows (the counts relate only to the specific DB2ENTRY definition that you select):

Count of Calls The number of DB2 calls generated by transactions since the CICS-DB2 attach was started.

Current Trans Active The number of transactions that are currently active in DB2 or are waiting for a thread (TWAIT=YES).

Count of Commits The number of commits completed since the attach facility was started.

Current TCBs Active The number of currently active threads.

Count of Authorizations The number of times authorization has been invoked for the threads defined to the transactions.

Current Total TCB Time The amount of CPU time in seconds that has been used by currently active threads.

Count of Aborts The number of aborts experienced by the transactions since the attach facility was started.

Current Waits/Ovflw Pool The number of transactions that are waiting for a thread, or that have overflowed into the pool.

Count of RO Commits The number of read-only commits since the attach facility was started.

Highwater Trans Active The largest number of concurrent transactions active since the attach facility was started.

Count of Waits/Ovflw Pool The number of times that all available threads for the entry were busy, and the transaction had to wait or be diverted to the pool (depending on the setting of Thread Wait).

11.5 IMS attach at thread level

From the Summary of DB2 activity panel shown in Figure 11-1 on page 181, you can reach the IMS thread summary panel by placing the cursor on the IMS connection type summary line and pressing PF11 (zoom).

Figure 11-5 shows the IMS thread summary panel. The top part of the panel uses the **IMSA** command to provide a summary of IMS connection information. It displays one summary line for each IMS subsystem (control region) with an active connection to the DB2 subsystem.

```
>         THREAD ACTIVITY:  Enter a selection letter on the top line.

> A-ALL   B-TSO   C-CICS   *-IMS       E-BACKGROUND  F-DIST ALLIED  G-DIST DBAC
> H-UTIL  I-INACT J-FILTER K-FUNCTIONS L-STORED PROC M-TRIGGERS     N-SYSPLEX
> O-ENCLAVES
===============================================================================
>                               IMS THREAD SUMMARY
 IMSA
+         IMS ID           CPU%          Connection          Threads
+       --------          -----          ----------          -------
+       IMS1               00.6                  43               13
 THDI
+ *
+ Elapsed      Planname Jobid    CPU% Status       GetPg Update Commit Connid
+ ----------   -------- -------- ---- ------------ ----- ------ ------ --------
+ 00:39:20.8   PL001    IMS1#020 00.0 NOT-IN-DB2       4      0      1 IMS1
+ 00:37:05.9   PL001    IMS1#021 00.0 NOT-IN-DB2       4      0      1 IMS1
+ 00:21:34.6   PL001    IMS1#022 00.0 NOT-IN-DB2       4      0      1 IMS1
```

Figure 11-5 IMS Thread Summary

Following the summary of IMS connections at IMS subsystem level in Figure 11-5 is a summary of active IMS threads ordered by thread elapsed time, produced using the **THDI** command. This display of active IMS threads differs very little from the standard active thread display in terms of the fields available.

11.6 IMS connection monitoring

The IMS connection detail panel shown in Figure 11-6 can be reached by placing the cursor on the required IMS subsystem name on the top part of the IMS thread summary panel, and pressing PF11 (zoom).

```
>                    IMS CONNECTION DETAIL INFORMATION

 ISSM
+
+  Region   Type  Crc  Lit   Rtt       Module    Err  Psb       Status
+ --------  ----  ---  ----  --------  --------  ---  --------  -----------
+ IMS1CTL   CNTL   ?   SYS1  None      DSNMIN10   R     N/A     NOT-IN-DB2
+ IMS1#041  MPP    -   SYS1  N3325     DSNMIN10   R   N3325     NOT-IN-DB2
+ IMS1#032  MPP    -   SYS1  PA020     DSNMIN10   R   PA020     NOT-IN-DB2
+ IMS1#030  MPP    -   SYS1  PA020     DSNMIN10   R   PA020     NOT-IN-DB2
+ IMS1#007  MPP    -   SYS1  MM3100    DSNMIN10   R   MM3100    NOT-IN-DB2
+ IMS1#017  MPP    -   SYS1  N3140     DSNMIN10   R   N3140     NOT-IN-DB2
+ IMS1#012  MPP    -   SYS1  R0301     DSNMIN10   R   R0301     NOT-IN-DB2
+ IMS1#016  MPP    -   SYS1  N3140     DSNMIN10   R   N3140     NOT-IN-DB2
+ IMS1#002  MPP    -   SYS1  MM3110    DSNMIN10   R   MM3110    NOT-IN-DB2
+ IMS1#033  MPP    -   SYS1  PA020     DSNMIN10   R   PA020     NOT-IN-DB2
+ IMS1#008  MPP    -   SYS1  MM3100    DSNMIN10   R   MM3100    NOT-IN-DB2
+ IMS1#031  MPP    -   SYS1  PA020     DSNMIN10   R   PA020     NOT-IN-DB2
+ E001X01   BMP    -   SYS1  E001      DSNMIN10   R   E001      NOT-IN-DB2
```

Figure 11-6 IMS Connection Detail

The IMS connection detail panel displays the IMS control region for the selected IMS subsystem, all dependent regions connected to DB2, and connection related information. The information on this panel allows you to monitor all the IMS connection activities.

This information shown is:

- The IMS region name followed by region type
- The command recognition character
- The language token interface
- The resource translation table
- The interface control module
- The error option specification

Additionally, for dependent regions, it shows the active application name (PSB) and the status of the dependent region. The status is blank if the region is not active in DB2.

Part 4

What’s new for OMEGAMON users

In this part of the book, we describe some of the most important functions of DB2 PE, now included in the converged Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS, which could be of great interest for a former OMEGAMON for z/OS user.

This part of the book contains the following chapters:

Please refer to *IBM DB2 Performance Expert for z/OS Version 2,* SG24-6867-01, Chapter 4, “Snapshot monitoring,” and Chapter 7, “Optimizing buffer pool resources,” respectively, for information on two other important functions:

- Snapshot history
- Buffer Pool Analysis

12

DB2 Connect monitoring

In this chapter, we describe the functionality of OMEGAMON XE for DB2 PE for monitoring DB2 Connect servers and DCS applications that connect through DB2 Connect servers to DB2 subsystems. You can use the information that you obtain through these monitoring activities to evaluate the health of a DB2 Connect server and to make changes necessary to tune the system.

If you have used DB2 Performance Monitor or DB2 Performance Expert, you know about this functionality and the user-interface support provided in the Performance Expert client. In the converged product, DB2 Connect monitoring is also possible from the VTAM and Tivoli Enterprise Portal (TEP) user interfaces.

We discuss the following topics:

- How DB2 Connect works
- How to set up Performance Expert agents and OMEGAMON Server for collecting and storing DB2 Connect performance data
- How to view the collected data from various perspectives in the VTAM and TEP user interfaces, and how to identify gateway problems
- How DB2 Connect monitoring works in a data-sharing environment

Note: Special thanks go to Holger Karn for providing valuable input for this chapter.

12.1 DB2 Connect servers

DB2 Connect provides fast and robust connectivity to mainframe databases for e-business and other applications running under UNIX® and Windows operating systems.

DB2 Connect has several connection solutions. DB2 Connect Personal Edition provides direct connectivity to host database servers, while DB2 Connect Enterprise Edition provides indirect connectivity that allows clients to access host database servers through the DB2 Connect server.

12.1.1 Overview

A DB2 Connect server enables multiple clients to connect to host data and can significantly reduce the effort that is required to establish and maintain access to enterprise data. Figure 12-1illustrates an environment in which DB2 clients and applications make an indirect connection to a host database server through a DB2 Connect server.

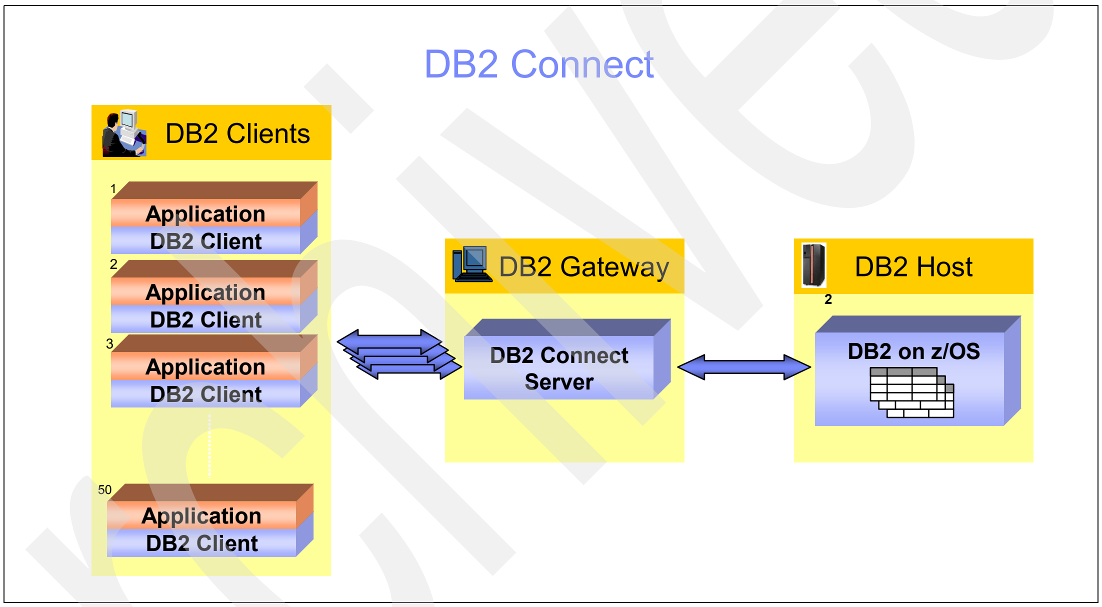


Figure 12-1 Connecting applications to a DB2 host via DB2 Connect server

DB2 Connect supports various distributed application scenarios and provides a highly scalable communication infrastructure. For more information on how you can make host data directly available to your Personal Computer or LAN-based workstations with DB2 Connect see *DB2 Connect User's Guide Version 8,* SC09-4835.

12.1.2 Connection pooling and connection concentrator

Connection pooling and *connection concentrator* are two important features of a DB2 Connect server.

In a distributed application employing DB2 Connect gateways, a client request for DB2 data is passed to the gateway and handled by so-called agents. On behalf of the client, a DB2 Connect agent connects to the DB2 server at the host and a thread is created in DB2 for fulfilling the request.

Connection pooling is a technique that allows the reuse of an established connection infrastructure for subsequent connections. Connection concentrator is a technique that allows the sharing of an agent that owns a connection and a DB2 thread across thousands of simultaneously executing transactions.

Figure 12-2 illustrates how connection pooling and connection concentrator work together and which database parameters can be used in that context.

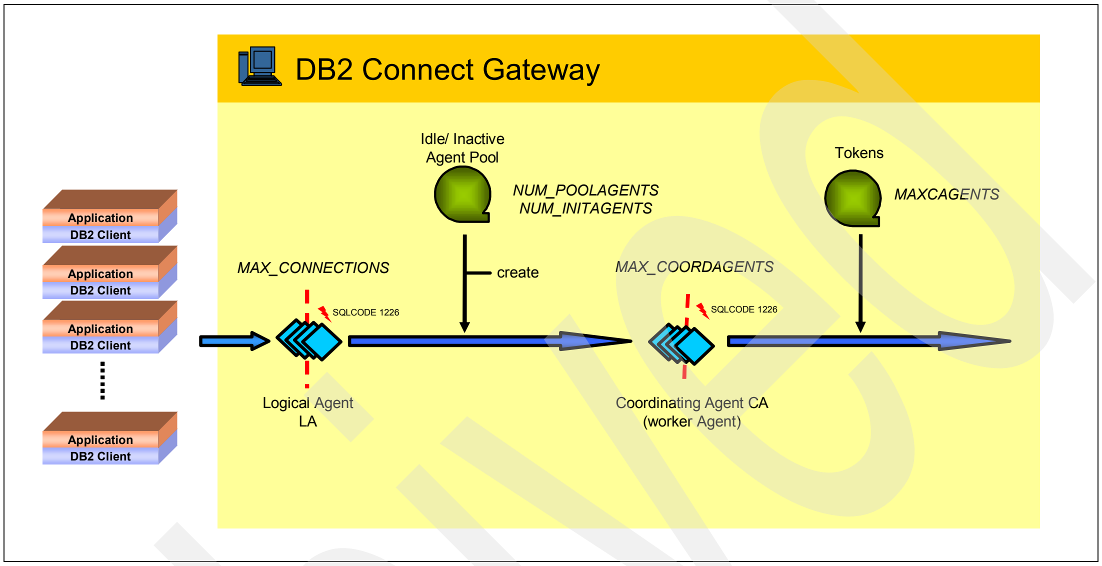


Figure 12-2 Connection pooling and connection concentrator

The connection concentrator splits the agent into two entities, a logical agent (LA) and a worker or coordinating agent (CA). Logical agents represent an application; coordinating agents are physical entities that own a connection and a DB2 thread for executing application requests, but which have no permanent attachment to any given application. Worker agents associate with logical agents to perform transactions. At the transaction boundary (COMMIT/ROLLBACK), the worker agents end the association and return to the pool of inactive agents where they keep their connection and a DB2 thread. This pool also contains idle agents, which are initialized by DB2 Connect and do not own a connection.

Agents are also put back into the pool, when the application issues a disconnect request. When a client requests a connection, it can be provided from this pool of ready connections. This is the concept of connection pooling, which significantly reduces the overhead typically spent on opening and closing these connections. Connection pooling saves the cost of establishing a connection when one is no longer needed by a terminating application.

Connection concentrator, on the other hand, allows DB2 Connect to make a connection available to an application as soon as another application has finished a transaction and does not require that other application to disconnect.

There are several database manager parameters for the DB2 Connect Server that you can configure for controlling and tuning connection pooling and connection concentrator:

NUM_POOLAGENTS This parameter denotes the maximum number of idle and inactive agents you wish the system to maintain in the pool.

NUM_INITAGENTS This parameter denotes the number of idle agents DB2 Connect creates at start up time before any clients make a request. These agents do not have a connection to a host database server but are used to avoid overhead of creating new agents.

MAX_CONNECTIONS This parameter sets the maximum number of logical agents and limits the number of incoming DB2 client applications.

MAX_COORDAGENTS This parameter defines the maximum number of coordinator agents. Connection concentrator is active if MAX_CONNECTIONS is higher than MAX_COORDAGENTS. Connection pooling is always active since DB2 Connect V6 and cannot be deactivated.

MAXAGENTS This parameter sets the maximum number of agents that can exist in the system.

MAXCAGENTS This parameter sets the maximum number of coordinator agents that can run concurrently.

If you want to see how your DB2 Connect server is configured (see Figure 12-3), attach to the DB2 Connect instance and enter the DB2 command, **`GET DBM CONFIG`**.

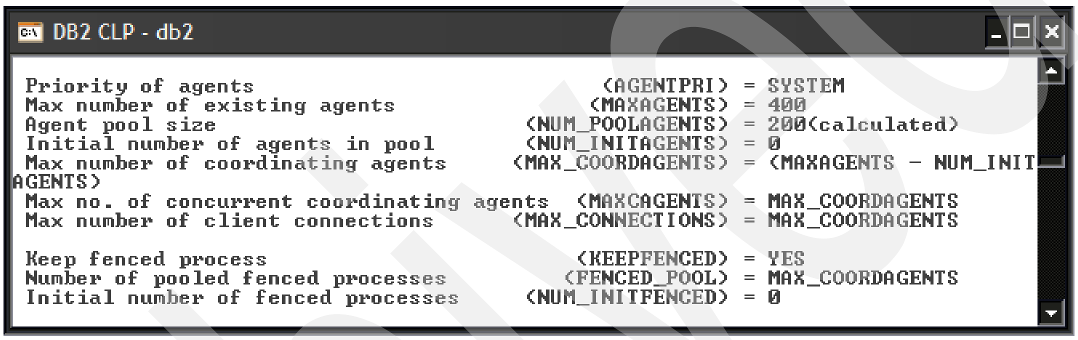


Figure 12-3 D2 Connect parameters

Figure 12-4 shows how the parameters are numerically related to each other.

```
NUM_POOLAGENTS  +  MAX_COORDAGENTS  <=  MAXAGENTS  <=  MAX_CONNECTIONS
          =                    =
          V                    V
 NUM_INITAGENTS           MAXCAGENTS
```

Figure 12-4 Relationship of DBM parameters

These relations (together with Figure 12-2) are intended to help you to understand how DB2 Connect assigns agents to incoming client requests, if connection concentrator is activated:

1. If the maximum number of incoming connections (MAX_CONNECTIONS) is reached, then it returns with an SQL error.
2. If a worker agent is available from the pool and if MAX_COORDAGENTS is not exceeded, then it assigns the worker agent to the incoming request. The agent will perform its task if the maximum number of concurrently running worker agents MAXCAGENTS is not reached. The value shown in Figure 12-11 for label *From Pool* represents the number of connections serviced without the overhead of creating a new server agent task.
3. If number of worker agents is below MAX_COORDAGENTS and if no worker agent is available from the pool, then it creates a new agent. The value *Create Empty Pool* is the number of agents created because there were no agent that could be re-assigned in the pool.

12.2 Collect and store DB2 Connect performance data

To monitor the activities of a DB2 Connect server and the connections of distributed applications, you must collect and store DB2 Connect performance data that can be viewed from a user interface.

Figure 12-5 shows the components for collecting and storing DB2 Connect data:

- Performance Expert Agent for DB2 Connect Monitoring (PE Agent)
- Performance Warehouse (PWH) tables used for DB2 Connect monitoring
- OMEGAMON Server with history datasets used for DB2 Connect monitoring
- Interfaces for the configuration of PE Agent and OMEGAMON Server.

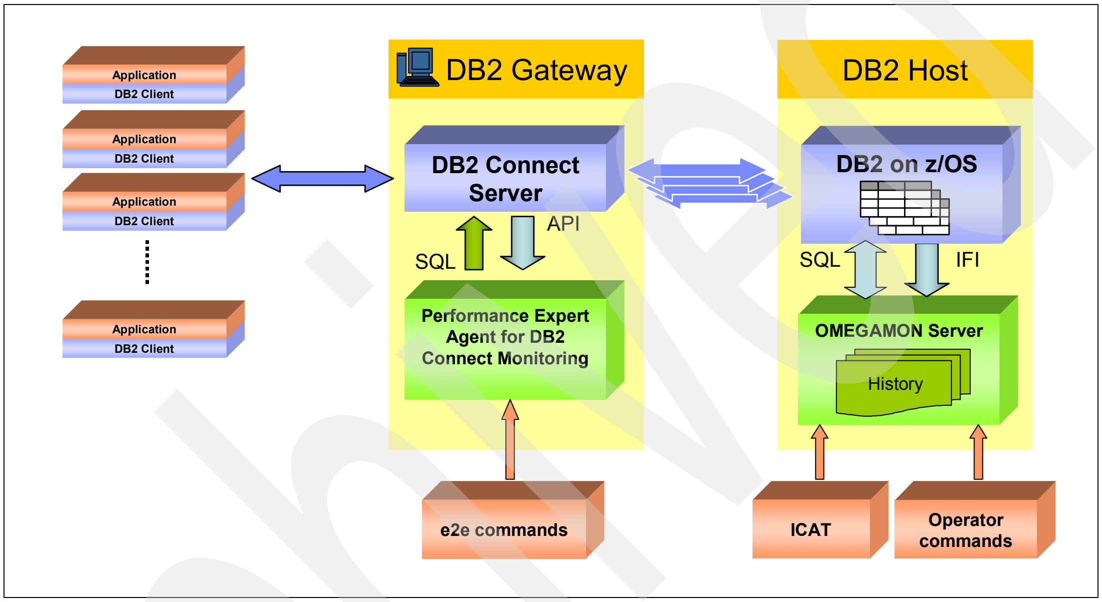


Figure 12-5 DB2 Connect monitoring components

The Performance Expert Agent periodically collects snapshots of performance-related information about the DB2 Connect gateway and the connections of distributed applications and stores it in the Performance Warehouse of a dedicated DB2 subsystem using a JDBC™ connection. While it periodically monitors and refreshes the data on the host, the PE Server subtask in the OMEGAMON Server retrieves the data from the database tables, correlates the data with other DB2 performance data, stores it in its history datasets, and makes it available to the user interfaces.

12.2.1 Performance Expert Agent for DB2 Connect Monitoring

You can install and run the Performance Expert Agent for DB2 Connect Monitoring (PE Agent) on a Windows, UNIX, or Linux® system where the DB2 Connect server resides. The agents are distributed with the product installation libraries.

For example, the binary files for the PE Agents on Windows systems are contained in the following datasets, where *hlq* denotes the high-level qualifier of your SMP/E target libraries:

- 'hlq.TKO2WS01(FPEKAGNT)'

  This dataset contains the program files for Performance Expert Agent for DB2 Connect Monitoring.

- 'hlq.TKO2WS01(FPEKWN32)'

  This dataset contains the win32.jar JRE file that is required for Performance Expert Agent for DB2 Connect Monitoring.

For details on how to download, install, and configure the agents, see *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Configuration and Customization,* GC18-9637.

A Performance Expert Agent stores the collected data in Performance Warehouse tables of a DB2 subsystem that is monitored by an OMEGAMON Server and an associated PE Server subtask. You must register the OMEGAMON Server (that is, the PE Server instance within the OMEGAMON Server and the related DB2 subsystem) to the PE Agent, such that the agent knows where to store the DB2 Connect trace data.

The PE Agent comes with a command line interface called *e2e command interface*, which you can use to register, maintain and deregister PE Server instances and the associated DB2 subsystem. Each command must be preceded by a -- prefix. The **addhost** command for registering a server requires the host name, port number and location name of the DB2 subsystem, the user ID and password to access the Performance Warehouse such as:

```
e2e --addhost 9.12.6.9 38050 DB8F Y paolor2 xxxxxxxx
```

User ID and password are encrypted in the e2e.ini file of your agent configuration directory.

Figure 12-6 lists the available commands in the e2e interface.

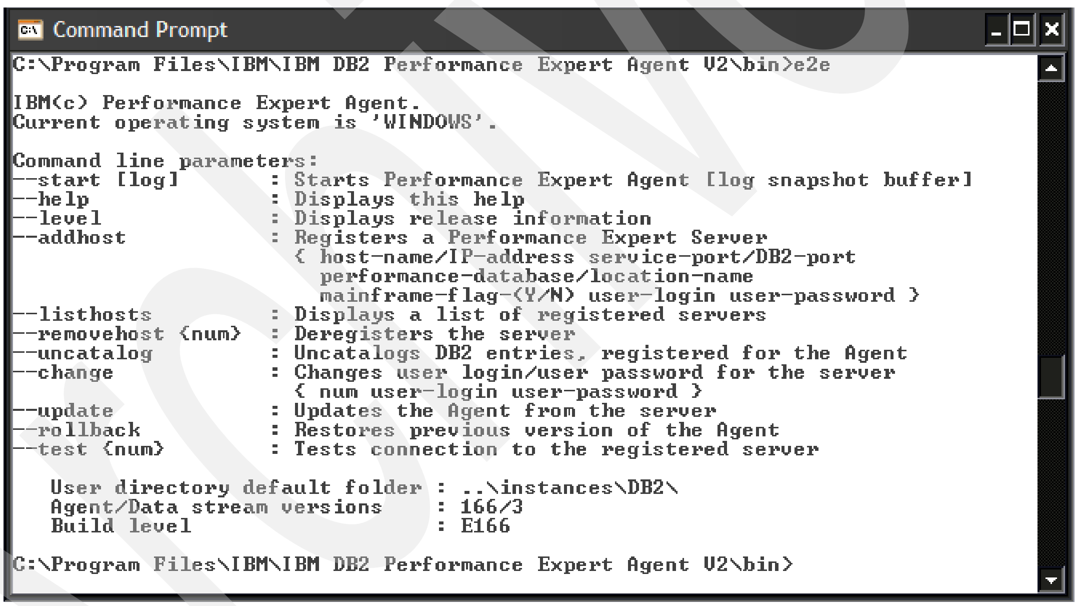


Figure 12-6 e2e command line interface

When an agent starts up, it uses JDBC to update the Performance Warehouse with information about its status.

As part of each measurement cycle, the PE Agent collects the snapshot of all distributed applications that are currently active, the snapshot of the database, the snapshot of the processes that are running in the DB2 Connect server, their CPU time and working set, and updates the PWH with this information.

During each measurement cycle, a sample SQL statement is sent from the PE Agent to the host and its execution timings are traced and computed. This enables the PE Agent to determine the time spent in network and the time spent in CLI. These computed values are also stored in the database.

Notes:

- Do not mix up Performance Expert Agents for DB2 Connect Monitoring and Tivoli Enterprise Monitoring Agents.
- A graphical user interface for submitting e2e commands will be provided by the PTF for APAR PK18535.

12.2.2 OMEGAMON Server

The DB2 Connect trace data available in the Performance Warehouse can be retrieved and stored in history datasets by the OMEGAMON Server. On request, the OMEGAMON Server provides this data to the user interfaces.

To enable the OMEGAMON Server for these tasks, the snapshot history for the DB2 subsystem (containing the Performance Warehouse with DB2 Connect data) must be configured appropriately.

This can be done in the ICAT configuration (see bold parameters) as shown in Figure 12-7.

```
KD260SH OMEGAMON XE for DB2 PE Snapshot History configuration (1) SSID: D871
Command ===>

  Specify the snapshot history collection:

  Keep snapshot history          ==> Y     (Y, N)
  Snapshot history archive size  ==> 16___ (>4MB)

  Specify the snapshot history collection:
                                               Interval
      Statistics               ==> Y (Y, N)      120__ (1-86400 in seconds)
      Dynamic Statement Cache  ==> Y (Y, N)      300__ (1-86400 in seconds)
      Data Set Statistics      ==> Y (Y, N)      300__ (1-86400 in seconds)
      System Parameters        ==> Y (Y, N)      300__ (1-86400 in seconds)
      Thread                   ==> Y (Y, N)      60___ (1-86400 in seconds)
          Include Locking      ==> N (Y, N)
          Include Stmt Text    ==> N (Y, N)
      DB2 Connect Application  ==> Y (Y, N)      60___ (1-86400 in seconds)
      DB2 Connect System       ==> Y (Y, N)      120__ (1-86400 in seconds)
Enter=Next  F1=Help  F3=Back
```

Figure 12-7 Snapshot History configuration

Alternatively, operator commands can be issued to the OMEGAMON Server and its PE Server subtasks:

F cccccccc,F PESERVER,F db2ssid,option

Where:

- cccccccc is the started task name that you specify for the OMEGAMON Server during the configuration within ICAT
- db2ssid is the identifier of the DB2 subsystem on which the PE Server subtask runs
- option is the subcommand for the PE Server subtask.

Subcommands for the PE Server subtask enabling DB2 Connect Monitoring are:

- SNAPSHOTHISTORY=Y

  Activates snapshot history processing.

- SHDB2CONNECTAPPLICATION=(Y,interval)

  Specifies that DB2 Connect application data is collected if the Performance Expert Agent for DB2 Connect Monitoring is installed, where interval is the time interval between two consecutive snapshots in seconds.

  Recommendation: For correlation reasons, the interval value should be identical to the interval value of the subcommand SHTHREAD.

- SHDB2CONNECTSYSTEM=(Y,interval)

  Specifies that DB2 Connect system data is collected if Performance Expert Agent for DB2 Connect Monitoring is installed, where interval is the time interval between two consecutive snapshots in seconds.

  Recommendation: For correlation reasons, the interval value should be identical to the interval value of the subcommand SHSTATISTICS.

The DB2 Connect snapshot history parameters are stored internally in the PE Server instance and are also transferred to the PWH table HISTORYDATA (see rows with keyword DB2C_SYSTEM and DB2C_APPLICATION). This data is used to synchronize performance data collection of the PE Agent with the history collection of the OMEGAMON Server.

One Performance Expert Agent for DB2 Connect Monitoring can send data to one or more OMEGAMON Servers, and one OMEGAMON Server can receive data from one or more Performance Expert Agents for DB2 Connect Monitoring. Therefore, you should plan in your configuration which OMEGAMON Server is responsible for which DB2 Connect Server (see also “DB2 Connect Monitoring in a data-sharing environment” on page 220).

12.3 Monitor DB2 Connect servers

To monitor the activities of a DB2 Connect gateway and the connections of DCS applications, your system must meet the following criteria:

- Performance Expert Agent for DB2 Connect Monitoring is installed on the system on which the DB2 Connect gateway runs, and is configured.
- At least one OMEGAMON Server is registered for a Performance Expert Agent for DB2 Connect Monitoring.
- Performance Expert Agent for DB2 Connect Monitoring and OMEGAMON Server are started. According to the configuration parameters, agent and server collect and store DB2 Connect statistics and applications trace data in the Performance Warehouse tables and the history datasets at specified time intervals.

> **Note:** The PE Agent activities can be tracked in the e2elog.instance file in the agent configuration directory.

If you have used DB2 Performance Monitor or DB2 Performance Expert, you know about the user-interface support provided in the Performance Expert client. For more details, see:

- Chapter 8 of *IBM DB2 Performance Expert for z/OS Version 2,* SG24-6867-01
- *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Monitoring Performance from the Performance Expert Client,* SC18-9640.

In the converged product, the performance of a DB2 Connect gateway can be also monitored from the VTAM interface and the Tivoli Enterprise Portal (TEP). The interfaces enable you to view the collected data from various perspectives and helps identify bottlenecks in the gateway (Figure 12-8). The next sections explain the DB2 Connect monitoring functionality that is now available in VTAM and TEP.

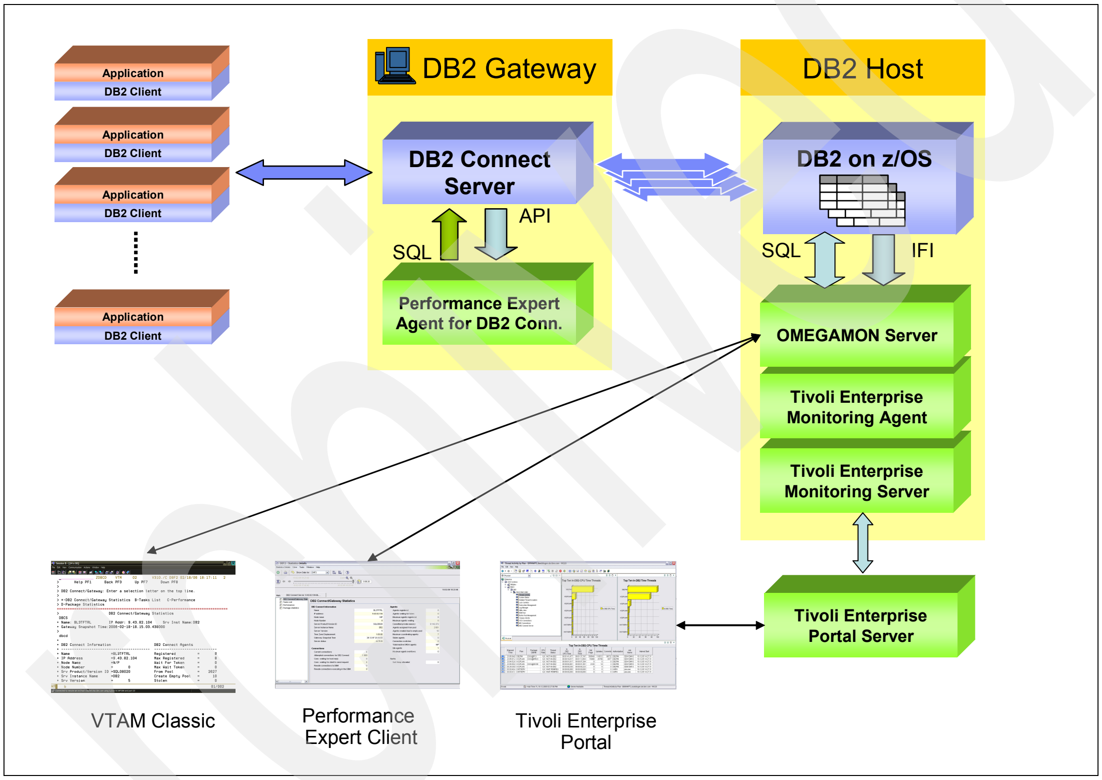


Figure 12-8 User interfaces for DB2 Connect monitoring

12.3.1 Monitoring gateway statistics

The performance of a DB2 Connect gateway can be monitored on various panels, where you can find:

- Statistics about the DB2 Connect gateway, including details about the number of agents and pooled agents, the connections waiting for the host to reply, and the connections waiting for the client to send a request.
- Statistics about the processes at the selected DB2 Connect gateway, such as the CPU and the working set.
- The amount of time that is required to process a sample SQL statement.
- Statistical details about the packages received at the DB2 Connect gateway.

VTAM

To view all DB2 Connect servers that serve as DB2 Connect gateways and are registered to the DB2 subsystem which is currently monitored by the VTAM interface, select the option **G DB2 CONNECT SERVER** in the VTAM main menu, and panel ZDB2C appears as shown in Figure 12-9.

```
 ________________ ZDB2C    VTM     O2       V310./C SDH1 02/17/06  1:09:25   2
>       Help PF1      Back PF3      Up PF7      Down PF8       Zoom PF11
> G.
===============================================================================
>                            DB2 CONNECT SERVER

 DB2C
+ Name            IP Address      Node Name  Status          Server Name
+ --------------- --------------- ---------- --------------- ---------------
+ BL3TFTRL        9.43.82.104     N/P        ACTIVE          DB2
===============================================================================
```

Figure 12-9 DB2 Connect server BL3TFTRL registered to subsystem SDH1

You can zoom into the listed servers to get more detail statistics information. The panels have a header as shown in Figure 12-10, which displays the navigation options A, B, C, D. The panel, which is currently selected, is marked with a '*' and an appropriate heading is displayed.

```
 ________________ ZDBCD    VTM     O2       V310./C SDH1 02/17/06  1:10:14
>       Help PF1      Back PF3      Up PF7      Down PF8
>
> DB2 Connect/Gateway: Enter a selection letter on the top line.
>
> *-DB2 Connect/Gateway Statistics  B-Tasks List   C-Performance
> D-Package Statistics
===========================================================================
>                        DB2 Connect/Gateway Statistics
 DBCS
+ Name: BL3TFTRL        IP Addr: 9.43.82.104     Srv Inst Name:DB2
+ Gateway Snapshot Time:2006-02-17-01.08.10.650000
```

Figure 12-10 Common header for DB2 Connect gateway statistics panels

The DB2 Connect/Gateway Statistics panel (Figure 12-11) shows important parameters that are related to connection pooling and connection concentrator.

```
 + DB2 Connect Information                    DB2 Connect Agents
+ ----------------------------------------  -----------------------------
+ Name                   =BL3TFTRL          Registered          =       3
+ IP Address             =9.43.82.104       Max Registered      =       3
+ Node Name              =N/P               Wait For Token      =       0
+ Node Number            =       0          Max Wait Token      =       0
+ Srv Product/Version ID =SQL08020          From Pool           =     535
+ Srv Instance Name      =DB2               Create Empty Pool   =       5
+ Srv Version            =       5          Stolen              =       0
+ Time Zone Displacement = 1h 0mn           Idle                =       0
+ Server Status          =ACTIVE            Max Coordinating    =       3
+                                           Max Overflow        =       0
+                                           Inactive DRDA       =N/P
+                                           Connection Switch   =       0
+                                           Private Memory      = 8142848
+
+ Connections                               Sorts
+ --------------------------------------    -----------------------------
+ Current                       =       0   Sort Heap Allocated =       0
+ Attempted                     =     471
+ Wait for Host Reply           =       0
+ Wait Client Send Request      =       0
+ Remote Connection             =       0
+ Remote Conn Executing in DBM  =       0
```

Figure 12-11 DB2 Connect/Gateway Statistics

Option B, DB2 Connect/Gateway Task List, shows the information about the tasks running at the selected DB2 Connect gateway such as the CPU and the working set. Normally, the percentage of CPU utilized by the DB2 Connect agents will be low. In UNIX operating systems, the agent is a UNIX process while in Windows operating system, an agent is a thread.

Occasionally there may be spikes in the CPU utilization (especially the percentage of kernel CPU utilization), occurring mainly due to the creation of an agent task. Hence it is important to size the agent pool size correctly. Generally, the total CPU utilization for DB2 Connect agents should be less than 25%. Sometimes certain busy applications could consume more processor resources. In general you should review the configuration if the DB2 Connect CPU utilization is consistently higher than 50%.

An important feature is the representation of the time spent in DB2 Connect, time spent in the host, and the time spent in the network. For those purposes, a sample SQL is sent to the host and the various events are timed. This is a very useful metric to know because in a peak load situation you can determine where exactly the bottleneck originates. Take a look at Figure 12-12.

```
>                     DB2 Connect/Gateway Performance
 DBCS
+ Name: BL3TFTRL        IP Addr: 9.43.82.104     Srv Inst Name:DB2
+ Gateway Snapshot Time:2006-02-17-01.08.10.650000
 dbcp
+
+ Times for Sample SQL Statement
+ --------------------------------------------
+ Total Statement Time        =00:00:00.176585
+ Time in DB2 Connect         =00:00:00.000364
+ Time on DB2 Host            =00:00:00.001209
+ Time in Network Connection  =00:00:00.175012
```

Figure 12-12 DB2 Connect/Gateway performance for sample SQL statement

In this case, the time spent in the network is almost 99%. During a peak load situation you can monitor the network effectiveness and throughput based on the details you see here. In a well configured DB2 host and the DB2 Connect gateway, a major component of the elapsed time will be the time spent in network. The overall performance of the application is likely to deteriorate due to a poor network topology, especially if the application involves frequent repeated fetches over the network. Such applications are likely to benefit by deploying stored procedures.

When you suspect a network performance such as this, the first thing to do is to ping the host from the DB2 Connect gateway server, as shown in Figure 12-13.

Figure 12-13 Network response from DB2 Connect gateway to the host

The **ping** command shows that the time taken for a 1500 byte packet from the DB2 Connect gateway server to the host takes about 180 milliseconds. For this example covering the VTAM interface, the DB2 host server is located 6000 miles away from the DB2 Connect gateway server in the residency lab. In a real life situation, such deployment needs immediate attention. Ideally the DB2 Connect gateway should be in proximity to the host and they should be architected in such a way that the network response time is less than a millisecond. An occasional worst case response should not be more than three milliseconds. If it is greater than this, then you need to review the network topology and parameters.

You can also use the **ping** command from the DB2 command prompt as shown in Figure 12-14. You have to connect to the database and then issue the **ping** command.

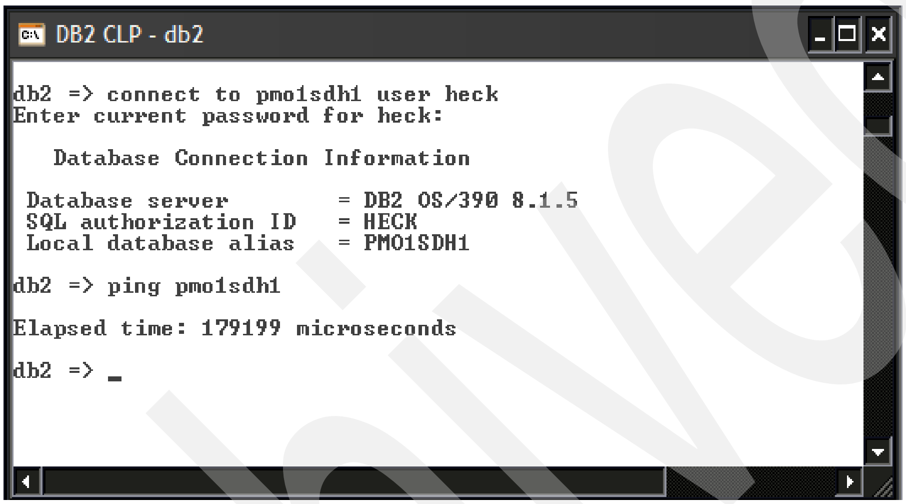


Figure 12-14 DB2 Connect gateway server - Database ping

Among other things, the DBM configuration parameters such as RQRIOBLK, query heap size, and TCP window sizes can affect the network performance and throughput.

Statistical details about the packages received at the DB2 Connect gateway can be viewed on the DB2 Connect/Gateway Package Statistics panel.

Tivoli Enterprise Portal

DB2 Connect server statistics can also be monitored from the Tivoli Enterprise Portal (TEP). The TEP workspaces contain data comparable to the VTAM panels.

The following screenshots are all taken with respect to monitored DB2 systems in Poughkeepsie, so network times are different from Boeblingen network times in “VTAM” on page 209.

Figure 12-15 shows the node, DB2 Connect Server, related to a DB2 subsystem in the tree of monitored DB2 subsystems and the workspace that you see when clicking on that node.

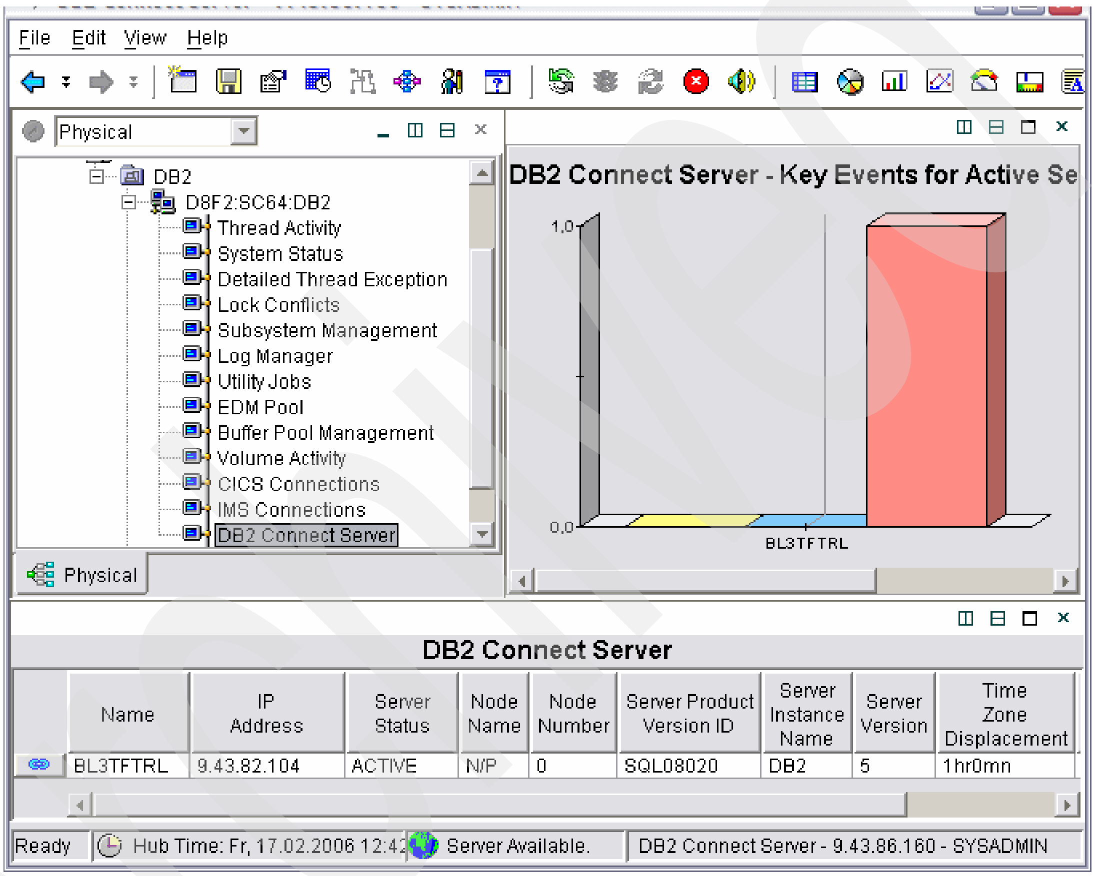


Figure 12-15 DB2 Connect server registered with D8F2

Right-click a listed server to get a menu of available perspectives of the gateway performance data:

- DB2 Connect/Gateway Statistics
- Tasks List
- Performance
- Package Statistics

Left-clicking brings you to the first menu item in the menu list. Select **Gateway Statistics** and the panel shown in Figure 12-16 appears.

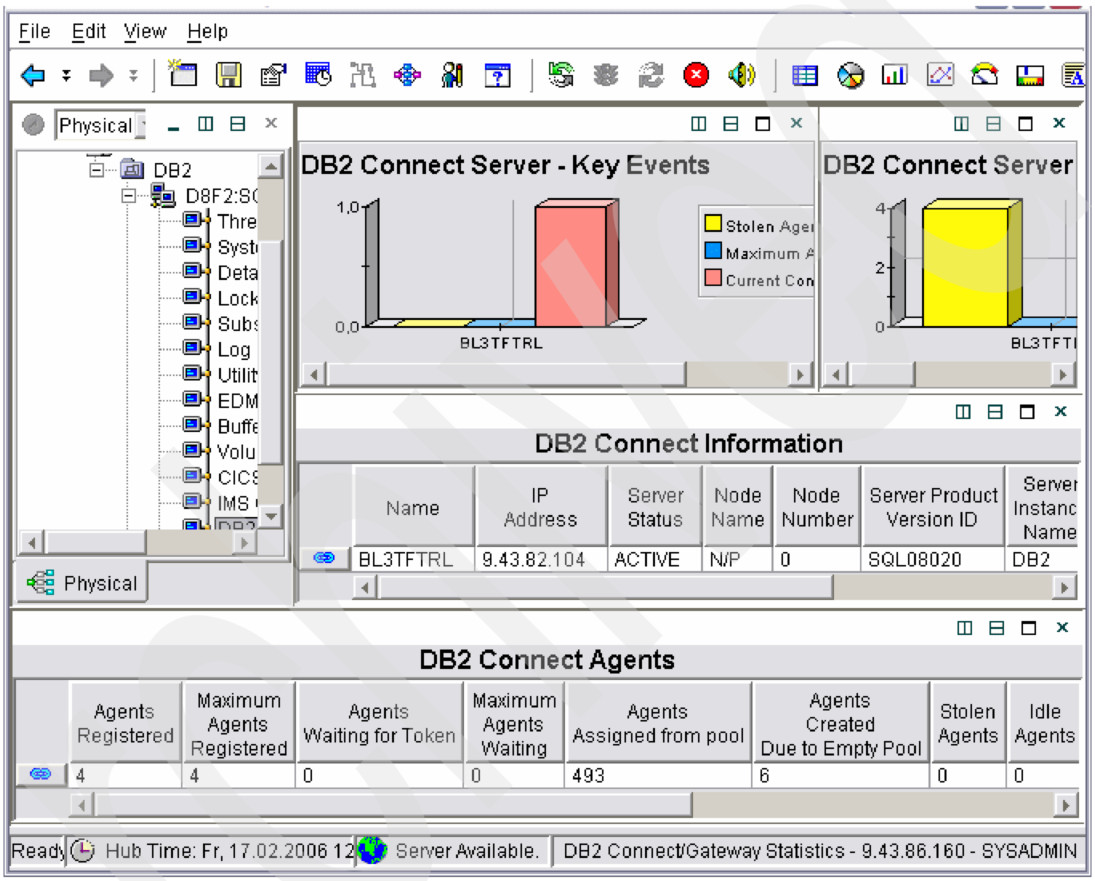


Figure 12-16 DB2 Connect/Gateway statistics

Right-click one of the chain symbols and navigate within the perspectives. See the window for the sample SQL statement execution in Figure 12-17. Left-clicking brings you back to the entry window listing the available DB2 Connect servers (DCS).

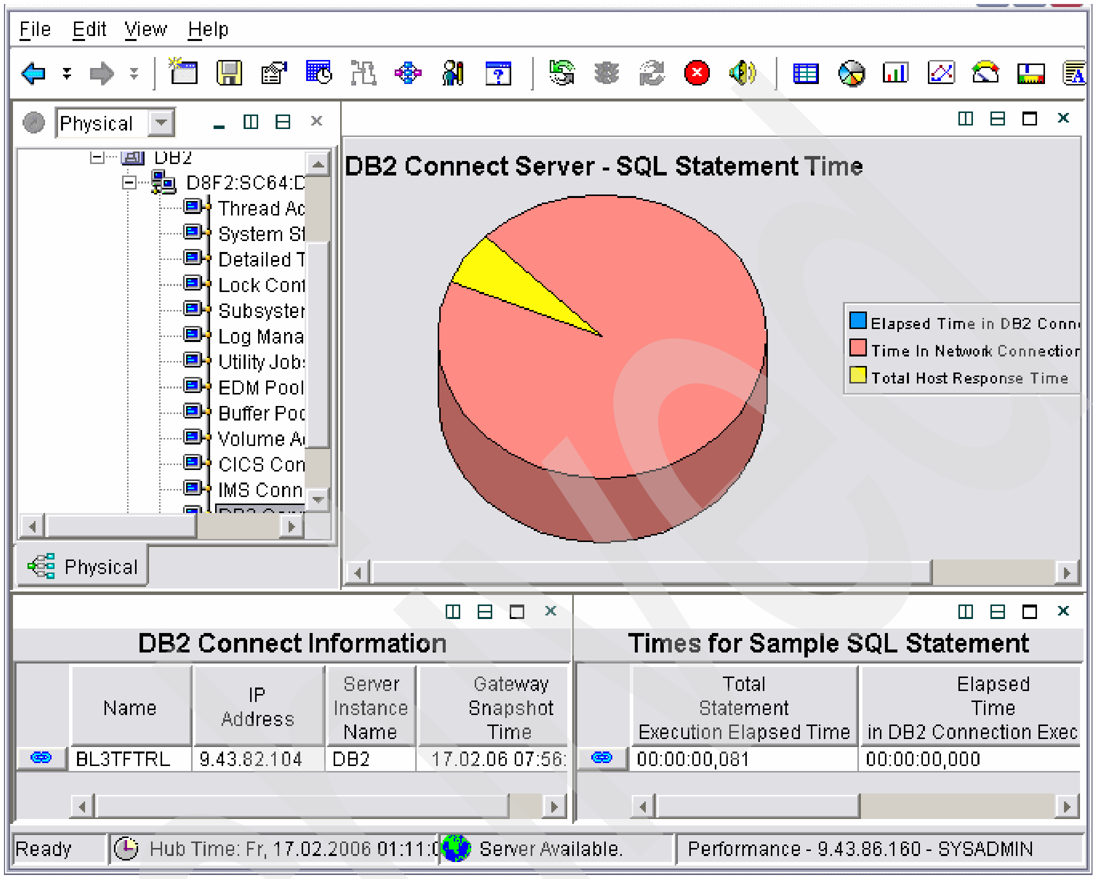


Figure 12-17 DB2 Connect performance for sample SQL statement

12.3.2 Monitoring DCS applications

In the VTAM and TEP interface, you can view information about DCS applications that are related to DB2 threads. Starting from the Thread Activity panel in VTAM and the Thread Activity node in the TEP workspace, you can monitor the DB2 Connect agents. It is possible to view the performance information for an active agent and if necessary, look at the corresponding activity for the distributed thread in the host, seamlessly.

VTAM

In the VTAM main menu, you can select option **T-THREAD ACTIVITY** and then option **G-DIST DBAC** to get all database access threads (DBAT) that are currently coming through a DB2 Connect gateway to your monitored DB2 subsystem at the host.

You can see some overview information for the DBAT threads such as elapsed times and workstation names if set by the applications. To see more detailed information about a particular thread, place the cursor on the line containing the name of the thread and press PF11 to get distributed thread details, distributed TCP/IP data and distributed SQL statistics. Figure 12-18 shows a sample SQL statistics section.

```
 +                         Distributed SQL Statistics
+
+ Remote Location Name = 9.43.82.104      Remote Location Luname =
+
+ Protocol Used        =                  Conversations Queued   =          0
+ Block Mode Switches  =            0     Message Buffer Rows    =      28689
+ Bind Remote Access   =            0     Max Allocated Conv     =          0
+ Conv Allocated       =            0     Conv Deallocated       =          0
+ Indoubt/Remote       =            0     Commit/Remote          =          0
+ Rollback/Remote      =            0
+
+        Tran     SQL     Row Message     Byte  Commit   Abort    Conv  Blocks
+     ------- ------- ------- ------- -------- ------- ------- ------- -------
+ Sent       0       0       0     679 41905180       0       0       0     673
+ Rec'd      1     676       0     679    76155       0       0       1       0
+
+                               Last 2-Phase                  Commit Backout
+ 2-PHASE COMMIT:      Prepare   Agent   Commit Backout  Forget    Resp    Resp
+                      ------- ------- -------- ------- ------- ------- -------
+          Sent              0       0        0       0       0       0       0
+          Rec'd             0       0        0       0       0       0       0
```

Figure 12-18   SQL statistics of a DBAT thread

In this panel, you should select option **V-DB2 CON SRV** to drill further down and get information about the agent in the DB2 Connect server that is connected to the DBAT thread at the host. You will see network statistics, the host response time with CPU distribution, the network send / receive byte count, etc., as shown below. The following panels have a common header (see Figure 12-19) and show the options A to D for selecting the DB2 Connect information.

```
 ________________ ZTDBC    VTM     O2       V310./C D8F2 02/17/06 16:22:40   2
> Help PF1                                                           Back PF3
>
>       THREAD INFORMATION:  Enter a selection letter on the top line.
>
> *-DB2 Connect Server  B-Overview   C-Statement Info   D-Package Statistics
>
===============================================================================
>                         DB2 Connect Server
 PLAN
+ Thread:  Plan=DISTSERV  Connid=SERVER   Corrid=DB2BP.EXE    Authid=PAOLOR2
+ Dist  :  Type=DATABASE ACCESS, Luwid=G92B5268.EC04.060217212041=820
```

Figure 12-19   Common header for DB2 Connect data

Option **A-DB2 Connect Server** provides an overview, and you can see the name and the IP address of the DB2 Connect gateway.

Option **B-Overview** presents an overview about the agent in the DB2 Connect gateway. You can find application information such as the application ID, and data about the client and the DB2 host such as the client process name, the authorization id, the host response time, the start time stamp and the time spent in DB2 Connect execution. The application ID stands for the identifier that is generated when the application connects to the database at the database manager. Figure 12-20 shows the client and DB2 host sections of this panel.

```
+ Client
+ -----------------------------------------------------------------
+ DB2 Connect First Connection      = 2006-02-17-16.20.38.281000
+ Unit of Work Start Timestamp      = 2006-02-17-16.20.38.845000
+ Unit of Work Stop Timestamp       = N/P
+ Previous UOW Completion Timestamp = N/P
+ Unit of Work Completion Status    = N/P
+ Elapsed Time DB2CONN Execution    = 00:01:42.545433
+ Most Recent UOW Elapsed Time      = 00:00:00.000000
+
+ DB2 Host
+ -----------------------------------------------------------------------------
+ Database Name         = DB8F             Inbound Byte Sent        =        0
+ Product/Version ID    = DSN08015         Inbound Byte Received    =   121256
+ Response Time         = 00:02:32.689111  Outbound Byte Sent       =   130584
+ Stmt Exec Elapsed Time= 00:00:03.036973  Outbound Byte Received   = 72374252
```

Figure 12-20 Client and DB2 Host data on the overview panel

Option **C-Statement Info** displays data about SQL statements, CPU time distribution and network statistics for the thread (see Figure 12-21).

```
+ Time
+ ---------------------------------------------------------------
+ Statement Start Timestamp         = 2006-02-17-16.20.38.920000
+ Statement Stop Timestamp          = 2006-02-17-16.26.56.045000
+ Time Spent on Gateway Processing  = 00:02:01.405649
+ Host Response Time                = 00:03:00.460477
+ Most Recent Stmt Elapsed Time     = 00:06:17.124732
+ Stmt Elapsed Execution Time       = 00:00:03.670419
+ Local: System CPU Time            = N/P
+ Local: User CPU Time              = N/P
+
+ Network Statistics
+ --------------------------------------------
+ Inbound Number of Bytes Sent      =        0
+ Inbound Number of Bytes Received  =   143794
+ Outbound Number of Bytes Sent     =   154884
+ Outbound Number of Bytes Received = 86122589
```

Figure 12-21 Time and network statistics section on the Statement Info panel

Option **D-Package Statistics** presents the outbound and inbound data and the network time taken to transmit the data (see Figure 12-22).

```
+                   Send   Received   Send Top   Rec'd Top  Send Bot   Rec'd Bot
+ Outbound      --------   --------   --------   ---------  --------   ---------
+ Data               340       8923        168        4254        86         415
+
+
+                    128        256        512        1024      2048        4096
+ Outbound      --------   --------   --------   --------   --------   --------
+ Send Data            2          1          0          0          0          0
+ Rec'd Data           0          0          1          0          0          0
+
+
+                   8192      16384      31999      64000      GT64K
+ Outbound      --------   --------   --------   --------   --------
+ Send Data            0          0          0          0          0
+ Rec'd Data           2          0          0          0          0
+
+
+                   2 ms       4 ms       8 ms      16 ms      32 ms    GT32 ms
+ Network       --------   --------   --------   --------   --------   --------
+ Time          N/P               0   N/P               0   N/P        N/P
```

Figure 12-22 Package Statistics panel

TEP

In the TEP workspaces, you should start from the Thread Activity node to identify distributed applications that are currently coming through a DB2 Connect gateway to your monitored DB2 subsystem at the host (see Figure 12-23).

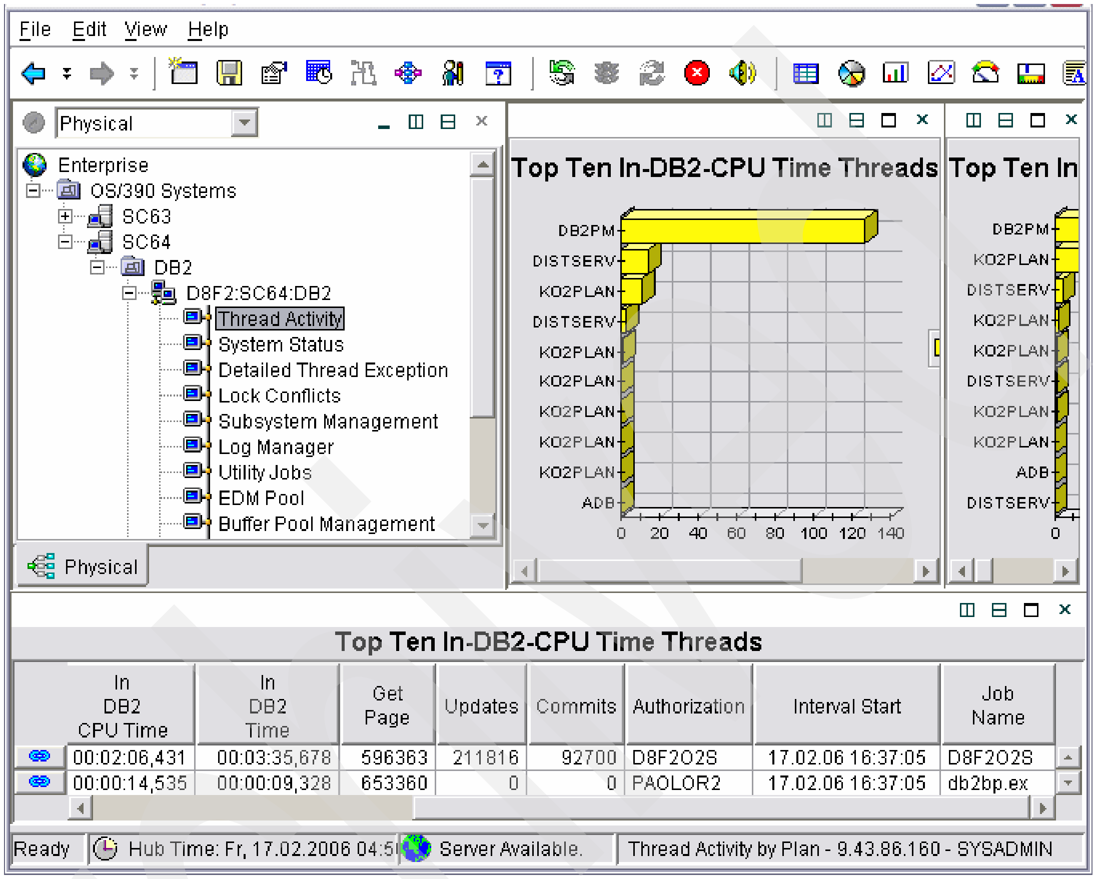


Figure 12-23 Thread Activity window

Here, you see an application with plan DISTSERV, which consumes high CPU, and you may want to analyze in more detail. A left-click on the chain symbol for the thread in the TOP Ten In-DB2-CPU Time Threads brings you to the Thread Detail window (see Figure 12-24).

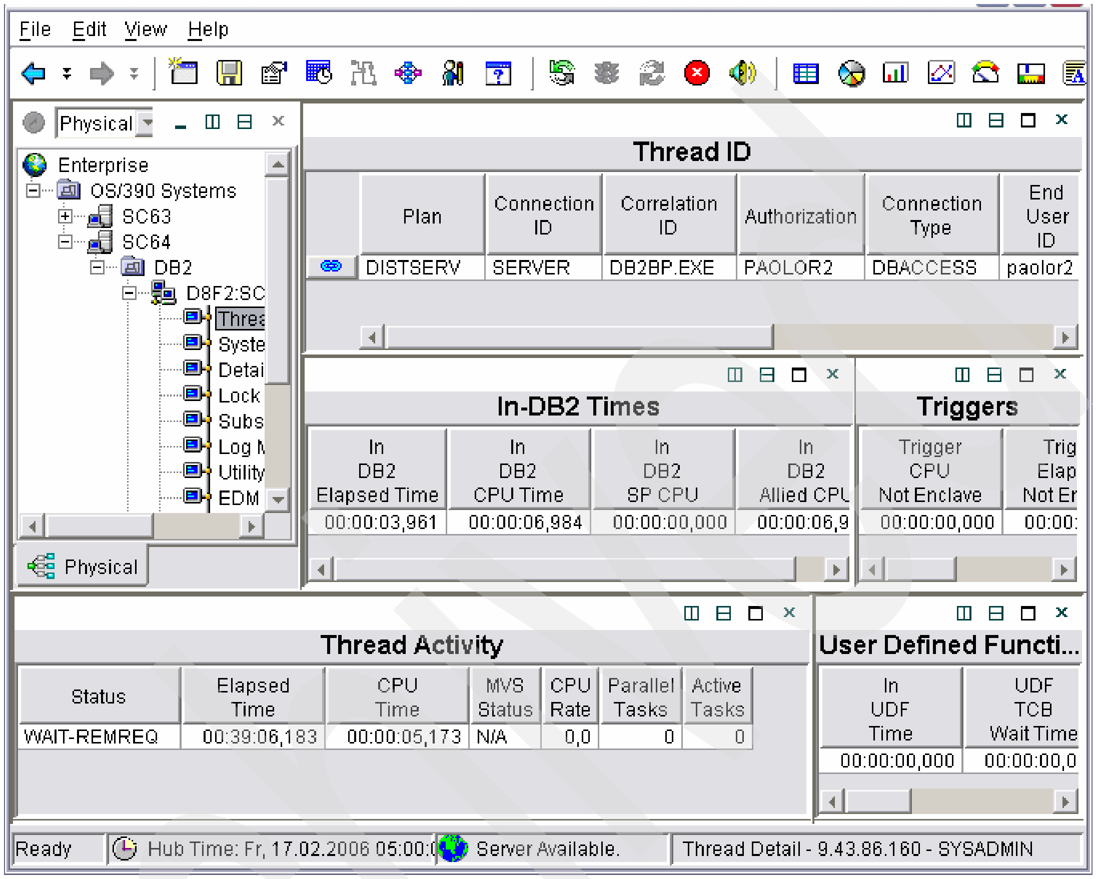


Figure 12-24 Thread details of distributed application

Right-clicking the chain for the DISTSERV thread opens a menu where you can select the **Distributed Thread Detail** menu item to get more information on the distributed application (see Figure 12-25).

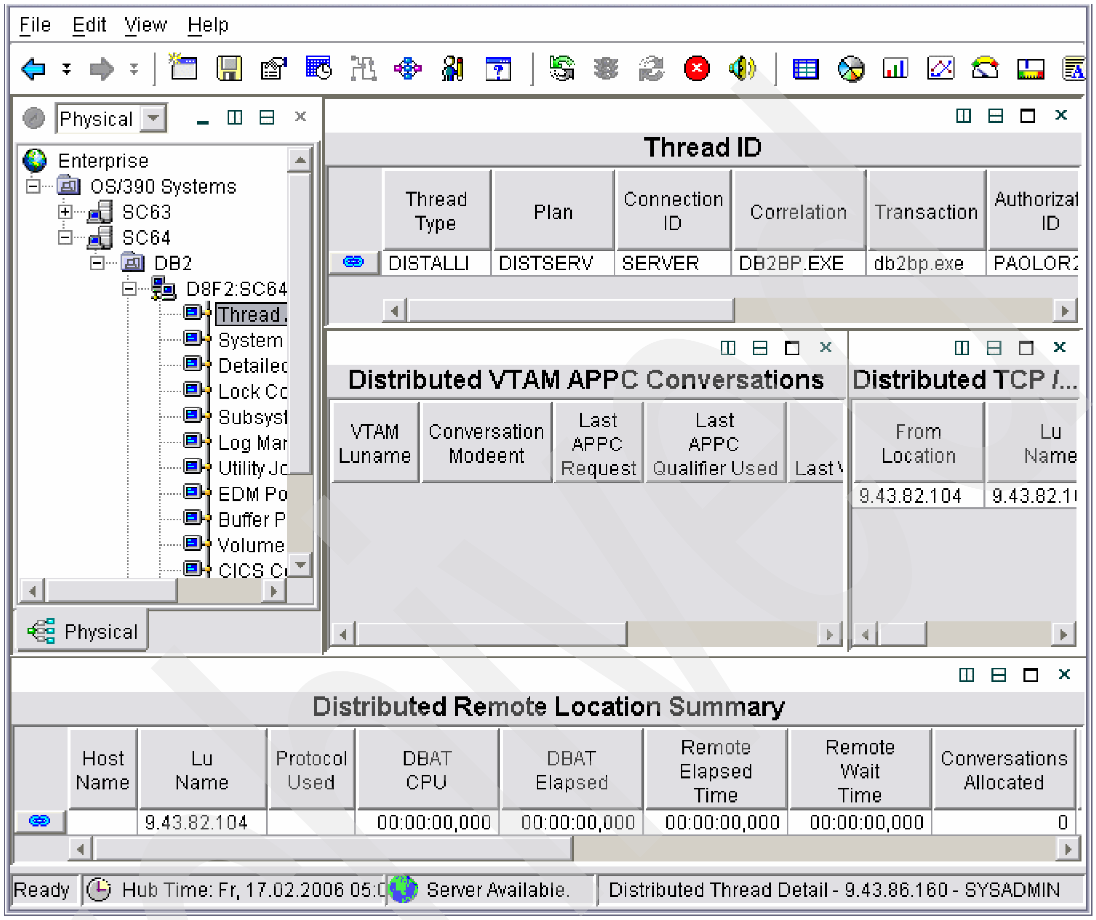


Figure 12-25 Distributed thread details of distributed application

A right-click on the chain in the Thread ID table (where Thread Type is equal to DSNALLI) opens a menu where you select **DB2 Connect Server**. This link navigates you to an overview window that contains information about the agent in the DB2 Connect server which is connected to the monitored DBAT thread at the host. In addition, a table for the DB2 Connect server is displayed.

A right-click on the agent thread with plan DISTSERV in the Thread/Application ID table displays a menu that allows you to drill down to the various agent perspectives as shown in Figure 12-26.

Note: A right-click on the chain symbol related to the DB2 Connect server leads you to the agent perspectives and not to the server statistics, as you might expect.

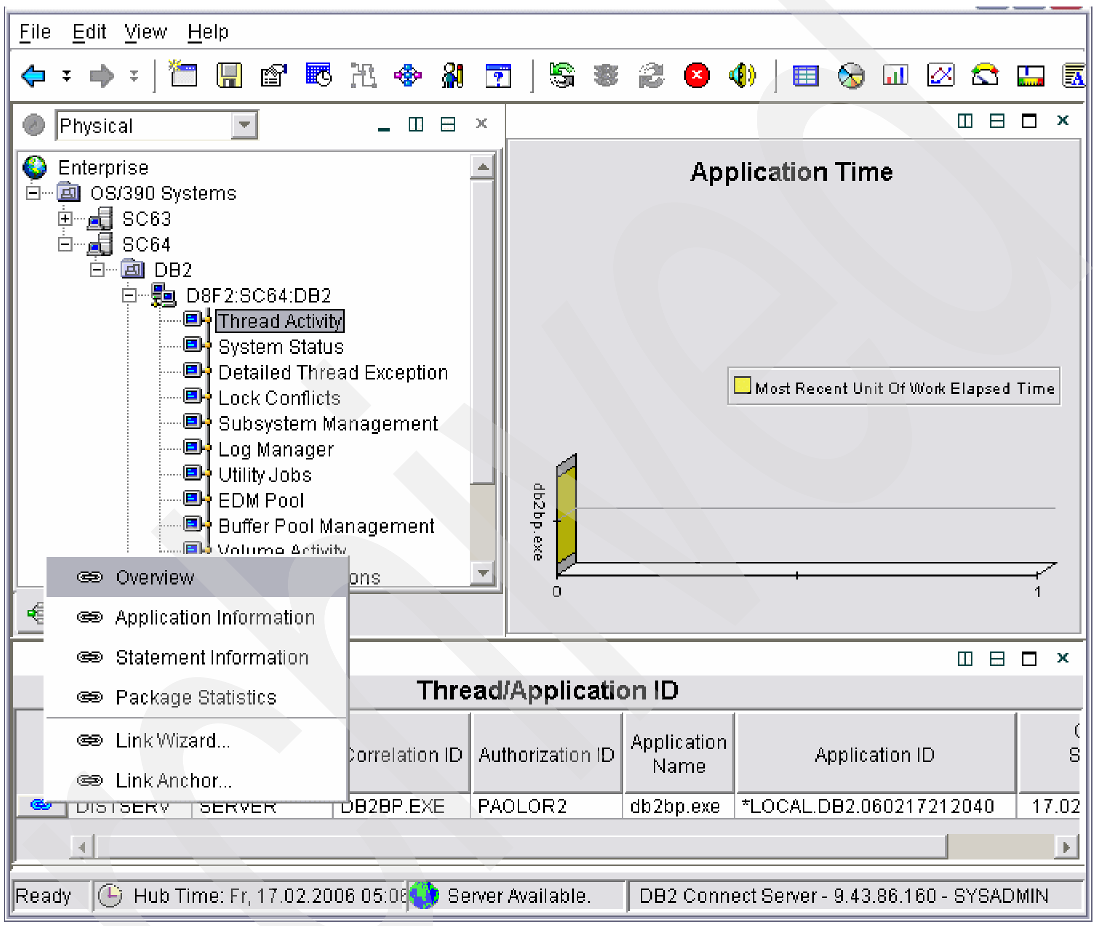


Figure 12-26 Thread/Application ID

The various views on the agent in the DB2 Connect server contain the same data elements as the VTAM panels. See the following figures, which show the windows providing overview, application, statement, and package statistics data. See Figure 12-27 for the agent overview workspace.

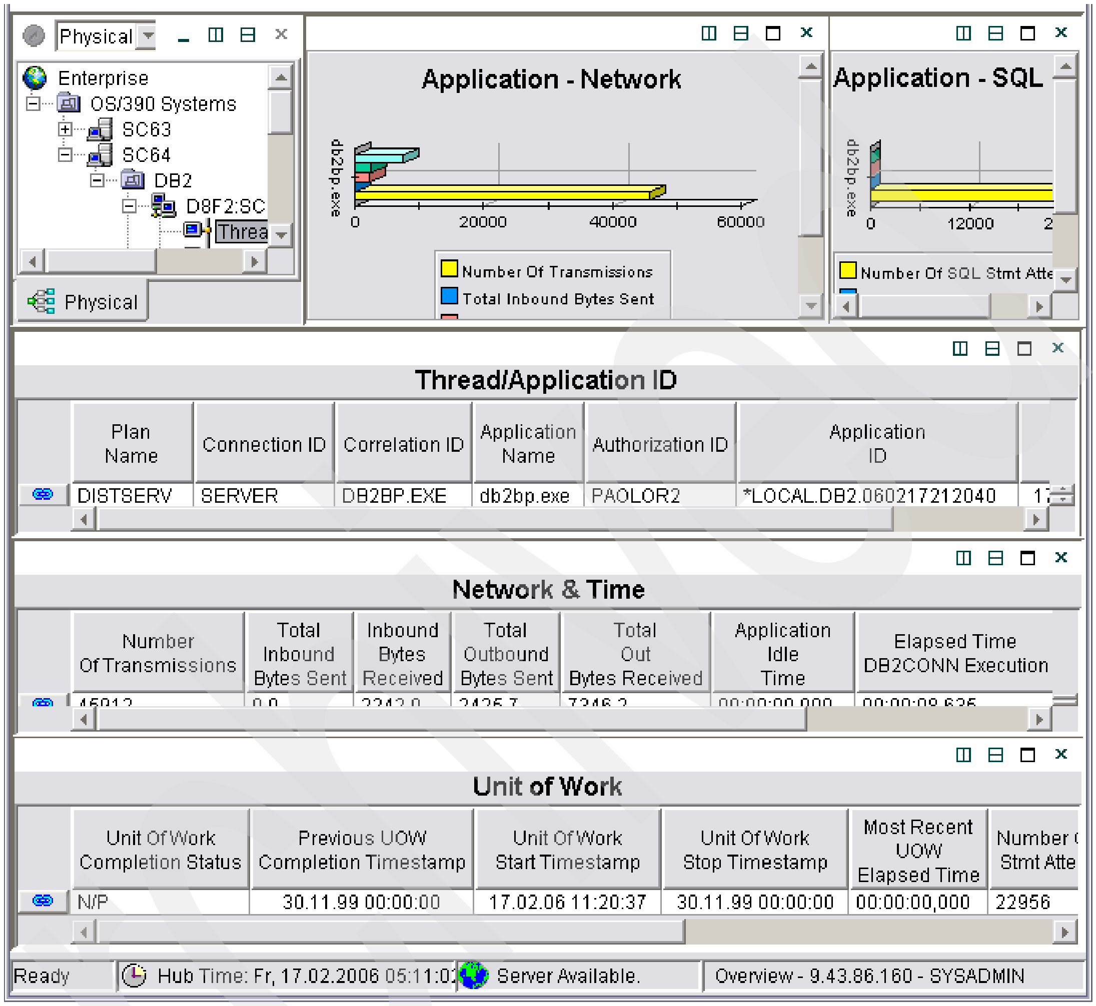


Figure 12-27 Agent overview

See Figure 12-28 for the application information workspace.

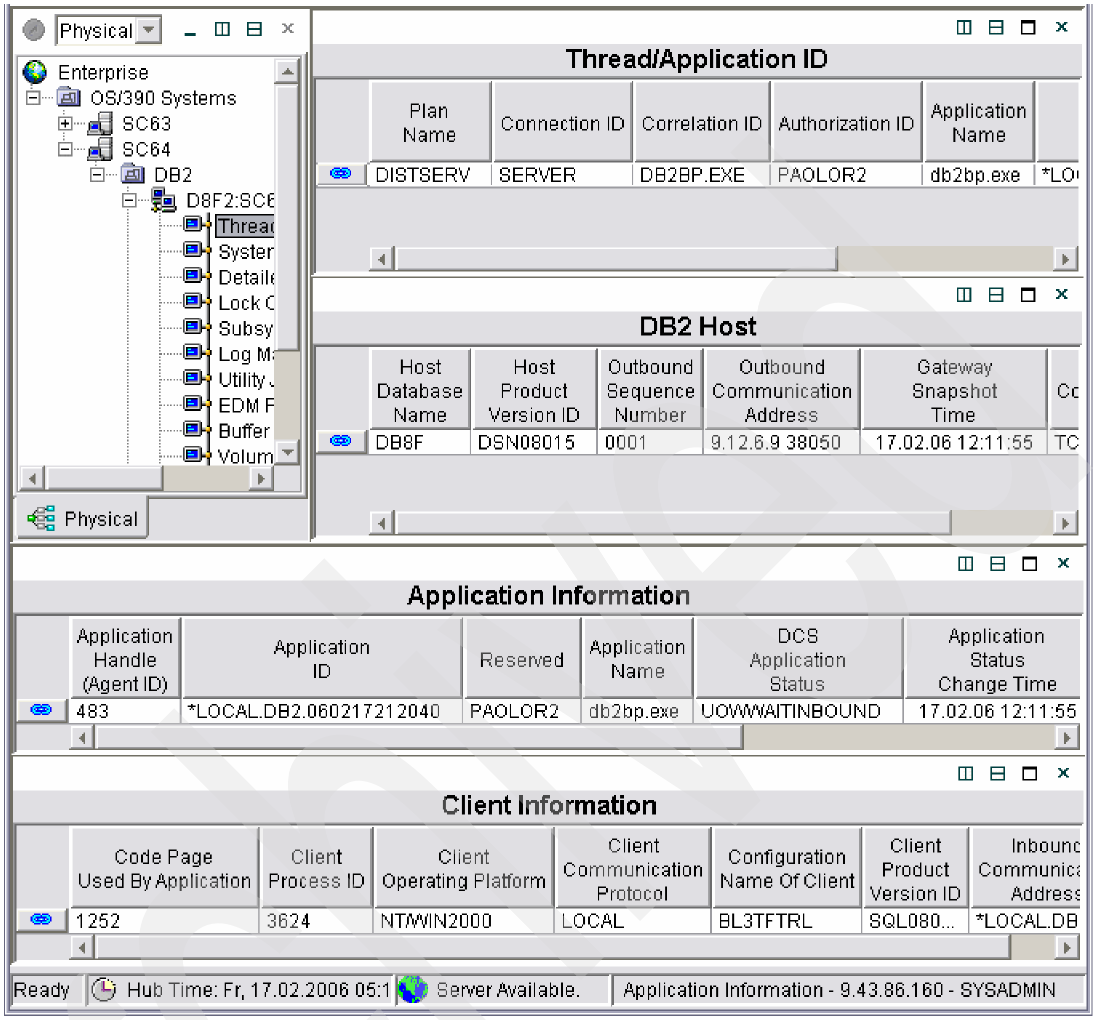


Figure 12-28 Agent application information

See Figure 12-29 for the statement information workspace.

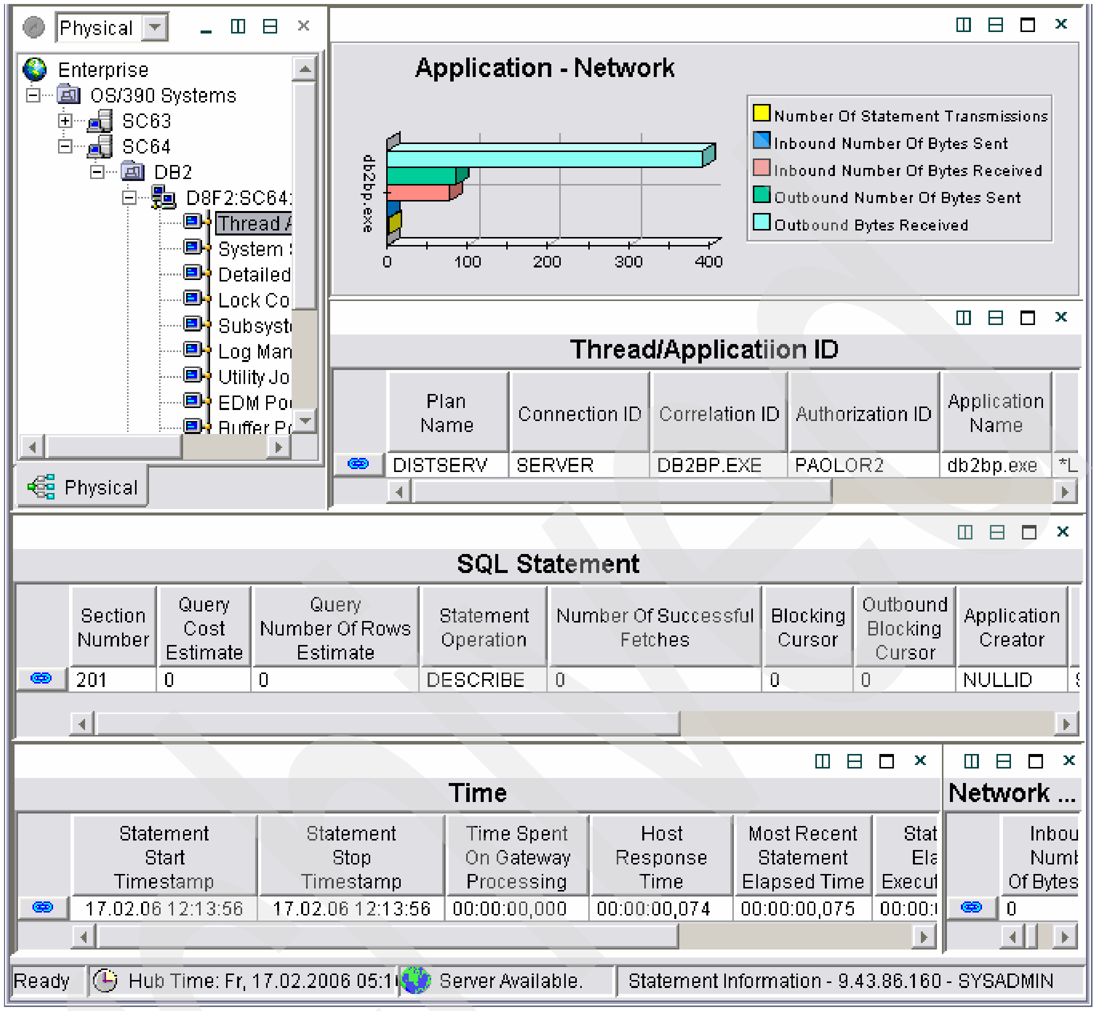


Figure 12-29 Agent statement information

See Figure 12-30 for the package statistics workspace.

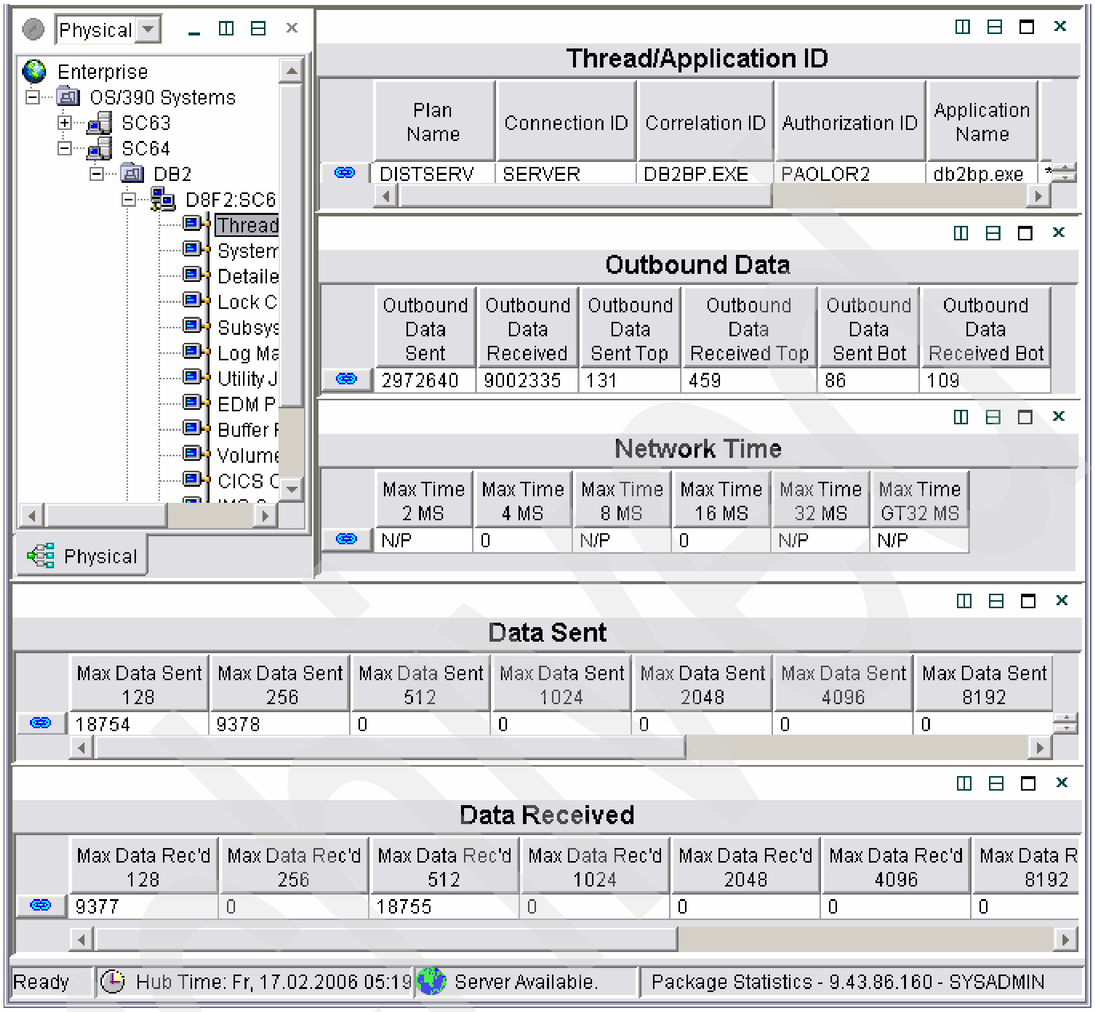


Figure 12-30   Agent package statistics

In the TEP user interface, you can alternatively navigate to DB2 Connect agent data:

1. Choose a distributed thread in the Thread Activity window, right-click, and select the menu item **Distributed Database Access Thread Summary**.
2. Right-click a distributed thread in the Distributed Database Access Thread Summary table and select the menu item **Distributed Thread Detail**.
3. Right-click the thread in the Thread ID table and choose option **DB2 Connect Server**.

12.4 DB2 Connect Monitoring in a data-sharing environment

In the OMEGAMON Server, DB2 Connect Monitoring is accomplished by a PE Server instance and its associated Performance Warehouse and snapshot history datasets. In a data-sharing environment, you need to be aware of some constraints that may affect DB2 Connect monitoring:

- For providing most online and snapshot history data to the user interfaces, it is sufficient to configure one PE Server instance for one member of the data-sharing group in ICAT. This is also true for data-sharing groups, where members reside on different LPARs in a SYSPLEX environment. It is recommended to configure only one PE Server for data-sharing groups.
- If you configure two or more PE Server instances for members in the data-sharing group, only one instance should have an activated Performance Warehouse in its configuration (see also Figure 14-3 on page 260).
- If you configure two or more PE Server instances for members in the data-sharing group, you must know about the effects of the snapshot history parameters for DB2 Connect Monitoring:
  - In DB2 Performance Expert, only one PE Server instance should define history collection intervals for DB2 Connect. A PE Client connects to that instance to get DB2 Connect data. Other configurations may invalidate the agent and server setup.
  - In OMEGAMON XE for DB2 PE, two or more PE Server instances can be defined with history collection intervals for DB2 Connect. However, the instance that comes up first will retrieve DB2 Connect data from the PE Agent and will store the data in its history datasets. The user interfaces have to connect to the member associated with that PE Server instance in order to get DB2 Connect monitoring data. If that PE Server instance is stopped, another instance (eligible for DB2 Connect monitoring) will take over the task and the user interfaces have to connect to this instance.

> **Note:** This functionality is made available by PTF UK12283 for APAR PK20717.

13

Reporting

In this chapter we describe the reporting functions of OMEGAMON XE for DB2 PE that generate reports out of collected DB2 trace data.

If you have used DB2 Performance Monitor or DB2 Performance Expert, you know about its useful and flexible reporting. Because of its comprehensiveness, the former DB2 Performance Expert reporting component has been chosen for the converged product, instead of OMEGAMON for DB2 on z/OS Historical Reporter.

This chapter explains the various reporting functions and shows how Historical Reporter users can migrate to the new reporting component.

We discuss the following topics:

- Overview of the reporting functions
- How to collect trace data
- How to generate reports
- How to use the report language
- How to use the report layouts

13.1 Overview of the reporting functions

This section provides an overview of the powerful reporting functions of the tool, very useful for performance analysis. This component gives you the ability to perform a thorough analysis considering every element that can affect your DB2 subsystem and your application environment (OLTP transactions and batch jobs).

Using the reporting functions, you are able to verify exactly what happened in your system. As a result, you are able to indicate and examine the causes of critical situations, and address DB2 performance problems efficiently.

The reporting functions allow you to obtain a wide variety of reports based on the collection of DB2 performance trace data. You can process the DB2 event trace data externalized by DB2 to SMF, GTF, or externalized to a sequential dataset by the Collect Report Data (CRD) feature of OMEGAMON XE for DB2 PE. The reporting functions can also work on other product-specific datasets.

Figure 13-1 shows the valid input/output datasets for the reporting functions. Very importantly, the reporting functions prepare the data that can be loaded to the Performance Database and the Performance Warehouse.

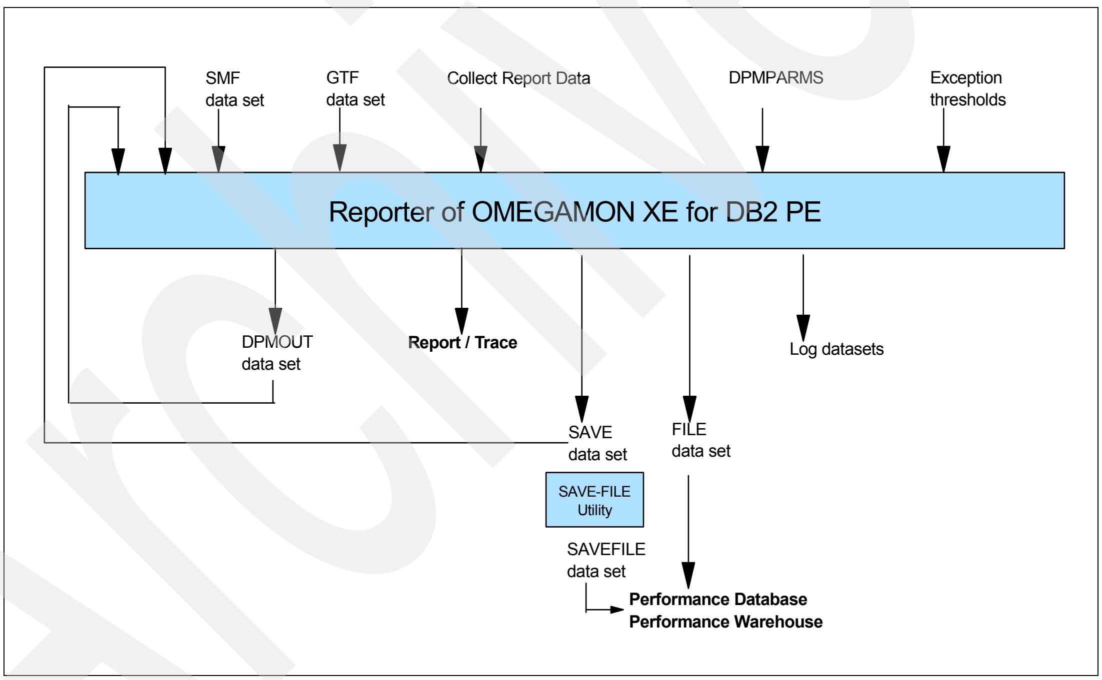


Figure 13-1 Input and output reporting datasets

Table 13-1 lists the reports that can be generated, and the DB2 trace types and classes that record the associated trace data.

Table 13-1 Reports and input data

Report	Trace type	Class
ACCOUNTING	ACCOUNTING	1, 2, 3, 5, 7, 8
AUDIT	AUDIT	1, 2, 3, 4, 5, 6, 7, 8
I/O ACTIVITY	PERFORMANCE	4, 5, 21

Report	Trace type	Class
LOCKING	STATISTICS	3
	PERFORMANCE	4, 6, 7, 17, 20, 21
RECOR TRACE	ALL	ALL
SQL ACTIVITY	ACCOUNTING	1, 2, 3, 5, 7, 8
	PERFORMANCE	2, 3, 4, 6, 8, 9, 10, 13, 16, 17, 30, 31, 32
STATISTICS	STATISTICS	1, 3, 4, 5, 6, 8
SYSTEM PARAMETERS	PERFORMANCE	ANY
	STATISTICS	5
UTILITY	ACCOUNTING	1
	PERFORMANCE	3, 4, 6, 10, 13, 16, 17

The reports (also called *report sets* in the product documentation) can be briefly explained as follows:

Accounting The accounting report set is based on DB2 accounting records, which are written when a thread terminates (that is at commit time), a thread is reused, or a DBAT becomes inactive. Local and distributed DB2 activity associated with a thread and/or DBRM/Package is contained.

Audit The audit report provides information about usage of auditable objects and authorization management, such as authorization changes, authorization control (GRANTs and REVOKEs of privileges), authorization failures, DML and DDL statements against auditable DB2 tables.

I/O Activity The **I/O Activity** report shows detailed information about buffer pool usage, **EDM pool** usage, active log, archive log and bootstrap dataset, and also cross invalidation in a data-sharing environment.

Locking The locking report provides detailed information about DB2 transaction locks, suspensions, timeouts, deadlocks, DB2 drain and claim, DB2 lock avoidance and related data, and page latches. It also provides information regarding global locks in a data-sharing environment.

Record Trace The record trace report formats most IFCID trace records. Records can be shown in a short, long, or dump format. This represents the most detailed level of performance data reporting.

SQL Activity The SQL Activity report provides information about SQL statements within a thread. Reports show details of SQL statement activity, such as workload highlights, scan, RID list, query parallelism activity, sort and I/O activities, lock suspensions, page locking, exit activity, data capture activity, and prepare (minibind) information.

Statistics The statistics report set is based on DB2 statistics trace records and contains information about SQL usage, EDM pool, subsystem services, open/close activity, log activity, plan/package processing, authorization management, locking activity, data sharing locking and group buffer pool activity, query parallelism, CPU times, buffer pool activity, and DDF activity at the DB2 subsystem level.

System Parameters The system parameters report provides information about the configuration parameters of a DB2 subsystem.

Utility The utility reports shows detailed information about executions of DB2 utilities such as LOAD, REORG or RUNSTATS.

For more information, see:

- Section 5.8 of *IBM DB2 Performance Expert for z/OS Version 2,* SG24-6867-01
- *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Report Command Reference,* SC18-9643
- *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Report Reference,* SC18-9642
- *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Reporting User's Guide,* SC18-9641

13.2 How to collect trace data

DB2 trace allows you to trace and record subsystem data and events. There are five different types of trace: STATISTICS, ACCOUNTING, PERFORMANCE, AUDIT and MONITOR. The easiest way to read and interpret the DB2 trace data is through the reporting functions.

DB2 provides commands such as START TRACE, STOP TRACE, DISPLAY TRACE and MODIFY TRACE for controlling the collection of trace data. Several parameters can be specified to further quality the scope of a trace. Specific events within a trace type can be traced as well as events with specific DB2 plans, authorization IDs, resource manage IDs and location. The destination to which trace is sent can also be controlled. The DB2 trace data can go to one of the following destinations:

- System Management Facility (SMF)
- Generalized Trace Facility (GTF)
- OP buffer

When you install DB2, you can request that any trace type and class start automatically when DB2 starts.

To monitor your systems, it is important to have some DB2 performance traces permanently turned on. Accounting and statistics reports are extremely useful because you get the most comprehensive information for the least cost. The default destination for statistics and accounting traces is SMF.

> **Tip:** Recommended traces to start for routine monitoring are accounting class 1, 2, 3, 7 and 8 and statistics class 1, 3, 5, 6 and 8.

Performance traces, on the other hand, have a very high cost and are not permanently turned on. They are started only for monitoring specific time periods or applications. Aspects of trace data collection are thoroughly discussed in section 5.7 of *IBM DB2 Performance Expert for z/OS Version 2,* SG24-6867-01.

OMEGAMON XE for DB2 PE provides interfaces to collect trace data for immediate problem analysis (the so-called Collect Report Data (CRD) function). This function starts the DB2 traces, and directs DB2 to write the data into the OP Buffer. CRD reads the trace data from the OP buffer and saves it to a sequential TSO dataset, which can then be analyzed by the reporting functions.

You are not required to have any knowledge of which DB2 traces need to be started for specific report sets. Data collection can be automatically started and stopped on the basis of time periods or events of interest. You can also view the status of traces and messages. To collect data, you require the necessary DB2 authority to start and stop DB2 traces.

CRD can be used from the ISPF interface and the Performance Expert client.

> **Note:** The near-term history component supports the collection of trace data with emphasis on recently completed threads. This data can be viewed from the Classic Interface in online mode but cannot be read and interpreted by the reporting functions. CRD is the choice of collecting general and problem-specific trace data for the reporting functions.

13.2.1 Use the Collect Report Data function from ISPF

You can access the collect report data panels from within the ISPF main panel by first selecting option **3. View online DB2 activity - PE ISPF OLM** and then by selecting option **6a. Collect Report Data - General** or by typing COLLECT on the command line.

The Collect Report Data General panel is the first panel you see. To collect performance data, you first have to configure a collect task. You can configure and start up to four independent collect tasks. For each collect task you must specify:

- The type of data you want to gather:

  Select the report sets for which you want data to be collected. A DB2 trace is started for each report set selected. You can collect data for the following report sets:

  – Accounting
  – Audit
  – I/O Activity
  – Locking
  – Record Trace
  – SQL Activity
  – Statistics
  – System Parameters
  – Utility Activity.

  You can also collect specific data types, IFCIDs, and limit the data by requesting location, plan name, and authorization ID.

- Trace start and stop criteria:

  When the criteria are met, the collect task is automatically started or stopped. The criteria to start collecting data can be, for example, time, periodic exception, exception event. The criteria to stop collecting data can be elapsed time, number of records collected, thread termination, or a specified number of records collected for a specific IFCID.

Figure 13-2 shows a sample configuration for a collect task that will trigger the immediate collection of accounting IFCIDs 3 and 239 once the task has been activated.

```
 Trace Configuration

Task Description . . . . . . : Collect Task A
                                                            More:     +
Trigger by . . . . . . . . . . 4  1=Time
                                  2=Periodic exception
                                  3=Exception event
                                  4=Immediate Start

Enter one or more selection characters to start DB2 traces for specific
DB2 PM report sets or overtype with a blank to delete the selection.

/  Accounting
   Audit
   I/O Activity
   Locking
   Record Trace
   SQL Activity

Command ===>
 F1=Help     F2=Split    F3=Exit     F7=Up       F8=Down     F9=Swap
F12=Cancel  F16=Look
```

Figure 13-2 Collect task configuration for immediate Accounting traces

- Output dataset name:

  Figure 13-3 shows that the DB2 trace is directed to a new dataset `PAOLOR2.ACCT.CRD`. The trace collection stops automatically after 5 minutes.

```
 Trigger Immediately

 Task Description . . . . : Collect Task A
                                                            More:     +
 Output Data Set for DB2 trace data to be written to
   Name . . . . . . . . . . 'PAOLOR2.ACCT.CRD'
   Disposition  . . . . . . 3  1=Append
                               2=Overwrite
                               3=New

 Start the DB2 traces immediately

 Stop the DB2 traces when any of the following conditions occur
 /  Elapsed time  . . . . . . . . . . . . . . . 300    (seconds)
    Number of records collected . . . . . . . . 0

 Additional stop conditions
    Thread termination

 Command ===>
  F1=Help     F2=Split    F3=Exit     F4=Prompt   F7=Up       F8=Down
  F9=Swap    F12=Cancel  F16=Look
```

Figure 13-3 Output dataset definition and the stop condition for the collect task

After a collect task has been configured, you have to activate it. The task then triggers the appropriate DB2 traces to start and stop when the trace start and stop criteria have been met, and writes the collected data to a dataset you have specified. You can even stop DB2 traces immediately by stopping the collect task.

In our sample, the activation of the task starts the DB2 traces immediately and the status field in the Collect Report Data General panel shows `Collecting data`. You are then able to display the status of the trace collection in more detail using the **Display** option. Figure 13-4 shows that traces for Accounting classes 1,2,3,5,7,8 are active and are written by DB2 to OP3. After 5 minutes of trace collection, the collect tasks stops and the status field shows `Data available`. The dataset PAOLOR2.ACCT.CRD can be now read by the reporting functions to create an accounting report.

```
Trace Status Summary            Row 1 to 6 of 10

   Display Status Detail
   Display messages for this task

Task Description . . . . : Collect Task A
Data Set Name  . . . . . : 'PAOLOR2.ACCT.CRD'
Data Set Status  . . . . : Open
DB2 Trace Data Started . : 06/02/09 14:15:26.90
Records Read . . . . . . : 4

Active traces for this destination
DSNW127I  -D8F2 CURRENT TRACE ACTIVITY IS -
TNO TYPE   CLASS        DEST QUAL
07  ACCTG  01,02,03,05, OP3  NO
07         07,08
*********END OF DISPLAY TRACE SUMMARY DATA*********
DSNW143I  -D8F2 CURRENT TRACE QUALIFICATIONS ARE -

Command ===>
 F1=Help     F2=Split    F3=Exit     F7=Up       F8=Down     F9=Swap
F12=Cancel  F16=Look
```

Figure 13-4 Status of task collecting accounting trace records

13.2.2 Use the Collect Report Data function from PWH client

The configuration of collect tasks can also be performed from the workstation via the Performance Warehouse graphical user interface, which is launched from the Performance Expert client. Collect tasks are defined as steps that can be combined with other steps in PWH processes.

For example, you might want to collect performance data to collect performance data on a regular basis, for example, to collect accounting data between peak time 10 a.m. and 12 noon every day from a specific DB2 subsystem. To perform this task, you

1. Define a process consisting of a single CRD (collect report data) step and specify the step properties.
2. Schedule the process to run periodically and activate it. During process execution, the trace data is collected and written to a sequential dataset. This dataset can be processed by the reporting functions.

Figure 13-5 on page 228 through Figure 13-7 on page 230 illustrate the definition and execution of a Performance Warehouse process containing a CRD step.

You can copy the sample process, *DB2PM.Templates.Collect Report Data*, from the Public process group folder to your personal process group and customize the step properties to your needs (Figure 13-5).

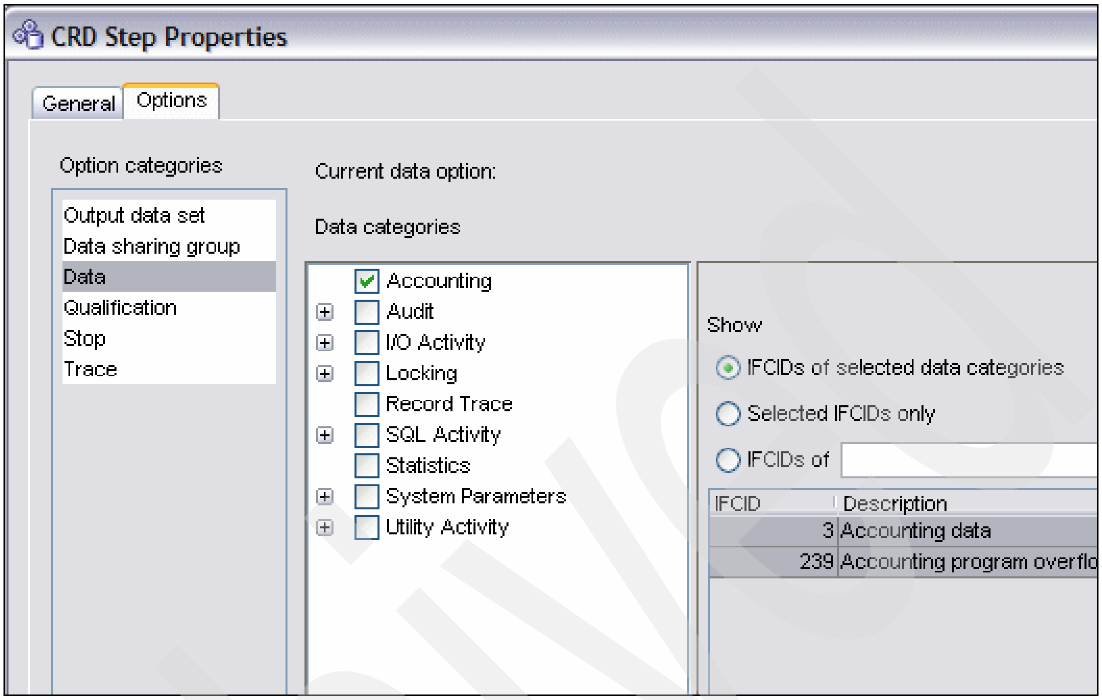


Figure 13-5 Definition of CRD Step properties

The configuration options in the properties windows are organized by categories. The folders pane lists the categories and the contents pane reflects the current definitions for a category that you can modify to your needs.

Note: The *data sharing group* option allows you to collect data for one member or all members if the Performance Warehouse resides in a data-sharing group.

The execution of the process is activated by setting the *Schedule of process* and the *Status* field in the Process Properties window (see Figure 13-6).

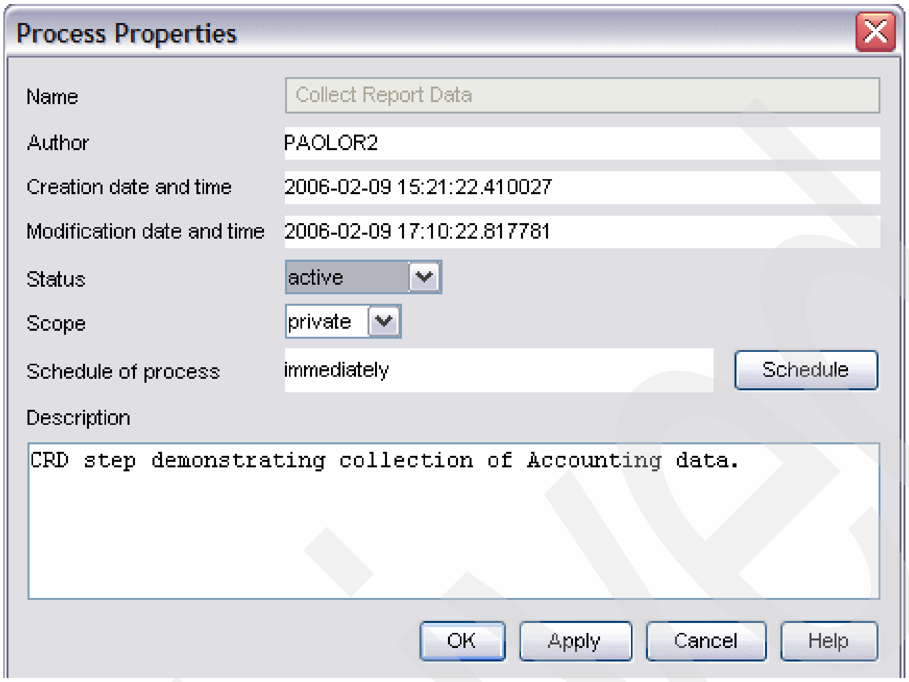


Figure 13-6 Start of the collect process

The status of a process execution can be viewed in the Process Execution Details window as shown in Figure 13-7. When the process terminates, the status of the execution changes to *FINISHED*.

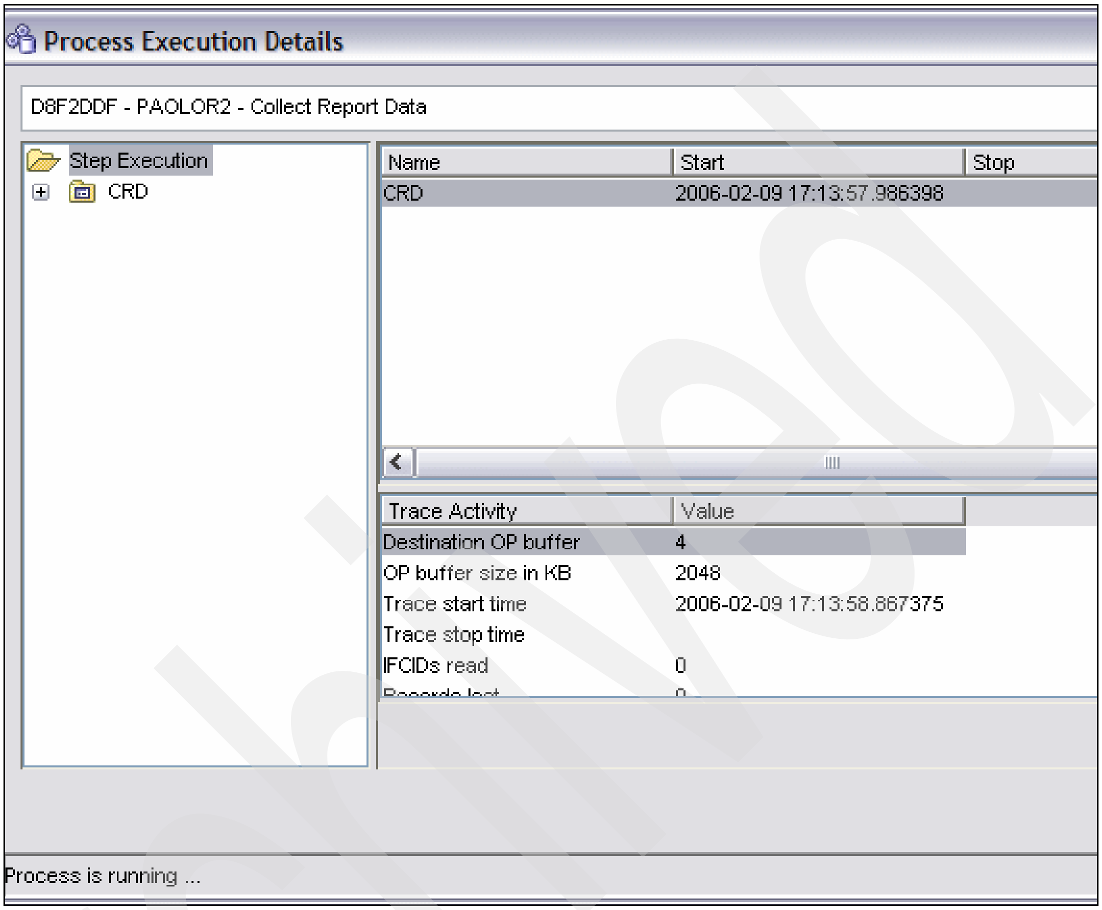


Figure 13-7 Status of the process collecting accounting data

Notes:

- To configure a CRD step for a monitored DB2 subsystem, a Performance Warehouse (PWH) must be configured and created in that subsystem. The PWH provides the infrastructure for the definition and execution of processes containing CRD steps.
- To execute a CRD step, the OMEGAMON Server and the PE Server instance associated with the DB2 subsystem must run with an activated PWH..

If you have MONITOR1 and MONITOR2 privileges, you can create and view SQL Activity reports for a specific DB2 thread. This function can invoked from the Performance Expert client and uses the infrastructure of the PWH to configure and execute CRD steps (see 13.3.4, "Using the PE client to create SQL Activity reports").

For more information, how to work with Performance Warehouse see *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Monitoring Performance from the Performance Expert Client,* SC18-9640.

13.3 How to generate reports

Once SMF, GTF or trace data collected by the CRD function is available, it can be read and interpreted by the reporting functions. OMEGAMON XE for DB2 PE provides the following alternatives for invoking the reporting functions:

- Using the Interactive Reporting Facility (IRF) from the ISPF interface
- Using a self-defined job with JCL statements and report commands
- Using the Performance Warehouse client
- Using the Performance Expert client to create SQL Activity reports.

13.3.1 Using the Interactive Report Facility

The Interactive Report Facility (IRF) provides a series of panels at the ISPF interface that lets you interactively specify and create the desired reports. After validation of your specifications, the IRF automatically generates a batch report command stream with JCL statements and the commands, subcommands, options and keywords that match your selections for the requested reports. This job can then be executed in foreground or background mode to produce the reports.

Example of producing an accounting report

The following steps show how you can request a long accounting report from the trace data you previously collected with the CRD function.

When you invoke the **1. Create and execute reporting commands** option from the ISPF main menu, a panel is displayed, from which you select which reports you want to create, the functions to apply to the selected reports, and additional functions to be reflected in the batch report command stream. The panel provides a matrix for selecting report sets (Accounting, Statistics, and so on) and the functions (Reduce, Report, and so on) to be applied to the selected report sets.

Enter `RESET` in the command line to start from scratch, select **Accounting Report** with a '/' symbol, and press Enter.

A selections panel appears, where you enter a'/' and a user comment for your report. See Figure 13-8.

```
Accounting REPORT Selections

Select one or more reports, then press Enter. Overtype with space to
delete any report. Request EXIT when complete.

   DDname    User Comment
/            My first Accounting report









Command ===>
 F1=Help    F2=Split   F3=Exit    F6=Browse  F7=Up      F8=Down    F9=Swap
```

Figure 13-8 Define the first accounting report

Then press Enter and a panel appears with ACRPTDD as the default DDname for the report. Specify the report parameters as shown in Figure 13-9.

```
Accounting REPORT

Update fields as required, then press Enter.

                                                                More:     +
User comment  . . . . . . . My first Accounting report
DDname  . . . . . . . . . . ACRPTDD

Scope . . . . . . . . . . . 1       1=member 2=group

Layout  . . . . . . . . . . LONG    User-tailored Report Format

Exception . . . . . . . . . 2  1=yes   2=no

Select to change values or overtype with space to use default.

   Top Entries
   Order Selections

                                       YY  MM  DD  HH  MM  SS  TH
Command ===>
 F1=Help      F2=Split     F3=Exit      F6=Browse    F9=Swap     F10=Global
F11=Inclexcl F12=Cancel
```

Figure 13-9 Update report properties

Press Enter and exit the next panel to return to the matrix of reports and functions. Now press F5=COMPOSE to provide additional dataset information. Here, specify `PAOLOR2.ACCT.CRD` as the required input dataset, which contains the previously collected accounting trace data. This shown in Figure 13-10. If you leave ACRPTDD empty, the report output is directed to SYSOUT.

```
  Background DDname Selections

  Update the data set information. Select one or more data sets to add
  extended information, then press Enter. Your TSO prefix is added to data
  set names not enclosed in single quotes. Press Enter to continue when
  complete.


      DDname       Data Set Information                             Required
      INPUTDD      'PAOLOR2.ACCT.CRD'                               *
      EXCPTDD
      EXTRCDD1
      EXFILDD1
      ACRPTDD
      DPMLOG       SYSOUT=A
      DPMOUTDD
      DPMPARMS
      JOBSUMDD
      JSSRSDD
      SYSOUT       SYSOUT=A

 Command ===>
   F1=Help    F2=Split   F3=Exit    F7=Up      F8=Down    F9=Swap   F12=Cancel
```

Figure 13-10 Update dataset information

Press Enter to complete and submit the job. Select option **1. Browse the generated job stream** and see the job stream generated by IRF. Notice that the STEPLIB entry points to the runtime library RKANMOD of your installation. The SYSIN DD contains the report language elements as specified in the previous panels. See Example 13-1.

Example 13-1 Job stream created by IRF

```
//PEV310  EXEC PGM=DB2PM
//STEPLIB  DD  DSN=OMEGASYS.DB8A.SC63RTE.RKANMOD,DISP=SHR
//INPUTDD  DD  DSN=PAOLOR2.ACCT.CRD,DISP=SHR
//DPMLOG   DD  SYSOUT=A
//SYSOUT   DD  SYSOUT=A
//SYSIN    DD  *
ACCOUNTING
            REPORT                    /*My first Accounting report*/
                     DDNAME(ACRPTDD)
                     LAYOUT(LONG)
                     NOEXCEPTION
                     SCOPE(MEMBER)
EXEC
```

Add the job statement information to your job as required in your environment (see Example 13-2) and then submit the job with option **4. Submit the job stream for background execution**.

Example 13-2 Job statement information

```
Job statement information
//PAOACC JOB (999,POK),'PAOLOR2',CLASS=A,MSGCLASS=T,
// NOTIFY=PAOLOR2,TIME=1440,REGION=0M
/*JOBPARM L=999,SYSAFF=SC64
```

See the job output for the report generation job (here: PAOACC) and select **DDNAME ACRPTDD**, which contains the accounting report for the collected trace data. See an extract of the report in Figure 13-11.

```
  LOCATION: DB8F                      OMEGAMON XE FOR DB2 PERFORMANCE EXPERT (
       GROUP: DB8FU                              ACCOUNTING REPORT - LONG
      MEMBER: D8F2
   SUBSYSTEM: D8F2                                ORDER: PRIMAUTH-PLANNAME
 DB2 VERSION: V8                                      SCOPE: MEMBER

 PRIMAUTH: PAOLOR2  PLANNAME: ADB

 ELAPSED TIME DISTRIBUTION                                            CLASS 2 TIME
 ----------------------------------------------------------------     ------------
 APPL   !=================================================> 100%    CPU    !====
 DB2    !                                                             NOTACC !====
 SUSP   !                                                             SUSP   !====

 AVERAGE       APPL(CL.1)  DB2 (CL.2)  IFI (CL.5)    CLASS 3 SUSPENSIONS   AVERAG
 ------------  ----------  ----------  ----------    --------------------  ------
 ELAPSED TIME   26.631219    0.010256         N/P    LOCK/LATCH(DB2+IRLM)      0.
  NONNESTED     26.631219    0.010256         N/A    SYNCHRON. I/O             0.
  STORED PROC    0.000000    0.000000         N/A     DATABASE I/O             0.
  UDF            0.000000    0.000000         N/A     LOG WRITE I/O            0.
  TRIGGER        0.000000    0.000000         N/A    OTHER READ I/O            0.
                                                     OTHER WRTE I/O            0.
 CPU TIME        0.007971    0.003860         N/P    SER.TASK SWTCH            0.
  AGENT          0.007971    0.003860         N/A     UPDATE COMMIT            0.
   NONNESTED     0.007971    0.003860         N/P     OPEN/CLOSE               0.
   STORED PRC    0.000000    0.000000         N/A     SYSLGRNG REC             0.
   UDF           0.000000    0.000000         N/A     EXT/DEL/DEF              0.
   TRIGGER       0.000000    0.000000         N/A     OTHER SERVICE            0.
```

Figure 13-11 Extract of sample accounting report

Select option **3. Store the job stream for future use**, which allows you to reuse and customize the IRF generated job stream. For example, store the job in dataset `PAOLOR2.REPORT.JCL(ACCTLONG)`.

13.3.2 Using a self-defined job

The Interactive Report Facility (IRF) can be considered as good starting point for creating sample reports.

Depending on your requirements, you may want to set up a range of report jobs, and maintain and run them independently of IRF. You can use the report jobs created and saved with IRF and customize them to your needs.

Example 13-3 shows a more complex command stream for the reporting functions, which:

- Creates multiple accounting and statistics reports in the same job
- Uses temporary work datasets
- Exploits the report language elements for reducing input data
- Creates SAVE and FILE datasets.

Example 13-3 A more complex command stream for reporting

```
//PAOACST JOB (999,POK),'PAOLOR2',CLASS=A,MSGCLASS=T,
// NOTIFY=PAOLOR2,TIME=1440,REGION=0M
/*JOBPARM L=999,SYSAFF=SC64
//PEV310  EXEC PGM=DB2PM
//STEPLIB  DD  DSN=OMEGASYS.DB8A.SC63RTE.RKANMOD,DISP=SHR
//*****************************************************************
//INPUTDD  DD  DISP=SHR,DSN=PAOLOR2.ACCT.CRD
//*****************************************************************
//ACRPTDD  DD  SYSOUT=*
//* ACC SAVE DATA SET (VSAM) CREATED WITH IDCAMS
//ACSAVDD1 DD  DSN=PAOLOR2.REPORT.ACSAV1,DISP=SHR
//DPMOUTDD DD  SYSOUT=*
//DPMLOG   DD  SYSOUT=*
//JOBSUMDD DD  SYSOUT=*
//SYSOUT   DD  SYSOUT=*
//* SORT WORK FILES:
//SORTWK01 DD SPACE=(CYL,(50,50)),UNIT=SYSDA
//SORTWK02 DD SPACE=(CYL,(50,50)),UNIT=SYSDA
//STFILDD  DD  SYSOUT=*
//STTRCDD  DD  SYSOUT=*
//*****************************************************************
//SYSIN    DD  *
GLOBAL
INCLUDE (LOCATION(DB8F))
*
ACCOUNTING
  REDUCE
     FROM(02/09/06,18:30:00)
     TO(02/09/06,20:30:00)
     INTERVAL(60)
     INCLUDE(CORRNAME(*))
     EXCLUDE(PACKAGE(*))
*
  REPORT
     SCOPE(MEMBER)
     LAYOUT(LONG)
     TOP(30 INDB2ET INDB2PT)
     ORDER(CORRNAME)
     DDNAME (ACRPTDD)
*
  SAVE DDNAME(ACSAVDD1)
*
STATISTICS
   TRACE
      LAYOUT(SHORT)
      DDNAME (STTRCDD)
*
   FILE DDNAME(STFILDD)
*
EXEC
```

13.3.3 Using the PWH client

The definition and creation of accounting and statistics reports can also be controlled from the workstation via the Performance Warehouse user interface, which is launched from the Performance Expert client.

The workstation interface allows you to interactively compose a report command stream. This relieves you of the need to know the exact syntax of the report language. Report generation tasks are defined as steps that can be combined with other steps in Performance Warehouse processes.

For example, you might want to create statistics reports based on SMF datasets that are continuously collected in your environment.

To perform this task, follow these steps:

1. Define a process consisting of a single report step and specify the report properties. If the SMF data is stored and maintained in a generation data group (GDG), specify the recent GDG(0) dataset as input dataset.
2. Schedule the process to run periodically according to the times the SMF switches take place and activate the process. During process execution, the statistics reports are created from the input dataset and stored in the Performance Warehouse. The reports can be viewed from the PWH client at the workstation, later on.

Or, you might want to create accounting and statistics reports regularly based on a dataset that contains the performance records for a specific time frame and DB2 location in question.

To perform this task, follow these steps:

1. Configure a CRD step as explained in 13.2.2, “Use the Collect Report Data function from PWH client” and select the accounting and statistics performance records that produce meaningful results.
2. Define an accounting report step.
3. Set up a statistics report step.
4. Schedule the process containing the CRD and report steps to run periodically and activate it. During process execution, the trace data is collected and accounting and statistics reports are generated.

Figure 13-12 on page 237 through Figure 13-14 on page 239 illustrate the definition and execution of a PWH process containing a statistics report step.

You can copy the sample process, *DB2PM.Templates.Statistics Report 1*, from the Public process group folder to your personal process group and customize the step properties to your needs (Figure 13-14).

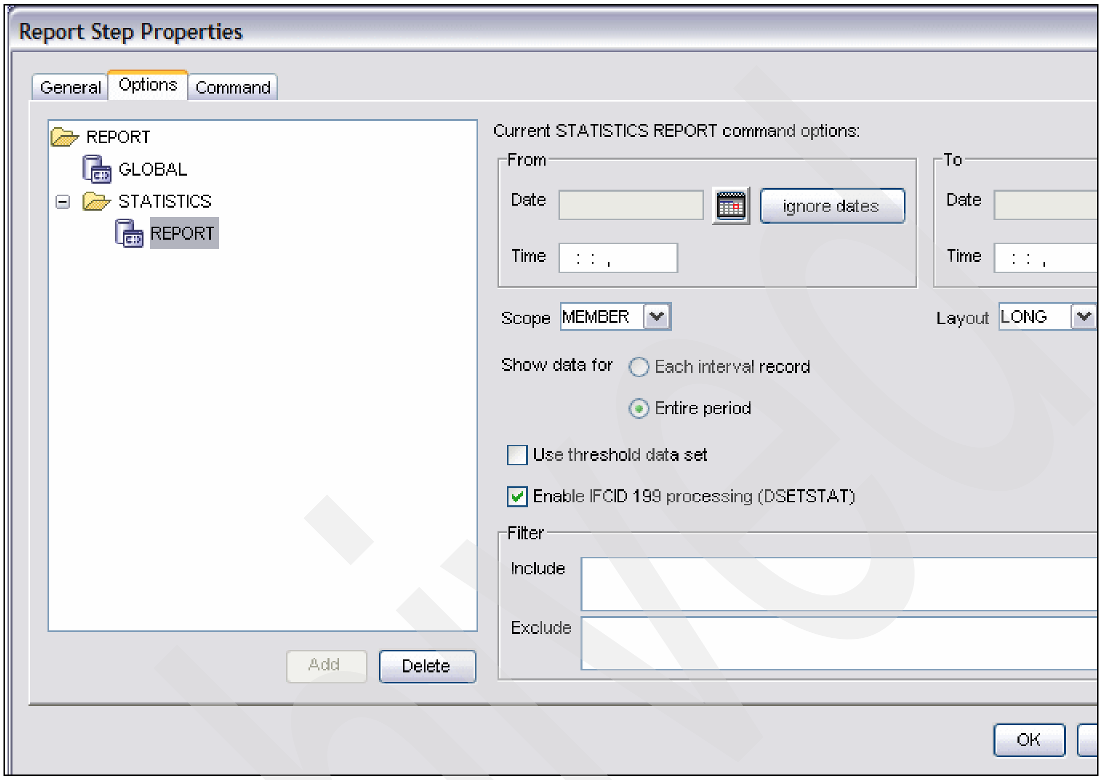


Figure 13-12 Configuration of a report step

The report options pane is organized as a tree, where each node corresponds to a report set subcommand. The contents pane reflects the current options of a subcommand that you can modify to your needs. For example, in the GLOBAL node you can specify the input dataset for the report. In Figure 13-12, you see the contents of the REPORT node and the associated options.

The execution of the process containing the report step is activated by setting the *Schedule of process* and the *Status* field in the *Process Properties* window. The status of the process execution can be viewed in the *Process Execution Details* window. When the process terminates, the status of the execution changes to *FINISHED* and you can see the output datasets that have been created and stored in the Performance Warehouse (see Figure 13-13).

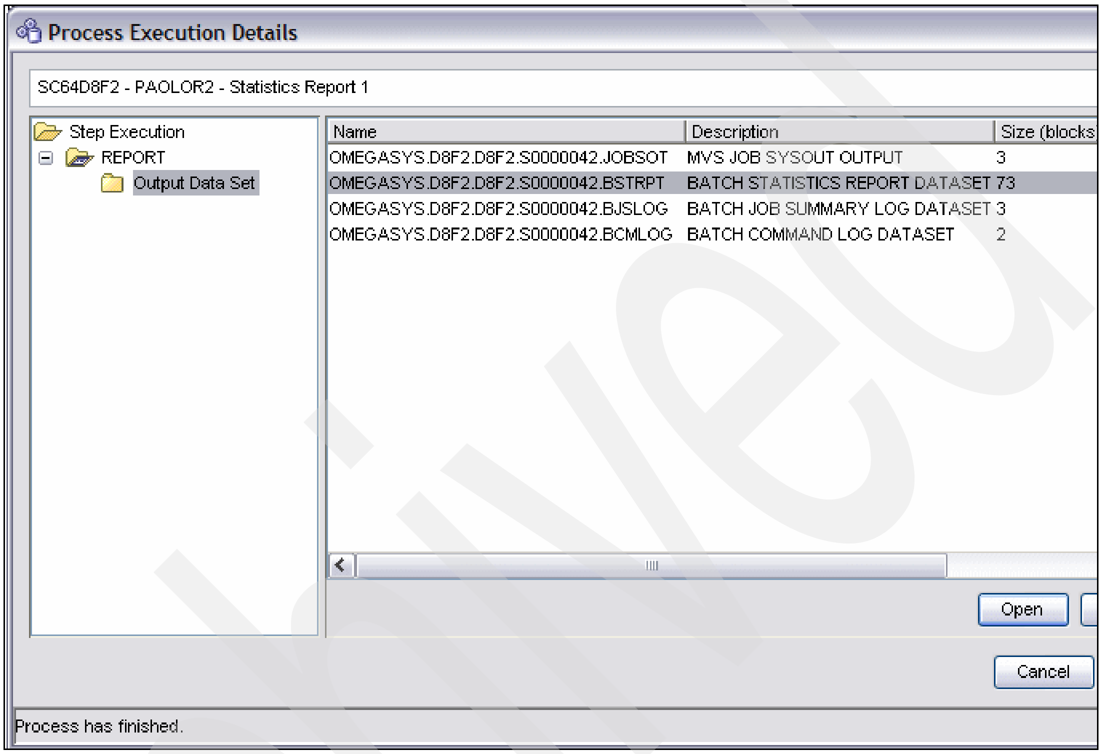
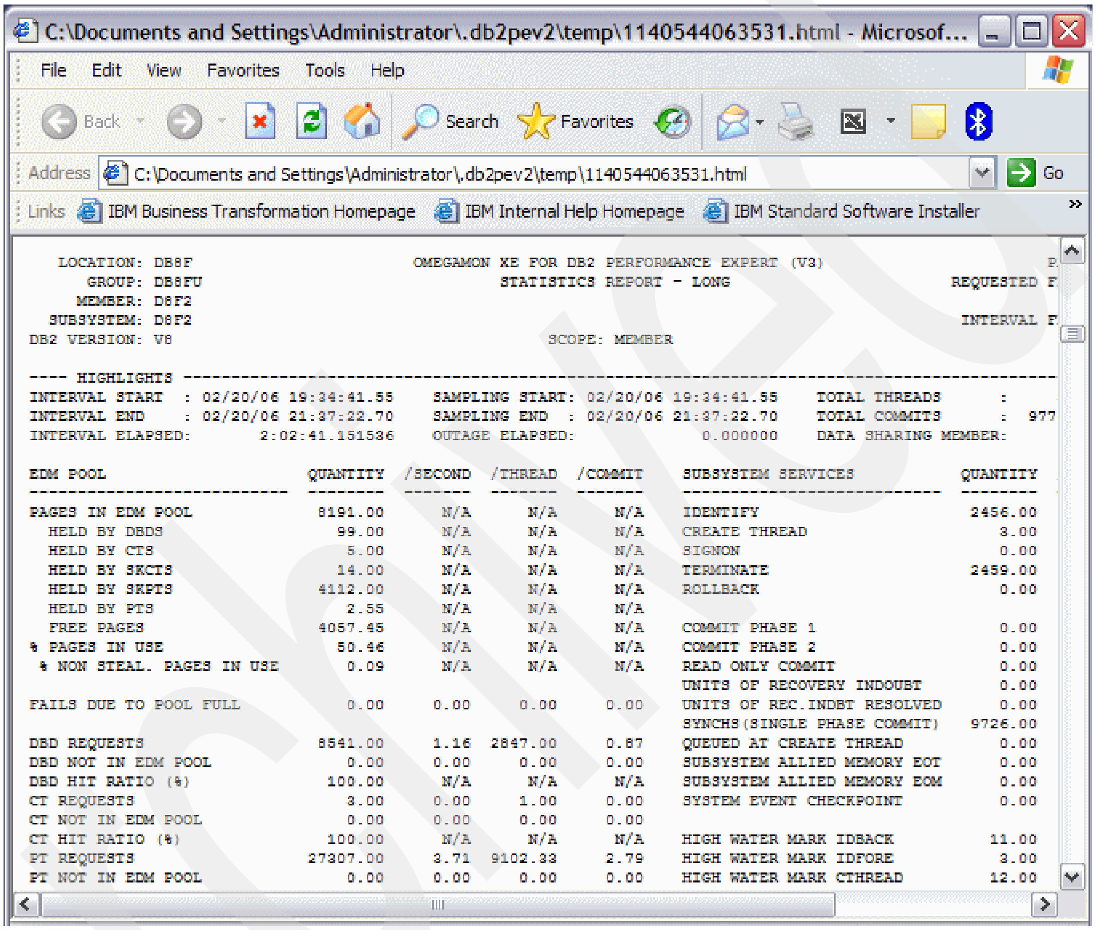


Figure 13-13 Output datasets of report step

Select the statistics report dataset and click the **Open** button to open a browser window, which contains the report (see Figure 13-14). The other output datasets contain log data and are useful for analyzing the process execution.

In the Process Execution Details window, the individual output datasets can be deleted from the Performance Warehouse. If you delete the finished process execution from the Process Executions folder, all output datasets for the process execution are removed from the PWH.




```
   LOCATION: DB8F                    OMEGAMON XE FOR DB2 PERFORMANCE EXPERT (V3)                       P
      GROUP: DB8FU                            STATISTICS REPORT - LONG                         REQUESTED F
     MEMBER: D8F2
  SUBSYSTEM: D8F2                                                                               INTERVAL F
DB2 VERSION: V8                                         SCOPE: MEMBER

---- HIGHLIGHTS ----------------------------------------------------------------------------------------------
INTERVAL START   : 02/20/06 19:34:41.55    SAMPLING START: 02/20/06 19:34:41.55    TOTAL THREADS       :
INTERVAL END     : 02/20/06 21:37:22.70    SAMPLING END  : 02/20/06 21:37:22.70    TOTAL COMMITS       :   977
INTERVAL ELAPSED:        2:02:41.151536    OUTAGE ELAPSED:               0.000000  DATA SHARING MEMBER:

EDM POOL                     QUANTITY  /SECOND  /THREAD  /COMMIT   SUBSYSTEM SERVICES            QUANTITY
---------------------------  --------  -------  -------  -------   ---------------------------   --------
PAGES IN EDM POOL             8191.00      N/A      N/A      N/A   IDENTIFY                       2456.00
  HELD BY DBDS                  99.00      N/A      N/A      N/A   CREATE THREAD                     3.00
  HELD BY CTS                    5.00      N/A      N/A      N/A   SIGNON                            0.00
  HELD BY SKCTS                 14.00      N/A      N/A      N/A   TERMINATE                      2459.00
  HELD BY SKPTS               4112.00      N/A      N/A      N/A   ROLLBACK                          0.00
  HELD BY PTS                    2.55      N/A      N/A      N/A
  FREE PAGES                  4057.45      N/A      N/A      N/A   COMMIT PHASE 1                    0.00
% PAGES IN USE                  50.46      N/A      N/A      N/A   COMMIT PHASE 2                    0.00
 % NON STEAL. PAGES IN USE       0.09      N/A      N/A      N/A   READ ONLY COMMIT                  0.00
                                                                   UNITS OF RECOVERY INDOUBT         0.00
FAILS DUE TO POOL FULL           0.00     0.00     0.00     0.00   UNITS OF REC.INDBT RESOLVED       0.00
                                                                   SYNCHS(SINGLE PHASE COMMIT)    9726.00
DBD REQUESTS                  8541.00     1.16  2847.00     0.87   QUEUED AT CREATE THREAD           0.00
DBD NOT IN EDM POOL              0.00     0.00     0.00     0.00   SUBSYSTEM ALLIED MEMORY EOT       0.00
DBD HIT RATIO (%)              100.00      N/A      N/A      N/A   SUBSYSTEM ALLIED MEMORY EOM       0.00
CT REQUESTS                      3.00     0.00     1.00     0.00   SYSTEM EVENT CHECKPOINT           0.00
CT NOT IN EDM POOL               0.00     0.00     0.00     0.00
CT HIT RATIO (%)               100.00      N/A      N/A      N/A   HIGH WATER MARK IDBACK           11.00
PT REQUESTS                  27307.00     3.71  9102.33     2.79   HIGH WATER MARK IDFORE            3.00
PT NOT IN EDM POOL               0.00     0.00     0.00     0.00   HIGH WATER MARK CTHREAD          12.00
```

Figure 13-14 Sample statistics report

13.3.4 Using the PE client to create SQL Activity reports

SQL Activity reports provide detailed information for analyzing the performance of a DB2 application. The overhead to collect performance trace data for these reports is very high and hundreds of thousands of trace records can be written in a minute or less for a single SQL statement. Therefore, the creation of SQL Activity reports requires special precaution.

Using the PE client, you can configure and start performance traces for a specific DB2 thread from the Thread Summary or Thread Details window in a flexible manner. You can specify the criteria for the data collection to stop and the SQL activity events you want to see in the SQL Activity report. When trace collection and report creation are finished, the SQL Activity report is downloaded and shown in a Web browser window. Such reports are downloaded as temporary files on the workstation and are deleted when you leave OMEGAMON XE for DB2 PE. To save these files, use the save functions provided by your Web browser.

Note: This functionality uses the PWH infrastructure to submit a CRD step and call the reporting functions. Therefore, a Performance Warehouse (PWH) must be configured and created for the monitored DB2 subsystem. The OMEGAMON Server and the PE Server instance associated with the DB2 subsystem must run with an activated PWH.

For more details, see:

- Section 4.4.3 of *IBM DB2 Performance Expert for z/OS Version 2,* SG24-6867-01
- *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Monitoring Performance from the Performance Expert Client,* SC18-9640.

13.4 How to use the report language

The generation of reports and other datasets can be controlled by commands and options in the SYSIN DD statement. These statement elements are part of a report language that describes which data is shown in the reports and how the data is represented. As shown in 13.3, “How to generate reports” on 231, there are various alternatives to create and submit a command stream.

The structure of the report commands in the SYSIN DD statement is listed in Example 13-4 .

Example 13-4 Sample SYSIN DD contents

```
  //SYSIN    DD  *
    GROUP/LIST commands
    GLOBAL
       subcommand
           option
    reportset
       subcommand
           option
       subcommand
           option
    reportset
       subcommand
           option
 EXEC
```

The concept of *OMEGAMON XE for DB2 PE identifiers* plays an important role in the report language. DB2 trace records contain identifiers that OMEGAMON XE for DB2 PE uses to group data, order reports, identify trace records, and to include or exclude specific data. These identifiers describe the object on which OMEGAMON XE for DB2 PE is reporting. For more details on OMEGAMON XE for DB2 PE identifiers, see Chapter 1 in the *BM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Report Command Reference,* SC18-9643.

To produce reports, you need to specify:

- GROUP and LIST commands that define a group of identifiers for INCLUDE/EXCLUDE
- One GLOBAL command with subcommands and options that filters data and sets defaults and is valid for all report sets in the command stream.
- A report set command for each report with subcommands for Reports/Traces/File/Save and various options
- EXEC as last command to start execution

You can reduce the amount of data with GLOBAL command and reporting variations:

- FROM/TO: This limits the amount of records to be processed by time.
- INCLUDE/EXCLUDE: This limits the amount of records to be processed for the report.
- TOP (accounting only): This shows top resources consumers.
- Exception processing: Events that have been identified as a potential problem.

Notes:

- In the *BM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Report Command Reference,* SC18-9643, the GLOBAL command is presented as *auxiliary* command. Because of its importance, it would be worth having a dedicated section for this command.
- For mapping the SET GLOBAL command of OMEGAMON for DB2 on z/OS Historical Reporter to the GLOBAL command of OMEGAMON XE for DB2, see Chapter 7, *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Getting Started,* GC18-9634.

For additional information, please refer to *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Report Command Reference,* SC18-9643.

Report set commands

The report set command specifies the type of report you want to create. Table 13-2 indicates the report set commands.

Table 13-2 Report set commands

Command	Report Set
ACCOUNTING	Accounting report set
STATISTICS	Statistics report set
EXPLAIN	Explain report set
SQLACTIVITY	SQL activity report set
UTILITY	Utility activity report set
LOCKING	Locking report set
IOACTIVITY	I/O activity report set
RECTRACE	Record trace report set
AUDIT	Audit report set
SYSPARMS	System parameters report set

Note: You can specify more than one report set command in a command stream. For example, you can create accounting and statistic reports in one execution. This is a very important performance feature, because the input dataset needs only to be read once by the reporting functions for multiple reports.

Report set subcommands

Use the following subcommands to specify how to you want the data to be presented:

- REDUCE: Use this command to aggregate statistics and accounting DB2 events. REDUCE consolidates DB2 events with the same DB2 PE identifiers (like plan name and authorization ID) into one. You may save the reduced using the SAVE command.
- REPORT: Use this command to generate reports. Use LAYOUT or LEVEL options to specify the amount of detail and ORDER to indicate how you want the data to be *summarized*. Use the EXCEPTION option to produce reports containing values outside user-specified limits.
- TRACE: Use this command to produce listings that show *individual DB2 events*, usually in the order of occurrence. Use the LAYOUT or LEVEL options of TRACE to specify the amount of detail. Use the EXCEPTION option to produce traces containing only values outside user-specified limits.
- FILE: Use this command to store data about individual DB2 events in sequential datasets that can be used with the DB2 load utility. Use the EXCEPTION option to produce datasets containing only values outside user-specified limits.
- SAVE: Use this command to save reduced data into VSAM datasets. You can combine it with new data to produce long-term reports. You can also convert the dataset into a sequential dataset that can be loaded into the Performance Database and/or in Performance Warehouse tables using the save-file utility.
- RESTORE: Use this command to include previously saved data.

13.5 How to use the report layouts

The LAYOUT subcommand is available for the accounting and the statistics report sets and specifies the name of a report layout to be applied in a report generation. A report layout determines which blocks of data and which fields within the blocks are included, and their relative order.

Report layouts can be tailored according to your requirements using the user-tailored reporting (UTR) feature. In previous versions of DB2 Performance Monitor and DB2 Performance Expert, two default layouts SHORT and LONG have been supplied for usage and tailoring. With OMEGAMON XE for DB2 PE, additional layouts are available that help you identify report data that was previously shown in the reports of the OMEGAMON Historical Reporter.

Table 13-3 shows the LAYOUT subcommand options of OMEGAMON XE for DB2 PE that you must specify in your JCL to generate data previously found in accounting reports of the OMEGAMON Historical Reporter.

Table 13-3 Finding information in OMEGAMON XE for DB2 PE Accounting reports

OMEGAMON Historical Reporter: Report subjects for ACCOUNTING report type	OMEGAMON XE for DB2 PE LAYOUT subcommand	ACCOUNTING trace or report	Other commands of OMEGAMON XE for DB2 PE
BUFFER_POOL_ACTIVITY	POOL	TRACE	
DETAIL	LONG	TRACE	Use the REPORT and/or TRACE subcommand of the SQLACTIVITY command
DISTRIBUTED_DATA_FACILITY_ ACTIVITY	DDF	TRACE	

OMEGAMON Historical Reporter: Report subjects for ACCOUNTING report type	**OMEGAMON XE for DB2 PE LAYOUT subcommand**	**ACCOUNTING trace or report**	**Other commands of OMEGAMON XE for DB2 PE**
GLOBAL_LOCK_ACTIVITY	LOCK	TRACE	
GROUP_BUFFER_POOL_ACTIVITY	POOL	TRACE	
LOCK_ACTIVITY	LOCK	TRACE	
PACKAGE_DETAIL	LONG PACK	TRACE	
PARALLEL_TASKS	PTA STP TIME	TRACE	
RESOURCE_LIMIT_FACILITY_ SUMMARY	RLF	REPORT	
SCAN_ACTIVITY		TRACE	Use the REPORT or TRACE subcommand of the SQLACTIVITY command
SORT_ACTIVITY		TRACE	Use the REPORT or TRACE subcommand of the SQLACTIVITY command
SQL_ACTIVITY	For DML Statements, DDL Statements, DCL Statements: SQL For Miscellaneous Statements: SQL For Miscellaneous Statements traces: SQL, PTA, TIME For Query Parallelism: PTA For Rid Pool Access: RID For Stored Procedure: STP	TRACE	
SUMMARY	LONG	REPORT	Use the REPORT or TRACE subcommand of the SQLACTIVITY command
TERMINATION_SUMMARY	PTA SQL TERM TIME	REPORT	
TIME_SUMMARY	PTA STP TIME	REPORT	

To create an accounting trace report for parallel tasks, you specify the LAYOUT command as shown in Example 13-5.

Example 13-5 Use LAYOUT command

```
//SYSIN    DD  *
ACCOUNTING
  TRACE
     LAYOUT(PTA)
```

The default layouts that OMEGAMON XE for DB2 PE provides for the accounting and statistics reports and traces should meet your requirements most of the time. However, if you have to create your own layouts, you can use the user-tailored reporting (UTR) feature. The UTR feature can be particularly useful if you want to remove fields from a report for which you have no responsibility, reduce the volume of data to highlight key fields, or provide more detail concerning particular aspects of the report. With the UTR feature you can:

- Add entire blocks and individual fields to an existing layout.
- Remove entire blocks and individual fields from an existing layout.
- Change the relative positions of blocks and fields in an existing layout.
- Change block and field labels.

The UTR feature can be invoked by option **5. Customize report and trace layouts** in the ISPF main menu. If you want to modify an existing layout for accounting reports, then specify a DPMPARMS dataset for storing the modified layout and select option **1. Accounting report**.

The panel in Example 13-15 below shows the default layouts provided for accounting reports (see also Figure 13-3 on page 242) which are available for user-tailoring.

```
  UTR Layout Selection              Row 1 to 13 of 13

  Report set  . . . . . : Accounting Report

  Select a Layout then press Enter.

      Layout  Description
      DDF     Distributed Data Facility
      LOCK    Locking
      LONG    Long
      PACK    Package
      POOL    Buffer Pools
      PTA     Parallel Task Activity
      RID     RID List
      RLF     Resource Limit Facility
      SHORT   Short
      SQL     SQL
      STP     Stored Procedures
      TERM    Application Termination
      TIME    CPU Times
 ******************************Bottomofdata********************************
  Command ===>
   F1=Help    F2=Split   F3=Exit    F7=Up      F8=Down    F9=Swap   F12=Cancel
```

Figure 13-15 Layouts for tailoring accounting reports

For more information, see:

- *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Getting Started,* GC18-9634
- *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Reporting User's Guide,* SC18-9641

14

Performance Database and Performance Warehouse

In this chapter we describe the Performance Database (PDB) and Performance Warehouse (PWH) components of OMEGAMON XE for DB2 PE that can hold raw and aggregated DB2 trace information spanning a long period of time. Long-term history data can help you in performance tuning activities.

If you have used DB2 Performance Monitor or DB2 Performance Expert, you know about its useful and flexible performance databases. Because of their comprehensiveness and their close relationship to the reporting functions, the former Performance Database and Performance Warehouse of DB2 Performance Expert have been chosen for the converged product, instead of OMEGAMON Historical Tables. Extractor and Summarizer components, which were strongly related to the Historical Tables, are not supported anymore.

This chapter explains the various functions of the Performance Database and Performance Warehouse components and shows how users of Historical Tables can migrate to the new databases.

We discuss the following topics:

- What data can be stored in the performance databases and how the Historical Tables can be mapped to the database tables in OMEGAMON XE for DB2 PE
- How database tables can be created and maintained
- How data can be loaded
- How data can be analyzed
- How users can benefit from both PDB and PWH.

14.1 Overview

In this section we give an overview of the Performance Database (PDB) and Performance Warehouse (PWH) components that provide repository functions for storing and analyzing performance data.

Long-term history data helps you evaluate the performance for a specified period of time. It is based on collected DB2 trace data which can be stored as raw or reduced information in the Performance Database or the Performance Warehouse database. The reporting functions are used to prepare the data to be loaded into the PDB or PWH. You can create reports or perform analysis on the data contained in the databases using SQL queries or predefined rules-of-thumb.

14.1.1 What is the Performance Database?

The Performance Database (PDB) is a DB2 database that can hold aggregated DB2 trace information spanning a long period of time. The performance data can come from the following data groups:

- Accounting
- Audit
- Locking
- Record traces (IFCIDs 22, 63, 96 and 125)
- Statistics and System parameters
- Batch, periodic, and display exceptions.

To help you build the Performance Database, OMEGAMON XE for DB2 PE provides the sample library RKO2SAMP with Data Definition Language (DDL) definitions (called C-parts). Additional meta data files are provided that describe the table columns (called B-parts). These files are in a format that allows theirs contents to be loaded into meta data tables for querying the table layouts.

Performance data to be loaded into the Performance Database and Performance Warehouse must be prepared by the reporting functions using the report language subcommands FILE and SAVE. The FILE subcommand generates sequential datasets that can be loaded to a database directly. The SAVE subcommand creates VSAM datasets that must be converted with save-file utilities to sequential datasets that are in loadable format. The sample library RKO2SAMP provides meta data files that describe the record layouts of the sequential datasets (so-called D-parts). These files are in a format that allows theirs contents to be loaded into meta data tables for querying the record layouts. And, DB2 load statements are available in RKO2SAMP that can be used with a LOAD utility to store the sequential dataset contents into the database tables (called L-parts).

Since the PDB is delivered via sample DDL statements, you can modify the tables to your needs. For example, you may want to focus on specific columns and therefore, you do not create all columns. Or, you may decide to introduce new columns which are derived from existing columns. If you make such changes, the LOAD parts must be modified accordingly. You must not change the load positions in a LOAD statement, since these are tightly connected to the format of the FILE and SAVE datasets written by the reporting functions.

For more detailed information on the Performance Database structure and the entities stored in the database tables see Chapter 19 of *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Reporting User's Guide,* SC18-9641.

14.1.2 What is the Performance Warehouse?

The Performance Warehouse (PWH) provides an infrastructure at the OMEGAMON Server and at the workstation to automate performance analysis tasks. It introduces the concept of processes which represent single or recurring tasks such as loading DB2 data into the Performance Warehouse or generating reports. The definition of processes and analysis tasks can be performed at the workstation via the Performance Warehouse graphical user interface, which is launched from the Performance Expert client.

The Performance Warehouse consists of DB2 tables to save the accounting and statistics performance counters which are the most relevant counters for analyzing performance problems. The tables are nearly identical to the tables in the Performance Database. It also consists of DB2 tables used by internal services. The Performance Warehouse provides a server component that automatically creates and maintains the DB2 tables.

The server component contains also a process engine which is responsible for executing the processes defined and scheduled at the workstation. The processes can consist of several steps that

- Collect trace data
- Create reports and generate load files for the load step
- Load data into the tables using the DB2 Load utility
- Combine any of the above steps.

Sample process templates are provided to help you get started. You can trace process executions from your workstation.

The Performance Warehouse supports additional activities at the Performance Expert client for:

- Collecting trace data for selected DB2 threads and creating SQL Activity reports
- Monitoring DB2 Connect gateways and applications.

Besides process-related functionality, the workstation also offers analysis support in form of rules-of-thumb and SQL performance queries. The provided rules-of-thumb (ROT) and SQL queries can be used to analyze the data stored in the accounting and statistics data. Rules-of-thumb represent expert rules that help you identify more complex performance problems and provide you with tuning recommendations. For example, you can combine several performance counters in a value expression and relate this expression to problem and warning thresholds. The provided ROT and the SQL queries can be adapted or you can create your own ROT and SQL queries by using table column wizards for your convenience. Sample rules-of-thumb and queries are provided to help you get started.

For more information, how to work with Performance Warehouse see Part 3 of *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Monitoring Performance from the Performance Expert Client,* SC18-9640.

Table 14-1 shows which report data (see also Chapter 13, “Reporting” on page 221) can be stored in the Performance Database and Performance Warehouse.

Table 14-1 Report Sets and where they can be stored

Report Set	PWH/PM Datasets	PM Database	Performance Warehouse
Accounting	Y	Y	Y
Audit	Y	Y	N

Report Set	PWH/PM Datasets	PM Database	Performance Warehouse
I/O Activity	Y	N	N
Locking	Y	Y	N
Record Trace	Y	Y	N
SQL Activity	Y	N	N
Statistics	Y	Y	Y
System Parameters	Y	Y	N
Utility Activity	Y	N	N

Note: The Performance Warehouse supports the accounting SAVE tables but not the accounting FILE tables.

14.2 Mapping Historical tables to Performance Database tables

This section shows how to map OMEGAMON Historical Tables to Performance Database tables.

14.2.1 Mapping tables

The former OMEGAMON provides Historical Tables for following categories, which are also supported by the Performance Database:

- Accounting Detail
- Accounting Summary
- Audit
- Statistics
- System Parameters

Data can be loaded into the Historical Tables with the Extractor and the Summarizer components. Table 14-2 below lists the database tables populated by the Extractor and correspondent tables in the Performance Database. This relationship holds, because the tables are filled from the same IFCID data sources (see Table 14-4 on page 252) and equivalent entities are stored in the database tables (see Table 14-5 on page 253).

Table 14-2 Extractor output tables

Type	Historical Table	OM XE PE Table
Accounting Detail	H2ACCT	DB2PMFACCT_GENERAL
	H2ACCTBP	DB2PMFACCT_BUFFER
	H2ACCTDDF	DB2PMFACCT_DDF
	H2ACCTGBP	DB2PMFACCT_GBUFFER
	H2ACCTPKG	DB2PMFACCT_PROGRAM

Accounting Sum	H2ACCT_SUM	DB2PMSACCT_GENERAL
	H2ACCTBP_SUM	DB2PMSACCT_BUFFER
	H2ACCTDDF_SUM	DB2PMSACCT_DDF
	H2ACCTGBP_SUM	DB2PMSACCT_GBUFFER
	H2ACCTPKG_SUM	DB2PMSACCT_PROGRAM
Audit	H2AUDIT_BIND	DB2PMFAUDT_BINDNR
	H2AUDIT_CHANGEID	DB2PMFAUDT_AUTHCHG
	H2AUDIT_CONTROL	DB2PMFAUDT_AUTHCTR
	H2AUDIT_DDL	DB2PMFAUDT_DDL
	H2AUDIT_DML	DB2PMFAUDT_DML
	H2AUDIT_FAIL	DB2PMFAUDT_AUTHFAI
	H2AUDIT_UTILITY	DB2PMFAUDT_UTILITY
Statistics	H2STATS	DB2PM_STAT_GENERAL
	H2STATBP	DB2PM_STAT_BUFFER
	H2STATDDF	DB2PM_STAT_DDF
	H2STATGBP	DB2PM_STAT_GBUFFER
System Parameter	H2PARAM	DB2PMSYSPAR_106

Table 14-3 lists the database tables populated by the Summarizer and correspondent tables in the Performance Database. Summarization functionality in OMEGAMON XE for DB2 PE is accomplished by the reporting functions. For the accounting and statistics domain, the reporting functions provide the REDUCE/SAVE subcommands which reduce and aggregate the performance data to a large extent, and create SAVE datasets. SAVE datasets can be converted and loaded to the database tables for subsequent analysis. Using self-defined SQL queries, the data in the PDB can be further summarized.

Table 14-3 Summarizer output tables

Type	Historical Table	OM XE PE Table
Accounting	H2ACCT_SUM*	DB2PMSACCT_GENERAL
	H2ACCTBP_SUM*	DB2PMSACCT_BUFFER
	H2ACCTDDF_SUM*	DB2PMSACCT_DDF
	H2ACCTGBP_SUM*	DB2PMSACCT_GBUFFER
	H2ACCTPKG_SUM*	DB2PMSACCT_PROGRAM
Statistics	H2STAT_SUM*	DB2PM_STAT_GENERAL
	H2STATBP_SUM*	DB2PM_STAT_BUFFER
	H2STATDDF_SUM*	DB2PM_STAT_DDF
	H2STATGBP_SUM*	DB2PM_STAT_GBUFFER
Note: The asterisk (*) denotes HOUR, DAY, WEEK, and MONTH tables.		

Note: The Performance Database supports additional tables.

Table 14-4 shows that Historical Tables and the correspondent Performance Database tables contain mostly data from the same IFCIDs. The only exception holds for tables H2ACCT and H2ACCT_SUM which could store data from IFCID 18, 95, 96. This is also reflected in the accounting reports in OMEGAMON XE for DB2 PE which do not contain data from these IFCIDs. These IFCIDs are covered in SQL Activity reports.

The Historical Tables have not been updated since DB2 V4, while the PDB tables have been maintained in sync up to DB2 V8. This explains why most PDB tables have more columns than the Historical Tables.

Table 14-4 IFCID sources and number of columns

Historical Table	IFCID	#C	OM XE PE Table	IFCID	#C
H2ACCT	3,18, 95,96	191	DB2PMFACCT_GENERAL	3	314
H2ACCTBP	3	24	DB2PMFACCT_BUFFER	3	29
H2ACCTDDF	3	56	DB2PMFACCT_DDF	3	64
H2ACCTGBP	3	17	DB2PMFACCT_GBUFFER	3	37
H2ACCTPKG	3,239	43	DB2PMFACCT_PROGRAM	3,239	104
H2ACCT_SUM	3,18, 95,96	249	DB2PMSACCT_GENERAL	3	303
H2ACCTBP_SUM	3	26	DB2PMSACCT_BUFFER	3	35
H2ACCTDDF_SUM	3	59	DB2PMSACCT_DDF	3	72
H2ACCTGBP_SUM	3	19	DB2PMSACCT_GBUFFER	3	43
H2ACCTPKG_SUM	3,239	46	DB2PMSACCT_PROGRAM	3,239	115
H2AUDIT_BIND	145	32	DB2PMFAUDT_BINDNR	145	31
H2AUDIT_CHANGEID	55,83, 87,169	28	DB2PMFAUDT_AUTHCHG	55,83, 87,169	34
H2AUDIT_CONTROL	141	31	DB2PMFAUDT_AUTHCTR	141	29
H2AUDIT_DDL	142	32	DB2PMFAUDT_DDL	142	31
H2AUDIT_DML	143, 144	29	DB2PMFAUDT_DML	143,144	30
H2AUDIT_FAIL	140	33	DB2PMFAUDT_AUTHFAI	140	32
H2AUDIT_UTILITY	23-25, 105,107	21	DB2PMFAUDT_UTILITY	24,105, 107	31
H2STATS	1,2	268	DB2PM_STAT_GENERAL	1,2, 202, 225	463
H2STATBP	1,2, 202	80	DB2PM_STAT_BUFFER	1,2	70
H2STATDDF	1,2	45	DB2PM_STAT_DDF	1,2	50

Historical Table	IFCID	#C	OM XE PE Table	IFCID	#C
H2STATGBP	1,2, 230	35	DB2PM_STAT_GBUFFER	1,2	65
H2PARAM	106	130	DB2PMSYSPAR_106	106	227

14.2.2 Mapping columns

The former OMEGAMON product provides an access to a data dictionary through the Historical Main Menu. This data dictionary describes the Historical Tables. It includes the description of each column, the column's data type and information from which IFCID field the column values are derived. Column sources starting with character Q indicates that a DB2 field is stored in the columns, sources starting with character Z represents values calculated from other values.

Meta data about the Performance Database tables is provided in the so-called B-parts in dataset RKO2SAMP. This data can be loaded into meta data tables for querying the table layout. The B-parts also contain column descriptions and refer to the fields which are stored in the columns. The meta data in the Historical Tables data dictionary and the B-parts can be syntactically joined on the fields, the column values are derived from. This results in a table, which shows the columns that have identical data sources and are related to each other. Other columns can be related by a semantic analysis of their descriptions. And, there are columns that have no counterparts.

Table 14-5 shows a partial mapping of columns in tables H2ACCT and DB2PM_FACCT_GENERAL. The entries in these accounting database tables represent detail performance data about DB2 threads.

Table 14-5 Mapping columns from H2ACCT to DB2PMFACCT_GENERAL table

Historical table H2ACCT	H2 field name	OM XE PE table DB2PMFACCT_GENERAL	IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS field name
SYSID	ZWHMVSID	n/a	
DB2ID	QWHSSSID	SUBSYSTEM_ID	QWHSSSID
GROUP_NAME	QWHADSGN	GROUP_NAME	QWHADSGN
MEMBER_NAME	QWHAMEMN	MEMBER_NAME	QWHAMEMN
DB2_VERSION	ZWHSRN	DB2_REL	DB2REL
D2_LEVEL	ZWD2LEV	DB2PM_REL	DB2PMREL
START_TIMESTAMP	QWHSSTCK	CLASS1_TIME_BEG	QWACBSC
START_DATE	ZWHSDATE		
START_TIME	ZWHSTIME		
START_MONTH	ZWHSMNTH		
START_HOUR	ZWHSHOUR		
END_TIMESTAMP	ZWHESTCK	CLASS1_TIME_END	QWACESC

Historical table H2ACCT	H2 field name	OM XE PE table DB2PMFACCT_GENERAL	IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS field name
PLANNAME	QWHCPLAN	PLAN_NAME	QWHCPLAN
AUTHID	QWHCAID	PRIMAUTH	QWHCAID
ORIGINAL_AUTHID	QWHCOPID	ORIGAUTH	QWHCOPID
CONNID	QWHCCN	CONNECT_ID	QWHCCN
CORRID	QWHCCV	CORRNAME	ADCORNME
		CORRNUMBER	ADCORNMB
LOCATION_NAME	QWHSLOCN	LOCAL_LOCATION	QWHSLOCN
CONNECT_TYPE	QWHCATYP	CONNECT_TYPE	QWHCATYP
CICS_TOKEN	QWHCTOKN	CICS_NET_ID	ADCICSNI
		CICS_LUNAME	ADCICSLU
		CICS_INSTANCE_NBR	ADCICSIN
DDF_NETWORK_ID	QWHDNETI	NET_ID	QWHSNID
DDF_LU_NAME	QWHDLUMN	LUNAME	QWHSLUNM
DDF_INSTANCE	QWHDUNIQ	INSTANCE_NBR	QWHSLUUV
DDF_COMMIT_COUNT	QWHDCCNT	LUW_SEQNO	QWHSLUCC
DDF_REQ_LOCATION	QWHDRQNM	REQ_LOCATION	QWHDRQNM
DDF_SERVER	QWHDSVNM		
DDF_PRODUCT_ID	QWHDPRID	REQ_PRODUCT_ID	QWHDPRID
		TIMESTAMP	QWHSSTCK
		MAINPACK	ADMAINPK
		WLM_SERVICE_CLASS	QWACWLME
		CLIENT_ENDUSER	QWHCEUID
		CLIENT_WSNAME	QWHCEUWN
		CLIENT_TRANSACTION	QWHCEUTX
COMMIT	QWACCOMM	COMMIT	QWACCOMM
ABORT	QWACABRT	ROLLBACK	QWACABRT
ELAPSED_TIME	ZWACTSC	CLASS1_ELAPSED	ADRECETT
TCBTIME	ZWACTJST	CLASS1_CPU_TOTAL	
STPR_TCBTIME	QWACSPCP	CLASS1_CPU_STPROC	QWACSPCP
SRBTIME	ZWACTSRB		
IN_DB2_TIME	QWACASC	CLASS2_EL_NONNEST	QWACASC

We describe how PDB data can be analyzed by an SQL query at 14.5.1, “Analyze data in the Performance Database” on page 269. We provide a sample that translates a Historical Tables query (delivered with former OMEGAMON) into a PDB query. This sample is also very useful for mapping columns of Historical Tables to PDB tables.

14.3 How to create and maintain the database tables

The Performance Database tables can be created using the sample CREATE statements shipped in the RKO2SAMP library. You can customize the table definitions to your needs. If new columns are added to the tables during the life cycle of OMEGAMON XE for DB2 PE, then ALTER statements are provided in the same library.

The OMEGAMON Server creates and maintains the Performance Warehouse tables automatically in a DB2 subsystem which is configured and activated in the ICAT installation.

14.3.1 Create a Performance Database for accounting and statistics tables

In this section we illustrates how a database for accounting and statistics performance data can be created. We use and modify the sample parts provided in the RKO2SAMP library.

Example 14-1 shows the statements to create the database and the table spaces for each accounting and statistics table. We have chosen to create the accounting SAVE tables which will hold aggregated performance data of DB2 applications. And, we have chosen to create the statistics tables for storing system-wide statistics information. Each table resides in its own table space.

Example 14-1 Create database and table spaces

```
-- create database
CREATE DATABASE SC63PDB ;

-- create table spaces for Accounting SAVE tables
CREATE TABLESPACE TSPASBU IN SC63PDB BUFFERPOOL BP1 ;
CREATE TABLESPACE TSPASDF IN SC63PDB BUFFERPOOL BP1 ;
CREATE TABLESPACE TSPASGE IN SC63PDB BUFFERPOOL BP1 ;
CREATE TABLESPACE TSPASGP IN SC63PDB BUFFERPOOL BP1 ;
CREATE TABLESPACE TSPASPK IN SC63PDB BUFFERPOOL BP1 ;
CREATE TABLESPACE TSPASRF IN SC63PDB BUFFERPOOL BP1 ;

-- create table spaces for Statistics tables
CREATE TABLESPACE TSPSBUF IN SC63PDB BUFFERPOOL BP1 ;
CREATE TABLESPACE TSPSDDF IN SC63PDB BUFFERPOOL BP1 ;
CREATE TABLESPACE TSPSGBP IN SC63PDB BUFFERPOOL BP1 ;
CREATE TABLESPACE TSPSGEN IN SC63PDB BUFFERPOOL BP1 ;
CREATE TABLESPACE TSPSSET IN SC63PDB BUFFERPOOL BP1 ;
```

When the data base and table spaces are available, the sample CREATE TABLE statements (C-parts) can be adapted to the previous definitions and afterwards be executed via SPUFI. Therefore, you should copy the members DGOACS* and DGOSC* from the runtime library RKO2SAMP to a private RKO2SAMP dataset and modify the table space names (see Table 14-6).

Table 14-6 CREATE TABLE statements

C-part in RKO2SAMP	Table name	Create in table space
DGOACSBU	DB2PMSACCT_BUFFER	SC63PDB.TSPASBU

C-part in RKO2SAMP	Table name	Create in table space
DGOACSDF	DB2PMSACCT_DDF	SC63PDB.TSPASDF
DGOACSGE	DB2PMSACCT_GENERAL	SC63PDB.TSPASGE
DGOACSGP	DB2PMSACCT_GBUFFER	SC63PDB.TSPASGP
DGOACSPK	DB2PMSACCT_PROGRAM	SC63PDB.TSPASPK
DGOACSRF	DB2PMSACCT_RLF	SC63PDB.TSPASRF
DGOSCBUF	DB2PM_STAT_BUFFER	SC63PDB.TSPSBUF
DGOSCDDF	DB2PM_STAT_DDF	SC63PDB.TSPSDDF
DGOSCGBP	DB2PM_STAT_GBUFFER	SC63PDB.TSPSGBP
DGOSCGEN	DB2PM_STAT_GENERAL	SC63PDB.TSPSGEN
DGOSCSET	DB2PM_STAT_DATASET	SC63PDB.TSPSSET

Note: The statistics tables can hold FILE and SAVE data. Depending on your needs for storing both type of data, you may decide to create separate statistics FILE and SAVE tables as these are already defined for accounting data. You may then use similar naming conventions:

- TSPSF* = table space name for statistics FILE table
- TSPSS* = table space name for statistics SAVE table
- DB2PMFSTAT_* = table name for statistics FILE table
- DB2PMSSTAT_* = table name for statistics SAVE table.

In OMEGAMON XE for DB2 PE, new columns have been added to the accounting package tables DB2PMFACCT_PROGRAM and DB2PMSACCT_PROGRAM to provide enhanced support for the package level accounting fields in IFCIDs 3 and 239 (see also Chapter 15.3, “Package level accounting support in PDB/PWH”) during the life cycle of OMEGAMON XE for DB2 PE, the data model of the Performance Database may also change. To support the migration and maintenance of database tables, members DGO*UPDB are shipped in the RKO2SAMP library.

Example 14-2 shows some ALTER statements in member DGOAUPDB. The brackets for the keyword COLUMN indicate, that the keyword is needed in a DB2 V8 subsystem but not in a DB2 V7 subsystem.

Example 14-2 ALTER statements to add new columns

```
ALTER TABLE DB2PMSACCT_PROGRAM ADD [COLUMN] SELECT
            DECIMAL(15,0) NOT NULL WITH DEFAULT;
ALTER TABLE DB2PMSACCT_PROGRAM ADD [COLUMN] INSERT
            DECIMAL(15,0) NOT NULL WITH DEFAULT;
ALTER TABLE DB2PMSACCT_PROGRAM ADD [COLUMN] UPDATE
            DECIMAL(15,0) NOT NULL WITH DEFAULT;

ALTER TABLE DB2PMFACCT_PROGRAM ADD [COLUMN] SELECT
            INTEGER NOT NULL WITH DEFAULT;
ALTER TABLE DB2PMFACCT_PROGRAM ADD [COLUMN] INSERT
            INTEGER NOT NULL WITH DEFAULT;
ALTER TABLE DB2PMFACCT_PROGRAM ADD [COLUMN] UPDATE
            INTEGER NOT NULL WITH DEFAULT;
```

14.3.2 Create a Performance Warehouse

The OMEGAMON Server contains besides other subtasks Performance Expert Server (PE Server) instances. Each instance is assigned to a DB2 subsystem and provides data collection and monitoring services to the OMEGAMON Server and the user interfaces. A PE Server instance runs as a subtask in the address space of the OMEGAMON Server.

A PE Server creates and maintains a Performance Warehouse in the associated DB2 subsystem if this functionality is selected during the ICAT configuration of the DB2 subsystem. Figure 14-1shows the panel where the Performance Warehouse of subsystem D8F2 is activated.

The Performance Warehouse is composed of several table spaces, tables, indexes and views. The tables can be categorized as

- Performance data tables for accounting SAVE and statistics SAVE and FILE data. Table names are identical to the PDB table names used in the C-parts in library RKO2SAMP.
- Internal tables for the process engine and infrastructure.

In the ICAT panel of Figure 14-1 you can also specify the storage pool and buffer pools of the PWH tables.

```
OMEGAMON XE for DB2 PE Performance Warehouse configuration SSID D8F2
Command ===>

 Configure Performance Warehouse.

   Activate Performance Warehouse   ==> Y  (Y, N)
   Performance Warehouse job name   ==> D8F2PWH_

   Specify storage group and buffer pool to be used for PWH database:
                   Storage Group         Buffer Pool
     PROCESS          SYSDEFLT              _____BP0
     OUTPUT           SYSDEFLT              _____BP0
     CONTROL          SYSDEFLT              _____BP0
     ROT              SYSDEFLT              _____BP0
     ONLINE           SYSDEFLT              ___BP32K
     STAT             SYSDEFLT              _____BP0
     QRY              SYSDEFLT              ___BP32K
     ACCS             SYSDEFLT              _____BP0

   Specify storage group and buffer pool that are used for the
   PWH index space:

     Storage group  ==> SYSDEFLT     Buffer pool  ==> _____BP0
 Enter=Next  F1=Help  F3=Back
```

Figure 14-1 Activate PWH for a DB2 subsystem

PWH processes are executed by JCL jobs. In the definition panel, you can specify the corresponding job name, here D8F2PWH. ICAT creates a started job JCL D8F2PWH in library RDK2SAM.

To complete the PWH configuration, you have to:

1. Copy the Performance Warehouse started Job JCL{D8F2PWH} to your PROCLIB.
2. Edit the JCL and:
   - Update{jobacc}with your job accounting information.
   - Update{jobprogrammer}with the job owner name information.
3. Grant DB2 privileges to the Performance Warehouse RACF GROUPID/USERID{DB2PM} by running member {RKD2SAM(PWG1D8F2).}

The DB2 objects are now automatically created and maintained by the Performance Expert Server the first time it is started as subtask of the OMEGAMON Server. See a partial list of messages in the sample joblog in Figure 14-2.

```
..........................................................
FPEV5009I  D8F2 Creating new table PROCESSGROUP ...
FPEV5009I  D8F2  ... creating primary index
FPEV5009I  D8F2  ... creating unique indexes
FPEV5009I  D8F2  ... setting comments
FPEV5009I  D8F2 Creating new table PROCESS ...
FPEV5009I  D8F2  ... creating primary index
FPEV5009I  D8F2  ... creating unique indexes
FPEV5009I  D8F2  ... setting comments
FPEV5009I  D8F2 Creating new table STEP ...
FPEV5009I  D8F2  ... creating primary index
FPEV5009I  D8F2  ... creating unique indexes
FPEV5009I  D8F2  ... setting comments
.........................................................
FPEV5009I  D8F2 Creating new table DB2PM_STAT_DATASET ...
FPEV5009I  D8F2  ... creating primary index
FPEV5009I  D8F2  ... creating unique indexes
FPEV5009I  D8F2  ... setting comments
FPEV5009I  D8F2 Creating new table DB2PM_STAT_GBUFFER ...
FPEV5009I  D8F2  ... creating primary index
FPEV5009I  D8F2  ... creating unique indexes
FPEV5009I  D8F2  ... setting comments
..........................................................
```

Figure 14-2 Create Performance Warehouse tables

Once the tables have been created by the PE Server, you can set up the security for the Performance Warehouse users by editing and running member {RKD2SAM(PWG2D8F2).}

OMEGAMON XE for DB2 PE provides the maintenance of these databases, table spaces, tables, and indexes, but does not provide you with the automatic capability to image copy, reorg, and run runstats against these objects. You have to prepare and schedule these maintenance utilities if you determine that these tables are frequently used and especially if the data volume increases. In addition, OMEGAMON XE for DB2 PE does not estimate or maintain the size of the table spaces used by the PWH. We recommend you monitor these DB2 objects and run the appropriate utilities as you would for any other DB2 objects.

Depending on your needs, you can have a one or many Performance Warehouses in your monitored environment.

Figure 14-3 shows how the Performance Warehouses might be configured in a typical environment.

This scenario illustrates an environment that includes:

- Two z/OS systems LPAR A and LPAR B in a parallel sysplex environment. LPAR A represents a production system and LPAR B provides members for the data sharing group and a stand-alone DB2 subsystem DBB1, that is used for storing performance data.
- For each LPAR, there is one OMEGAMON Server running Performance Expert Server instances and other monitoring subtasks. In a data-sharing environment, you may have one PE server instance for the entire data-sharing group. For availability, you might design two PE server instances in each LPAR (as illustrated in the diagram) in case of a z/OS system failure.
- Each PE server instance on LPAR A runs a Performance Warehouse, because you may want to collect trace data regularly for peak times and store the data in sequential datasets. Or, in case of performance problems on the production system, you may need to react immediately and create SQL Activity reports.
- In a data-sharing group, only one Performance Expert Server instance may run and maintain a Performance Warehouse. Therefore, on LPAR B only one Performance Warehouse is defined for DB2 subsystem DBB1. You may want to use this Performance Warehouse as repository for storing and analyzing the accounting and statistics performance data collected with CRD on LPAR A.
- One Performance Expert client where you can
  - Define the CRD steps to collect regularly performance data for LPAR A
  - Create immediate SQL Activity reports for LPAR A
  - Define processes that regularly load the CRD data into the Performance Warehouse on LPAR B and create reports to be viewed at the workstation
  - Analyze the loaded data on LPAR B using rules-of-thumbs or SQL queries.

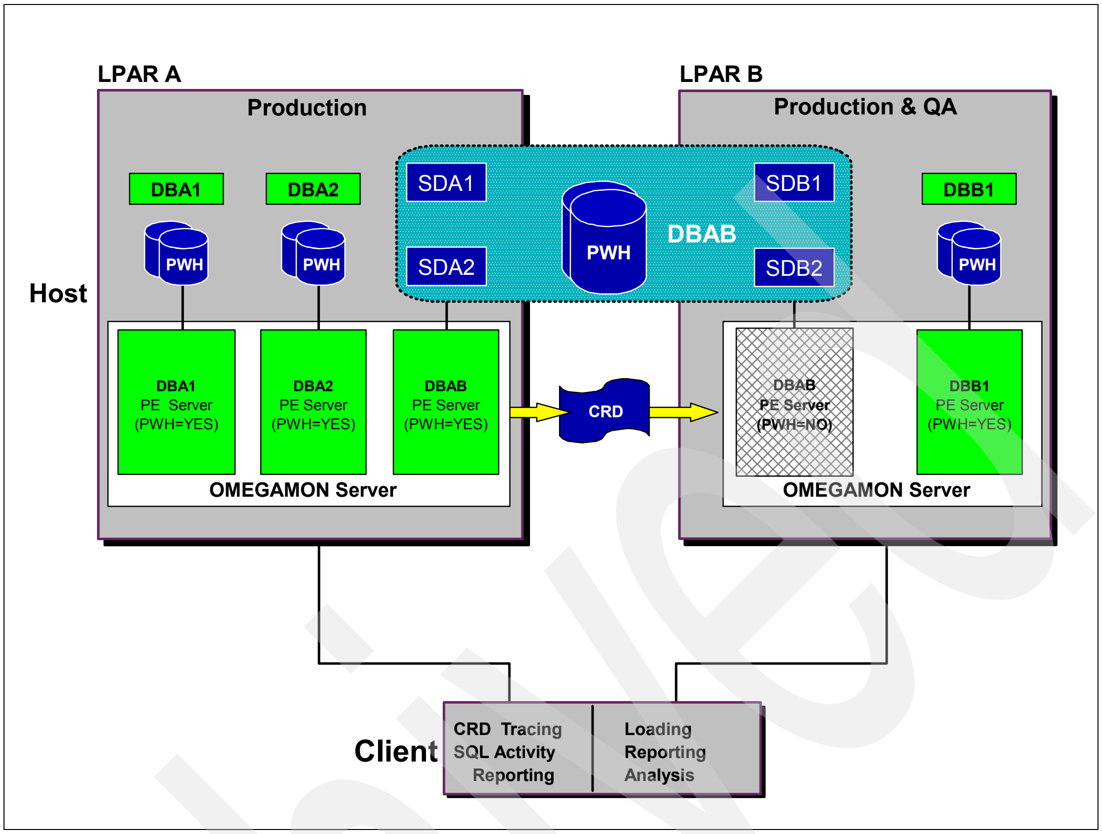


Figure 14-3 Performance Warehouse typical environment

14.4 How to load data into the tables

Performance data can be loaded into the Performance Database tables using the LOAD statements in the RKO2SAMP library together with a DB2 load utility. The data must be contained in FILE and converted SAVE datasets that have been created by the reporting functions.

In the Performance Warehouse, processes which contain load steps, perform the task of loading data into the warehouse tables. As for the Performance Database, the input data for the load step must have been processed by the reporting functions.

14.4.1 Load data into the Performance Database

In 13.1, "Overview of the reporting functions" on page 222, the flow of DB2 trace data (SMF, GTF or CRD) to the Performance Database is displayed. The data must be prepared by the reporting functions for subsequent load steps. There are two subcommands FILE and SAVE in the report language to control the generation of loadable data. Figure 14-4 refines the overview diagram and shows which report sets can be stored in FILE and SAVE tables in the Performance Database.

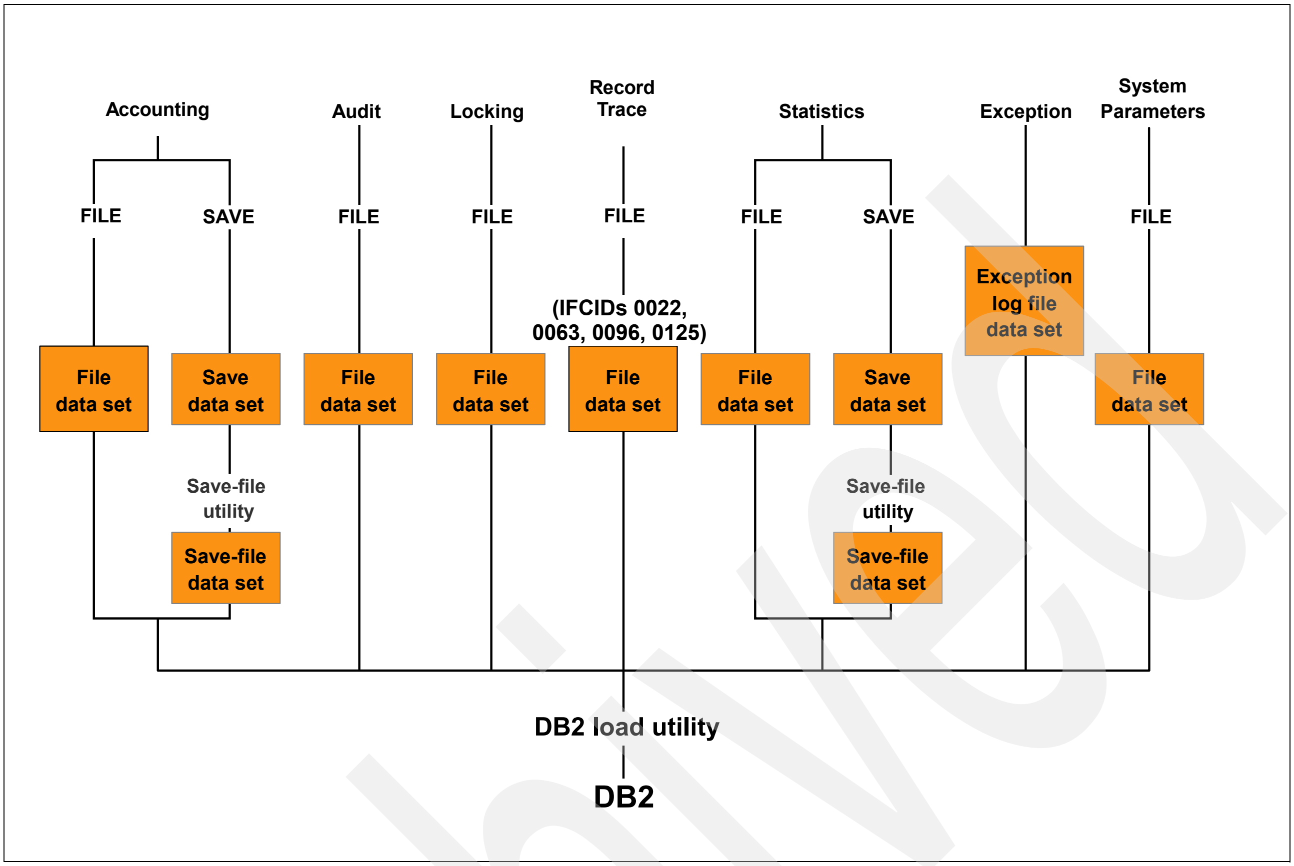


Figure 14-4 Logical flow of loading trace data into the Performance Database

The LOAD statements for both FILE and SAVE tables in the Performance Database are available in the RKO2SAMP library. If you have customized the CREATE TABLE statements in the C-parts to your needs, the L-parts must be updated accordingly. For example, if you have deleted columns in a table definition, then the columns must be also deleted in the related LOAD statement.

FILE tables

Use the FILE subcommand to store data about individual DB2 events in sequential datasets that can be used with the DB2 load utility. The granularity of this data is comparable to the data in the report listings which you can produce by specifying the TRACE option. FILE tables are available for following report sets:

- ACCOUNTING
- AUDIT
- EXCEPTION
- LOCKING
- RECTRACE
- STATISTICS
- SYSPARMS

Example 14-3 illustrates the commands to generate an accounting TRACE report filtered by the location DB8F and to write corresponding FILE data to DDNAME ACFILEDD1. The output dataset associated with ACFILDD1 is a sequential dataset.

Example 14-3 FILE subcommand for accounting data

```
//ACFILDD1 DD  DSN=PAOLOR2.REPORT.ACFILE,DISP=SHR
//ACRPTDD1 DD  SYSOUT=*
//SYSIN    DD  *
GLOBAL
INCLUDE (LOCATION(DB8F))
*
ACCOUNTING
  TRACE
     LAYOUT(LONG)
     DDNAME (ACRPTDD1)
  FILE DDNAME(ACFILDD1)
*
EXEC
```

The Performance Database SC63PDB as defined before contains only SAVE tables. In order to hold Accounting FILE data, table spaces and tables must be created using C-parts DGOACF* in library RKO2SAMP.

To load the Accounting FILE data into the newly created tables, you can use the L-parts DGOALF* in dataset RKO2SAMP. Be aware that the members contain the REPLACE clause for the LOAD control statement. You must change this clause to RESUME YES in order not to delete the data already loaded.

Example 14-4 shows a sample load job to load general accounting data into the Performance Database. SYSREC points to the FILE dataset, SYSIN refers to the LOAD statement dataset.

Example 14-4 Sample load job for accounting data from FILE dataset

```
//PAOLORLD JOB (999,POK),'DB2 LD',CLASS=A,MSGCLASS=T,
// NOTIFY=&SYSUID,TIME=1440,REGION=0M
/*JOBPARM SYSAFF=SC63
//*
//UTIL EXEC PGM=DSNUTILB,
//          PARM='DB8A,PAOLOAD1,'
//STEPLIB  DD  DISP=SHR,DSN=DB8A8.SDSNEXIT
//         DD  DISP=SHR,DSN=DB8A8.SDSNLOAD
//*
//SYSUDUMP DD  DUMMY
//UTPRINT  DD  SYSOUT=*
//SYSPRINT DD  SYSOUT=*
//SYSUT1 DD UNIT=SYSDA,SPACE=(4000,(20,20),,,ROUND)
//SORTOUT DD UNIT=SYSDA,SPACE=(4000,(20,20),,,ROUND)
//SYSERR DD UNIT=SYSDA,SPACE=(4000,(20,20),,,ROUND)
//SYSDISC DD UNIT=SYSDA,SPACE=(4000,(20,20),,,ROUND)
//SYSMAP DD UNIT=SYSDA,SPACE=(4000,(20,20),,,ROUND)
//*
//SYSREC DD DSN=PAOLOR2.REPORT.ACFILE,DISP=SHR
//SYSIN  DD DSN=PAOLOR2.PDB.RKO2SAMP(DGOALFGE),DISP=SHR
```

SAVE tables

Use the SAVE subcommand to save aggregated data into VSAM datasets (see also 14.4.3, “Aggregate trace data using REDUCE, SAVE and GROUP REDUCE”). The data in the VSAM datasets is usually more granular than the data in the report listings. This is due to the fact that you can use the ORDER option in the REPORT subcommand to further aggregate the data in the report listing.

You must convert the VSAM dataset into a sequential dataset using the save-file utility before the data can be loaded into the Performance Database. SAVE tables are available for following report sets:

- ACCOUNTING
- STATISTICS

Example 14-5 illustrates the commands to generate an accounting REPORT report using INCLUDE and REDUCE to reduce the data, which is further summarized by the ORDER(PLANNAME-PRIMAUTH) option in the report listing. The SAVE subcommand is used to store the reduced data (which is more granular than the ORDERed report data) to a VSAM dataset associated with DDNAME ACSAVDD1.

Example 14-5 SAVE subcommand for accounting data

```
//ACRPTDD1 DD  SYSOUT=*
//ACSAVDD1 DD  DSN=PAOLOR2.REPORT.ACSAV1,DISP=SHR
//SYSIN    DD  *
GLOBAL
INCLUDE (LOCATION(DB8F))
*
ACCOUNTING
  REDUCE
     FROM(02/09/06,18:30:00)
     TO(02/09/06,20:30:00)
     INTERVAL(60)
  REPORT
     SCOPE(MEMBER)
     LAYOUT(LONG)
     ORDER(PLANNAME-PRIMAUTH)
     DDNAME (ACRPTDD)
  SAVE DDNAME(ACSAVDD1)
*
EXEC
```

SAVE data which is stored in VSAM format, must be converted to sequential format compatible with the DB2 LOAD utility. The accounting and statistics save-file utilities (see sample jobs DGOPJACO and DGOPJSCO in the RKO2SAMP library) can be used for that purpose. Example 14-6 shows a sample convert job using the template of the save-file utility DGOPJACO.

Example 14-6 Convert SAVE dataset to sequential format

```
//PAOACONV JOB (999,POK),'CONVERT',CLASS=A,MSGCLASS=T,
// NOTIFY=&SYSUID,TIME=1440,REGION=0M
/*JOBPARM SYSAFF=SC63
//*
//CONVERT EXEC PGM=DGOPMICO,PARM=CONVERT
//STEPLIB  DD  DSN=OMEGASYS.DB8A.SC63RTE.RKANMOD,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//INPUT    DD  DSN=PAOLOR2.REPORT.ACSAV1,DISP=OLD
//OUTPUT   DD  DSN=PAOLOR2.REPORT.ACLOAD,DISP=(NEW,CATLG,DELETE),
//             UNIT=SYSDA,SPACE=(TRK,(15,15),RLSE),
//             DCB=(RECFM=VB,LRECL=4092,BLKSIZE=4096)
```

The sequential dataset can now be loaded to the accounting SAVE tables. Use a load job as in Example 14-4 on page 262 in which SYSREC points to PAOLOR2.REPORT.ACLOAD and SYSIN refers to the LOAD parts DGOALS* in your sample library.

Note: Use the GROUP BY clause of SQL to summarize data in the Performance Database comparable to the ORDER option in the report generation.

14.4.2 Load data into the Performance Warehouse

In Chapter 13, “Reporting” on page 221 we explain how Performance Warehouse processes can be configured for collecting trace data and creating reports which can be viewed from the workstation client. Such processes can be extended seamlessly by additional steps that:

- Convert SAVE datasets into load sequential datasets using the save-file utility
- Load sequential datasets (either FILE or converted SAVE datasets) into the Performance Warehouse.

Figure 14-5 illustrate all steps that can be combined in a PWH process.

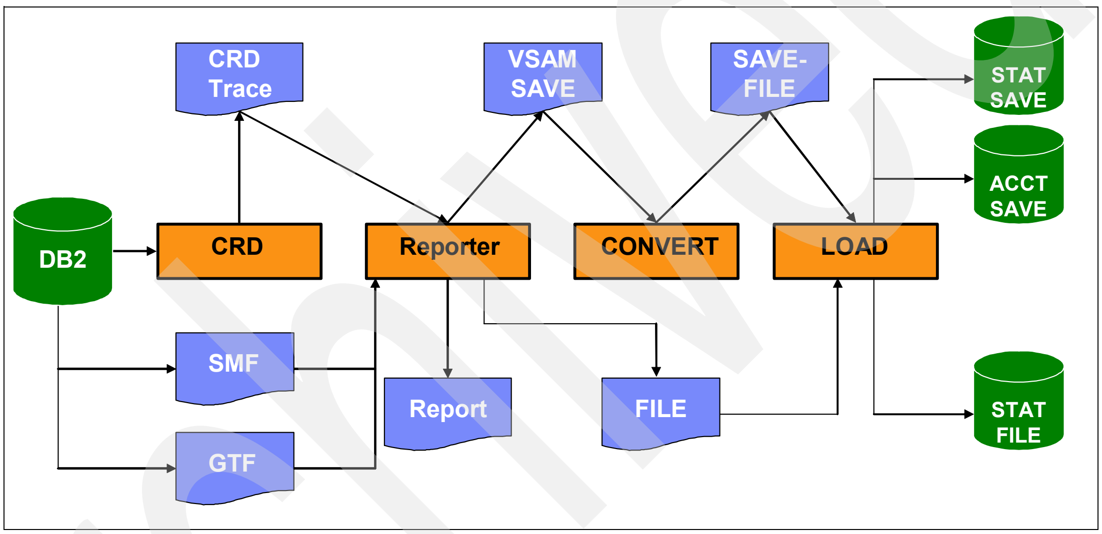


Figure 14-5 PWH processes including LOAD steps

Figure 14-6 shows the definition of a report step containing a SAVE and LOAD SAVE option.

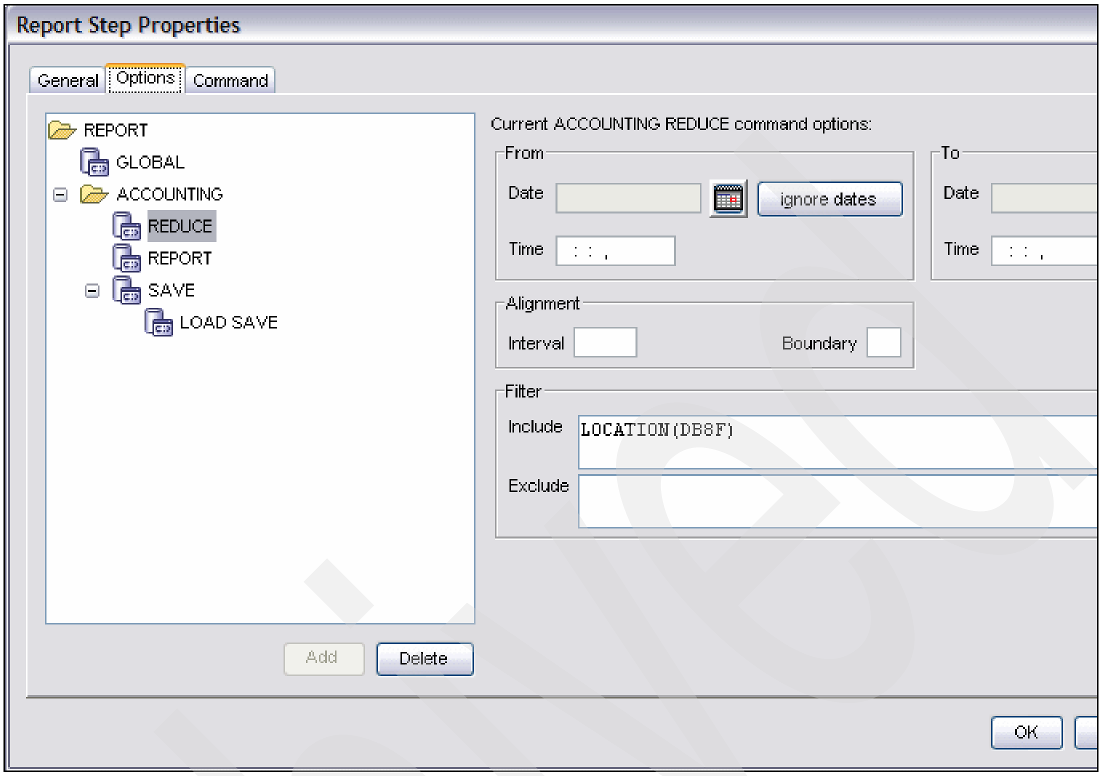


Figure 14-6 Report step containing a LOAD SAVE option

Figure 14-7 shows an execution of the process, which contains the report step from above. This step is preceded by a CRD step, the LOAD SAVE option triggers the execution of the save-file utility, and the collected trace data is loaded into the SAVE tables.

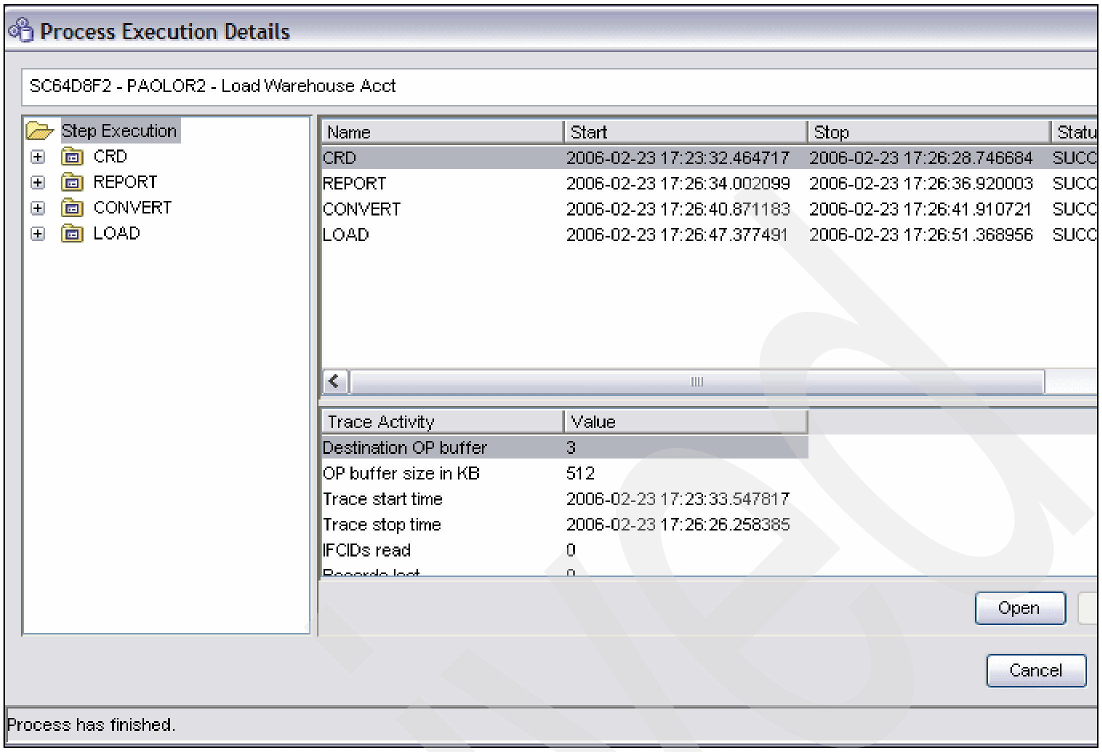


Figure 14-7 Process execution invoking the LOAD utility

In the Output Data Set folder of the LOAD step execution, you can look at the SYSPRINT and find out, how many rows have been loaded to the various SAVE tables.

14.4.3 Aggregate trace data using REDUCE, SAVE and GROUP REDUCE

The ACCOUNTING and STATISTICS commands provide the REDUCE subcommand which is a powerful means to filter and aggregate DB2 trace data. The reduced data can be further aggregated or sorted during report generation or can be stored into SAVE datasets. SAVE datasets can be converted and loaded into the SAVE tables in the Performance Database or Performance Warehouse for subsequent analysis.

Statistics

In the statistics domain, the REDUCE subcommand can be used with the INTERVAL option to accumulate and apportion DB2 statistics data from IFCIDs 1, 2, 199 and 225 into intervals.

When a DB2 statistics trace is active, DB2 maintains various statistics counters and externalizes them at regular intervals usually every 15 or 30 minutes depending on the system configuration. The counters provided in DB2 statistics IFCID records represent the DB2 activity between the time the record is externalized and the time the DB2 system was last started.

A counter is given in one of the following forms:

- As an accumulated value since the DB2 system was last started. For example, the total number of SELECT statements executed since the system was last started.
- As a current or snapshot value. For example, the number of open datasets at the time the DB2 statistics records pair was externalized.

- As a maximum or high water mark value the counter has reached since the time the system was last started. For example, the maximum number of open datasets at any time since the system was last started.

For externalizing the counters in statistics TRACE reports or FILE datasets, the reporting functions take consecutive DB2 IFCID records and depending on the type of a counter, it calculates deltas, or selects the current snapshot or high-water mark value. The values are stored in so-called *delta records*.

For externalizing the counters in statistics REPORT reports or SAVE datasets, you can specify the duration of an interval using the INTERVAL option of the REDUCE subcommand. The reporting functions uniformly distribute the delta records over the user-specified intervals. The term i*nterval record* is used for a set of counters describing the activity of a DB2 system in a user-specified period of time.

For example, if INTERVAL(60) is specified, 60-minute intervals are created over the period for which the DB2 statistics data is available and the DB2 data is distributed to these intervals. INTERVAL(0) specifies that only one interval record is created, starting with the first and ending with the last DB2 statistics record.

From a technical point of view, in a VSAM SAVE dataset, entries are aggregated with respect to a VSAM key, where a key contains a combination of individual OMEGAMON XE for DB2 PE identifiers used in the report language and of internal fields. For each OMEGAMON XE for DB2 PE identifier, there exists a correspondent column in the SAVE table. For example, the following columns contribute to the VSAM key of a SAVE record that is loaded to the statistics general table DB2PM_STAT_GENERAL:

- LOCAL_LOCATION
- GROUP_NAME
- SUBSYSTEM_ID
- MEMBER_NAME
- INTERVAL_TSTAMP
- BEGIN_REC_TSTAMP.

Accounting

In the accounting domain, the REDUCE subcommand can be used with the INTERVAL option to accumulate and apportion DB2 trace data from IFCIDs 3 and 239 into intervals.

The accounting report set is based on DB2 accounting records, which are written when a thread terminates (that is at commit time), a thread is reused, or a DBAT becomes inactive. Local and distributed DB2 activity associated with a thread and/or DBRM/Package is contained.

In the accounting TRACE reports and FILE datasets, individual DB2 threads are displayed with information such as elapsed, CPU, and wait times. An entry in the report or FILE datasets is referred to as a *logical accounting record* as it can consist of several physical accounting records (IFCIDs 3 and 239).

As for the statistics domain, you can use the REDUCE subcommand to reduce the volume of data that is input to the accounting REPORT and SAVE subcommands. REDUCE consolidates records with certain common characteristics into one record.

From a technical point of view, in a VSAM SAVE dataset, entries are aggregated with respect to a VSAM key, where a key contains a combination of individual OMEGAMON XE for DB2 PE identifiers used in the report language and of internal fields. For each OMEGAMON XE for DB2 PE identifier, there exists a correspondent column in the SAVE table.

For example, the following columns contribute to the VSAM key of a SAVE record that is loaded to the accounting general table DB2PMSACCT_GENERAL:

- LOCAL_LOCATION
- GROUP_NAME
- SUBSYSTEM_ID
- MEMBER_NAME
- REQ_LOCATION
- CONNECT_TYPE
- CONNECT_ID
- CORRNAME
- CORRNUMBER
- PLAN_NAME
- PRIMAUTH
- ORIGAUTH
- INTERVAL_TIME
- MAINPACK
- THREAD_TYPE
- CLIENT_ENDUSER
- CLIENT_WSNAME
- CLIENT_TRANSACTION.

GROUP REDUCE

OMEGAMON XE for DB2 PE provides an additional feature, that extends the basic REDUCE/SAVE functionality as outlined before. See Appendix A, *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Reporting User's Guide,* SC18-9641.

The functionality employs the report command GROUP in context with OMEGAMON XE for DB2 PE identifiers and allows you to decrease significantly the size of SAVE datasets. In a SAVE dataset, entries are aggregated with respect to a VSAM key, where a key is combined of separate OMEGAMON XE for DB2 PE identifiers. Because an identifier such as CORRNAME may have many different values, there can be a very large number of reduced records in an accounting SAVE dataset. If you never require separate entries for one or more identifiers, you can use GROUP to reduce the uniqueness of the key, and thus the number of reduced data records on the save dataset. During REDUCE processing, the group name is substituted for the original value on the key entry.

For example, in order to get rid of individual correlation names and numbers, you can specify a command stream as shown in Example 14-7. In this sample, all records in the reduced data have a correlation name of ALLCNM and a correlation number of ALLCNU.

Example 14-7 GROUP REDUCE

```
GROUP (CORRNAME (ALLCNM(*)))
GROUP (CORRNMBR (ALLCNU(*)))
ACCOUNTING
   REDUCE
     INCLUDE (CORRNAME (G(ALLCNM)))
     INCLUDE (CORRNMBR (G(ALLCNU)))
   SAVE
```

Using groups can significantly reduce the diversity of the unique identifier combinations in the save data, resulting in far fewer records, smaller save datasets, and improved performance.

14.5 How to analyze the data

Using the Performance Database and Performance Warehouse, you can analyze the health of your DB2 subsystems and applications and perform trend analysis or capacity planning based on historical data. You can write your own SQL queries and analysis applications, or you can use queries and rules-of-thumb provided in OMEGAMON XE for DB2 PE.

14.5.1 Analyze data in the Performance Database

In the sample library RKO2SAMP, a set of predefined queries and views for accounting and statistics tables is available in members DGOAQ*, DGOAV* and DGOSQ*.

Summarize accounting data with SQL

The REDUCE command consolidates accounting IFCID records with certain common characteristics (same VSAM key values) into one SAVE record. Data stored in SAVE tables can be further summarized using the SQL language element GROUP BY. For example, in order to calculate the sum of class1 elapsed times for all individual plans use the query in Example 14-8.

Example 14-8 Sample query for summarizing SAVE data.

```
SELECT
  SUM(CLASS1_ELAPSED) AS ELAPSED_TIME,
  PLAN_NAME
FROM DB2PMSACCT_GENERAL
GROUP BY PLAN_NAME
```

Note: The report subcommand option ORDER can be used to get summarized data in reports.

Calculate average times of applications

The REDUCE command aggregates accounting IFCID records with certain common characteristics (same VSAM key values) into one SAVE record. The column OCCURRENCES tracks the number of threads which are accumulated in one row in the SAVE table and can be used for calculating average times and events. For example, in order to calculate the average class1 elapsed time for all individual plans use the query in Example 14-9.

Example 14-9 Sample query for calculating average times

```
SELECT
  SUM(CLASS1_ELAPSED) / SUM(OCCURRENCES) AS AVG_ELAPSED_TIME,
  PLAN_NAME
FROM DB2PMSACCT_GENERAL
GROUP BY PLAN_NAME
```

Analyze virtual storage

Virtual storage consumption is an important topic in Performance Tuning. Please refer to *DB2 UDB for z/OS Version 8 Performance Topics,* SG24-6465 for considerations on using virtual storage counters for DB2 V8.

In the Statistics Reports, virtual storage counters and calculated fields derived from IFCID 225 are presented in the DBM1-related report blocks.

The IFCID 225 fields can also be stored in the Statistics General table. See Example 14-10 which shows the formula for calculating the thread footprint and the maximum number of possible threads.

Example 14-10 SQL query for virtual storage analysis in member DGOSQGEN

```
SELECT
        (TOT_VAR_STORAGE
        - TOT_AGENT_SYS_STRG
        - LOC_DYN_CACHE_POOL)
        /
        (ACT_ALLIED_THREADS
         + MAX_ACTIVE_DBATS)
AS THREAD_FOOTPRINT,

        (EXT_REG_SIZE_MAX
        - BIT_EXT_LOW_PRI_31
        - MIN((EXT_REG_SIZE_MAX/8),(200*1024*1024))
        - (TOT_GETM_STORAGE + TOT_FIXED_STORAGE + LOC_DYN_CACHE_POOL
           + TOT_AGENT_SYS_STRG))
        /
        ((TOT_VAR_STORAGE
        - TOT_AGENT_SYS_STRG
        - LOC_DYN_CACHE_POOL)
        /
        (ACT_ALLIED_THREADS
        + MAX_ACTIVE_DBATS))
AS MAX_THREADS

FROM DB2PM_STAT_GENERAL ;
```

Note that, based on findings from some SAP customers, the formulas for thread footprint and maximum number of possible threads have slightly been changed in OMEGAMON XE for DB2 PE.

The thread footprint (TF) is calculated as:

```
(QW0225VR - QW0225AS - QW0225SC) / (QW0225AT + QDSTHWAT)
```

The maximum number of possible threads is calculated as:

```
(QW0225RG - QW0225EL - MIN (QW0225RG / 8, 200*1024*1024) – (QW0225GM + QW0225FX + QW0225SC
+ QW0225AS)) / TF.
```

Query for OMEGAMON Historical tables translated to PDB

Sample queries for the Historical tables are delivered as datasets KO2Q* in the installation library TKANSAM. See Example 14-11, which translates the query in member KO2QDC1D to a PDB query. Note that column TIMESTAMP is used for joining the rows in the accounting general and DDF table. In the where-clause, the value for column THREAD_TYPE has been changed to ALLDDIST.

Example 14-11 Sample query mapped to PDB

```
SELECT
   GEN.SUBSYSTEM_ID      AS DB2ID,
   GEN.LOCAL_LOCATION    AS LOCATION_NAME,
   DDF.REMOTE_LOCATION   AS REMOTE_LOCATION,
   GEN.PLAN_NAME         AS PLANNAME,
   GEN.CONNECT_ID        AS CONNID,
   GEN.CORRNAME CONCAT
       GEN.CORRNUMBER    AS CORRID,
```

```
   GEN.PRIMAUTH          AS AUTHID,
   GEN.CLASS1_TIME_BEG   AS START_TIMESTAMP,
   GEN.COMMIT            AS COMMIT,
   GEN.ROLLBACK          AS ABORT,
   DDF.SQL_SENT          AS SQL_STMT_SENT,
   DDF.SQL_RCD           AS SQL_STMT_RECEIVED,
   DDF.ROWS_SENT         AS ROWS_SENT,
   DDF.ROWS_RCD          AS ROWS_RECEIVED,
   DDF.BYTES_SENT        AS BYTES_SENT,
   DDF.BYTES_RCD         AS BYTES_RECEIVED,
   DDF.CONV_SENT         AS CONV_SENT,
   DDF.CONV_RCD          AS CONV_RECEIVED,
   DDF.MSGS_SENT         AS MESSAGES_SENT,
   DDF.MSGS_RCD          AS MESSAGES_RECEIVED,
   DDF.TRANS_SENT        AS TRANS_SENT,
   DDF.TRANS_RCD         AS TRANS_RECEIVED,
   DDF.COMMITS_SENT      AS COMMITS_SENT,
   DDF.COMMITS_RCD       AS COMMITS_RECEIVED,
   DDF.ROLLBACKS_SENT    AS ABORTS_SENT,
   DDF.ROLLBACKS_RCD     AS ABORTS_RECEIVED,
   DDF.CONV_QUEUED      AS CONV_REQ_QUEUED,
   DDF.REQUESTOR_ELAPSED AS TOTAL_ELAP_TIME,
   DDF.SERVER_ELAPSED    AS REMOTE_ELAP_TIME,
   DDF.SERVER_CPU        AS REMOTE_CPU_TIME,
   GEN.NET_ID            AS DDF_NETWORK_ID,
   GEN.LUNAME            AS DDF_LU_NAME,
   GEN.INSTANCE_NBR      AS DDF_INSTANCE,
   GEN.LUW_SEQNO         AS DDF_COMMIT_COUNT,
   GEN.REQ_LOCATION      AS DDF_REQ_LOCATION,
   GEN.THREAD_TYPE       AS THREAD_TYPE
FROM DB2PMFACCT_GENERAL GEN, DB2PMFACCT_DDF DDF
WHERE GEN.THREAD_TYPE  ='ALLDDIST'       AND
      GEN.SUBSYSTEM_ID = &DB2ID          AND
      TIME(GEN.TIMESTAMP) >= &STARTTIME   AND
      TIME(GEN.TIMESTAMP) <= &ENDTIME     AND
      DATE(GEN.TIMESTAMP) >= &STARTDATE   AND
      DATE(GEN.TIMESTAMP) <= &ENDDATE     AND
      GEN.TIMESTAMP=DDF.TIMESTAMP
ORDER BY GEN.SUBSYSTEM_ID,
         GEN.LOCAL_LOCATION,
         GEN.PLAN_NAME,
         GEN.CONNECT_ID,
         GEN.CORRNAME,
         GEN.CORRNUMBER,
         GEN.PRIMAUTH,
         DDF.REMOTE_LOCATION,
         GEN.TIMESTAMP;
```

14.5.2 Analyze data in the Performance Warehouse

The Performance Warehouse provides a set of SQL queries and rules-of-thumb (ROTs) that help you to identify complex performance problems. Besides the standard queries and ROTs that come with the product, you may also define and use your own rules-of-thumb and write your own SQL queries to analyze the performance data.

Expert Analysis

Expert Analysis is a set of five queries provided in the Performance Warehouse which give you further help to interpret DB2 trace data for accounting and to understand how efficiently your applications work.

These queries examine some of the most important aspects of an application from a DB2 performance point of view:

- Class1 elapsed time
- Class 2 elapsed and CPU time
- Class 3 time
- Type of SQL statements

These are guidelines to help quickly monitor and tune your applications with little effort. Figure 14-8 illustrates the decision tree which is supported by the Expert Analysis queries.

In addition, you can adapt the Expert Analysis queries to your needs. For example, you can specify different class 1 elapsed time criteria, if you have to investigate online transactions as opposed to a batch workload.

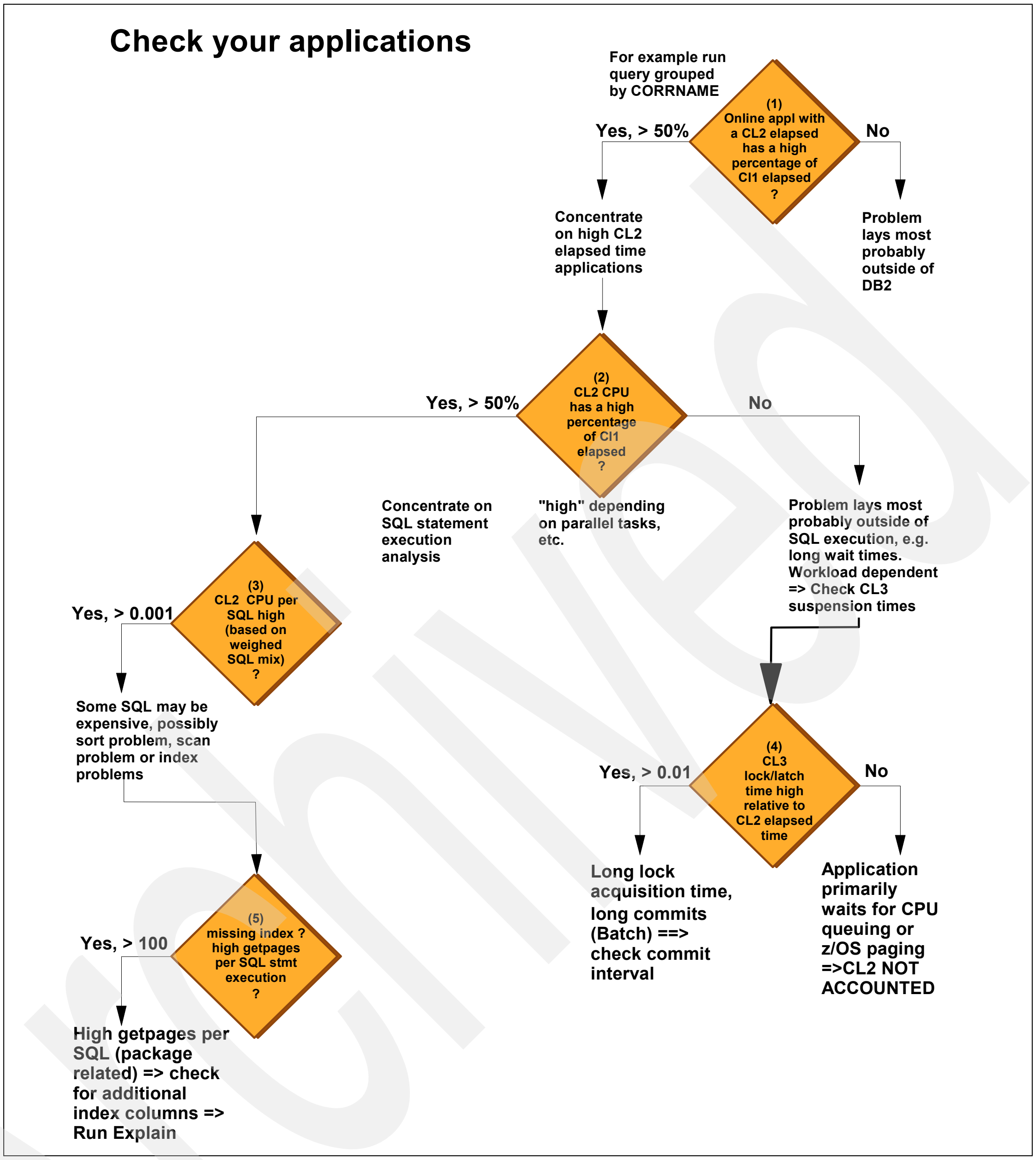


Figure 14-8 Diagnosing application performance issues with Expert Analysis queries

14.6 Scenario for using PDB and PWH

This section shows how users can benefit from a Performance Database (PDB) and Performance Warehouse (PWH) in their environment.

The Performance Warehouse was initially developed to help customers in setting up an easy-to-use database for accounting and statistics tables. Over time, the infrastructure of the warehouse has been extended and exploited for other tasks:

- Include CRD steps in PWH processes to collect trace data, on demand or regularly.
- Collect trace data for a DB2 thread and create a SQL Activity report for immediate problem analysis.
- Monitor DB2 Connect gateways and applications.

These functions help to monitor critical applications and systems and to *react* if performance problems occur. The data in the Performance Warehouse can be viewed as *short-term historical data*. Figure 14-3 on page 260 shows an environment where the PWH is used for that purpose.

The Performance Database needs to be set up and maintained by the customer, no automatic support is available in the product. On the other hand, lack of automation means full flexibility to organize the table layouts and the processes which populate and analyze the tables. The PDB is more suited to serve as *long-term historical* repository which can be customized to meet the specific requirements in a customer's environment. Historical data can be evaluated to detect performance trends and *proactively* tune the DB2 subsystems accordingly.

Figure 14-9 shows a scenario where PDB and PWH coexist as short- and long-term repository. The environment of Figure 14-3 is extended by another LPAR C that hosts a Performance Database. Performance Warehouses reside on LPAR A and B and are used for immediate problem analysis. Trace data is collected by CRD steps to monitor specific performance criteria that are relevant for the production environment. SMF data which is continuously collected from SMF is loaded into the Performance Database for long-term analysis. Accounting and statistics data in the PWH is also transferred to the PDB.

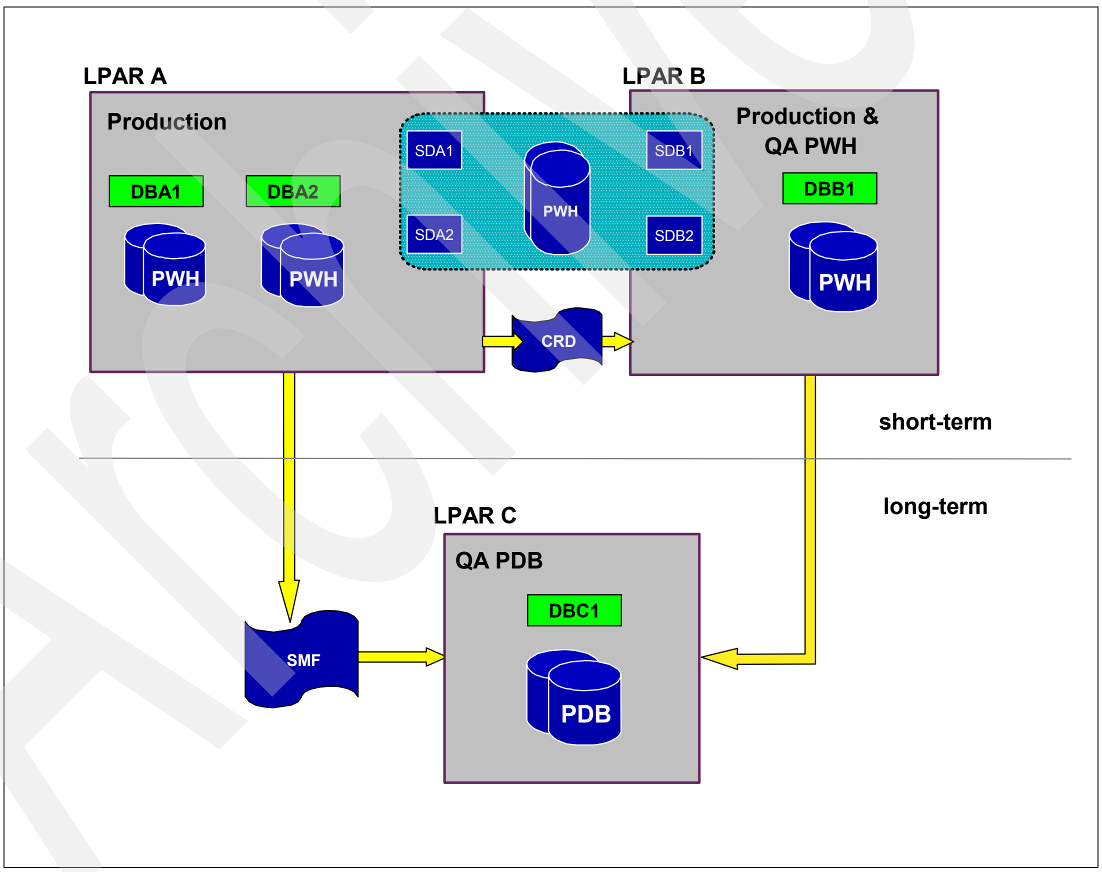


Figure 14-9 Scenario for PDB and PWH

Such environments can be set up for other tasks, such as tracking the performance of database applications during application development.

Part 5

User scenarios

In this part of the book, we discuss in detail the usage of new IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS in solving real life DB2 related issues.

This part of the book contains the following chapters:

15

Package level accounting

The reporting functions, the Performance Database, and the Performance Warehouse of OMEGAMON XE for DB2 PE provide enhanced support for accounting reports at the package level. This chapter describes the enhancements and shows how you can analyze the accounting package data in the reports and in the databases.

We discuss the following topics:

- DB2 V8 enhancements for IFCID 239
- Package level accounting support in reports
- Package level accounting support in PDB/PWH

15.1 DB2 V8 enhancements for IFCID 239

DB2 V8 introduces major changes to IFCID 3 and 239 by adding SQL, buffer pool, and locking counters at the package level. These details are very useful for analyzing the performance of your applications, especially when few plans with many packages are in place.

Here is a summary of the changes:

- The package section QPAC is removed from IFCID 3 and added to IFCID 239. Now, IFCID 3 contains only plan level accounting.
- IFCID 239 is extended to include SQL, locking, and buffer pool counters for package level accounting in addition to the QPAC (Class2, Class3) data that was already recorded in IFCID 3. The buffer pool counters represent a rollup of all buffer activities over a given package.
- Package level accounting is externalized only via IFCID 239.
- Package level accounting continues to be activated by starting Accounting Class 7 or Class 8.

Note: Online Monitor IFCIDs 0147 and 0148 are not changed. They both continue to return the QPAC section for the current package.

15.2 Package level accounting support in reports

In PE V2, the accounting reports already show the buffer pool activities at the package level. In Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS, the SQL and locking counters of IFCID 239 are also supported in the reports.

Figure 15-1 shows how the package-related report blocks are arranged in a standard accounting LONG report (the default ORDER is by PRIMAUTH-PLANNAME).

If you specify ORDER(PACKAGE) in your report command stream, then the blocks belonging to each package are arranged together.

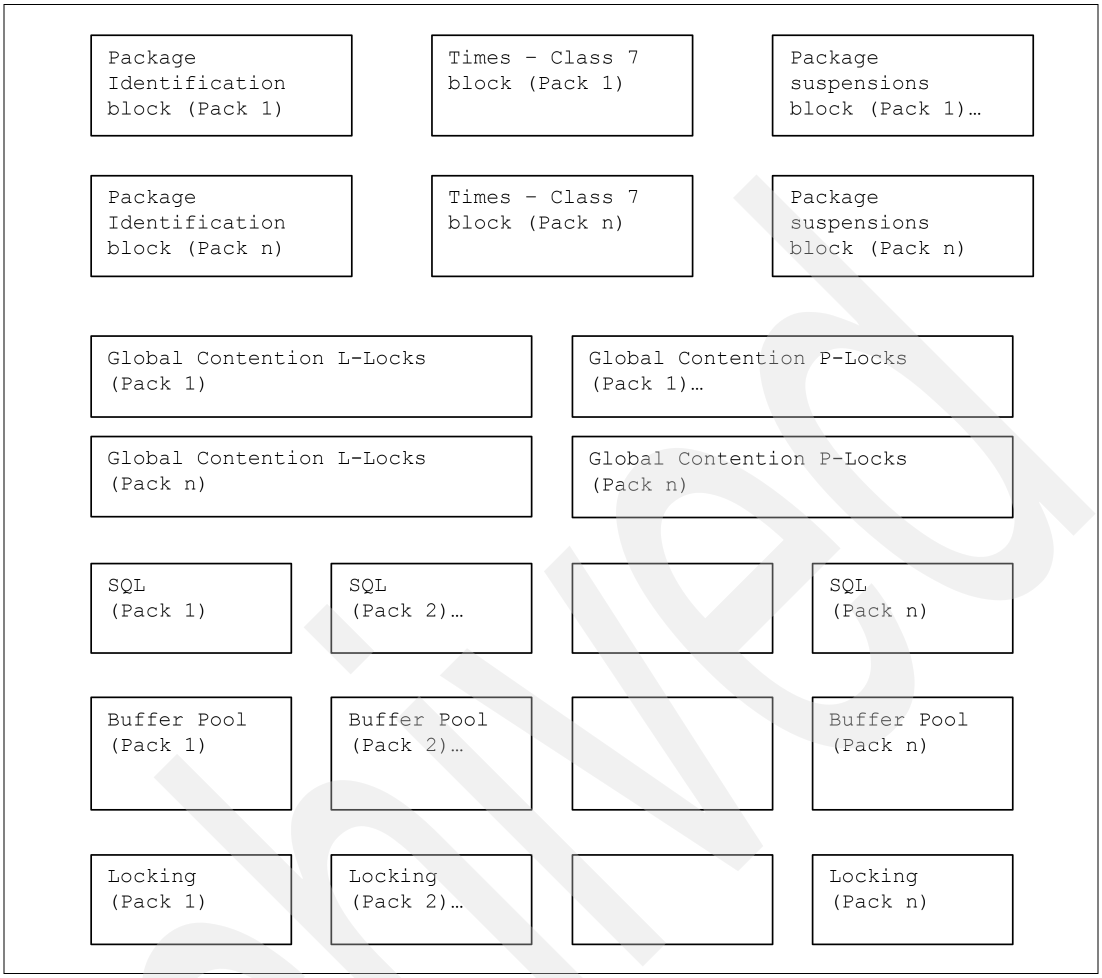


Figure 15-1 Package-related blocks in an accounting LONG report

Example 15-1 shows a portion of the CLASS 7 CONSUMERS summary block at the top of the package level accounting report. Notice that the program names are listed as well as the class 7 percent CPU time utilized. This information is useful to narrow down which package to focus your attention on when dealing with large CPU time consumption. Here the package DPTBAL may demand your attention since it is using the most CPU time within the plan.

Example 15-1 Class 7 consumers

```
!PROGRAM NAME                          CLASS 7 CONSUMERS
!DPTADD            !
!DPTANO            !
!DPTBAL            !=====================> 42%
!DPTDEL            !
!DPTMGR            !
!DPTSEL            !
!DPTUPD            !
!DPTUPR            !
!EMPADD            !> 1%
!EMPANO            !> 1%
!EMPDEL            !======> 13%
!EMPFND            !=> 2%
!EMPQRY            !=> 2%
!EMPUPD            !============> 24%
```

Example 15-2 provides a partial summary for all buffer pool activity related to package DPTBAL. Having the buffer pool information at the package level can provide useful performance measurements. This information is rolled up for all of the buffer pools used at the package level.

Example 15-2 Buffer pool activity at the package level

```
DPTBAL                AVERAGE      TOTAL
-------------------   --------   --------
BPOOL HIT RATIO (%)      99.89        N/A
GETPAGES               1581.25       6325
GETPAGES-FAILED           0.00          0
BUFFER UPDATES          275.25       1101
SYNCHRONOUS WRITE         0.00          0
SYNCHRONOUS READ          0.75          3
SEQ. PREFETCH REQS       20.75         83
LIST PREFETCH REQS       17.00         68
DYN. PREFETCH REQS        3.75         15
PAGES READ ASYNCHR.       1.00          4
```

Example 15-3 provides a partial summary for all locking activity related to package DPTBAL. Having the locking information at the package level can provide useful performance measurements.

Example 15-3 Locking activity at the package level

```
DPTBAL                  AVERAGE     TOTAL
---------------------  --------  --------
TIMEOUTS                   0.00         0
DEADLOCKS                  0.00         0
ESCAL.(SHARED)             0.00         0
ESCAL.(EXCLUS)             0.00         0
MAX PG/ROW LOCKS HELD      2.00         8
LOCK REQUEST             270.25      1081
UNLOCK REQUEST            79.75       319
QUERY REQUEST              0.00         0
CHANGE REQUEST            25.00       100
OTHER REQUEST              0.00         0
TOTAL SUSPENSIONS          2.50        10
 LOCK SUSPENSIONS          2.50        10
 IRLM LATCH SUSPENS.       0.00         0
 OTHER SUSPENS.            0.00         0
```

Example 15-4 provides a partial summary for all SQL activity related to package DPTBAL. Having the SQL information at the package level can provide useful performance measurements.

Example 15-4 SQL activity at the package level

```
DPTBAL               AVERAGE     TOTAL
------------------  --------  --------
SELECT                  8.50        34
INSERT                  4.25        17
UPDATE                118.25       473
DELETE                  0.00         0

DESCRIBE                0.00         0
PREPARE                 0.00         0
OPEN                    4.25        17
```

```
FETCH                    42.25       169
CLOSE                     0.00         0

LOCK TABLE                0.00         0
CALL                      4.50        18
```

15.3 Package level accounting support in PDB/PWH

In DB2 PE V2, the buffer pool activities at the package level are already externalized in the accounting SAVE table DB2PMSACCT_PROGRAM and in the accounting FILE table DB2PMFACCT_PROGRAM.

In OMEGAMON XE for DB2 PE, the SQL and locking counters of IFCID 239 are also supported in these tables. See the list of new columns in Example 15-5.

Example 15-5 New columns for package level accounting

```
Locking                      SQL activity
-------                      ------------
TIMEOUT                      SELECT
DEADLOCK                     INSERT
LOCK_ESC_SHARED              UPDATE
LOCK_ESC_EXCLUSIVE           DELETE
MAX_LOCKS_HELD               DESCRIBE
LOCK_REQ                     PREPARE
UNLOCK_REQ                   OPEN
QUERY_REQ                    FETCH
CHANGE_REQ                   CLOSE
OTHER_REQ                    LOCK_TABLE
SUSPEND_LOCK                 STPROC_CALL
SUSPEND_IRLM_LATCH
SUSPEND_OTHER
```

Performance Database

The CREATE statements DGOAC*PK in library RKO2SAMP contain the definition of the new columns. Meta data is available in members DGOAB*PK and DGOAD*PK. The LOAD statements in DGOAL*PK refer to the new package level columns.

If you already use the PROGRAM tables and you want to migrate existing tables, then you can apply the ALTER statements in member DGOAUPDB to add the new columns. The COLUMN keyword is needed for DB2 V8 but not for DB2 V7 subsystems.

Performance Warehouse

The Performance Warehouse supports the accounting SAVE table and the new columns are automatically created in new databases or added to an existing table DB2PMSACCT_PROGRAM when the OMEGAMON Server and associated PE server instance are started.

Sample queries

The query in Example 15-6 shows how accounting data in the reports can be calculated from the database tables with SQL.

Example 15-6 SQL query for calculating accounting data at the package level

```
SELECT
   SUM(FLOAT(SELECT))/COUNT(*), SUM(SELECT),
   SUM(FLOAT(INSERT))/COUNT(*), SUM(INSERT),
   SUM(FLOAT(UPDATE))/COUNT(*), SUM(UPDATE),
   SUM(FLOAT(DELETE))/COUNT(*), SUM(DELETE)
FROM DB2PMSACCT_PROGRAM
WHERE PCK_ID = 'DPTBAL'

Result row has been split:
COL0001          COL0002
                    *                *
--------------------- ----------------
8.500000000000000E+00               34

             COL0003          COL0004
--------------------- ----------------
4.250000000000000E+00               17

             COL0005          COL0006
--------------------- ----------------
1.182500000000000E+02              473

             COL0007          COL0008
--------------------- ----------------
0.000000000000000E+00                0
```

The query in Example 15-7 shows how you can create comma-separated values (CSV) from the database tables.

Example 15-7 SQL query for creating CSV data

```
SELECT CORRNAME
   CONCAT ';'
   CONCAT PCK_ID
   CONCAT ';'
   CONCAT RTRIM(TRANSLATE(CHAR(LOCK_REQ),' ','.'))
   CONCAT ';'
   CONCAT RTRIM(TRANSLATE(CHAR(UNLOCK_REQ),' ','.'))
FROM DB2PMSACCT_PROGRAM
ORDER BY CORRNAME, PCK_ID
```

The CSV data can be directed via SPUFI to a file, which can be post-processed and downloaded to a workstation. From there, you can import the file into a spreadsheet and create pivot tables or charts for visualizing performance data and trends. See Figure 15-2 for a chart that shows a lock request value contributing to the total value as presented in the report block.

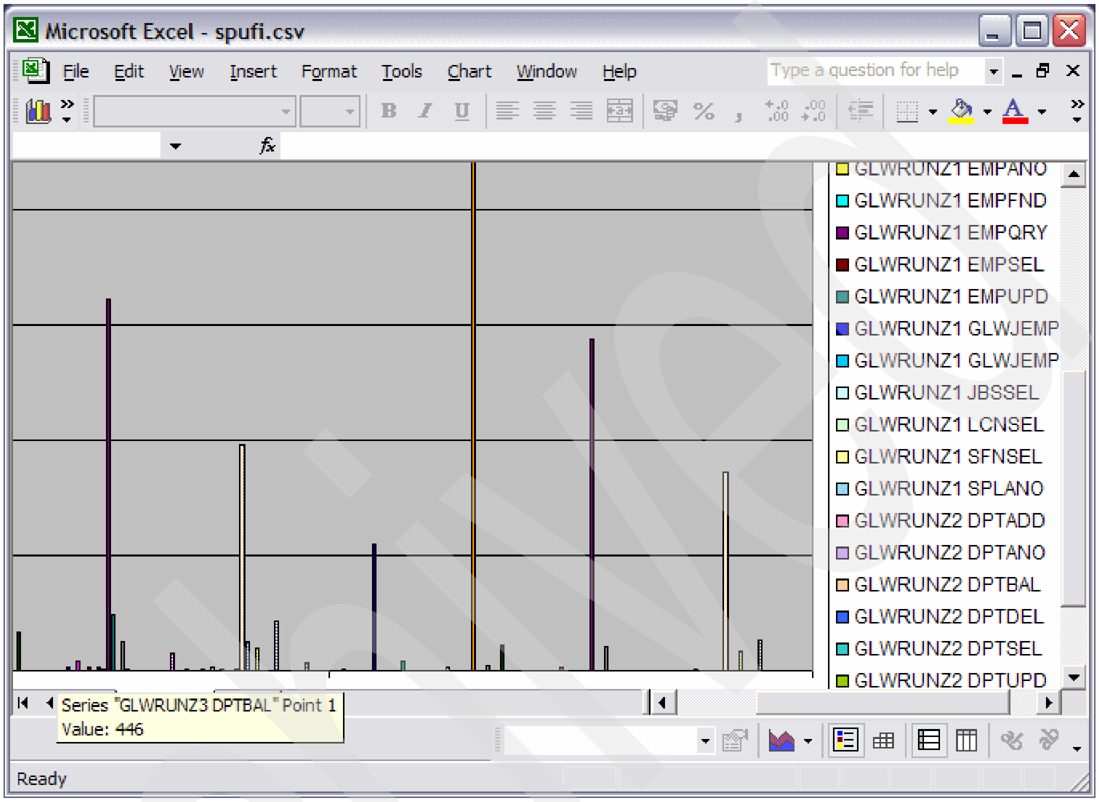


Figure 15-2 Spreadsheet for visualizing performance data

This technique can be applied to other data in the Performance Database or Performance Warehouse as well.

Notes:

- You can run DB2 sample program DSNTIAUL to create CSV datasets in batch mode.
- The locking report set provides a SPREADSHEETDD subcommand that writes lock suspension data into a CSV dataset. See the Report Reference for more details.

16

Solving common DB2 operational problems

In this chapter we describe how you can use OMEGAMON XE for DB2 PE to help solve some common DB2 problems that regularly occur in operational environments. The emphasis is on rapid and accurate problem solving in large and complex environments.

We discuss the following topics:

- General considerations
- Using exception triggering with screen logging
- How to recognize a looping DB2 thread
- How to cancel a DB2 thread safely
- How to find badly performing SQL
- Dealing with lock escalation problems
- Dealing with deadlocks and timeouts

16.1 General considerations

In this chapter we show how to use OMEGAMON XE for DB2 PE in practice to solve a variety of common DB2 operational problems. We concentrate on the use of the VTAM Classic Interface because speed of response is frequently essential to solving operational problems.

We devote 16.2, “Using exception triggering with screen logging” on page 287 to a detailed explanation of the setting up of exception triggering with screen logging. This is because this feature is difficult to set up correctly and therefore not widely used. However, it provides the ability to capture information that is extremely difficult to capture in any other way, and that can be vital to solving certain DB2 problems.

As we work through the scenarios below, we frequently tell you to check particular fields on OMEGAMON XE for DB2 PE panels and we are often moving quickly between panels. To help you identify these fields quickly and accurately, we refer to them in this standard way:

Look at: `field-name` on panel `panel-name` (`panel-id, menu-path`)

If we use Figure 16-1 as an example, we would refer to the field `Thread Status` in the following way:

Look at: `Thread Status` on panel `SQL CALL BEING EXECUTED` (`ZSQL, T.ZOOM.F`)

The `panel-id` is on the top line of the panel after the input field, and uniquely identifies the panel. The `menu-path` shows how to reach the panel from the OMEGAMON XE for DB2 PE Main Menu. In the `menu-path` individual letters separated by dots represent menu options and `ZOOM` represents placing the cursor on a detail line (usually a thread) and pressing Enter. So in the example, the panel would be reached with the sequence `T` → `ZOOM` → `F`.

```
 ________________ ZSQL     VTM     O2       V310./I DB8A 02/28/06 18:15:23   2
> Help PF1                                                             Back PF3

>       THREAD INFORMATION:  Enter a selection letter on the top line.

> A-THREAD DETAIL B-LOCK COUNTS C-LOCK WAITS     D-LOCKS OWNED  E-GLOBAL LOCKS
> *-CURRENT SQL   G-SQL COUNTS  H-DISTRIBUTED    I-BUFFER POOL  J-GROUP BP
> K-PACKAGES      L-RES LIMIT   M-PARALLEL TASKS N-UTILITY      O-OBJECTS
> P-CANCEL THREAD Q-DB2 CONSOLE R-DSN ACTIVITY   S-APPL TRACE   T-ENCLAVE
> U-LONG NAMES
===============================================================================
>                            SQL CALL BEING EXECUTED
 PLAN
+ Thread:  Plan=KO2PLAN   Connid=DB2CALL  Corrid=DB8AO2S      Authid=DB8AO2S
+ Attach:  BATCH          JOB Name=DB8AO2S                    JOB Asid= 175
+ Package: DGO§DB2I       Collection=KO2OM310
 call
+      SQL call is active, call information is as follows :
+
+      Thread Status  = WAIT-REMREQ    SQL Request Type     = STATIC
+      Total SQL Reqs =       12       SQL Call Type        = CLOSE CURSOR
+      SQL DBRM Name  = DGO§DB2I       SQL Statement Number = 00390
```

Figure 16-1 Example for References to Field Names on Panels

In the scenarios that follow, we generally give a number of indicators from different OMEGAMON XE for DB2 PE fields that you should check for a specific problem. You should always try to build up as complete a picture as possible for a suspected problem thread. It is easy to be misled if you rely on the information from a single field in diagnosing a problem, particularly as some fields can be easily misinterpreted or may even be displaying incorrect data under certain conditions.

16.2 Using exception triggering with screen logging

In general, exception analysis within OMEGAMON XE for DB2 PE is of limited usefulness in problem determination at application level for the following reasons:

- The exception is only visible during the cycle (a typical cycle is five seconds) in which it is triggered.
- Unless an exception is triggered over many consecutive cycles, there is not enough time to investigate the thread causing the exception before the condition disappears.
- Operational staff with possibly 50 or more DB2 subsystems to monitor cannot spend all their time watching exception panels for possible exception activity.
- It is very difficult to set meaningful exception thresholds for DB2 threads. Acceptable CPU usage, elapsed time, update rates and commit frequencies vary very widely from transaction to transaction.
- Problems often occur too infrequently and are too short-lived to be caught by manual monitoring of exception panels.

Despite these limitations there is one method of using exception triggering, which can be particularly helpful in solving certain DB2 operational problems. This is the use of exception triggering combined with automatic screen logging, which is a special feature of the OMEGAMON XE for DB2 PE Classic Interface.

When an exception has been set up with automatic screen logging, each time the exception occurs, a specified panel is automatically logged for the thread that caused the exception. This is either a default panel or one that you have built yourself. This means that all the information you need about the thread to help solve the problem can be automatically logged at the time the exception occurs and printed out later for analysis. Because you can build the panel that is logged yourself, you can capture for later analysis any information that is available in OMEGAMON XE for DB2 PE at the time the exception occurs.

Some examples of the kind of information you might want to capture for the thread that triggered the exception are:

- Identification details for the thread (plan, authid, correlation-id, jobname).
- The text of the SQL (dynamic or static) that was executing.
- Details of locks held by the thread.
- A lock the thread was waiting for.
- A count of SQL calls executed, listed by type.
- Details of all other threads with lock suspensions.
- Detailed buffer pool usage for the thread.
- Global lock contention.
- Parallel thread activity.
- Stored procedure and trigger statistics for the thread.

In this section we describe how to set up exception triggering with screen logging. Later sections will show some practical applications of the technique. The initial setup is fairly complicated. However, once the definitions are set up and stored in a user profile, the technique can be used whenever needed without much trouble by specifying the user profile suffix during logon.

To set up exception triggering with automatic screen logging, you need to carry out the following sequence of operations:

1. Define the threshold parameters and the exception logging facility (XLF) parameters for the exceptions you want to log.
2. Optionally define further rules for the exceptions to restrict when they trigger.
3. Optionally create a panel to be used for logging exceptions.
4. Set up the exception logging facility (XLF) printing options.
5. Set up the automatic screen facility (ASF) printing options.
6. When you have completed all the above changes, save the results in a user profile.
7. Optionally reduce the exception checking cycle from the default of five seconds. This may be necessary if the exception events you want to capture are occurring very rapidly.
8. Switch on exception logging and automatic screen logging as required.

We now describe these operations in more detail using the `WTRE` thread exception as an example (the `WTRE` exception warns when a DB2 thread is waiting for a resource for longer than the specified threshold value in seconds).

Defining the threshold and XLF parameters

From the OMEGAMON XE for DB2 PE Main Menu, type the menu path `P.B.B` to reach the Thread Exception Analysis Maintenance Menu shown in Figure 16-2.

```
> P.B.B
>          Type a selection letter on the top line and press ENTER.
===============================================================================
>                THREAD EXCEPTION ANALYSIS MAINTENANCE MENU
_ GX       Global Exclude Rules

_ A  ARCM  Thread backout processing is waiting for an archive tape mount
_ B  COMT  Ratio of system page updates to commits is greater than threshold
_ C  CTHD  Application is waiting for thread creation
_ D  DWAT  Time dist. allied thread has been waiting for response - remote SQL
_ E  ETIM  Elapsed time for a DB2 thread
```

Figure 16-2 Thread Exception Analysis Maintenance Menu

The full list of thread exceptions available in OMEGAMON XE for DB2 PE with descriptions is shown in Table 16-1.

Table 16-1 Table of Thread Exceptions

Exception	Description
ARCM	Thread backout processing is waiting for an archive tape mount
COMT	Ratio of system page updates to commits is greater than threshold
CTHD	Application is waiting for thread creation
DWAT	Time dist. allied thread has been waiting for response - remote SQL
ETIM	Elapsed time for a DB2 thread
GETP	Getpages/Read I/O is greater than threshold
IDBC	Amount of CPU time used by DB2 to process thread
IDBT	Time DB2 has been processing thread
INDB	Individual threads in indoubt status

Exception	Description
LKUS	Number of locks owned by an individual thread
PGUP	Number of page update requests per second made by a thread
PREF	Read sequential prefetch rate
RCPU	Amount of CPU time being used by a dist. DB access thread
RELM	Resource limiting threshold reached for a thread
RIO	Thread synchronous read I/O rate
TCPU	CPU rate of active threads
TRCV	Amount of data received by dist. threads from remote DB2
TSND	Amount of data sent by dist. threads to remote DB2
WCLM	Time thread waiting for drain of claims
WDLK	Time thread waiting for acquisition of Drain lock
WGLK	Time thread waiting because of global contention
WLGQ	Time thread waiting for ARCHIVE LOG MODE(QUIESCE)
WSPS	Time thread waiting for a TCB to schedule a stored procedure
WSRV	Time thread waiting for DB2 Service
WTRE	Time thread waiting for a resource

When you select the `WTRE` exception (time thread waiting for a resource) from the Thread Exception Analysis Maintenance Menu, you see the panel shown in Figure 16-3.

```
>              SET EXCEPTION ANALYSIS OPTIONS FOR WTRE EXCEPTION

> The WTRE exception warns when a DB2 thread is waiting for a resource for
> longer than the specified threshold value.

> To change the value of an exception option, type the new value over the
> current one.  Press ENTER to record the change.

 XACB LIST=WTRE
: WTRE
+      DISPLAY Parameters:   THRESHOLD Parameters:   XLF Parameters:
:        State=ON             Threshold=5              Auto=ON
:        Group=TH             Display=CLR3             Log=ON
:        Bell=OFF             Attribute=NONE           Limit=5 (5)
:      BOX Parameters:       CYCLE Parameters:         Repeat=YES
:        Boxchar=NO BOX       ExNcyc=0                 Persist=1
:        Boxclr=NONE          Stop=0 (0)               SL=ZTHLOG
:        Boxattr=NONE         Cumulative=0
```

Figure 16-3 Set Exception Analysis Options for the WTRE Exception

To set up the WTRE exception so that exception logging and automatic screen logging are enabled, at least the following fields on the panel need to be updated:

State Set to ON to enable background exception processing for the WTRE exception.

Threshold Set to the required threshold. In our example it is set to trigger an exception after the thread has waited five seconds for a DB2 resource (usually a lock). This field is not applicable if the exception relates to an action rather than a value.

Auto Set to ON to enable automatic screen logging.

Log Set to ON to enables automatic exception logging.

Limit Set to the number of times you want the exception to be logged. After this limit is reached, triggered exceptions will no longer be logged. In our example, logging will stop after five exceptions have been triggered. Use 0 to disable logging and NONE for no logging limit.

SL Set to the name of the panel to be logged when an exception is triggered. You can use the default panel for thread exceptions, which is ZTHLOG, or you can use your own panel name.

SS If SS=ZTHLOG is displayed this must be changed to SL=ZTHLOG to ensure that the panel is logged.

ExNcyc Set to 0 to ensure that the exception will be checked on every cycle (typically every five seconds).

Persist/Repeat Use Persist=1 with Repeat=YES to ensure that all exceptions are logged until the logging limit (Limit) is reached.

To change any of the above values, type the new value over the old one and press Enter. You will need to follow the above procedure for each exception that you want to log.

Defining further thread exception rules

For thread exceptions it is possible to define further rules to qualify when the exception is triggered. On the lower part of the Set Exception Analysis Options for WTRE Exception panel shown in Figure 16-4 are fields that enable you to set up rules to qualify the triggering of the WTRE exception more precisely. In this case we are defining RULE 01 to qualify exception triggering for the WTRE exception by PLAN, AUTHID, CONNTYPE, PACKDBRM and CORRID.

```
>              SET EXCEPTION ANALYSIS OPTIONS FOR WTRE EXCEPTION

XTHD WTRE
: FUNCTION=ADD(01)            ! (DIS/ADD/DEL)
+------------------------------------------------------------------------------
+ RULE=01 ! Thread Exception Rule Parameters:
:         !  PLAN=DSNREXX_     AUTHID=PAOLOR1_     CONNTYPE=BATCH__
:         !  CICS Parameters:  CICSTRAN=____       CICSCONN=________
:         !  IMS Parameters:   PSBNAME=________    IMSID=________
:         !  PACKDBRM=EMPQRY__
:         !  PAKCOLID=__________________
:         !  CORRID=GLWRUNY_____
:         !  REQLOC=________________
:         !  THRDTYPE=__
:         !  THRSHOLD=2_______ EXCLUDE=NO
```

Figure 16-4 Add a Rule for the WTRE Exception

FUNCTION Set to ADD to add a new rule when you press Enter. Use DEL to delete a rule and DIS to display existing rules.

PLAN The plan name for which this exception will trip.

AUTHID The authorization id for which this exception will trip.

CONNTYPE The connection type for which this exception will trip. Possible values are BATCH, CICS, DIST, IMS, TSO and UTILITY.

CICSTRAN The CICS transaction id for which this exception will trip.

CICSCONN The CICS job name for which this exception will trip.

PSBNAME The IMS PSB name for which this exception will trip.

IMSID The IMS subsystem name for which this exception will trip.

PACKDBRM The name of the package or DBRM for which this exception will trip.

PACKAGECOLID The name of the collection for which this exception will trip.

CORRID The name of the correlation id for which this exception will trip.

REQLOC The name of the requesting location for which exception will trip.

THRDTYPE The number from the following list for which exception will trip:
1 -- Normal threads
2 -- Allied Distributed threads
3 -- Command threads
4 -- Distributed Database Access threads
5 -- INDOUBT threads
6 -- Parallel task threads
8 -- System threads

THRSHOLD The threshold that the exception is to use when the rule you are defining applies. To avoid the exception being triggered outside of this rule you should set the default threshold value set by the XACB command to a high value such as 999999 (See Figure 16-3).

EXCLUDE Specifies whether to exclude (YES) or to include (NO) threads with the specified criteria in exception analysis reporting.

Creating a panel to be used for logging exceptions

You can either use the default panel for logging thread exceptions (ZTHLOG) or create your own panel containing commands for the precise information you require. You can find details of how OMEGAMON XE for DB2 PE commands work, and how to build and save your own panels in Chapter 6, “VTAM interface” on page 87.

> **Restriction:** In OMEGAMON XE for DB2 PE V3.1.0 at the time of writing this book the supplied ZTHLOG panel does not log any information when called as the .RTN command appears before the PLAN command. You can use the panel shown in Figure 16-5 as a substitute to get started.

Figure 16-5 shows an example of a user defined Thread exception log screen. The following OMEGAMON XE for DB2 PE commands are used to give detailed information about the thread that triggered the exception:

.LOGPOP/.LOC Necessary for correct panel navigation.

PLAN The major command PLAN is used to give plan identification details. This is followed by a number of related minor commands.

call Details and text for the active SQL call (both static and dynamic calls can be shown).

act Thread activity details, including elapsed, CPU, and wait times.

own All locks and claims owned by the thread.

wait Details of any lock the thread may be waiting for.

sqls Current SQL counts for the thread displayed by type.

.RTN Must be the last command on the panel to ensure that logging completes correctly.

```
______________   ZTHLOG
 .LOGPOP
 .LOC
 ======
>                        Thread Exception Log Screen
 PLAN
 ======
 call
 ======
 act
 ======
 own
 ======
 wait
 ======
 sqls
 ======
 .RTN
```

Figure 16-5 User Defined Thread Exception Log Screen

Setting up printing options

From the OMEGAMON XE for DB2 PE Main Menu, type the menu path `P.I.A` to reach the Set XLF Printer and Routing Options panel shown in Figure 16-6.

```
>.XLFOUT
 OUTP  XLFLOG
+                            !----- Pending ------!----- Current ------!
 copy                        ! 1                  ! 1                  !
 ddnm                        ! (*DYNAMIC)         ! (*DYNAMIC)         !
 dest                        ! *NONE*             ! *NONE*             !
 dstu                        ! *None*             ! *None*             !
 fold                        ! YES                ! YES                !
 form                        ! *None*             ! *None*             !
 hold                        ! NO                 ! NO                 !
 id1                         ! 'D8F202S '         ! 'D8F202S '         !
 id2                         ! '                ' ! '                ' !
 id3                         ! '                ' ! '                ' !
 id4                         ! '                ' ! '                ' !
 lnct                        ! 60                 ! 60                 !
 sout                        ! A                  ! A                  !
```

Figure 16-6 Set XLF Printer and Routing Options

Use this panel to define your options for printing the exception log.

To change an option, type the new value to the right of the option name and press Enter. The new value appears in the Pending column. To make the value the Current value, remove the **>** preceding `.XLFOUT` and press Enter.

The menu path `P.I.E` can be used in an identical way to set the printing options for the Automatic Screen logging.

Saving the changes in a user profile

Because of the complicated nature of the initial setting up for exception logging, you probably want to save your changes in a user profile. You can then use the definitions easily in a later session by specifying the user profile suffix during logon.

Full details of how to create and manage user profiles can be found in Chapter 6, "VTAM interface" on page 87.

If the default exception checking cycle time of five seconds is too slow to catch the exceptions you are interested in, you can alter the field `INTERVAL` on the Set Control Function Options panel (menu path `P.A.B`).

Switching on logging and printing the log

Once you have set up the definitions in the way described you can use the following menu paths to switch logging on and off, and to print the logs. It is not necessary to switch off logging before you print the logs.

`P.I.B` Turn on Exception Logging Facility

`P.I.C` Turn off Exception Logging Facility

`P.I.D` Print Exception Logging Facility log (XLFLOG)

`P.I.F` Turn on Automatic Screen Facility

`P.I.G` Turn off Automatic Screen Facility

`P.I.H` Print Automatic Screen Facility log (REPORT)

You need to switch on both exception logging (XLF) and automatic screen logging (ASF) to see full details of the exception threads in the log.

16.3 How to recognize a looping DB2 thread

It would be possible to set up exception logging, as described in 16.2, "Using exception triggering with screen logging" on page 287, to trigger threads with very high CPU usage, in an attempt to identify looping DB2 threads. Unfortunately, the only two thread exception conditions available that might help both have problems:

IDBC The `IDBC` exception warns when the amount of CPU time used by DB2 to process a thread is greater than the specified threshold value. Since program loops generally occur in the code outside DB2, this exception is unlikely to be triggered.

TCPU The `TCPU` exception warns when the CPU utilized for an address space that has DB2 connections and threads is > n%. Unfortunately a high CPU% rate by itself occurs frequently in DB2 threads for short periods of time, and is not a clear indication of a looping thread.

You could use the TCPU exception to give you warning of a thread that might be looping. More likely, you will be asked to investigate a thread to see if it is looping, which has been identified for other reasons (unexpectedly long run time, or an application no longer reacting as expected). However it comes to your attention, you need to look at a DB2 thread very carefully in OMEGAMON XE for DB2 PE before you can decide with any certainty whether it is looping.

We look at two types of looping threads with very different characteristics:

- Thread is in a hard loop outside DB2
- Thread is in a logical loop executing SQL

Thread is in a hard loop outside DB2

If a program is in a hard loop outside of DB2, it will generally be executing a small range of program code repeatedly, with no file activity or SQL access. This can occur after a program change when a logic error has been accidentally introduced into the program or, if no program change has occurred, because of an unexpected combination of data that is not handled correctly by the program.

You should check the following fields in OMEGAMON XE for DB2 PE to determine whether a thread is in a hard loop outside DB2:

1. Look at fields CPU and Status on panel ALL THREADS CONNECTED TO DB2 (ZALLT, T.A) after using PF10 to sort on the CPU column. The thread currently using the most CPU appears at the top of the list. See Figure 16-7.

```
>                      ALL THREADS CONNECTED TO DB2
 THDA                                                                   FLTR ON
+                        *
+ Elapsed      Planname  CPU    Status       GetPg   Update  Commit  Jobname
+ ----------   --------  -----  ------------ ------  ------  ------  --------
+ 00:01:42.3   DSNREXX   42.0%  NOT-IN-DB2   300007  276510      45  GLWRUNLP
```

Figure 16-7 All Threads Connected to DB2

For a thread looping outside of DB2 you would expect to see consistently over several minutes:

```
Status          NOT-IN-DB2
CPU             > 80%
```
(CPU rate on one processor)

You need to be careful when looking at the field CPU. It may represent the CPU rate as a percentage of one processor (CP), or it may represent the rate spread over the number of processors available on your LPAR, depending on how your installation is configured. In the example in Figure 16-7 the displayed CPU rate of 42% actually represents a CPU rate of 84% on one processor, as the LPAR has two processors available.

2. Press PF11 to zoom to the THREAD DETAIL panel for the thread you suspect is looping, as shown in Figure 16-8.

```
>                                THREAD DETAIL
 PLAN
+ Thread:  Plan=DSNREXX   Connid=DB2CALL  Corrid=GLWRUNLP     Authid=PAOLOR1
+ Attach:  BATCH          JOB Name=GLWRUNLP                   JOB Asid=  29
+ Package: DSNREXX        Collection=DSNREXX
 act
+ Thread Activity                         User Defined Functions
+ ------------------------------------   ------------------------------------
+ DB2 Status           =    NOT-IN-DB2   TCB Time (SQL)         = 00:00:00.000
+ MVS Status           =     USING CPU   Wait for TCB Time      = 00:00:00.000
+ Total Elapsed Time   =  00:02:04.756   Elapsed Time           = 00:00:00.000
+ CPU Utilization      =         41.3%   Elapsed Time (SQL)     = 00:00:00.000
+ Total CPU Time       =  00:01:21.860   SQL Events             =            0
```

Figure 16-8 Thread Detail

Look at fields `DB2 Status`, `MVS Status`, `Total Elapsed Time`, `CPU Utilization`, and `Total CPU Time` on the panel `THREAD DETAIL (ZTDTL, T.ZOOM.A)`. For a thread looping outside of DB2 you would expect to see consistently over several minutes:

```
DB2 Status          NOT-IN-DB2
MVS Status          USING CPU
CPU Utilization     > 80%
```
(CPU rate on one processor)

If you are not sure whether the `CPU Utilization` rate is being divided by the number of processors available on the LPAR, compare the change in `Total Elapsed Time` with the change in `Total CPU Time` over a period of one minute to get an estimate of the CPU rate on one processor.

If the LPAR is heavily loaded with work, you should also take into account the Workload Manager (WLM) dispatching priority of your thread. If the priority of your thread is very low and the thread is often waiting for CPU resources, the `CPU Utilization` may be lower than 80% even for a looping thread.

3. Look at fields `Thread Status` and `Total SQL Reqs` on panel `SQL CALL BEING EXECUTED (ZSQL, T.ZOOM.F)` as shown in Figure 16-9.

```
>                         SQL CALL BEING EXECUTED
 PLAN
+ Thread:  Plan=DSNREXX   Connid=DB2CALL  Corrid=GLWRUNLP     Authid=PAOLOR1
+ Attach:  BATCH          JOB Name=GLWRUNLP                   JOB Asid=  29
+ Package: DSNREXX        Collection=DSNREXX
 call
+       SQL call is active, call information is as follows :
+
+       Thread Status  = WAIT-REMREQ    SQL Request Type     = STATIC
+       Total SQL Reqs =   498556       SQL Call Type        = CALL
+       SQL DBRM Name  = DSNREXX        SQL Statement Number = 02232
```

Figure 16-9 SQL Call Being Executed

For a thread looping outside of DB2 you would expect to see consistently over several minutes:

```
Thread Status       WAIT-REMREQ
Total SQL Reqs      No change in the count of SQL requests
```

4. Look at field `Getpage Requests` on panel `THREAD BUFFER POOL ACTIVITY (ZBUF,T.ZOOM.I)` as shown in Figure 16-10.

```
>                        THREAD BUFFER POOL ACTIVITY
 PLAN
+ Thread:  Plan=DSNREXX   Connid=DB2CALL  Corrid=GLWRUNLP     Authid=PAOLOR1
+ Attach:  BATCH          JOB Name=GLWRUNLP                   JOB Asid=  29
+ Package: DSNREXX        Collection=DSNREXX
 buf
+ Buffer Pool: ALL
+
+ Getpage Requests       =     300007   Failed Getpage Requests    =          0
+ Synchronous Read I/O   =          1   Getpage/Read I/O           =  300007.00
+ Page Updates           =     276510   Seq Prefetch Requests      =       2121
+ List Prefetch Requests =        268   Dynamic Prefetch Requests  =         72
+ Prefetch Pages Read    =         40   Prefetch Pages In Hiperpool=          0
```

Figure 16-10 Thread Buffer Pool Activity

For a thread looping outside of DB2 you would expect to see consistently over several minutes that the value of `Getpage Requests` does not increase. In fact, you would not expect any of the counters on this panel to increase for a thread looping outside of DB2.

5. Look at the fields on panel `DISTRIBUTED THREAD DETAIL (ZTDIST,T.ZOOM.H)` as shown in Figure 16-11.

```
>                        DISTRIBUTED THREAD DETAIL
PLAN
+ Thread:  Plan=DSNREXX   Connid=DB2CALL  Corrid=GLWRUNLP     Authid=PAOLOR1
+ Attach:  BATCH          JOB Name=GLWRUNLP                   JOB Asid=  29
+ Package: DSNREXX        Collection=DSNREXX
rsum
+      Not A Distributed Thread
```

Figure 16-11 Distributed Thread Detail

If this panel shows that the thread has been active for a long time executing an SQL call on a remote DB2 subsystem, this might account for the lack of activity on the DB2 you are monitoring. However, in this case you would not expect to see consistently high CPU usage with `DB2 Status = NOT-IN-DB2`.

6. If you have access to OMEGAMON XE for z/OS you can confirm that the suspected thread is looping using the `Inspect CSECT CPU Utilization` panel for the thread. A looping thread will usually show 90-100% CPU usage over a very small address range within the looping program. This panel also gives you the name of the looping program (load module) and the offset where the loop is occurring, which can be used in diagnosing the program error.

If after making these checks you decide that the thread is looping, you need to consider the implications of cancelling the thread. These issues are covered in 16.4, "How to cancel a DB2 thread safely" on page 299.

Thread is in a logical loop executing SQL

It is possible for a DB2 thread to be in a loop where it is repeatedly executing the same SQL call. This typically occurs when an SQL call returns an unexpected SQLCODE (for example from a timeout or unavailable resource) that is not correctly handled by the program logic. This kind of loop is much harder to identify than a hard loop outside of DB2, but there are a number of good indications that you can obtain through OMEGAMON XE for DB2 PE. For a thread that is looping while executing an SQL call, you may have to confirm you diagnosis with someone who knows the program logic well, such as a developer.

Bear in mind that if the SQL is being executed in an unintentional loop, the values in the host variables are probably not changing. This means that the SQL will be repeatedly accessing the same pages in the DB2 tables and indexes.

You should check the following fields in OMEGAMON XE for DB2 PE to determine whether a thread is in a logical loop executing SQL:

1. From the `ALL THREADS CONNECTED TO DB2 (ZALLT, T.A)` panel press `PF11` to zoom to the `THREAD DETAIL` panel for the thread you suspect is looping, as shown in Figure 16-12.

```
>                                THREAD DETAIL
 PLAN
+ Thread:  Plan=DSNESPCS  Connid=TSO      Corrid=PAOLOR1      Authid=PAOLOR1
+ Attach:  TSO           Userid=PAOLOR1                       Region asid= 16K
+ Package: DSNESM68      Collection=DSNESPCS
 act
+ Thread Activity                         User Defined Functions
+ -------------------------------------   ------------------------------------
+ DB2 Status           =   IN-SQL-CALL    TCB Time (SQL)         = 00:00:00.000
+ MVS Status           =     USING CPU    Wait for TCB Time      = 00:00:00.000
+ Total Elapsed Time   =  00:01:40.343    Elapsed Time           = 00:00:00.000
+ CPU Utilization      =         43.0%    Elapsed Time (SQL)     = 00:00:00.000
+ Total CPU Time       =  00:01:25.882    SQL Events             =            0
```

Figure 16-12 Thread Detail

   Look at fields `DB2 Status, MVS Status and CPU Utilization` on panel `THREAD DETAIL (ZTDTL, T.ZOOM.A)`. For a thread looping while executing SQL you would expect to see consistently over several minutes:

   `DB2 Status` `IN-SQL-CALL`
   `MVS Status` `USING CPU`
   `CPU Utilization` > 80% (CPU rate on one processor)

   You need to be careful when looking at the field `CPU Utilization`. It may represent the CPU rate as a percentage of one processor (CP), or it may represent the rate spread over the number of processors available on your LPAR, depending on how your installation is configured. In the example in Figure 16-12 the displayed CPU rate of 43% actually represents a CPU rate of 86% on one processor, as the LPAR has two processors available.

   Depending on the SQL being repeatedly executed in the loop, the CPU rate may be lower than shown above, but it should be high and consistent.

2. Look at fields `Thread Status, Total SQL Reqs, SQL Statement Number and SQL Text` on panel `SQL CALL BEING EXECUTED (ZSQL, T.ZOOM.F)` as shown in Figure 16-13.

```
>                            SQL CALL BEING EXECUTED
 PLAN
+ Thread:  Plan=DSNESPCS  Connid=TSO      Corrid=PAOLOR1      Authid=PAOLOR1
+ Attach:  TSO            Userid=PAOLOR1                      Region asid= 16K
+ Package: DSNESM68       Collection=DSNESPCS
 call
+      SQL call is active, call information is as follows :
+
+      Thread Status  = IN-SQL-CALL    SQL Request Type     = DYNAMIC
+      Total SQL Reqs =     2103       SQL Call Type        = FETCH
+      SQL DBRM Name  = DSNESM68       SQL Statement Number = 00201
+      Collection ID  = DSNESPCS
+
+      SELECT COUNT(*) FROM PAOLOR4.EMPL
```

Figure 16-13 SQL Call Being Executed

For a thread looping while executing SQL you would expect to see consistently over several minutes:

`Thread Status`	`IN-SQL-CALL`
`Total SQL Reqs`	consistently increasing at a high rate
`SQL Statement Num.`	not changing
`SQL Text`	not changing

If the thread is looping on one SQL call, you should see the same `SQL Text` all the time. Make sure that the field `Total SQL Reqs` is consistently increasing at a high rate (more than 100 calls per second). If the `Total SQL Reqs` count is not increasing, it just means that DB2 is taking a long time to execute the SQL statement (usually `OPEN CURSOR`) and in this case it is not looping although the `SQL Text` is not changing.

3. Look at fields `Getpage Requests`, `Synchronous Read I/O`, `Seq Prefetch Requests`, `List Prefetch Requests` and `Dynamic Prefetch Requests` on panel `THREAD BUFFER POOL ACTIVITY (ZBUF,T.ZOOM.I)` as shown in Figure 16-14.

```
>                          THREAD BUFFER POOL ACTIVITY
 PLAN
+ Thread:  Plan=DSNESPCS  Connid=TSO      Corrid=PAOLOR1      Authid=PAOLOR1
+ Attach:  TSO            Userid=PAOLOR1                      Region asid= 16K
+ Package: DSNESM68       Collection=DSNESPCS
 buf
+ Buffer Pool: BP0
+
+ Getpage Requests       =     643417  Failed Getpage Requests    =         0
+ Synchronous Read I/O   =          0  Getpage/Read I/O           =  643417.00
+ Page Updates           =          0  Seq Prefetch Requests      =         0
+ List Prefetch Requests =          0  Dynamic Prefetch Requests  =         0
+ Prefetch Pages Read    =          0  Prefetch Pages In Hiperpool=         0
```

Figure 16-14 Thread Buffer Pool Activity

For a thread looping while executing SQL you would expect to see consistently over several minutes:

`Getpage Requests`	Either not changing at all, or increasing very rapidly (depending on the SQL being executed)
`Synchronous Read I/O`	Not increasing as all pages are in the buffer

`Seq Prefetch Requests`	Not increasing as all pages are in the buffer
`List Prefetch Requests`	Not increasing as all pages are in the buffer
`Dynamic Prefetch Requests`	Not increasing as all pages are in the buffer

Because we do not expect the host variables in the program to be changing when SQL is being executed in an unintentional loop, the referenced pages will already be in the buffer. So we would not expect to see any I/O activity. In some circumstances (for example a repeated FETCH with SQLCODE = 100) there is also no get page activity.

4. If the SQL statement is an `INSERT, UPDATE or DELETE` statement, then it is probably failing each time with a negative SQLCODE when the thread is in a SQL loop. In this case the field `Total SQL Reqs` on panel `SQL CALL BEING EXECUTED` will be increasing at a high rate (the SQL calls are registered even though they are failing with a negative SQLCODE). However, as the updates are failing, there will be no change in the number of page-level or row-level X locks being taken. You can check the locks by looking at field `Percent NUMLKUS` on panel `LOCKS/CLAIMS OWNED BY A THREAD (ZLOCKO, T.ZOOM.D)`.
5. If your analysis shows that an `INSERT` or `DELETE` statement may be repeatedly failing with a negative SQLCODE or SQLCODE = 100 in a loop, you can check your diagnosis using the following SQL query for the DB2 table `owner.table_name` referenced in the `INSERT` or `DELETE`:

```
SELECT COUNT(*)
FROM owner.table_name
WITH UR ;
```

If the SQL is repeatedly failing in a loop, then the COUNT(*) will not change with repeated executions.

If after making these checks you decide that the thread is looping, you need to consider the implications of cancelling the thread. These issues are covered in 16.4, “How to cancel a DB2 thread safely” on page 299.

16.4 How to cancel a DB2 thread safely

The most common reasons for needing to cancel a DB2 thread are:

- You have determined that the thread is in a loop.
- The thread is not in a loop but is holding resources (such as locks) needed by a higher priority transaction or job.
- The thread has been running for an unacceptably long time.
- Your system is under stress and you need to remove low priority work.
- A job was started at the wrong time or with the wrong parameters.
- The job or transaction is producing errors.
- You need to bind a package held by the job or transaction.

Whatever the reason, the need to cancel DB2 threads occurs regularly in an operational environment, and it is important that you are able to assess the likely impact of cancelling the thread.

How long will the thread take to back out?

If you are intending to cancel a DB2 thread, the most important question you need to answer is how long the backout will take. You need to try and estimate how much update activity has occurred in DB2 since the last time the thread issued a commit. If the thread also contains distributed access you need to try and assess the amount of update activity at the remote DB2 subsystem. To help you answer these questions you should look at the following fields in OMEGAMON XE for DB2 PE:

1. Look at fields `Insert, Delete and Update` on panel `CURRENT SQL COUNTS (ZTSCNT, T.ZOOM.G)` as shown in Figure 16-15.

```
>                              CURRENT SQL COUNTS
 PLAN
+ Thread:  Plan=AX888B    Connid=BATCH    Corrid=AX888K01     Authid=BATCH
+ Attach:  BATCH          JOB Name=AX888K01                   JOB Asid=  59
+ Package: AX888B         Collection=AX
 sqls
+ Commit         =       0  Abort          =       0  Select          =  578169
+ Open Cursor    =  237296  Close Cursor   =  237295  Fetch           =  408455
+ Insert         =  148149  Delete         =       0  Update          =       0
```

Figure 16-15 Current SQL Counts

If fields `Insert, Delete, and Update` all have a count of zero, then no update activity has occurred on the local DB2 subsystem (we are excluding the possibility of DDL here as this would not normally be part of scheduled operational activity). If there has also been no distributed update activity, then there is nothing for DB2 to back out and cancelling the thread should not cause any problems. If there has been update activity, as in Figure 16-15, then you need to investigate further.

2. Look at field `Commit` on panel `CURRENT SQL COUNTS (ZTSCNT, T.ZOOM.G)` as shown in Figure 16-15. Check how quickly the commit count is going up. If the thread is cancelled DB2 will only back out to the last commit point. If commits are occurring once every minute or more often, then the backout time will be correspondingly short and cancelling the thread should cause no problems. If commits occur very infrequently or not at all, then you need to investigate further.

3. Look at field `Total Elapsed Time` on panel `THREAD DETAIL (ZTDTL, T.ZOOM.A)` as shown in Figure 16-16.

```
>                                 THREAD DETAIL
 PLAN
act
+ Thread Activity                        User Defined Functions
+ ---------------------------------------  ------------------------------------
+ DB2 Status          =   IN-SQL-CALL   TCB Time (SQL)         = 00:00:00.000
+ MVS Status          =     WAIT-MISC   Wait for TCB Time      = 00:00:00.000
+ Total Elapsed Time  =  00:35:15.051   Elapsed Time           = 00:00:00.000
+ CPU Utilization     =         01.6%   Elapsed Time (SQL)     = 00:00:00.000
+ Total CPU Time      =  00:06:50.981   SQL Events             =            0
```

Figure 16-16 Thread Detail

If the `Total Elapsed Time` for the thread is not large (less than 15 minutes) then the maximum total backout time is not likely to be much longer than this (it could be much less, depending on the update activity), as archive logs should not be needed for the backout. If the `Total Elapsed Time` is much longer than 15 minutes, then you need to investigate further.

4. If you have established that the thread has been running for a long time without committing, and has executed `INSERTS, DELETES or UPDATES`, then you need to try and assess how much update activity has taken place. The real issue here is the number of log records that DB2 has written without committing. The most reliable way to find this out is to check for DB2 message `DSNJ031I` in the DB2 MSTR address space as shown in Example 16-1.

Example 16-1 Message DSNJ031I

```
05.39.43 STC12926  DSNJ031I  ?DB8A DSNJW001 WARNING - UNCOMMITTED UR  453
   453                       HAS WRITTEN 1000000 LOG RECORDS -
   453                       CORRELATION NAME = JOB00A
   453                       CONNECTION ID  = BATCH
   453                       LUWID = NET0001.DB8A.BE614B66E7EE = 16911
   453                       PLAN NAME = DSNTEP2
   453                       AUTHID = BATCH
   453                       END USER ID = *
   453                       TRANSACTION NAME = *
   453                       WORKSTATION NAME = *

10.39.25 STC12926  DSNJ031I  ?DB8A DSNJW001 WARNING - UNCOMMITTED UR  257
   257                       HAS WRITTEN 14000000 LOG RECORDS -
   257                       CORRELATION NAME = JOB00A
   257                       CONNECTION ID  = BATCH
   257                       LUWID = NET0001.DB8A.BE614B66E7EE = 16911
   257                       PLAN NAME = DSNTEP2
   257                       AUTHID = BATCH
   257                       END USER ID = *
   257                       TRANSACTION NAME = *
   257                       WORKSTATION NAME = *
```

You will only see this message if DSNZPARM parameter URLGWTH has been changed from its default value of zero. The parameter URLGWTH multiplied by 1000 specifies the number of log records that can occur in an uncommitted unit of recovery (UR) before message DSNJ031I is written. In Example 16-1 the DSNZPARM parameter URLGWTH = 1000 and message DSNJ031I is written for every 1000000 uncommitted log records in the UR. You can also see from Example 16-1 that the thread shown has not committed for at least five hours, and in this time has written 14000000 log records. A cancel of this thread is likely to require a very long backout period, possibly lasting several hours.

Cancelling the thread

If a thread has performed updates within DB2 it is generally safer to cancel the thread from within DB2 using the CANCEL THREAD command than to cancel the job directly with an operator command. This is because DB2 can terminate the thread in a more controlled way through the CANCEL THREAD command. You can use OMEGAMON XE for DB2 PE to help you cancel a DB2 thread. Look at field Token on panel CANCEL THREAD (ZTCANT, T.ZOOM.P) as shown in Figure 16-17.

```
>                                CANCEL THREAD
 PLAN
+ Thread:  Plan=KO2PLAN   Connid=DB2CALL  Corrid=D8F202S      Authid=D8F202S
+ Attach:  BATCH          JOB Name=D8F202S                    JOB Asid= 166
+ Package: DGO§WR2C       Collection=KO2OM310
 tcmd
+    Press ENTER to CANCEL the thread with Token=000398
+    Press PF3 to terminate the CANCEL request.
```

Figure 16-17 Cancel Thread

To cancel the thread shown on this panel, you would use the Token displayed and issue the DB2 command:

-CANCEL THREAD(398)

Or you can press Enter on the Cancel Thread panel to get OMEGAMON XE for DB2 PE to issue this command for you.

Restriction: In OMEGAMON XE for DB2 PE V3.1.0, at the time of writing this book, the Cancel Thread panel is not generating the correct syntax for the `CANCEL THREAD` command, and the command fails without notifying you of any error. Having identified the token for the thread, you have to issue the `CANCEL THREAD` command yourself from the DB2 Console panel.

Cancelling without backout

If you are expecting a very long backout time after issuing a `CANCEL THREAD` command, you can reissue the command as:

```
-CANCEL THREAD(token) NOBACKOUT
```

Because no further backout will be executed for the thread after this command is issued, the thread will terminate almost immediately. You need to be aware that any tablespaces updated by the thread since the last commit point are placed in `RECOVER PENDING` status after issuing a cancel command with `NOBACKOUT`. You need to be sure that the data in these tablespaces is unimportant or can be recovered by other means.

Reducing backout time

If you decide that for data consistency reasons a cancelled thread must be allowed to fully back out, you may be able to reduce the backout time if your archive logs are HSM migrated. Look at the archive log allocations using panel `SYSTEM SERVICES ADDRESS SPACE - ALLOCATED DDNAMES AND DATASETS` (`ZPEEKAB`, `M.A.B`) as shown in Figure 16-18.

```
>                       ALLOCATED DDNAMES AND DATASETS

 PEEK  D8F2MSTR ASID=137, collected at 15:25:46
 ddns
+     DDname   Addr VolSer Sta,Dsp DSN
+     STEPLIB  A907 SBOXEA SHR,KEE DB8F8.SDSNEXIT
+              A907 SBOXEA SHR,KEE DB8F8.SDSNLOAD
+     BSDS1    A907 SBOXEA SHR,KEE DB8FU.D8F2.BSDS01
+     BSDS2    AA07 SBOXEB SHR,KEE DB8FU.D8F2.BSDS02
+     SYS00001 AA07 SBOXEB SHR,KEE DB8FU.D8F2.LOGCOPY1.DS01
+     SYS00002 AA07 SBOXEB SHR,KEE DB8FU.D8F2.LOGCOPY1.DS02
+     SYS00003 AA07 SBOXEB SHR,KEE DB8FU.D8F2.LOGCOPY1.DS03
```

Figure 16-18 Allocated DDNAMES and Datasets

This panel shows you the currently allocated archive logs (if any) that DB2 is using for backout. If your archive logs are HSM migrated, you can recall them just before DB2 needs them to reduce the wait for data set recalls.

16.5 How to find badly performing SQL

Before you can use tools like EXPLAIN to analyze badly performing SQL, you need to identify the SQL that is causing the problem. In a complex operational environment with a mix of batch and online work, this can be the most difficult part of the problem solving task. Depending on how long the thread exists, you need to take different approaches to try and find badly performing SQL.

Examining DB2 threads that exist for more than one minute

If the DB2 thread exists for more than one minute (a batch job or a long running transaction) you have a reasonable chance of seeing the SQL being executed. The longer the runtime of the thread, the better are your chances of identifying the badly performing SQL. Once the thread has terminated, you will no longer be able to see the text of the SQL call that was being executed. This means you need to navigate as quickly as possible through the panels when you become aware of a possible SQL problem. Often you only have a few seconds to find the SQL before the thread terminates. Figure 16-9 shows an example of an SQL call being executed.

```
>                          SQL CALL BEING EXECUTED
 PLAN
+ Thread:  Plan=AX01B    Connid=BATCH    Corrid=AX01B02     Authid=BAT
+ Attach:  BATCH          JOB Name=AX01B02                   JOB Asid= 127
+ Package: AX01B         Collection=C1
 call
+      SQL call is active, call information is as follows :
+
+      Thread Status  = IN-SQL-CALL    SQL Request Type     = STATIC
+      Total SQL Reqs =    23862       SQL Call Type        = SELECT
+      SQL DBRM Name  = AX01B          SQL Statement Number = 20321
+      Collection ID  = C1
+
+      SELECT ACCT , STATUS INTO : H , : H FROM TAB01 WHERE COUNT = : H
+      AND ACCT = : H AND ACCT_TYPE = 'A' AND ENT_ID = : H
+      QUERYNO 100
```

Figure 16-19   SQL Call Being Executed

When you are on panel `SQL CALL BEING EXECUTED` (`ZSQL, T.ZOOM.F`) as shown i n Figure 16-19, press Enter repeatedly (about once per second) and watch the fields `Thread Status`, `Total SQL Reqs` and `SQL Statement Number`:

- If the value of field `Thread Status` indicates that the thread is active in an SQL call each time you press `Enter` (typically showing status `IN-SQL-CALL, WAIT-SYNC-IO or WAIT-ASYNCRD`), the value of field `Total SQL Reqs` is not increasing, and field `SQL Statement Number` is not changing, then DB2 has been busy the whole time executing the text of the SQL call shown on the panel with the `SQL Statement Number` as shown. Unless you know that this SQL call is expected to take a long time, then you have identified your problem SQL and can copy it to an EXPLAIN tool for analysis.
- If, as you repeatedly press Enter, the value of field `Total SQL Reqs` is rapidly increasing (at 1000 requests per second or more), then it is unlikely that you have a badly performing SQL call. However, if you notice that the same SQL text and `SQL Statement Number` are showing all the time, this indicates that this SQL call is being repeatedly executed. In this case, although there is probably no serious problem with the SQL call, it might be worth analyzing it to see if there is potential for tuning.
- If you are watching a long running batch job, be aware that the pattern of SQL calls may change as the job progresses. For instance, it is common to open cursors at the beginning of a batch job with SQL that may take several minutes to execute the OPEN calls. If you only watch the start of the job you may get a false impression that these OPEN calls are causing a problem.
- If the `Total SQL Reqs` are rapidly rising (more than 1000 requests per second) and the SQL text is constantly changing as you repeatedly press Enter, then it is unlikely that you have a problem with the SQL calls in the thread.

Examining DB2 threads that exist for less than one minute

For DB2 threads that exist for less than a minute, it is extremely difficult to look at the SQL calls before the thread terminates. A typical problem would be an online transaction that normally runs in under a second, which is now taking 10 to 20 seconds to complete. Although badly performing SQL is the most likely cause in this case, it can be extremely difficult to look at the SQL using the panel `SQL CALL BEING EXECUTED` (`ZSQL, T.ZOOM.F`) before the thread terminates. All of the following techniques for such difficult cases have problems associated with them, but they can produce useful results.

- If you know that the SQL causing the problem is likely to be dynamic SQL, you should switch on near-term history collection, if it is not already started, as described in 9.2, "How to access and control near-term history collection" on page 138. You can then look at the complete accounting record information for the thread after it has terminated. This will include the text of any dynamic SQL calls executed. However, it will not show you the elapsed or CPU time for the individual SQL calls, nor can you see details of any static SQL executed.
- You can use the Application Trace Facility (ATF) as described in Chapter 10, "Application Trace Facility" on page 163. This will give the elapsed and CPU time for both static and dynamic SQL in the thread, although it does not show the SQL text.

  The main disadvantage of this approach is that it is often difficult to qualify the trace sufficiently to prevent a hugh volume of trace output being generated within a few seconds of starting the trace. It can then be very difficult to identify the thread you are interested in, or the trace buffer may have filled and become inactive before recording the thread you want to see.
- The third approach is to use exception triggering with screen logging as described in 16.2, "Using exception triggering with screen logging" on page 287. Set up the `IDBT` exception "Time DB2 has been processing thread" with a rule specifying the transaction you want to capture and a threshold value of one second (field `THRSHOLD in RULE=01`). All of the commands you specify in the `ZTHLOG` panel, including the command "`call`" to specify the SQL text of static and dynamic SQL calls, will be written to the OMEGAMON XE for DB2 PE log each time the exception is triggered.

  Figure 16-20 shows how to set up the `IDBT` exception and associated rule. Note that you should set the field `Threshold` in the top part of the panel to a high value (like 999999) so that the exception is only triggered by threads that meet the conditions specified in `RULE=01`. This approach is more complicated to set up than the other two, but can sometimes provide the required information when the other approaches fail.

```
>              SET EXCEPTION ANALYSIS OPTIONS FOR IDBT EXCEPTION
: IDBT
+      DISPLAY Parameters:   THRESHOLD Parameters:   XLF Parameters:
:        State=ON             Threshold=999999        Auto=ON
:        Group=TH             Display=CLR3            Log=ON
:        Bell=OFF             Attribute=NONE          Limit=5 (5)
:      BOX Parameters:       CYCLE Parameters:        Repeat=YES
:        Boxchar=NO BOX       ExNcyc=0                Persist=1
:        Boxclr=NONE          Stop=0 (0)              SL=ZTHLOG
XTHD IDBT
: FUNCTION=ADD(01)             ! (DIS/ADD/DEL)
+-------------------------------------------------------------------------------
+ RULE=01 ! Thread Exception Rule Parameters:
:         !  PLAN=planname     AUTHID=authid _     CONNTYPE=________
:         !  CICS Parameters:  CICSTRAN=tranname   CICSCONN=________
:         !  IMS Parameters:   PSBNAME=________    IMSID=________
:         !  PACKDBRM=packname
:         !  PAKCOLID=__________________
:         !  CORRID=_____________
:         !  REQLOC=________________
:         !  THRDTYPE=__
:         !  THRSHOLD=1_______  EXCLUDE=NO
```

Figure 16-20 Set Exception Analysis Options for IDBT Exception

16.6 Dealing with lock escalation problems

Lock escalation problems occur typically in batch jobs that have heavy update activity and commit infrequently. There are two DSNZPARM parameters that define how the lock escalation process operates:

- NUMLKTS defines the default value for the maximum number of page/row locks that a thread can hold on a single table space. This default value will be used when the table space is defined with LOCKMAX=SYSTEM. When this limit is reached for a table space, DB2 will try to replace the page/row level locks with a single table space level lock (unless the table space is defined with LOCKMAX = 0). This process is known as lock escalation. A typical value for NUMLKTS is 5000.
- NUMLKUS defines the total number of page/row locks across all table spaces that a thread can hold. When this limit is reached SQLCODE = -904 is returned to the program, which normally causes the program to abend. A typical value for NUMLKUS is 20000.

You can check the values of these parameters by looking at fields Max/Tablespace (NUMLKTS) and Max/User (NUMLKUS) on panel DSNZPARM IRLM PARAMETERS (ZPRLM, R.H.F) as shown in Figure 16-21.

```
>                           DSNZPARM IRLM PARAMETERS
 DSNZ
+ DSNZPARM Module = DSNZPARM            Assembly Date = 11/20/05
+ Initial  Module = DSNZPARM            Assembly Date = 11/20/05
+ Previous Module = DSNZPARM            Assembly Date = 11/20/05

 prlm
+   Max/Tablespace (NUMLKTS)   =   5000   IRLM Proc (IRLMPRC)       = DB01IRLM
+   Max/User (NUMLKUS)         =  20000   IRLM Subsys (IRLMSID)     =     IR01
+   Timeout Interval (IRLMRWT) =     10   Auto Start Wait (IRLMSWT) =       60
+   Auto Start IRLM (IRLMAUT)  =    YES   Utility Timeout (UTIMOUT) =       12
+   Number Local/Global Cycles =      0   Wait Time Local Deadlock  =        0
+   MVS Lock Table Hash Entry=        0   IRLM Maximum CSA Usage    =        0
+   PC Yes Specified           =    ...   Pending Number Hash Entries =      0
+   Timeout Interval           =      0
```

Figure 16-21 DSNZPARM IRLM Parameters

To prevent a batch job abending with SQLCODE = -904 because the NUMLKUS limit has been reached, it is important that the individual table spaces reach their NUMLKTS limit first, and have their page/row level locks replaced by table space level locks (lock escalation).

To see how many page/row level locks are being held for individual table spaces and what percentage of NUMLKUS a thread has reached, look at panel LOCKS/CLAIMS OWNED BY A THREAD (ZLOCKO, T.ZOOM.D) as shown in Figure 16-22.

```
                         LOCKS/CLAIMS OWNED BY A THREAD
 PLAN
+ Thread:  Plan=AX110     Connid=CICSA01  Corrid=ENTRAX010035 Authid=CICS
+ Attach:  CICS   JOB=CICSA01   Tran=AX01  Task#=29741  Term=NONE  Type=ENTY
+ Package: AX467          Collection=C1
 own
+                         Lock Ownership Information
+      Percent NUMLKUS       = 70.8       Total  Locks Owned        = 14152
+      Total Catalog Locks   = 1          Pageset and Dataset Locks = 55
+      Catalog Pageset Locks = 1          Page/Row Locks            = 13931
+      Catalog Page/Row Locks= 0          Directory and Other Locks = 166
+      Bind ACQUIRE option   = USE        Bind RELEASE option       = DEALLOC
+      ISOLATION option      = Cursor Stability
+
+      Type    Level       Resource                            Number
+      ----    -----       ------------------------------------ ------
+      DPAG        X       DB=AX111D        PS=AXLWDS             133
+                  X       DB=AX111D        PS=AXBCPS            2356
+                  U       DB=AX111D        PS=AXRDPS            4527
+                  X       DB=AX111D        PS=AXESTS            3458
+                  X       DB=AX111D        PS=AXNRVS            3457
```

Figure 16-22 Locks/Claims Owned by a Thread

From this panel you can identify which table spaces are holding a large number of page/row level locks that have not been escalated. If necessary, to prevent NUMLKUS being reached, you may need to alter some of the table spaces to reduce the LOCKMAX value below the default value of NUMLKTS. These table spaces will then have their locks escalated to table space level at a lower lock count threshold, helping to prevent NUMLKUS being reached.

To reduce the `value of the LOCKMAX` parameter for a table space (the value at which lock escalation occurs) to 1000 execute the following SQL:

```
ALTER TABLESPACE dbname.tsname
LOCKMAX 1000 ;
```

If a batch job repeatedly experiences problems because the `NUMLKUS` limit is reached, then consider taking one of the following courses of action:

- Increase the commit frequency for the program.
- Add commit processing to the program if it has none.
- Explicitly lock some of the table spaces in the program using the `LOCK TABLESPACE` statement.

16.7 Dealing with deadlocks and timeouts

The first place to look if you are experiencing deadlocks or timeouts is panel `LOCKING CONFLICTS`, which is option L from the OMEGAMON XE for DB2 PE Main Menu. This is shown in Figure 16-23.

```
>                              LOCKING CONFLICTS

 XLOK
+Stat Plan     Corrid       Type Lvl Resource
+---- -------- ------------ ---- --- ----------------------------------------
+ OWN AXC10B   AXC10L01     DPAG   X DB=AERTD   PS=AERTS    PG=03C
+WAIT AX780B   AX780S01     DPAG   S DB=AERTD   PS=AERTS    PG=03C
+WAIT AX346B   AX346R01     DPAG   S DB=AERTD   PS=AERTS    PG=03C
```

Figure 16-23 Locking Conflicts

From this panel you can position the cursor on a thread that is waiting for a lock and press `PF11 (ZOOM)` to go to the thread detail panel. Then select Option F to see the SQL call that is waiting for the lock. Knowing the SQL call that is waiting for a lock will often help to determine why the lock wait is occurring, especially if you use EXPLAIN to show the access path for the call.

In practice, the above procedure can be very difficult to carry out, for the following reasons:

- Deadlocks and timeouts usually occur very infrequently, so they are hard to detect.
- The locking conflict only persists for a few seconds before one of the conflicting threads receives a timeout condition, again making detection very difficult.
- For the above reasons, even after you see a locking conflict, there is not generally enough time to trace it through to the `SQL CALL BEING EXECUTED` panel before the thread terminates.

Because of these problems a very useful approach is to use exception triggering with screen logging as described in 16.2, "Using exception triggering with screen logging" on page 287. Set up the `WTRE` exception "Time thread waiting for a resource" with a rule specifying the transaction you want to capture and a threshold value of five seconds (field `THRSHOLD in RULE=01`). All the commands you specify in the `ZTHLOG` panel, including the command **"**`call`**"** to specify the SQL text of static and dynamic SQL calls, will be written to the OMEGAMON XE for DB2 PE log each time the exception is triggered.

Figure 16-24 shows how to set up the `WTRE` exception and associated rule. Note that you should set the field `Threshold` in the top part of the panel to a high value (like 999999) so that the exception is only triggered by threads that meet the conditions specified in `RULE=01`.

```
>             SET EXCEPTION ANALYSIS OPTIONS FOR WTRE EXCEPTION
: WTRE
+      DISPLAY Parameters:   THRESHOLD Parameters:   XLF Parameters:
:        State=ON             Threshold=999999        Auto=ON
:        Group=TH             Display=CLR3            Log=ON
:        Bell=OFF             Attribute=NONE          Limit=5 (5)
:      BOX Parameters:       CYCLE Parameters:        Repeat=YES
:        Boxchar=NO BOX       ExNcyc=0                Persist=1
:        Boxclr=NONE          Stop=0 (0)              SL=ZTHLOG
XTHD WTRE
: FUNCTION=ADD(01)             ! (DIS/ADD/DEL)
+-------------------------------------------------------------------------------
+ RULE=01 ! Thread Exception Rule Parameters:
:         !  PLAN=planname     AUTHID=authid _     CONNTYPE=________
:         !  CICS Parameters:  CICSTRAN=tranname   CICSCONN=________
:         !  IMS Parameters:   PSBNAME=________    IMSID=________
:         !  PACKDBRM=_________
:         !  PAKCOLID=__________________
:         !  CORRID=_____________
:         !  REQLOC=________________
:         !  THRDTYPE=__
:         !  THRSHOLD=5_______ EXCLUDE=NO
```

Figure 16-24   Set Exception Analysis Options for WTRE Exception

Once this exception triggering has been set up, it can be left running unattended for several hours until the deadlock or timeout you want to capture occurs. You can then analyze the details of the thread, including which SQL call was being executed at the time of the deadlock or timeout, using the panels captured in the OMEGAMON XE for DB2 PE log.

Part 6

Appendixes

This part of the book contains two appendixes:

- Appendix A, “DB2 Workload Generator” on page 311 briefly describes the setup of the workload generator used for our tests.
- Appendix B, “Additional material” on page 315 describes how to locate and download the DB2 Workload Generator.

A

DB2 Workload Generator

This appendix describes the DB2 Workload Generator V1. This tool was written to provide a simple way of creating and driving a substantial workload on a DB2 for z/OS database.

IBM DB2 Workload Generator V1

The purpose of this tool is to create and drive a workload on a DB2 for z/OS database.

The generator is designed to be run from either TSO Batch or Windows, although the only action permitted from within Windows is RUN.

The tool can be used with all the DB2 tools:

Change Management Create a development and production database, make changes to development, and compare the difference.

Performance management Run the workload and observe with IBM Tivoli OMEGAMON XE for DB2 Performance Monitor / Expert, capture the SQL for analysis with QM (Query Monitor).

Recovery management Examine the logs with LAT (Log Analysis Tool), use Change Accumulation Tool to update image copies.

This tool is composed of five principal tables (GLWTDPT, GLWTEMP, GLWTPRJ, GLWTPJA and GLWTEPA) and 12 supporting tables, 14 Stored Procedures, and 12 supporting procedures, and finally, exploit views, referential integrity, partitioned tables, and triggers.

Before you run, you need to customize the GLWRUN member to adapt to your environment.

There are four possible actions that you can use:

- RUN — Run the workload.
- BUILD — Create DB2 objects (database, table space, etc..., bind packages, define GDG for image copies).
- DROP — Drop the database if created by the same user, drop stored procedure, free package, and delete GDGs and image copies.
- BIND — Bind the packages.

The job example is shown in Example A-1.

Example: A-1 Job example of DB2 Workload Generator with action RUN

```
//*****************************************************************
//WORKLOAD EXEC PGM=IKJEFT01,DYNAMNBR=100
//*****************************************************************
//STEPLIB  DD DSN=DB8A8.SDSNLOAD,DISP=SHR
//SYSEXEC  DD DSN=PAOLOR4.GLW.EXEC.CUST,DISP=SHR
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSTSIN  DD *
%GLWRUN DB2SSID(DB8A) SCHEMA(GLWSAMP) ACTION(RUN)          -
        RUNTIME(3)  RUNMODE(RANDOM) RUNPROF(STANDARD)  -
        WAITTIME(0)        -
        BUILDRI(DB2)       -
        GDGLIMIT(0)        -
        EXPLAIN(Y)             -
        WLMENV(DB8AWLM1)       -
        LOG(SUMMARY)
```

The output of this job is shown in Example A-2.

Example: A-2 Output of DB2 Workload Generator job with action RUN

```
GLWR141I: Execution of Stored Procedures
    1 EMPADD   19:35:48 ; Objno:    3024 ; Tran:    6107 ; SQLrc: 0
    2 EMPADD   19:35:48 ; Objno:    3025 ; Tran:    6108 ; SQLrc: 0
    3 DPTADD   19:35:48 ; Objno:     308 ; Tran:    6109 ; SQLrc: 0
    4 EMPADD   19:35:48 ; Objno:    3026 ; Tran:    6110 ; SQLrc: 0
...
...
Table row count before and after report
Table name                      Before      After Difference
GLWSAMP.GLWTDPT                    234        404        170
GLWSAMP.GLWTEMP                   2935       4831       1896
GLWSAMP.GLWTENG                      0          0          0
GLWSAMP.GLWTEPA                  17914      30514      12600
GLWSAMP.GLWTJBS                     15         15          0
GLWSAMP.GLWTLCN                    130        130          0
GLWSAMP.GLWTLNG                      0          0          0
GLWSAMP.GLWTLNM                    351        351          0
GLWSAMP.GLWTPGW                     84         84          0
GLWSAMP.GLWTPJA                   7002      11862       4860
...
...
...

GLWR142I: Stored Procedure call summary
DPTADD   ran   208 times at an average elapsed of  0.009 seconds
DPTBAL   ran   171 times at an average elapsed of  0.011 seconds
DPTDEL   ran    38 times at an average elapsed of  1.721 seconds
DPTLCK   did not run
DPTMGR   ran   184 times at an average elapsed of  0.009 seconds
DPTUPD   ran   166 times at an average elapsed of  0.008 seconds
DPTUPR   ran   204 times at an average elapsed of  0.008 seconds
EMPADD   ran  1925 times at an average elapsed of  0.012 seconds
EMPDEL   ran    30 times at an average elapsed of  0.012 seconds
EMPFND   ran    83 times at an average elapsed of  0.015 seconds
EMPQRY   ran   117 times at an average elapsed of  0.251 seconds
EMPUPD   ran   152 times at an average elapsed of  0.012 seconds
PRJADD   ran   373 times at an average elapsed of  0.138 seconds
PRJUPD   ran   171 times at an average elapsed of  0.009 seconds

Total calls= 3822 ; Runtime=     3.0 Minutes
Transaction rate=    20.9 trans/sec
```

B

Additional material

This redbook refers to additional material that can be downloaded from the Internet as described below.

Locating the Web material

The Web material associated with this redbook is available in softcopy on the Internet from the IBM Redbooks Web server. Point your Web browser to:

ftp://www.redbooks.ibm.com/redbooks/SG247224

Alternatively, you can go to the IBM Redbooks Web site at:

ibm.com/redbooks

Select the **Additional materials** and open the directory that corresponds with the redbook form number, SG247224.

Using the Web material

The additional Web material that accompanies this redbook includes the following files:

File name	*Description*
GLW.V1R2.zip	Zipped version 1.2 of the DB2 Workload Generator

System requirements for downloading the Web material

The following system configuration is recommended:

Hard disk space:	13 MB
Operating System:	Windows
Processor:	Intel 386 or higher
Memory:	16 MB

How to use the Web material

Create a subdirectory (folder) on your workstation, and unzip the contents of the Web material zip file into this folder.

See Appendix A, "DB2 Workload Generator" on page 311 for the next steps.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information on ordering these publications, see "How to get IBM Redbooks" on page 318. Note that some of the documents referenced here may be available in softcopy only.

- *DB2 UDB for z/OS Version 8 Performance Topics,* SG24-6465-00
- *IBM DB2 Performance Expert for z/OS Version 2,* SG24-6867-01
- *IBM Tivoli OMEGAMON V3.1 Deep Dive on z/OS*, SG24-7155-00

Other publications

These publications are also relevant as further information sources:

- *Monitoring Performance from the OMEGAMON Classic Interface*, SC18-9659
- *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Getting Started,* GC18-9634
- *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Configuration and Customization,* GC18-9637
- *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Messages,* GC18-9638
- *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Monitoring Performance from ISPF,* SC18-9639
- *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Monitoring Performance from the Performance Expert Client,* SC18-9640
- *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Monitoring Performance from the OMEGAMON Classic Interface,* SC18-9659
- *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Report Command Reference,* SC18-9643
- *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Report Reference,* SC18-9642
- *IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS; IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS: Reporting User's Guide,* SC18-9641
- *Program Directory for IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS,* GI10-8698

- *Program Directory for IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS,* GI10-8699
- *z/OS V1R7.0 Security Server RACF Command Language Reference,* SA22-7687
- *DB2 Connect User's Guide Version 8,* SC09-4835

Online resources

These Web sites and URLs are also relevant as further information sources:

- Web site for IBM Tools for DB2 for z/OS, DB2 on Linux, UNIX, Windows, IMS, and Information Integration:

  `http://www.ibm.com/software/data/db2imstools/`

- Tivoli software information center:

  `http://publib.boulder.ibm.com/tividd/td/tdprodlist.html`

How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications, and Additional Materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Abbreviations and acronyms

AC	autonomic computing
ACS	automatic class selection
AIX®	Advanced Interactive eXecutive from IBM
APAR	authorized program analysis report
API	application programming interface
AR	application requester
ARM	automatic restart manager
AS	application server
ASCII	American National Standard Code for Information Interchange
B2B	business-to-business
BCDS	DFSMShsm™ backup control data set
BCRS	business continuity recovery services
BI	Business Intelligence
BLOB	binary large objects
BPA	buffer pool analysis
BSDS	boot strap data set
CBU	Capacity BackUp
CCA	channel connection address
CCA	client configuration assistant
CCP	collect CPU parallel
CCSID	coded character set identifier
CD	compact disk
CDW	central data warehouse
CEC	central electronics complex
CF	coupling facility
CFCC	coupling facility control code
CFRM	coupling facility resource management
CICS	Customer Information Control System
CLI	call level interface
CLOB	character large object
CLP	command line processor
CMOS	complementary metal oxide semiconductor
CP	central processor
CPU	central processing unit
CRCR	conditional restart control record
CRD	collect report data
CRUD	create, retrieve, update or delete
CSA	common storage area
CSF	Integrated Cryptographic Service Facility
CTE	common table expression
CTT	created temporary table
CUoD	Capacity Upgrade on Demand
DAC	discretionary access control
DASD	direct access storage device
DB	database
DB2	Database 2™
DB2 PE	DB2 Performance Expert
DBA	database administrator
DBAT	database access thread
DBCLOB	double-byte character large object
DBCS	double-byte character set
DBD	database descriptor
DBID	database identifier
DBM1	database master address space
DBRM	database request module
DCL	data control language
DDCS	distributed database connection services
DDF	distributed data facility
DDL	data definition language
DDL	data definition language
DES	Data Encryption Standard
DLL	dynamic load library manipulation language
DML	data manipulation language
DNS	domain name server
DPSI	data partitioning secondary index
DRDA®	Distributed Relational Data Architecture
DSC	dynamic statement cache, local or global
DSNZPARMs	DB2's system configuration parameters
DSS	decision support systems
DTT	declared temporary tables

DWDM	dense wavelength division multiplexer
DWT	deferred write threshold
DUCK	dynamic user configuration kernel
EA	extended addressability
EAI	enterprise application integration
EAS	Enterprise Application Solution
EBCDIC	extended binary coded decimal interchange code
ECS	enhanced catalog sharing
ECSA	extended common storage area
EDM	environmental descriptor manager
EJB™	Enterprise JavaBean
ELB	extended long busy
ENFM	enable-new-function mode
ERP	enterprise resource planning
ERP	error recovery procedure
ESA	Enterprise Systems Architecture
ESP	Enterprise Solution Package
ESS	Enterprise Storage Server®
ETR	external throughput rate, an elapsed time measure, focuses on system capacity
EWLC	Entry Workload License Charges
EWLM	Enterprise Workload Manager
FIFO	first in first out
FLA	fast log apply
FTD	functional track directory
FTP	File Transfer Program
GB	gigabyte (1,073,741,824 bytes)
GDDM	IBM Graphical Data Display Manager
GBP	group buffer pool
GDPS®	Geographically Dispersed Parallel Sysplex™
GLBA	Gramm-Leach-Bliley Act of 1999
GRS	global resource serialization
GUI	graphical user interface
HALDB	High Availability Large Databases
HPJ	high performance Java
HTTP	Hypertext Transfer Protocol
HW	hardware
I/O	input/output
IBM	International Business Machines Corporation
ICF	internal coupling facility
ICF	integrated catalog facility
ICMF	integrated coupling migration facility
ICSF	Integrated Cryptographic Service Facility
IDE	integrated development environments
IFCID	instrumentation facility component identifier
IFI	Instrumentation Facility Interface
IFL	Integrated Facility for Linux
IGS	IBM Global Services
IMS	Information Management System
IORP	I/O Request Priority
IPLA	IBM Program Licence Agreement
IRD	Intelligent Resource Director
IRLM	internal resource lock manager
IRWW	IBM Relational Warehouse Workload
ISPF	interactive system productivity facility
ISV	independent software vendor
IT	information technology
ITR	internal throughput rate, a processor time measure, focuses on processor capacity
ITSO	International Technical Support Organization
IVP	installation verification process
J2EE™	Java 2 Enterprise Edition
JDBC	Java Database Connectivity
JFS	journaled file systems
JNDI	Java Naming and Directory Interface
JTA	Java Transaction API
JTS	Java Transaction Service
JVM™	Java Virtual Machine
KB	kilobyte (1,024 bytes)
LCU	Logical Control Unit
LDAP	Lightweight Directory Access Protocol
LOB	large object
LPAR	logical partition
LPL	logical page list
LRECL	logical record length
LRSN	log record sequence number
LRU	least recently used

LSS	logical subsystem
LUW	logical unit of work
LVM	logical volume manager
MAC	mandatory access control
MB	megabyte (1,048,576 bytes)
MBps	megabytes per second
MLS	multi-level security
MQT	materialized query table
MTBF	mean time between failures
MVS	Multiple Virtual Storage
NALC	New Application License Charge
NFM	new-function mode
NFS	Network File System
NPI	non-partitioning index
NPSI	nonpartitioned secondary index
NVS	non volatile storage
ODB	object descriptor in DBD
ODBC	Open Database Connectivity
ODS	Operational Data Store
OLE	Object Link Embedded
OLTP	online transaction processing
OP	Online performance
OS/390	Operating System/390®
OSC	optimizer service center
PAV	parallel access volume
PCICA	Peripheral Component Interface Cryptographic Accelerator
PCICC	PCI Cryptographic Coprocessor
PDS	partitioned data set
PIB	parallel index build
PPRC	Peer-to-Peer Remote Copy
PR/SM™	Processor Resource/System Manager
PSID	pageset identifier
PSP	preventive service planning
PTF	program temporary fix
PUNC	possibly uncommitted
PWH	Performance Warehouse
QA	Quality Assurance
QMF	Query Management Facility
QoS	Quality of Service
QPP	Quality Partnership Program
RACF	Resource Access Control Facility
RAS	reliability, availability and serviceability
RBA	relative byte address
RBLP	recovery base log point
RDBMS	relational database management system
RDS	relational data system
RECFM	record format
RI	Referential Integrity
RID	record identifier
ROI	return on investment
RPO	recovery point objective
RR	repeatable read
RRS	resource recovery services
RRSAF	resource recovery services attach facility
RS	read stability
RTO	recovery time objective
RTS	real-time statistics
SAN	storage area networks
SBCS	store single byte character set
SCUBA	self contained underwater breathing apparatus
SDM	System Data Mover
SDP	Software Development Platform
SLA	service-level agreement
SMIT	System Management Interface Tool
SOA	service-oriented architecture
SOAP	Simple Object Access Protocol
SPL	selective partition locking
SQL	Structured Query Language
SQLJ	Structured Query Language for Java
SRM	Service Request Manager
SSL	Secure Sockets Layer
SU	Service Unit
TCO	total cost of ownership
TPF	Transaction Processing Facility
UA	Unit Addresses
UCB	Unit Control Block
UDB	Universal Database
UDF	user-defined functions
UDT	user-defined data types
UOW	unit of work
UR	unit of recovery
USS	UNIX System Services

vCF	virtual coupling facility
VIPA	Virtual IP Addressing
VLDB	very large database
VM	virtual machine
VSE	Virtual Storage Extended
VSIP	Visual Studio® Integrator Program
VWLC	Variable Workload License Charges
wizards	Web-based assistants
WLC	Workload License Charges
WLM	Workload Manager
WSDL	Web Services Definition Language
WTO	write to operator
XA	Extended Architecture
XML	Extensible Markup Language
XRC	eXtended Remote Copy
z800	zSeries 800
z890	zSeries 890
z990	zSeries 990
zAAP	zSeries Application Assist Processor
zELC	zSeries Entry License Charge

Index

Numerics

A

B

C

D

E

F

G

H

I

P

Q

A Deep Blue View of DB2 Performance: IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS

(0.5” spine)
0.475”<->0.873”
250 <-> 459 pages

A Deep Blue View of DB2 Performance

IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS

Explore the combined functions of the new performance tool

Detect issues and apply thresholds in realistic scenarios

Choose the right function for the problem on hand

IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS represents the effort on converging OMEGAMON XE for DB2 and DB2 Performance Expert into one product that retains the best features of each. This new tool gives you a single, comprehensive tool to help assess the efficiency of and optimize performance from your DB2 Universal Database in the z/OS environment. It automates the analysis of your database performance in real time and also adds expert database analysis functions to help you maximize performance and enhance productivity.

The main functions of this tool allow you to:

- Monitor, analyze, and tune the performance of IBM DB2 Universal Database and DB2 applications on z/OS
- Improve productivity with meaningful views of performance
- Quickly and easily identify performance bottlenecks using predefined rules of thumb
- Enjoy substantial breadth and depth in monitoring DB2 environments by combining batch-reporting capabilities with real-time monitoring and historical tracking functions
- Support an enterprise-wide integrated systems management strategy activated by the IBM Tivoli OMEGAMON XE family
- Store performance data and analysis tools in a performance warehouse

The software combines the sophisticated reporting, monitoring, and buffer pool analysis features of the IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS and IBM DB2 Buffer Pool Analyzer products.

This IBM Redbook will help you install and understand the main functions of the product, clarify the differences, and point out the advantages if you had one of the pre-existing products already in use.

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

SG24-7224-00 ISBN 0738497371