Tivoli software

IBM

IBM Tivoli Monitoring for Databases

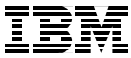
Database Management Made Simple

- Extensive installation and migration examples
- In-depth coverage of resource models and tasks
- Integration to TBSM and TEDW

Budi Darmawan
Bart Jacob
Edson Manoel
Ebrahim Rahim
Baswa Shaker Shamshabad

Redbooks

ibm.com/redbooks



International Technical Support Organization

IBM Tivoli Monitoring for Databases
Database Management Made Simple

October 2002

SG24-6613-00

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

First Edition (October 2002)

This edition applies to Version 5.1 of IBM Tivoli Monitoring for Databases (product number 5724-B96).

Note: This book is based on a pre-GA version of a product and may not apply when the product becomes generally available. We recommend that you consult the product documentation or follow-on versions of this redbook for more current information.

Contents

Figures

Tables

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®	OS/2®	Tivoli Enterprise™
AIX 5L™	OS/390®	Tivoli Enterprise Console®
Database 2™	Redbooks™	Tivoli Management
DB2®	Redbooks(logo)™ 	Environment®
DB2 Universal Database™	S/390®	TME®
IBM®	SP™	WebSphere®
Informix®	Tivoli®	

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

Lotus®	Notes®	Word Pro®

The following terms are trademarks of other companies:

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

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

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

C-bus is a trademark of Corollary, Inc. in the United States, other countries, or both.

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

SET, SET Secure Electronic Transaction, and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

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

Preface

This IBM Redbook will help you install, tailor and configure the new IBM Tivoli Monitoring for Databases Version 5.1.0 (5724-B96). This product actually consists of the following three modules:

- IBM Tivoli Monitoring for Databases: DB2
- IBM Tivoli Monitoring for Databases: Informix
- IBM Tivoli Monitoring for Databases: Oracle

In this redbook, we show the installation and migration steps for the product, listing and customizing the monitoring feature, and discussing some operational scenarios. We also provide integration of the IBM Tivoli Monitoring for Databases with other Tivoli product to provide higher level of management, such as real time business impact management with Tivoli Business Systems Manager and Tivoli Enterprise Console, and predictive management with Tivoli Enterprise Data Warehouse.

This redbook also discuss some detailed product structure with tips on performing troubleshooting for the IBM Tivoli Monitoring for Databases.

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, Austin Center.

Budi Darmawan is a Consulting IT Specialist at the International Technical Support Organization, Austin Center. He writes extensively and teaches IBM classes worldwide on all areas of systems management and database. Before joining the ITSO three years ago, Budi worked in IBM Global Services Integrated Technology Services in IBM Indonesia as Technical lead and Solution Architect. His areas of expertise including Tivoli solutions, database administration, business intelligence, and OS/390 administration.

Bart Jacob is a Senior Technical Staff Member at IBM Corp - International Technical Support Organization, Austin Center. He has 21 years of experience providing technical support across a variety of IBM products and technologies, including communications, object-oriented software development, and systems management. He has ten years of experience at the ITSO, where he has been writing IBM Redbooks and creating and teaching workshops around the world on

a variety of topics. He holds an MS degree in Numerical Analysis from Syracuse University.

Edson Manoel is a Software Engineer at IBM Corporation - International Technical Support Organization, Austin Center. He applies his extensive field experience as an IT Specialist to his work at the ITSO in the Systems Management area. Prior to joining the ITSO, Edson worked in the IBM Software Group as a Tivoli Technology Ambassador and in IBM Brasil Professional Services Organization as a Certified IT Specialist. He was involved in numerous projects, designing and implementing systems management solutions for IBM customers and Business Partners. Edson holds a BSc degree in Applied Mathematics from Universidade de Sao Paulo, Brazil.

Ebrahim Rahim is a Tivoli Certified IT Specialist from South Africa. He has two years, nine months experience in systems management at IBM South Africa. His areas of expertise include Tivoli Framework, Inventory and Distributed Monitoring. He holds an ND IT qualification, and is currently busy with his Btech IT.

Baswa Shaker Shamshabad is a Senior Consultant with G4 Technologies Inc., a Boston based IBM Business Partner. He has 11 years of experience in Systems Integration. His areas of expertise include WebSphere family products (WebSphere MQ/WMQI/Workflow), DB2, and Tivoli. He worked as an Enterprise Application Integration solution architect in Asia Pacific, India, and the U.S. He has written extensively on Enterprise Application Integration and Information Systems security. He holds a master degree in finance from IIF, Delhi and MBA (Systems) from NITIE, Mumbai. He can be reached at shaker@g4soft.com.

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

Morten Moeller and Wade Wallace
International Technical Support Organization, Austin Center

Julio Perez
Resident of the IBM Tivoli Monitoring for Business Integration project

Shajeed Massimiliano
Resident of the IBM Tivoli Monitoring for Web Infrastructure project

Erica Smith, Ndu Emunchay, Reid Minyen, Tony Rusho
IBM Tivoli Monitoring for Databases Product Introduction team

Bertina Jones
Product manager for IBM Tivoli Monitoring for Databases

Merleen Gray
Development manager for IBM Tivoli Monitoring for Databases

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 Internet note to:

  redbook@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. 0SJB Building 003 Internal Zip 2834
  11400 Burnet Road
  Austin, Texas 78758-3493

1

Introduction

This chapter introduces the concept of IBM Tivoli Monitoring for Databases and describes the overall product positioning within the Tivoli product structure. The discussion is comprised of:

- 1.1, “Database management” on page 2 discusses specific issues of database management that generally arise in an enterprise.
- 1.2, “Tivoli systems management product” on page 6 provides a background on the Tivoli product set that performs the overall systems management process.
- 1.3, “IBM Tivoli Monitoring” on page 8 specifically discusses the IBM Tivoli Monitoring product.
- 1.4, “Overview of IBM Tivoli Monitoring for Databases” on page 11 provides an overview of the IBM Tivoli Monitoring for Databases product and its relation to other products and components.
- 1.5, “Project environment and setup” on page 12 shows our environment and machines where we perform this testing.
- 1.6, “Document organization” on page 14 lists the overall structure of this redbook.

1.1 Database management

Databases are the heart of most business applications today. They provide a unique set of capabilities that provide a repository for persistent data that can be accessed quickly and reliably. The information contained in the database feeds various parts of any application in an enterprise. It is one of the most critical components of information technology.

Most of the databases used in the industry are relational database management system (RDBMS). These databases are accessed using a standard set of Structured Query Language (SQL). Generic interfaces for these databases, such as Open Database Connectivity (ODBC) or Java Database Connectivity (JDBC) are used widely. In this redbook, we will only cover relational database management.

As a critical component, database systems must:

- Have a fast response time
- Have timely and controlled availability of data
- Have efficient operations with the proper utilization of available resources
- Have reliability and data integrity protected from component failure or unauthorized access
- Keep pace with the organizational growth

A database administrator (DBA) is usually responsible for ensuring continuous and efficient operation of the databases. This database administration is a major issue in today's business environment.

These database management requirements need a tool to ensure that database operation and monitoring can be handled quickly, efficiently, and correctly by operation personnel. To achieve these objectives, one needs a systems management tool, such as IBM Tivoli Monitoring for Databases, to meet the wide ranging needs that depend on the environment, such as:

- The capability to monitor various performance metric of database
- A consistent interface to operate the database (to avoid errors)
- The ability to integrate to other monitoring and reporting tools

These capabilities will allow a database administrator to:

- Reduce the database response time
- Increase the database throughput
- Increase the processing efficiency of a database instance

- Tune resources, such as the CPU, memory, I/O, and so on
- Have uninterrupted availability of data to users
- Maintain data integrity
- Have easy monitoring of the database health

1.1.1 Architecting e-business application infrastructures

In a typical e-business environment, the application infrastructure consists of multiple tiers, and the communication between these tiers are restricted. Databases may be part of such a solution, providing communications both within a tier and across tiers.

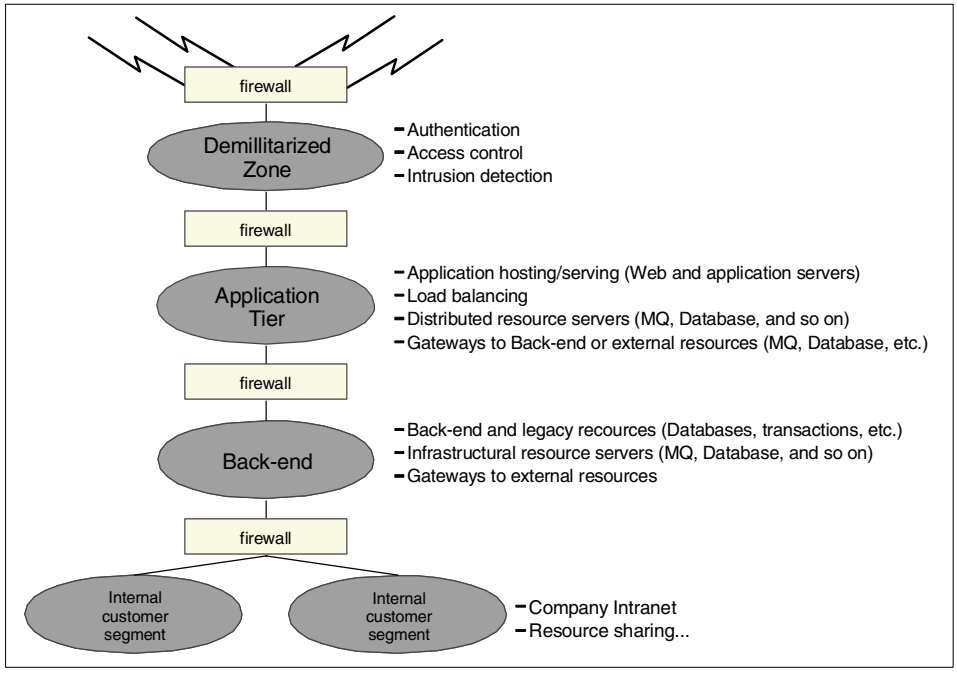


Figure 1-1 A typical e-business application infrastructure

This model architecture (shown in Figure 1-1) is a proven way of providing secure, scalable, and highly available access to company data with a minimum of exposure to security violations. However, the actual components, application servers, and infrastructure resources may vary depending upon the nature of the applications, company policies, the requirements for availability and performance, and the capabilities of the technologies used.

In case you are in the e-business hosting area, or you have to support multiple lines of business that require strict separation, the conceptual architecture shown in Figure 1-1 on page 3 may be even more complex.

To help design the most adequate architecture for a specific set of e-business applications, IBM has published a set of Patterns for e-business that may be used to speed up the process of developing e-business applications and deploying the infrastructure to host them.

The concept behind the Patterns for e-business are to reuse tested and proven architecture to help solve 80% of the business problem. IBM has gathered experiences from more than 20,000 engagements and compiled these experiences into a repository of reusable assets with associated guidelines. These reusable assets allow a solution architect to start with a business problem and drill down through the Patterns layered asset model. By drilling down using the Patterns for e-business layered asset model, the architect quickly identifies the Business, Application, Runtime patterns, and the Product mapping that help build the proposed solution. These solutions may require databases, such as DB2, Oracle, or Informix. Further details on Patterns for e-business may be found in Appendix B, “Patterns for e-business” on page 165.

For a full understanding of Patterns for e-business and how they might apply to business integration, please refer to the book *Patterns for e-business: A strategy for Reuse* by Jonathan Adams, et al.

1.1.2 Managing e-business applications using Tivoli

The Patterns for e-business helps with the design of e-business applications by breaking them down into functional units so that they may be more easily implemented. However, the patterns provide only a little assistance on how to manage these complex application environments. Tivoli solutions provide the end-to-end management that is required to successfully deploy and maintain these applications and the large number of components on which they are built.

When designing the systems management infrastructure that is needed to manage the e-business applications, it must be kept in mind that the determining factor for the application architecture is the nature of the application itself. This determines the application infrastructure and the technologies used. However, it does not hurt if the solution architect consults with systems management specialists while designing the application.

The systems management solution will ideally manage the various application resources seamlessly, without any impact to the e-business application, while observing the company policies on networking use, security, and so on.

Management of e-business applications are therefore best achieved by establishing yet another networking tier, parallel to the application tier, in which all the systems management components can be hosted without influencing the applications. Naturally, since the management applications have to communicate with the resources that need to be managed, the two meet on the networking wires, and on the machines hosting the various e-business application resources.

Using the Tivoli product set, you can establish the central components in the management tier, and utilize proxies and agents present in the DMZ and application tiers, as shown in Figure 1-2.

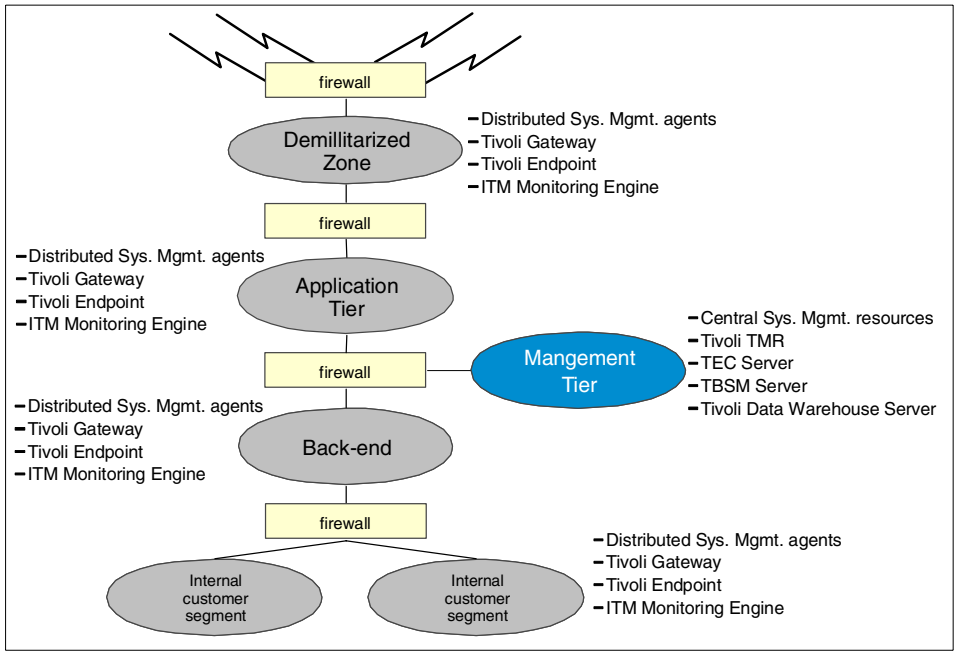


Figure 1-2 A typical Tivoli managed e-business application infrastructure

Implementing the management infrastructure in this fashion, there is minimal interference between the application and the management systems, and the access to or from the various network segments is manageable since the communication flow is between a limited number of nodes using well-known communication ports.

1.2 Tivoli systems management product

Let us take a look at how Tivoli solutions provide comprehensive system management for the enterprise and how IBM Tivoli Monitoring for Databases fits into the overall architecture.

Tivoli provides an end-to-end systems management solution for the whole enterprise. Tivoli products are organized into categories, as shown in Figure 1-3.

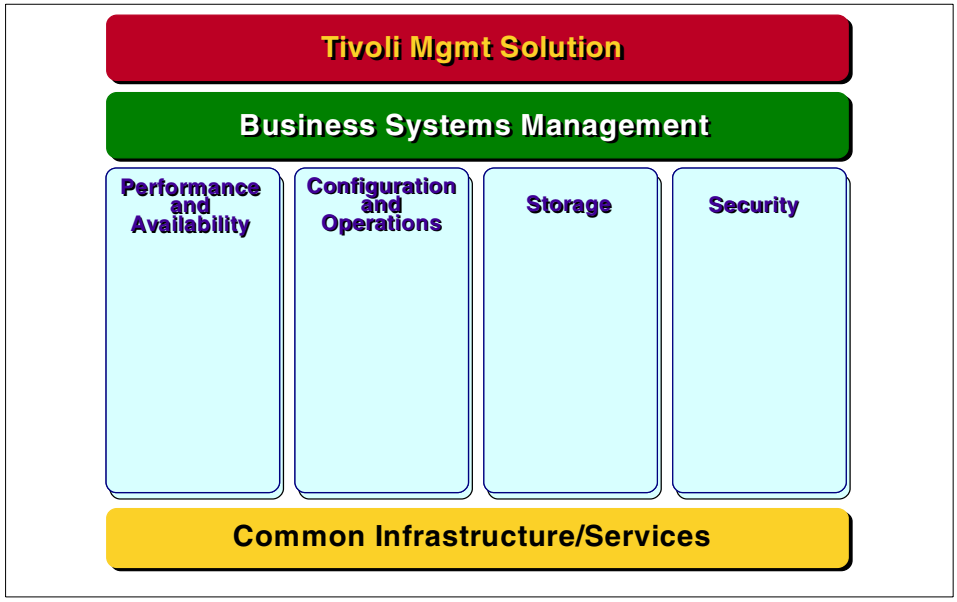


Figure 1-3 Tivoli product categories

Underlying the Tivoli solution set is a group of common services and infrastructure that provide consistency across Tivoli management applications, as well as enabling integration.

Within the Tivoli product family, there are specific solutions that target four primary disciplines of systems management: Performance and Availability, Configuration and Operation, Storage Management, and Security. Products within each of these areas have been made available over the years, and though they continue to be enhanced, have become accepted solutions in enterprises around the world. With these core capabilities in place, IBM has been able to focus on building applications that take advantage of these pillars to provide true business systems management solutions. A typical business application depends not only on the hardware and networking, but also on software ranging from the operating systems to middleware such as databases, web servers, and messaging to the application themselves. Providing a suite of solutions, such as

the IBM Tivoli Monitoring components, allows an IT department to provide management of the entire business system in a consistent way from a central site using an integrated set of tools. By providing an end-to-end set of solutions build on a common foundation, enterprises can manage the ever increasing complexity of their IT infrastructure with reduced staff and increasing efficiency.

Within the performance and availability pillar in Figure 1-3 on page 6, the Tivoli product suite is structured as shown in Figure 1-4.

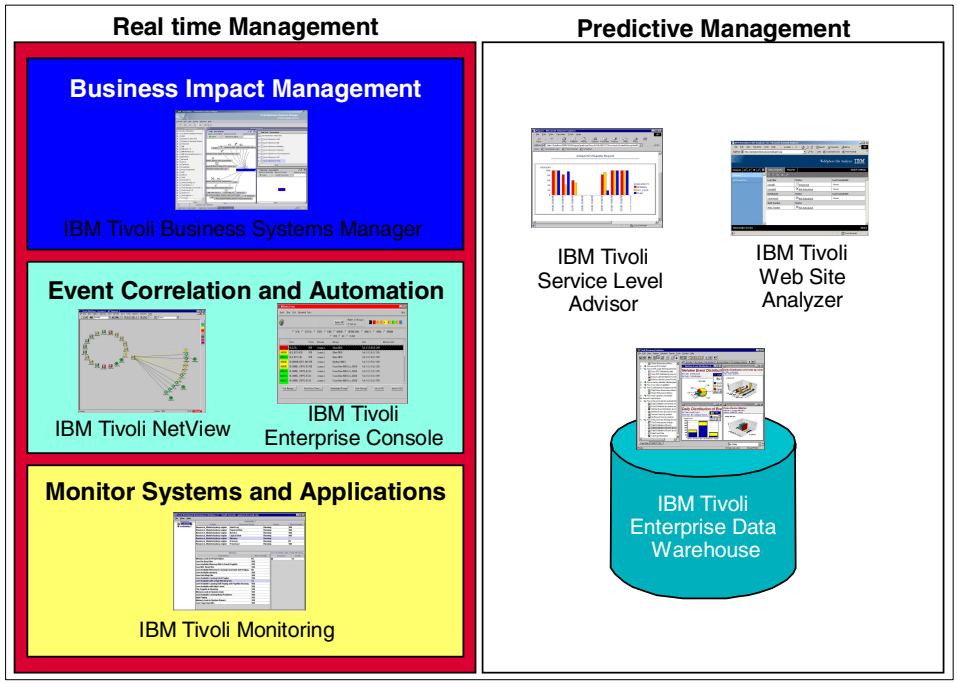


Figure 1-4 Tivoli performance and availability monitoring application structure

In Figure 1-4, we see that the performance and availability application can be grouped in the way they process data, such as:

- Real-time management: The applications that provide real-time status of resources. These applications are mainly used by IT operation personnel to quickly and proactively resolve an outage.
- Predictive management: The applications that collect information and generate historical reports. These applications are mainly used by IT management and capacity planner to evaluate the performance and capacity of resources and also predict the growth of resource usage.

The applications can also be seen by the level of aggregation they provide. This mainly concerns the real-time monitoring, although some level of aggregation is also present in the historical analysis. The following levels exist:

- Monitoring systems: Applications that monitor, and possibly automate, a resolution of a single event from resources. Basic correlation can be performed at this level, but mostly only relates to a single resource instance. This level is very useful in testing and for feeding a higher level of monitoring. This level does not allow operation personnel to prioritize their work based on the importance of the event, nor does it gives operational width of how other related functions are performing.
- Event correlation and automation: Applications that correlate and automate events from multiple resources and can provide an list of relevant outages and perform a preliminary cause-effect analysis. This level is useful in most production environments where operation personnel need to fix all problems fast. They will have a list of open issues, where the system filters the unnecessary events. However, this level of monitoring does not show how an outage affects the business or which functions are impacted by the outage.
- Business impact management: Applications that aggregate individual events and show areas of business or functions' status. This level provides both details of single events, and you can also immediately see which functions are affected. Based on the affected functions, operation personnel can act to resolve outages that has the most impact on the business first.

As shown in Figure 1-4 on page 7, the IBM Tivoli Monitoring products fits in the device monitoring category for real-time monitoring. It feeds the data into the historical monitoring system for reporting and service level analysis. It also provides information for the higher level of monitoring represented using the Tivoli Enterprise Console and Tivoli Business Systems Manager.

1.3 IBM Tivoli Monitoring

The Tivoli Management Environment framework provides a means to manage distributed resources through centralized control and configuration. IBM Tivoli Monitoring 5.1 is now the backbone for availability monitoring across operating systems and applications and more tightly integrates into the Framework environment.

IBM Tivoli Monitoring 5.1 maintains many of the advances of Tivoli Distributed Monitoring (Advanced Edition) 4.1, such as integration with the Common Information Model (CIM) and Windows Management Instrumentation (WMI) to manage event and performance data closer to the source. More intelligent decisions are made based on the resource model approach introduced in Tivoli Distributed Monitoring (Advanced Edition) 4.1.

IBM Tivoli Monitoring 5.1 continues the paradigm in resource monitoring by reducing the processing of large amounts of raw data generated from sources at a single central location. Instead, IBM Tivoli Monitoring 5.1 uses Java-based technology to integrate with industry standard information models to correlate event data at the server level with other event data from monitors on the server. Overall, this will reduce the number of events at the central display. By performing functions locally at the managed resource, which includes local event analysis, event correlation, event aggregation, event management, and event logging, it is able to process the information intelligently to recognize critical situations.

IBM Tivoli Monitoring 5.1 continues the approach of building monitors based on resource models, which in turn are based on the CIM, which provides industry standards for vendors and application developers that wish to instrument their software.

Figure 1-5 on page 10 presents a high-level overview of the interaction between various components of IBM Tivoli Monitoring 5.1.

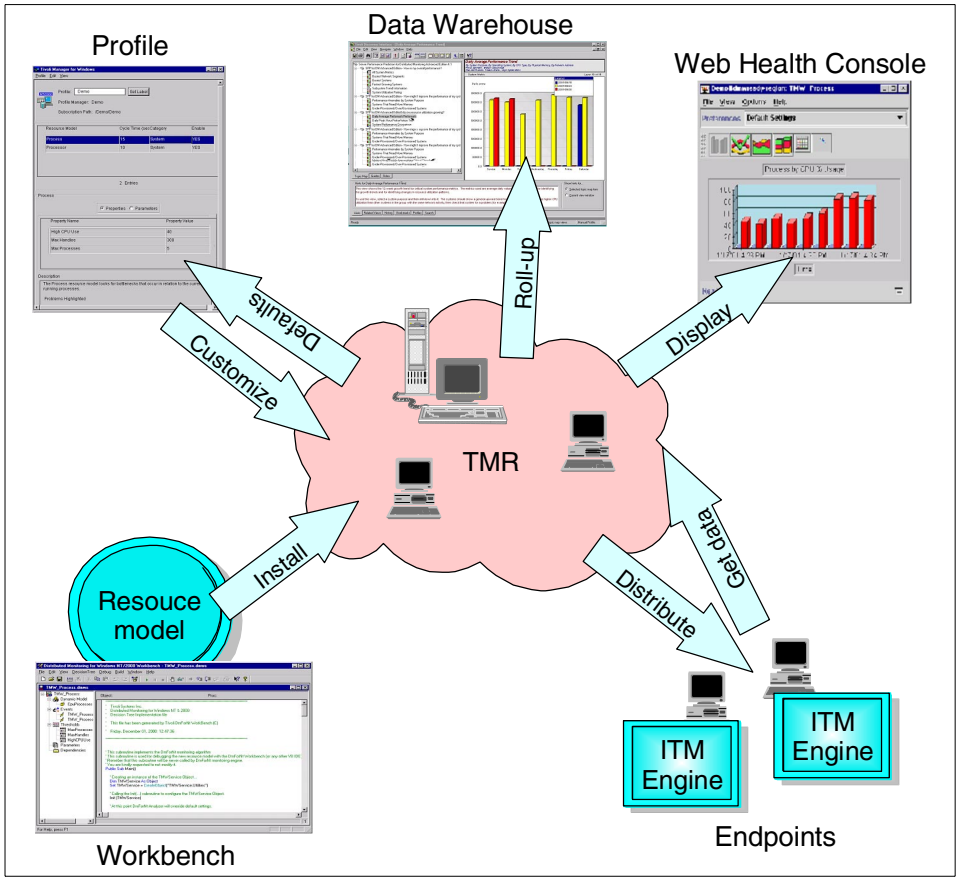


Figure 1-5 High-level overview

The IBM Tivoli Monitoring 5.1 profile contains, among other information, a resource model. The resource model is a collection of monitors that correlate among themselves before attempting to perform a notification action. The resource model is designed and created using the IBM Tivoli Monitoring Workbench.

The IBM Tivoli Monitoring 5.1 profile is distributed to the endpoints to monitor one or more resources (examples of typical resources are hard disk space, paging space, and process/service). Based on configuration settings in the IBM Tivoli Monitoring 5.1 profile, the engine runs on the endpoint and performs the necessary monitoring on the resources that are specified in the distributed resource model(s).

The Web Health Console obtains logged data from selected endpoints and displays the “health” of the endpoints for their resources. These data can be rolled up and analyzed using the Tivoli Enterprise Data Warehouse.

1.4 Overview of IBM Tivoli Monitoring for Databases

This redbook deals specifically with the IBM Tivoli Monitoring for Databases product family. IBM Tivoli Monitoring for Databases is one of a family of solutions based on the IBM Tivoli Monitoring product. IBM Tivoli Monitoring evolved from the Tivoli Distributed Monitoring product. With its new architecture based on resource models, IBM Tivoli Monitoring provides a solid foundation for the development of management solutions addressing the complex needs of today’s IT infrastructures. A set of modules built on top of IBM Tivoli Monitoring provide a comprehensive set of solutions for companies facing the challenge of becoming e-businesses. These modules are delivered through a set of offerings that currently include:

- IBM Tivoli Monitoring for Applications
- IBM Tivoli Monitoring for Business Integration
- IBM Tivoli Monitoring for Databases
- IBM Tivoli Monitoring for Messaging and Collaboration
- IBM Tivoli Monitoring for Web Infrastructure

The IBM Tivoli Monitoring for Databases comes with the following modules:

- IBM Tivoli Monitoring for Databases: DB2, which support DB2 version 7.*
- IBM Tivoli Monitoring for Databases: Oracle, which support Oracle version 8.1.7 and 9.0.1
- IBM Tivoli Monitoring for Databases: Informix, which support Informix version 7.31, 9.21 and 9.30

They help to ensure the availability and performance of critical applications in an integrated e-business environment. Its capabilities include:

- Auto-discovery of the resources to be monitored
- Problem identification, notification, and correction
- Automated best practices for management and operations
- Historical reporting through a centralized data warehouse

IBM Tivoli Monitoring for Databases provides enhanced capabilities over the predecessor products, Tivoli Manager for DB2 and Tivoli Manager for Oracle.

Each module comes with a set of publications. Those are:

- Release Notes, which provides the important information about IBM Tivoli Monitoring for Databases, such as version and level supported, software prerequisites and so on
- Installation and Setup Guide, which provides the tasks needed to implement the IBM Tivoli Monitoring for Databases
- User's Guide, which explains the day-to-day tasks of using the IBM Tivoli Monitoring for Databases products
- Reference Guide, which provides detailed information for resource model, task, and role within IBM Tivoli Monitoring for Databases
- Limitation and Workarounds Supplement, which provides last minute information on known problems and how to overcome them.

For a more detailed publication list, refer to "Related publications" on page 177.

1.5 Project environment and setup

In this section, we present the operating environment that we use so you can be familiar with the machine names and their roles in the document. The overview of our environment is shown in Figure 1-6 on page 13.

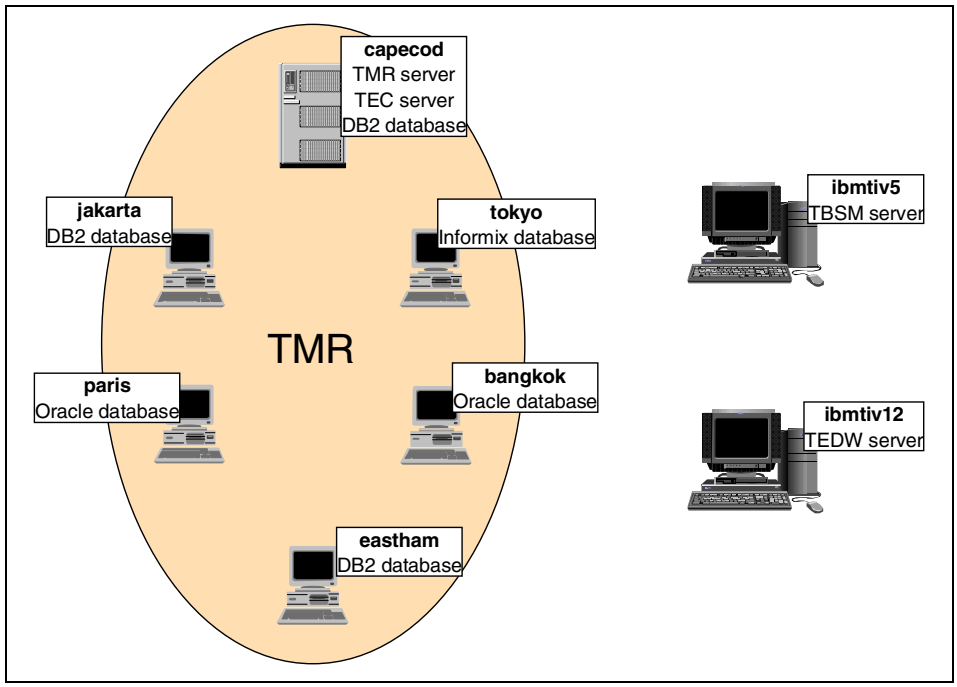


Figure 1-6 Project environment

Table 1-1 shows the list of the software installed in the machines.

Table 1-1 Machine environment

Machine name	Operating system	Function
capecod	AIX 5L	TMR and TEC server DB2 version 7.2 EEE database
jakarta	Windows 2000	DB2 version 7.2 EE database
paris	Windows 2000	Oracle version 9i
bangkok	Windows 2000	Oracle version 8i
tokyo	Windows 2000	Informix
eastham	AIX	Oracle version 9i
ibmtiv5	Windows NT	TBSM Server
ibmtiv12	Windows 2000	TEDW server

For more information on the various software levels that are installed on these machines, refer to 2.1, “Product requirements and prerequisites” on page 16.

1.6 Document organization

This redbook discusses the IBM Tivoli Monitoring for Databases products in the following chapters:

- Chapter 1, “Introduction” on page 1 (this chapter) introduces this redbook.
- Chapter 2, “Installation, configuration, and migration” on page 15 provides information and various installations and migrations for the IBM Tivoli Monitoring for Databases product family.
- Chapter 3, “Database monitoring and resource model” on page 45 discusses existing resource models and customizing them within the scope of the IBM Tivoli Monitoring for Databases products.
- Chapter 4, “Managing databases scenarios” on page 63 shows some example of operational usage of the IBM Tivoli Monitoring for Databases, based on the function required.
- Chapter 5, “Real-time monitoring” on page 83 integrates the IBM Tivoli Monitoring for Databases with functions from Tivoli Enterprise Console and Tivoli Business Systems Manager.
- Chapter 6, “Historical reporting” on page 101 shows Tivoli Enterprise Data Warehouse integration for IBM Tivoli Monitoring for Databases.
- Chapter 7, “Log files and troubleshooting” on page 151 gives some in-depth product structure and discussion of troubleshooting process for IBM Tivoli Monitoring for Databases.

2

Installation, configuration, and migration

This chapter discusses the issues for the installation and migration of IBM Tivoli Monitoring for Databases products. The discussion consists of:

- 2.1, “Product requirements and prerequisites” on page 16 discusses what need to be installed for the product to work.
- 2.2, “Product structure” on page 18 shows the product components as distributed from Tivoli.
- 2.3, “Installation steps overview” on page 19 shows the installation process for IBM Tivoli Monitoring for Databases; first, we discuss the generic steps, and then we provide specific steps for each product.
- 2.4, “Migration from Tivoli Manager for Database product” on page 38 shows the migration steps from the Tivoli Manager for DB2 or Tivoli Manager for Oracle.
- 2.5, “Rapid deployment installation support” on page 40 discusses the special installation for freshly installed machine to manage databases.
- 2.6, “Uninstallation process” on page 41 illustrates the process of removing the installation of the IBM Tivoli Monitoring for Databases.

2.1 Product requirements and prerequisites

IBM Tivoli Monitoring for Databases relies on several components for it to work properly. Primarily, it relies on:

- Tivoli Framework: As a framework based application that is deployed on the endpoints, it relies heavily for Tivoli Framework facility, such as Endpoint level, MDist distribution, and Task library structure.
- IBM Tivoli Monitoring Advanced Edition: IBM Tivoli Monitoring for Databases uses the latest monitoring technology that is available with the IBM Tivoli Monitoring. It uses resource model based monitors that allows local analysis and correlation on the endpoint level to facilitate more accurate monitoring scheme.
- IBM Tivoli Monitoring Component Services: Establishes a basic utilities for application management, including the necessary tools executable and object properties. This component replaces the Tivoli Application Services and Tivoli Application Proxy Services.

The software level that we used in our ITSO environment, related to the above requirement, are:

- Tivoli Management Framework version 3.7B, upgraded with Tivoli Management Framework version 3.7.1, and applies the following patches:
  - 3.6.1-TMF-0034
  - 3.6.1-TMF-0062
  - 3.7.1-TMF-0076
  - 3.7.1-TMF-0073
  - 3.7.1-TMF-0074
  - 3.7.1-TMF-0085
  - 3.7.1-TMF-0087

  Note: The above software level gives an endpoint level of 103, which can be verified using the **`wadminep <ep_name> view_version`** command. It is important that you do not use endpoint level 98, as there is a problem in MDist distribution related to that level.

- We install several Java related components from the Tivoli Management Framework Version 3.7B and 3.7.1 CDs (in the JAVA sub-directory of the CD) to enable MDist monitoring GUI with the following product and patches:
  - Tivoli Java Client Framework 3.7

  - Tivoli Java Client Framework 3.7.1
  - Java for Tivoli 3.7
  - Tivoli Java RDBMS Interface Module (JRIM) 3.7
  - Tivoli Java RDBMS Interface Module (JRIM) 3.7.1
  - JavaHelp for Tivoli 3.7
  - JavaHelp for Tivoli 3.7.1
  - Swing for Tivoli 3.7
  - Swing for Tivoli 3.7.1
  - Tivoli MDist 2 Graphical User Interface
  - Tivoli MDist 2 Graphical User Interface 3.7.1 Maintenance Release
  - Tivoli MDist 2 Graphical User Interface 3.7.1 001 Patch
- The IBM Tivoli Monitoring that is required is version 5.1.1; we use a pre-release (beta) version.
- IBM Tivoli Monitoring Component Services version 5.1.0 is also a pre-release (beta) version.

On the endpoints, the resource model deployed with the IBM Tivoli Monitoring for Databases monitors are written in JavaScript. This requires that Java Runtime Environment (JRE) Version 1.3.0 is installed on all endpoints to be managed. The following links are the image of the JRE that we use; these links need you to register to the IBM developerWorks community.

- For AIX, we use the JRE from the IBM developerWorks:

  `http://www6.software.ibm.com/dl/dka/priv/dka-h?S_PKG=dka130ww`
- For Windows, we use JRE from the WebSphere technology preview:

  `http://www6.software.ibm.com/dl/wspt/priv/wspt-h?S_PKG=pretechww`

Important: Since we were running a beta version of the software, check the latest software and hardware prerequisites for the Release Notes publication listed in 1.4, “Overview of IBM Tivoli Monitoring for Databases” on page 11 before you install the product. Also, review the Limitations and Workarounds Supplement manuals from the product documentation

Apart from the above requirement, we installed other Tivoli software that can be integrated to the IBM Tivoli Monitoring for Databases to perform additional functions:

- Tivoli Enterprise Console: We use Tivoli Enterprise Console Version 3.7.1 with the following components:
  - Tivoli Enterprise Console Server
  - Tivoli Enterprise Console
  - Tivoli Enterprise Console User Interface Server
  - Tivoli Enterprise Console FixPack 2
- Tivoli Business Systems Manager Distributed Edition installed with the Event Enablement feature
- IBM Tivoli Monitoring Extension for Enterprise Data Warehouse installed from the IBM Tivoli Monitoring version 5.1.1 CD-ROM

2.2 Product structure

The IBM Tivoli Monitoring for Databases product CD comes in several components. These components reside in the following sub-directories:

HCONSOLE Health Console language support installation; see Chapter 4, “Managing databases scenarios” on page 63 for a discussion of the Health Console.

PRODUCT Main product installation directory, used for fresh install of the product. We mainly focus on this installation method in 2.3, “Installation steps overview” on page 19.

SETUP Canned installation directory, which is used for a pristine installation for the IBM Tivoli Monitoring for Databases product. See 2.5, “Rapid deployment installation support” on page 40.

TBSM Tivoli Business Systems Manager integration installation directory; see Chapter 5, “Real-time monitoring” on page 83.

UNINSTALL Product uninstallation script (only for DB2 and Oracle); uninstallation discussion is provided in 2.6, “Uninstallation process” on page 41.

UPGRADE Patch install of the product, used for migration from Tivoli Manager for Database product (only for DB2 and Oracle); see 2.4, “Migration from Tivoli Manager for Database product” on page 38.

WORKBENCH	Source file for use with IBM Tivoli Monitoring Workbench to modify and create resource model; for resource model discussion and the usage of the files in this directory, refer to Chapter 3, “Database monitoring and resource model” on page 45.
tedw_apps_etl	Tivoli Enterprise Data Warehouse integration installation directory; see Chapter 6, “Historical reporting” on page 101.

2.3 Installation steps overview

This section discusses the installation and configuration process that we perform to install the IBM Tivoli Monitoring for Databases. The discussion in this section is a generic overview of the complete installation process. Specific steps for each module will be discussed in the sub-sections.

All Tivoli products can be installed from the desktop using Software Installation Service (SIS) or from the command line. If you are unfamiliar with the Tivoli installation process, refer to the *Tivoli Framework 3.7.1 User's Guide,* GC31-8433.

Tip: Before and after each installation steps, do not forget to create a Tivoli object database backup using the **`wbkupdb`** command or from the Tivoli Desktop menu **Desktop** -> **Backup...**. The backup file will allow you to revert back any failure or error caused by the installation.

The following installation description is performed using the Tivoli Desktop:

1. Install the appropriate IBM Tivoli Monitoring for Databases product (from the Tivoli Desktop, select **Desktop** -> **Install** -> **Install Product...**). The install product dialog shows the products available to install in Figure 2-1 on page 20.

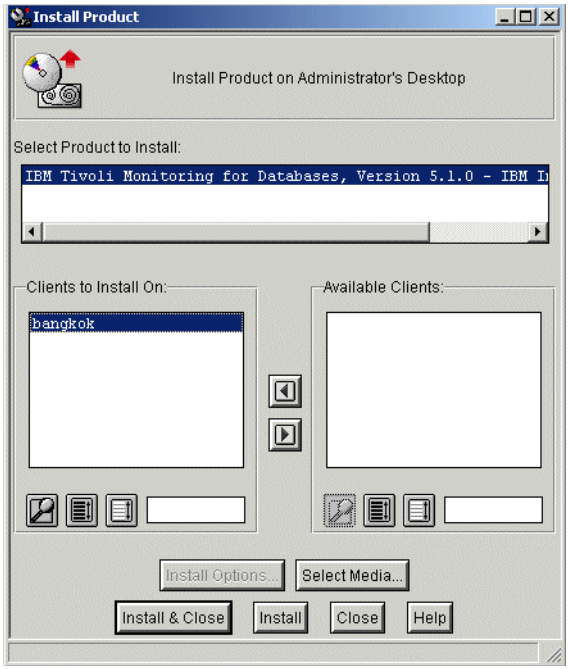


Figure 2-1 Install product dialog

You need to install the product at the TMR Server and all the Gateways in your environment where IBM Tivoli Monitoring for Databases will be used. Click on **Install**, and the product installation screen will show the action that will be performed; click on the **Continue Install** button. When the installation is completed, click on **Close**.

2. For all the endpoints that you want to manage, you need to have JRE version 1.3 installed. You will then need to run the DMLinkJRE task. This task can be run to multiple endpoints simultaneously if all these endpoints has JRE installed in the same directory. You may want to run this task separately for each platform, as they have different installation conventions. These are the steps that we perform:

   a. Select **Desktop** -> **Navigator...** from the Navigator window. Check the Task Library at the bottom of the Navigate To list.

   b. Open the IBM Tivoli Monitoring Tasks task library.

   c. Run the DMLinkJre task. Specify the endpoints for the Execute Task dialog (you need to selectively execute to all endpoint objects that has the same JRE installation directory). You can also identify endpoints using one or more profile managers. Click **Execute**.

Note: Ensure that the path where JRE is physically installed is the same on all selected endpoints.

d. On the DMLinkJre dialog shown in Figure 2-2, enter the complete path where JRE is installed. For example, C:/jre means that you can execute the **`C:/jre/bin/java -version`** command to get the JRE version.

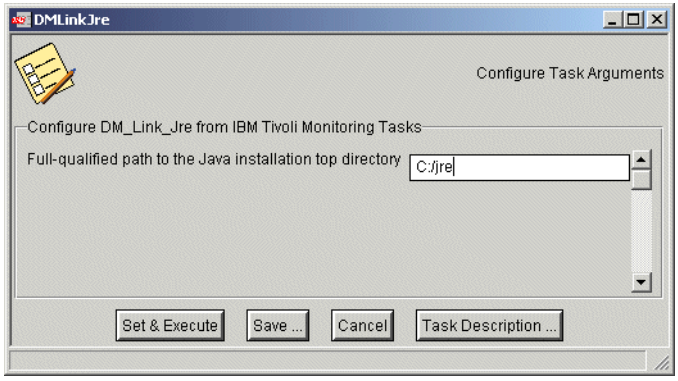


Figure 2-2 DMlinkJRE task argument

e. Click **Set & Execute**. JRE will be linked to the product on all selected endpoints. When the execution is successful, the result will be similar to Figure 2-3 on page 22.

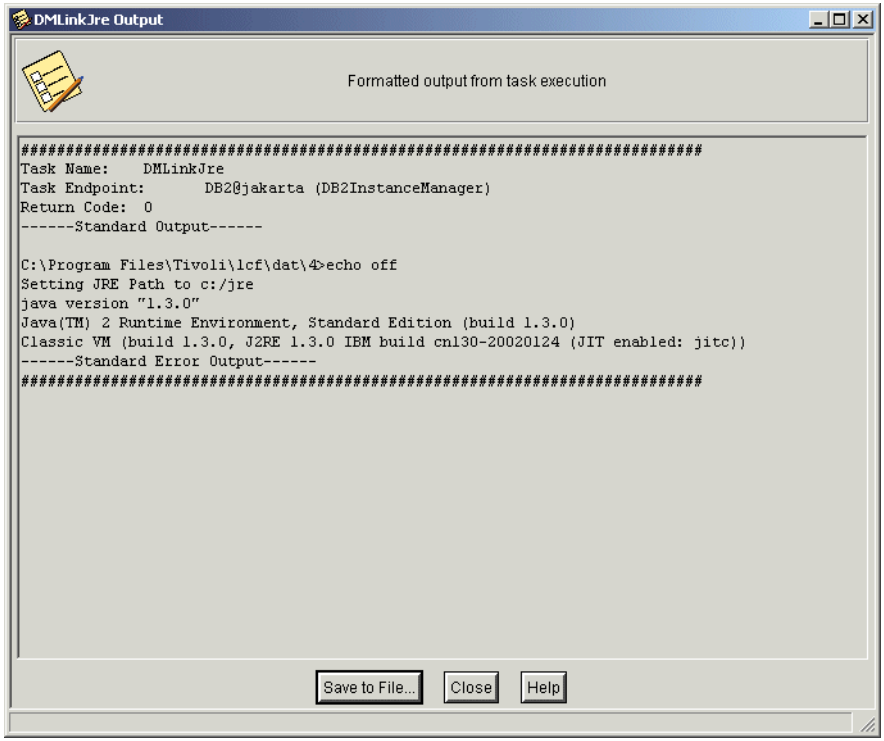


Figure 2-3 DMLinkJRE result

3. You may want to create additional Tivoli administrators to manage each database environment. See the *Tivoli Framework 3.7.1 User's Guide,* GC31-8433 for more information on creating an administrator. The IBM Tivoli Monitoring for Databases products have additional roles defined for the managing databases. The Tivoli administrator that will manage the database environment needs to have these roles assigned to them. Table 2-1 shows the list of roles for each product.

Table 2-1 List of needed authorization roles

	IBM Tivoli Monitoring for Databases		
	DB2	**Oracle**	**Informix**
System administrator	senior	senior	senior IBMInformix_super
Database administrator	db2_dba	oracle_dba admin	IBMInformix_admin
User	db2_user	oracle_user	IBMInformix_user

The above roles can be assigned to an administrator from the Tivoli desktop. Double-click the **Administrators** icon and right-click the **Administrator** icon and select **Edit Resource Roles...** or **Edit TMR Roles...**.

The following characterize the roles divisions:

- System administrator: Assigns Tivoli roles and resources based on the Policy Region structure
- Database administrator: Performs day-to-day database maintenance and monitoring
- User: Performs operation of the Tivoli environment, and needs the ability to check basic database health

4. In regards to setting up policy region for managing the IBM Tivoli Monitoring for Databases resources, the policy region is created when you install the product. The default policy region is called:

   - Monitoring for DB2
   - Monitoring for Oracle
   - Monitoring for IBM Informix

   For DB2, you need to link this policy region to an administrator desktop; in our environment, we run the following command:

   ```
   wln "@PolicyRegion:Monitoring for DB2" @Administrator:Root_TI7031
   ```

   The managed resources for the policy regions are shown in Table 2-2.

Table 2-2 Managed resources assignment

	DB2	Oracle	Informix
Common	ProfileManager Tmw2kProfile		
Specifics	DB2DatabaseManager DB2Discovery DB2Gateway DB2InstanceManager DB2PartitionGroupManager DB2PartitionManager	OracleDiscovery OracleDatabaseManager OracleInstanceManager OracleRoleManagerProfile OracleUserManagerProfile OracleResourceManagerProfile	IBMInformixServer InformixDiscovery InformixDiscoveryHBO

5. Additional tasks needed to enable IBM Tivoli Monitoring for Databases will be discussed in the following sub-sections for each components. The root desktop for when only IBM Tivoli Monitoring for Databases components are installed is shown in Figure 2-4 on page 24.

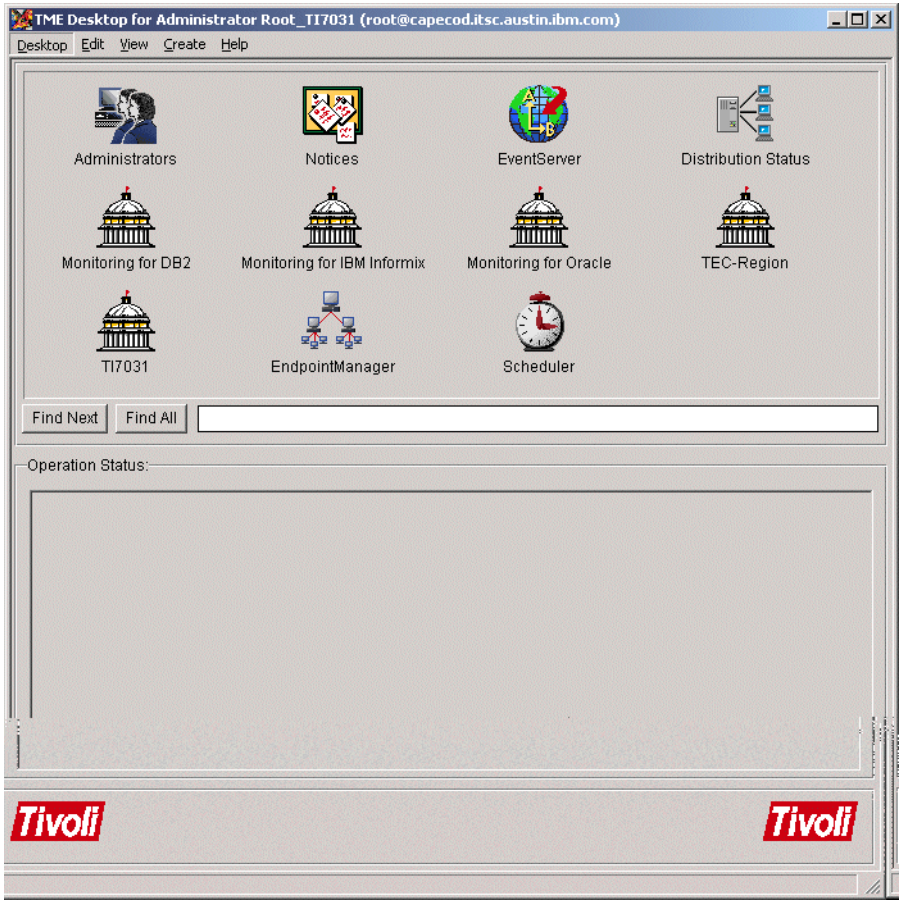


Figure 2-4 Tivoli Desktop with IBM Tivoli Monitoring for Databases

2.3.1 Configuring IBM Tivoli Monitoring for Databases: DB2

Once you have installed IBM Tivoli Monitoring for Databases: DB2, you need to perform the following configuration steps:

1. Creating the db2ecc user ID at the endpoint where you have DB2 database.
   - For a Windows 2000 machine, you use User management application from the Start menu by selecting **Start** -> **Programs** -> **Administrative Tools** -> **Computer Management**. In the Computer Management window, select **System Tools** -> **Local Users and Groups** -> **Users**. Then select **Action** -> **New User...**. Create the user ID db2ecc and specify the password. Click **Create** when done.

This user ID needs a special right to run a service called Tivoli-DB2 Monitoring Service. This is performed by selecting **Start** -> **Programs** -> **Administrative Tools** -> **Local Security Policy**, and then select **Local Policies** -> **User Rights Assignment** in the Local Security Settings window. Double-click on **Log on as a service** and the Local Security Policy Setting dialog will pop up. Click the **Add...** button and add the db2ecc account to the list, as shown in Figure 2-5. Click **OK** when done.

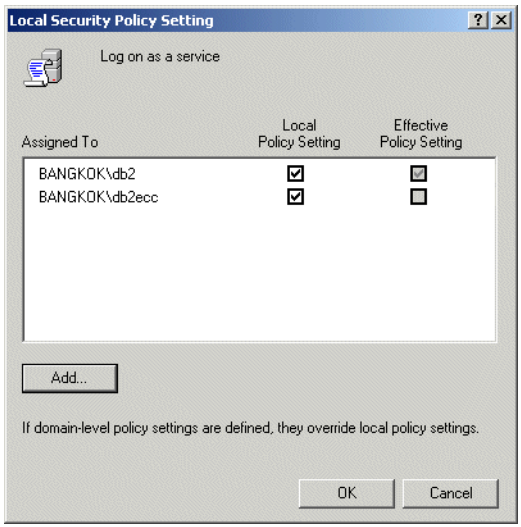


Figure 2-5 Local Security Policy Setting

– For AIX, create the db2ecc user ID using the **`smit user`** command and select Add User. The db2ecc user ID is used to monitor DB2. Monitoring requires SYSADM, SYSMAINT, or SYSCTRL authority within DB2. Assign db2ecc to a group with sufficient authority. For a typical installation, the db2iadm1 group is sufficient.

2. Perform resource discovery using the following steps:

   a. Right-click the DB2 discovery icon from the policy region and select **Edit endpoints**. From the list, select the endpoints on which you want to run the discovery (the machines that have DB2 installed) on, as shown in Figure 2-6 on page 26. Be sure to specify the password for db2ecc user ID for Windows and OS/2 platform.

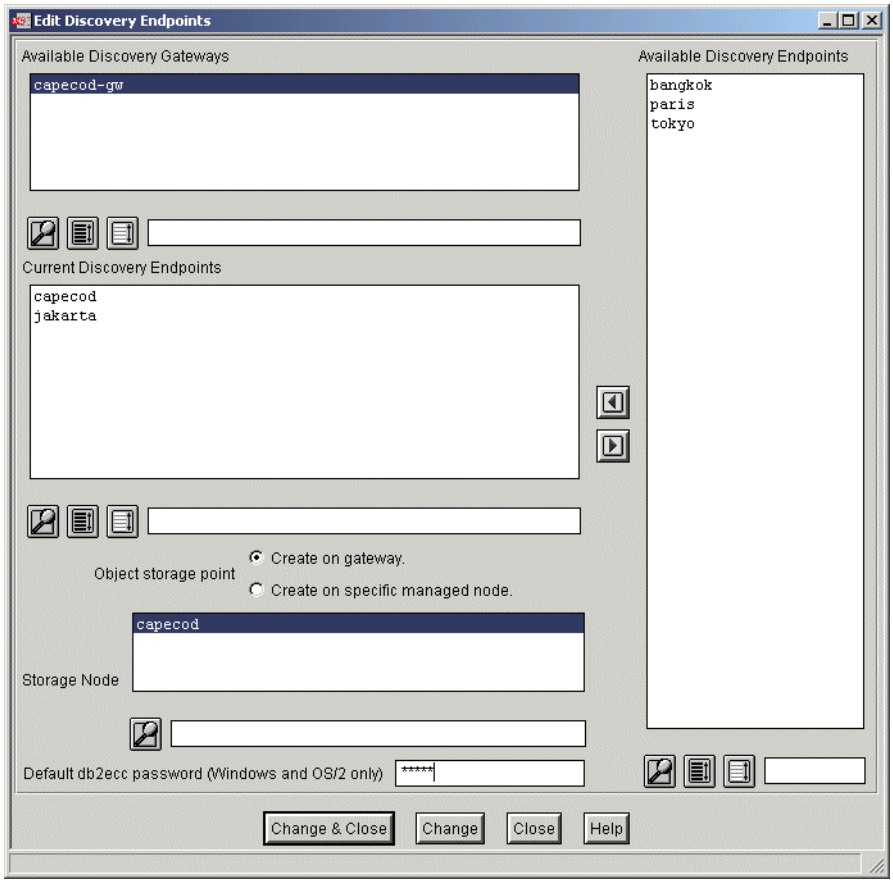


Figure 2-6 Edit task endpoints dialog

b. Run the discovery by double-clicking on the DB2 discovery icon. The discovery progress is shown in a dialog box, as shown in Figure 2-7 on page 27.

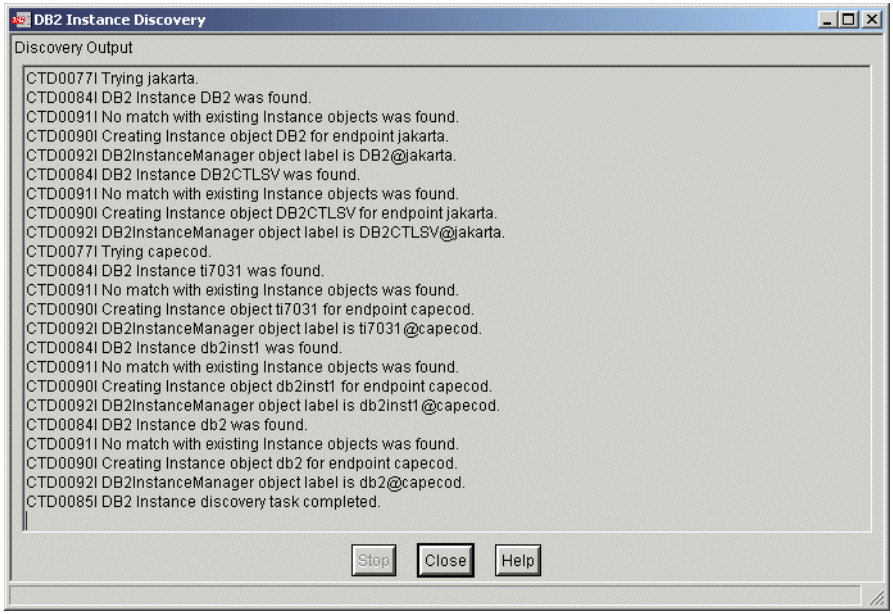


Figure 2-7 DB2 Instance discovery dialog

3. Once the discovery is completed, the policy region where you run the discovery from should have the instances icons, as shown in Figure 2-8.

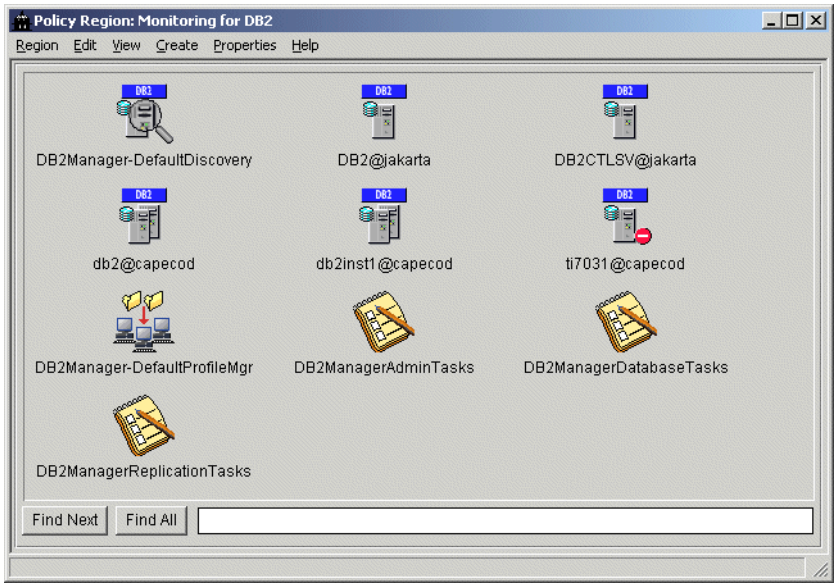


Figure 2-8 DB2 policy region

4. From each instance, you may want to discover the databases. You discover the databases by right-clicking on an instance icon and selecting **Discover Databases...**.

Restriction: The discovery cannot discover DB2 extended-enterprise edition databases as they are considered partitioned.

The list of databases in the instance will be shown; you can move the databases that you want to manage into the right area and click on **Add & Close**. The dialog is shown in Figure 2-9.

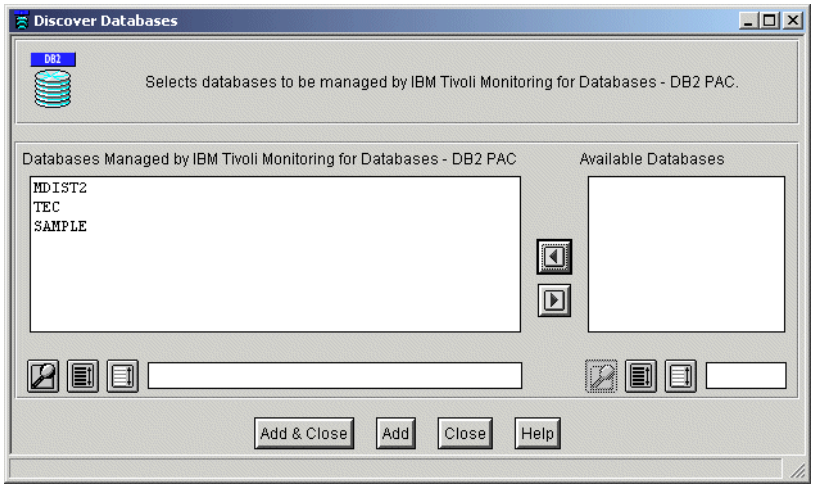


Figure 2-9 Discover database dialog

5. For a DB2 Extended-Enterprise edition, the databases can be created as DB2 Partition objects. Select **Create** -> **DB2PartitionManager**. The dialog will show all the DB2 EEE instances, as shown in Figure 2-10 on page 29.

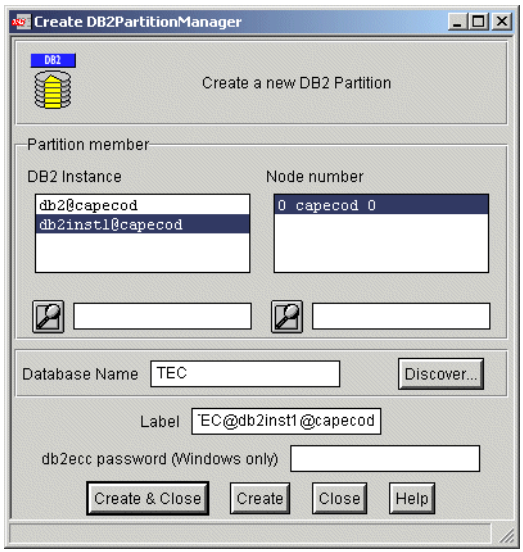


Figure 2-10 DB2 Partition creation

When you click **Discover...**, the databases belonging to that instance will be listed, as shown in Figure 2-11.

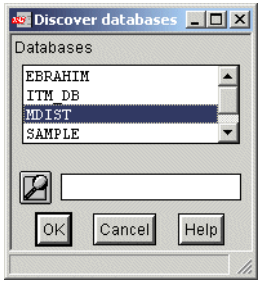


Figure 2-11 DB2 EEE database list

6. You can group DB2PartitionManager into a group as an object called DB2PartitionGroupManager. Select **Create** -> **DB2PartitionGroupManager** to create it. The creation dialog is shown in Figure 2-12 on page 30.

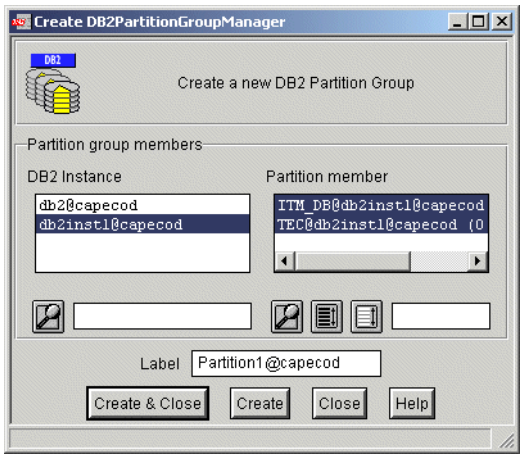


Figure 2-12 DB2 partition group manager creation

7. Now that we have all our managed objects, the policy region is now similar to the one shown in Figure 2-13.

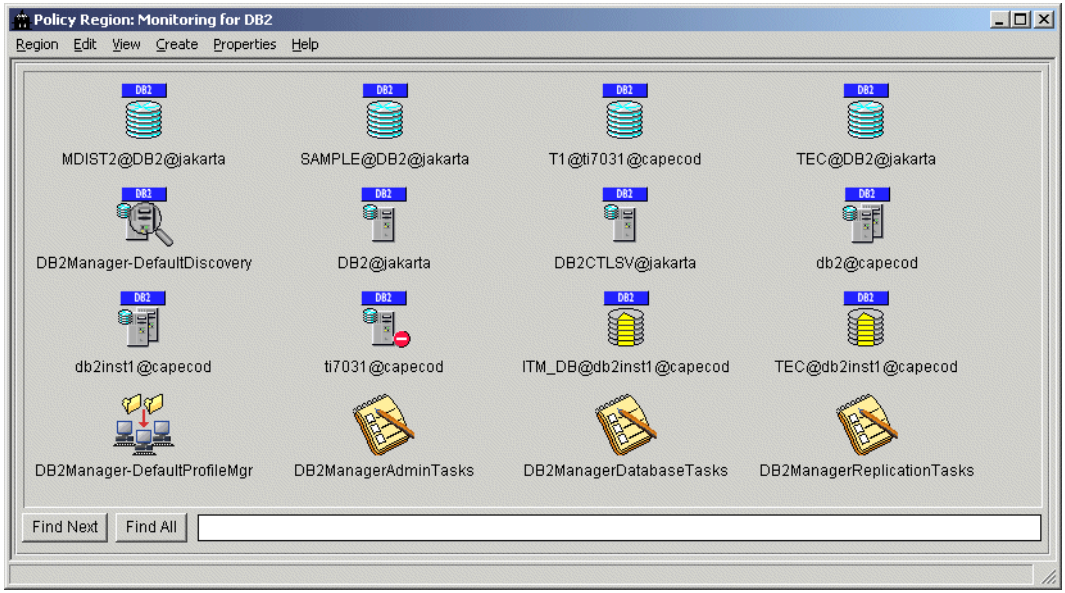


Figure 2-13 Monitoring for DB2 policy region - final

2.3.2 Configuring IBM Tivoli Monitoring for Databases: Oracle

For the IBM Tivoli Monitoring for Databases: Oracle, you need to perform the following actions:

1. Use the Monitoring for Oracle policy region from the Tivoli desktop, similar to the one shown in Figure 2-4 on page 24.
2. Right-click the Oracle discovery icon and select **Edit endpoints**. From the list, select the endpoints on which you want to run the discovery (the machines that have Oracle installed) on, as shown in Figure 2-14.

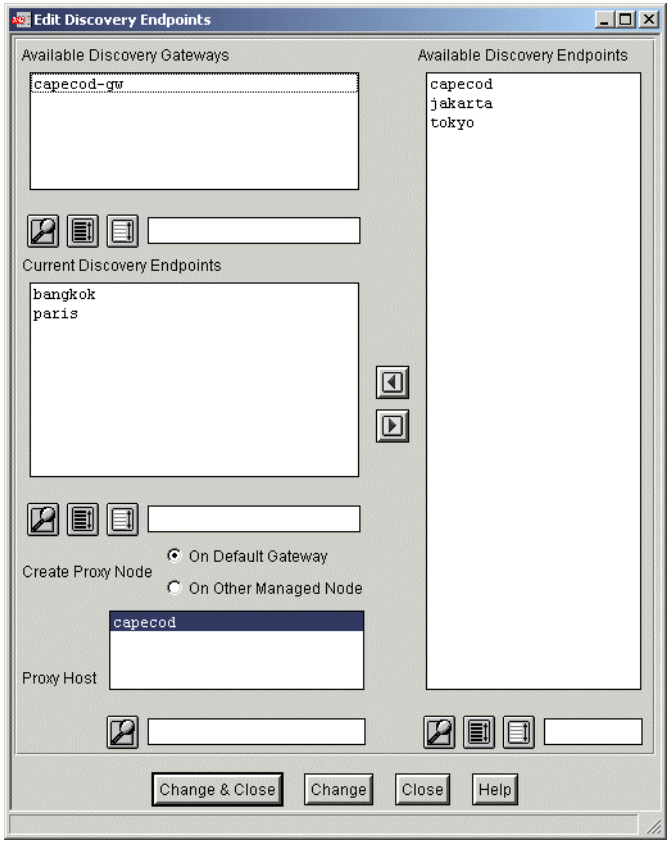


Figure 2-14 Edit endpoint for Oracle discovery

3. Run the discovery by right-clicking on the Oracle discovery icon and selecting **Run discovery**. When the discovery is completed, you will get a pop-up message similar to Figure 2-15 on page 32.

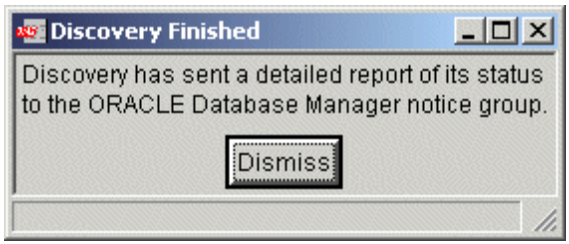


Figure 2-15 Oracle discovery pop-up notice

4. The result of the Oracle discovery process is stored in the Notices for ORACLE Database Manager notice group. A sample notice is shown in Figure 2-16.

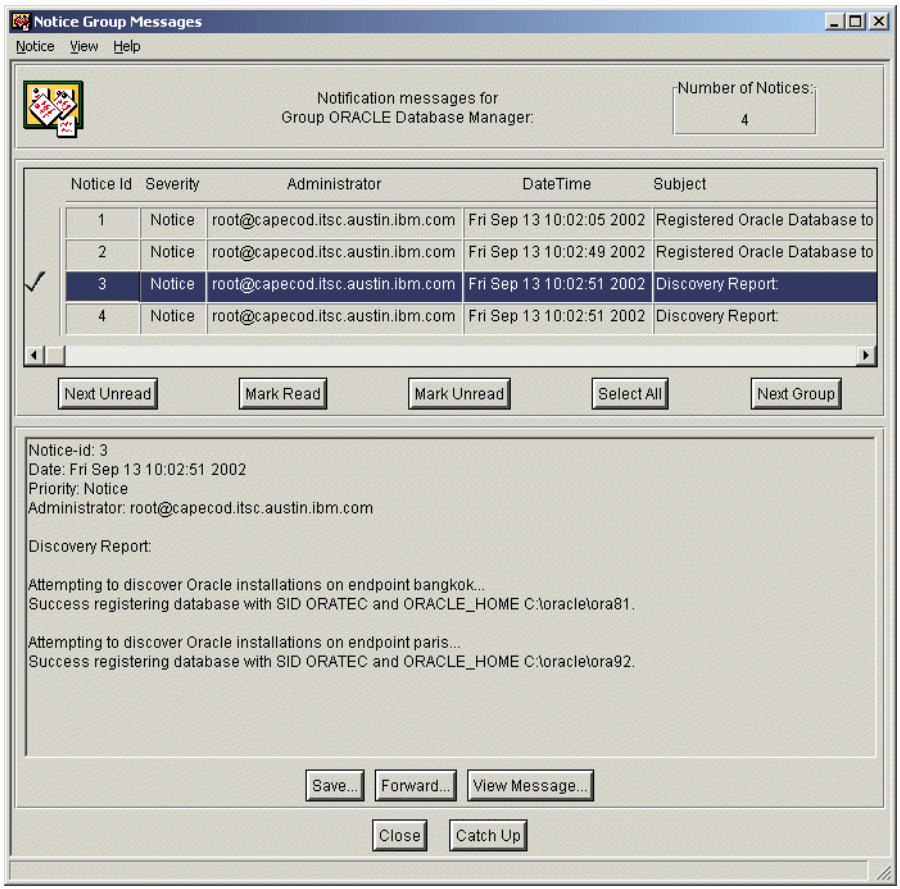


Figure 2-16 Notice for Oracle discovery

5. Once the discovery is complete, your Tivoli desktop should look like the one shown in Figure 2-17.

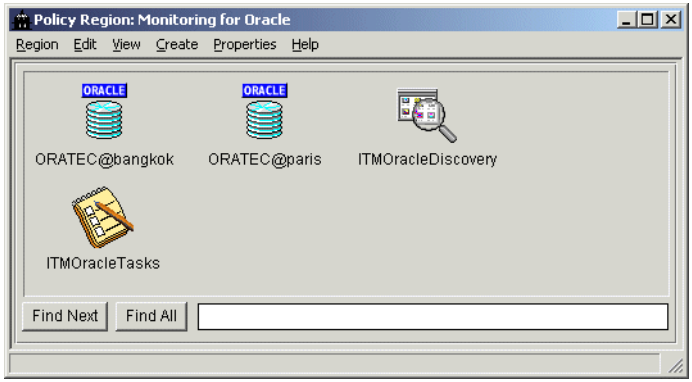


Figure 2-17 Oracle policy region

6. For Oracle 9i, the **`svrmgrl`** interface does not exist anymore. We cannot connect as the internal user ID; therefore, an explicit user ID and password for the system user ID needs to be specified. Right-click on the OracleDatabaseManager icon and select **Properties** -> **Edit Properties**. The Property windows is shown in Figure 2-18 on page 34. Put in the user ID and password for an authorized dba user, such as system. Click **Set**.

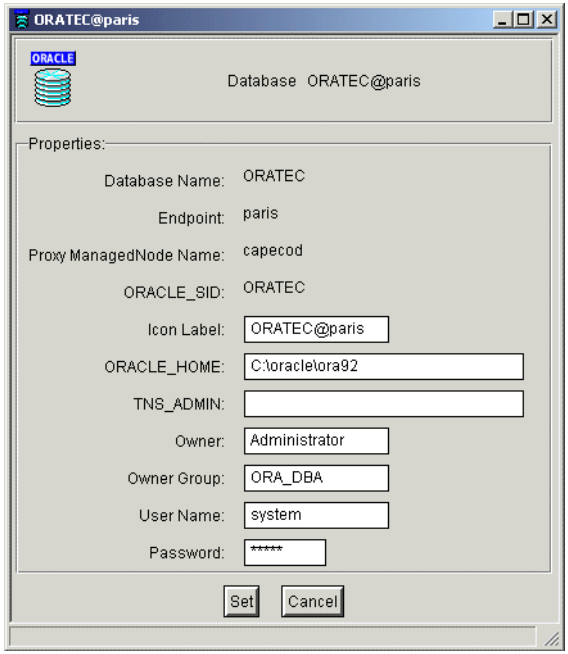


Figure 2-18 Oracle database property dialog

2.3.3 Configuring IBM Tivoli Monitoring for Databases: Informix

For the IBM Tivoli Monitoring for Databases: Informix, perform the following actions:

1. Once IBM Tivoli Monitoring for Databases: Informix is installed, a policy region called Monitoring for IBM Informix is automatically created. Within it are three policy regions, as shown in Figure 2-19 on page 35.

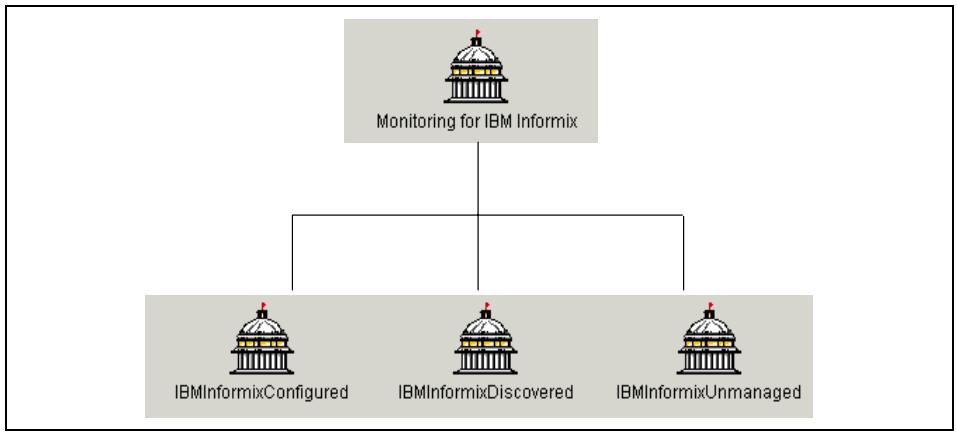


Figure 2-19 Informix policy regions breakdown

Unlike the other IBM Tivoli Monitoring for Databases products, the database resources for Informix are not created directly under the Monitoring for IBM Informix policy region. Depending on its status, these resources can be in one of the three policy region:

- IBMInformixDiscovered: This policy region is used to store the resources that has just been discovered by the discovery interface. The Informix Discovery object also resides in this policy region.
- IBMInformixUnmanaged: For those resources that does not need to be monitored, they can be moved to this policy region.
- IBMInformixConfigured: For those resources where monitoring action has been configured.

2. Discovery is performed using the discovery object in the IBMInformixDiscovered policy region. Right-click the IBMInformixDiscovery object on the desktop and select **Edit endpoints**. Select the endpoints on which you would like to run the discovery from the dialog shown in Figure 2-20 on page 36. Click **OK** when done.

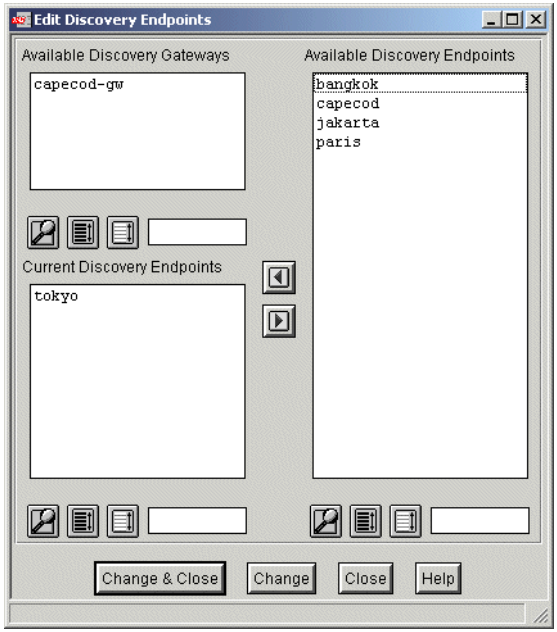


Figure 2-20 Edit endpoint for Informix discovery

3. Now run the discovery using the context menu of the discovery object and select **Run discovery**. The IBM Informix resources will be created under the IBMInformixDiscovered policy region. Our discovery process produces the region, as shown in Figure 2-21.

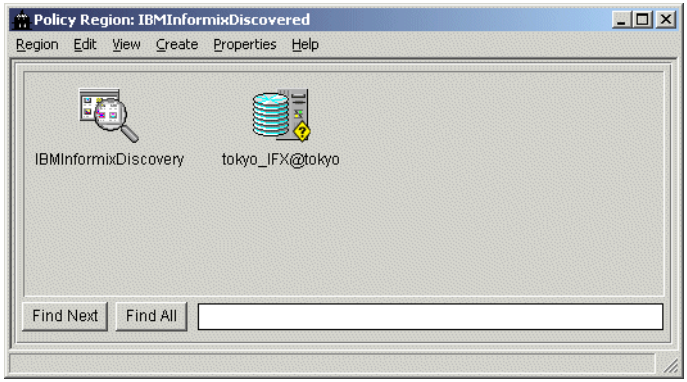


Figure 2-21 IBM Informix Discovered policy region

4. The database resources found in the IBMInformixDiscovered policy region will have a question mark on it. It indicates that the object has not been configured. IBM Tivoli Monitoring for Databases: Informix uses Java

Database Connectivity (JDBC) connection to obtain many of the monitoring metrics for the resource models. You must ensure that your Informix instance has JDBC configured. Once JDBC has been configured, record the JDBC port, jdbc driver name (which will be com.informix.jdbc.IfxDriver), and the full path to the JDBC jar file ifxjdbc.jar. These JDBC settings will be required when configuring the IBM Informix Server.

Right-click the icon and select **Configure IBMInformix Server**; the dialog is shown in Figure 2-22. You need to specify the user specification and drivers for the database. Click **Create And Close** when done and the object will be put under the IBMInformixConfigured policy region.

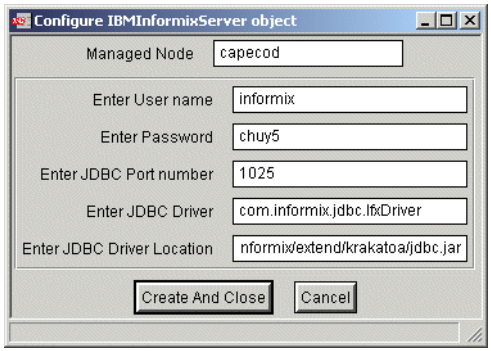


Figure 2-22 Informix server configuration dialog

5. Especially for Informix, if you have Tivoli Enterprise Console installed in your environment and you plan to use it for monitoring, you need to run the Send_TEC_Files_To_TEC task from the IBM Informix Server Tasks task library. The option for this task is the TEC server name, as shown in Figure 2-23 on page 38.

Attention: You need to run this task, even though your TEC is in the same TMR as your installation, or even if your TEC server is the same as the TMR server. This will copy the necessary files needed to run the Configure_TEC task.

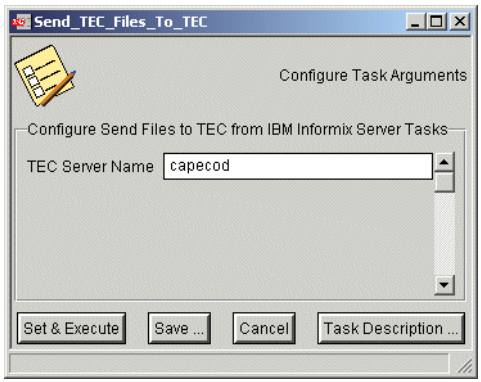


Figure 2-23 Send_TEC_Files_To_TEC argument

2.4 Migration from Tivoli Manager for Database product

The IBM Tivoli Monitoring for Databases products are the successor to the Tivoli Manager for Database products. The Tivoli Manager products are based on the Tivoli Distributed Monitoring, while the new IBM Tivoli Monitoring for Databases products are based on the IBM Tivoli Monitoring product.

In this section, we discuss the migration process and its consideration for migrating Tivoli Manager for DB2 Version 2.1 and Tivoli Manager for Oracle Version 2.1 to the IBM Tivoli Monitoring for Databases product. We would not cover Tivoli Manager for Informix, because the previous version available is Version 1.0, which is based on a Managed Node environment.

The migration is performed on the following product environment:

- Tivoli Framework Version 3.7.1 with FixPack 3 patches applied
- Tivoli Enterprise Console Version 3.7.1 with FixPack 2 patch applied
- Tivoli Distributed Monitoring Version 3.7
- Tivoli Manager for DB2 Version 2
- Tivoli Manager for Oracle Version 2

Note: Do not forget to create a backup of the Tivoli object database using the **wbkupdb** command before and after each installation step so that you can revert back in case of any error.

The migration process that we performed is:

1. Ensuring that all the functions of the Tivoli Manager for DB2 and Tivoli Manager for Oracle are working, such as the discovery and monitors. For more information on these products, refer to the Tivoli Manager for Databases manuals.
2. Install IBM Tivoli Monitoring Version 5.1.1 and IBM Tivoli Monitoring Component Services Version 5.1. Make sure that these product run and the functionality of the Tivoli Distributed Monitoring are still intact.
3. Perform the installation of the IBM Tivoli Monitoring for Databases as an upgrade patch using the Tivoli Desktop. Select **Desktop** -> **Install** -> **Install Patch...** and set the Media into the UPGRADE sub-directory in the IBM Tivoli Monitoring for Databases CD-ROM. You need to install this patch to TMR server and all gateways that has Tivoli Manager for Database installed.
4. You may need to install JRE 1.3.0 and link it to the IBM Tivoli Monitoring environment using the DMLinkJRE task, as discussed in 2.3, “Installation steps overview” on page 19.
5. The migration process actually preserves the database objects and converts them to the new version structure. Existing programs supplied with the Tivoli Manager for Database will still work. Therefore, both Tivoli Manager for Databases and IBM Tivoli Monitoring for Databases can co-exist. You may want to ensure that all database related resources are still working.
6. You may then start creating new monitoring profiles to replace the old Distributed Monitoring profiles. Assign Tmw2kProfile as a managed resource in the policy region and start creating the Tmw2kProfile object. Distribute the profile and check the result using the **wdmlseng -e <ep_name>** command. A sample output is shown in Example 2-1.

Example 2-1 Sample output from wdmlseng command

```
# wdmlseng -e jakarta

Forwarding the request to the engine...


The following profiles are running:

DB2@jakarta.DB2InstanceStatus#TI7031
    DB2InstanceStatus        :Running
DB2CTLSV@jakarta.DB2InstanceStatus#TI7031
    DB2InstanceStatus        :Running
```

7. Once you can ensure that the new monitor is working, the old monitoring engine is no longer needed at the endpoint. The following steps should be carried out to disable the Distributed Monitoring engine:
   a. Unsubscribe the profile from the endpoints, or disable the old profile and distribute the profile to the relevant endpoints.
   b. You need to clear the engine by issuing the **`wclreng <ep_name>`** command.
   c. Stop the engine on the endpoint by issuing the **`wstopeng <ep_name>`** command.

2.5 Rapid deployment installation support

The IBM Tivoli Monitoring for Databases products provides a set of Java installation scripts in the SETUP directory that allows the product to be installed from the rapid deployment installation.

The rapid deployment is a pre-packaged installation that is provided from the IBM Tivoli Monitoring for Databases Installation CD. You can invoke it with the **`setupDB.bat`** or **`setupDB.sh`** command.

The rapid deployment can be performed on a server which:

- Has no software installed for the Tivoli Management Framework.
- Has Tivoli Management Framework, Version 3.7, Revision A or Revision B installed.
- Has Tivoli Management Framework, Version 3.7.1, installed, but does not have the required patches.

There are two types of rapid deployment:

- A typical installation, which has minimal prompt and customization.
- A custom installation, which allows you to:
  - See all target computers in a single tree view for quick reference.
  - Control the naming of managed resources.
  - Control the name of target directories.
  - Be able to export a Custom installation plan.

For more information on rapid deployment installation, refer to *IBM Tivoli Monitoring for Databases Installation and Setup Guide Version 5.1.0*, GC23-4730.

2.6 Uninstallation process

The IBM Tivoli Monitoring for Databases: DB2 and IBM Tivoli Monitoring for Databases: Oracle provides a set of uninstallation scripts that will remove all objects related to IBM Tivoli Monitoring for Databases. This section shows how we run these installation scripts and remove products that are not removed by the script.

Note that this process does not clean up the Tivoli Enterprise Console rule bases from the class and ruleset files populated by IBM Tivoli Monitoring for Databases. For more information on this process, see Chapter 5, “Real-time monitoring” on page 83.

Important: It is important that you back up your database prior to the uninstallation using the **wbkupdb** command. After the uninstallation, it is recommended to recheck the databases using the **wchkdb -u** command.

2.6.1 IBM Tivoli Monitoring for Databases: DB2 uninstallation

The program for uninstallation for IBM Tivoli Monitoring for Databases: DB2 is called **wdb2uninstall**. It resides under the UNINSTALL directory of the IBM Tivoli Monitoring for Databases: DB2 CD. In a UNIX platform, you need to copy this script to disk and run **chmod +x wdb2uninstall** to add the execute flag for the file.

The syntax of this command is:

```
wdb2uninstall ITMDB2_51 | DB2_21 | DB2_11 | ALL
```

The uninstallation program stores the log under the $DBDIR directory called wdb2uninstall.log.

If you have created monitoring profiles in the IBM Tivoli Monitoring for Databases: DB2 profile managers, then these are not deleted. You need to clean these up manually. In our setup, all the profiles for DB2 have the word DB2, so we can find all these profiles and profile managers using the **wlookup** command and delete them manually using the **wdel** command. The command that we use to list the objects is shown in Example 2-3 on page 43. We also need to manually delete the DB2Manager-DefaultPolicyRegion, since it is not deleted by the uninstallation program, when there are objects that are not deleted. The manual operation is shown in Example 2-2.

Example 2-2 Manual removal of IBM Tivoli Monitoring for Databases: DB2 objects

```
# wlookup -ar Tmw2kProfile | grep DB2
DB2DatabaseMonitorTEDW  2035659666.1.1290#TMW2K::All#
DB2InstanceMonitorTEDW  2035659666.1.1288#TMW2K::All#
```

```
# wlookup -ar ProfileManager | grep DB2
DB2DBMgrs       2035659666.1.1258#TMF_CCMS::ProfileManager#
DB2DatabaseManagers     2035659666.1.1280#TMF_CCMS::ProfileManager#
DB2InstanceManagers     2035659666.1.1282#TMF_CCMS::ProfileManager#
DB2PartitionGroupManagers       2035659666.1.1284#TMF_CCMS::ProfileManager#
DB2PartitionManagers    2035659666.1.1283#TMF_CCMS::ProfileManager#
# wdel @Tmw2kProfile:DB2DatabaseMonitorTEDW
# wdel @Tmw2kProfile:DB2InstanceMonitorTEDW
# wdel @ProfileManager:DB2DBMgrs
# wdel @ProfileManager:DB2DatabaseManagers
# wdel @ProfileManager:DB2InstanceManagers
# wdel @ProfileManager:DB2PartitionGroupManagers
# wdel @ProfileManager:DB2PartitionManagers
# wdelpr "DB2Manager-DefaultPolicyRegion"
```

2.6.2 IBM Tivoli Monitoring for Databases: Oracle uninstallation

The program for uninstallation for IBM Tivoli Monitoring for Databases: Oracle is called **wouninstall**. It resides under the UNINSTALL directory of the IBM Tivoli Monitoring for Databases: Oracle CD. In a UNIX platform, you need to copy this script to disk and run **chmod +x wouninstall** to add the execute flag for the file.

The syntax of this command is:

```
wouninstall ALL | ITMORA51 | ORACLE2.0
```

If you previously migrated from Tivoli Manager for Oracle to IBM Tivoli Monitoring for Databases: Oracle, you need to uninstall ALL. You can uninstall just the IBM Tivoli Monitoring for Databases: Oracle objects and resource models or just the Tivoli Manager for Oracle objects.

The uninstallation program stores the log under the $DBDIR directory called wouninstall.log.

If you have created monitoring profiles in the IBM Tivoli Monitoring for Databases: Oracle profile managers, then these are not deleted. You need to clean these up manually. In our setup, all the profiles for Oracle have the word Ora, so we can find all these profiles and profile managers using the **wlookup** command and delete them manually using the **wdel** command. The command that we use to list the objects is shown in Example 2-3 on page 43. We also need to manually delete the Monitoring for Oracle policy region, since it is not deleted by the uninstallation program when there are objects that are not deleted.

Example 2-3 Listing and removing Oracle profiles

```
# wlookup -ar Tmw2kProfile | grep Ora
OracleInstanceTEDW      2035659666.1.1287#TMW2K::All#
OracleTEDW      2035659666.1.1285#TMW2K::All#
# wlookup -ar ProfileManager | grep Ora
OracleDatabaseManagers  2035659666.1.1277#TMF_CCMS::ProfileManager#
OracleInstanceManagers  2035659666.1.1278#TMF_CCMS::ProfileManager#
OracleMonitors  2035659666.1.1279#TMF_CCMS::ProfileManager#
# wdel @Tmw2kProfile:OracleInstanceTEDW
# wdel @Tmw2kProfile:OracleTEDW
# wdel @ProfileManager:OracleDatabaseManagers
# wdel @ProfileManager:OracleInstanceManagers
# wdel @ProfileManager:OracleMonitors
# wdelpr "Monitoring for Oracle"
```

2.6.3 IBM Tivoli Monitoring for Databases: Informix uninstallation

Uninstallation for IBM Tivoli Monitoring for Databases: Informix is performed using the **wuninst** command. You can list the component names that can be uninstalled by the **wuninst -list** command. The sample **wuninst** output in our environment is shown in Example 2-4.

Example 2-4 Listing of uninstallation objects

```
# wuninst -list
Creating Log File (/tmp/wuninst.log)...
-----------------------------------------------
    Uninstallable Products installed:
-----------------------------------------------
ACF
ITMCmptSvcs
ITMInformix
ITM_TEDW
JCF371
JRIM
JavaHelp
SSLA_1.1
Sentry2.0.2
Swing
TEC_JCONSOLE
TEC_SERVER
TEC_UI_SRVR
TMNT_3.6.2
mdist2gui
wuninst complete.
```

As shown in Example 2-4 on page 43, the IBM Tivoli Monitoring for Databases: Informix component is called ITMInformix. Therefore, the uninstallation can be called using:

```
wuninst ITMInformix
```

The uninstallation program stores a trace file in the temporary directory pointed to by the **wtemp** command (on AIX it is /tmp) called wifxuninst_<timestamp>.trc.

The uninstallation program for Informix actually removes all Tmw2kProfile objects that are empty, but it did not remove a user-defined Profile Manager that we create in the Monitoring for IBMInformix policy region. The profile manager needs to be deleted manually and also the policy region, since it cannot be deleted while there are still objects in there. The manual deletion is shown in Example 2-5.

Example 2-5 Manual removal of IBM Tivoli Monitoring for Databases: Informix objects

```
# wdel @ProfileManager:ITMIFX_PM
# wdelpr "Monitoring for IBM Informix"
```

An additional object that is not deleted is imp_InformixDiscovery::DiscoveryEPA. This can be deleted manually using the **wdel** command.

3

Database monitoring and resource model

When monitoring a database system, you need to consider the various aspects of database operation in order to get a complete view of the database's health. The resource models provided with the IBM Tivoli Monitoring for Databases use several metrices from the database to get a health indicator of the database.

This chapter explains the resource models available with IBM Tivoli Monitoring for Databases:

- 3.1, "Introducing resource model" on page 46 discusses the resource model concept and functions.
- 3.2, "Supplied resource models" on page 47 lists the supplied resource models provided by the IBM Tivoli Monitoring for Databases products.
- 3.3, "Customizing resource models" on page 52 shows some sample customization of the resource models.

3.1 Introducing resource model

IBM Tivoli Monitoring Version 5, which provides the basic environment for IBM Tivoli Monitoring for Databases, uses a set of resource models to understand the database health.

The resource model is a monitoring model that contains the program scheme necessary to determine what data is to be accessed from an endpoint at runtime and how this data is to be handled. in other words, the resource model is an equivalent implementation of monitors from previous editions of Tivoli Distributed Monitoring, albeit in a different way, using the object oriented modeling approach and integration with CIM. Each resource model obtains resource data from the endpoint it is distributed to, performs root cause analysis using a built-in algorithm, and reacts accordingly in the form of built-in actions or user-defined tasks.

The implementation of resource model can be seen in Figure 3-1.

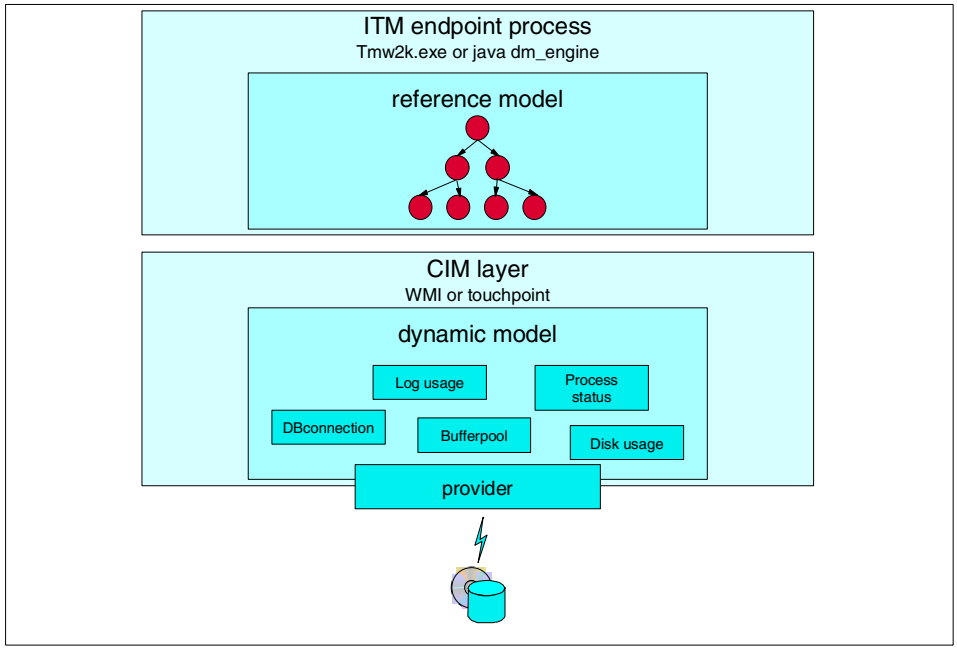


Figure 3-1 IBM Tivoli Monitoring resource model

As shown in Figure 3-1, the implementation of a resource model has two components: the dynamic model and the reference model.

- The dynamic model is the definition of a set of attributes that specify the current state of the resource. The attribute values typically will change

dynamically while the system is working. IBM Tivoli Monitoring 5.1 leverages on the WMI infrastructure on Windows systems, and the Touchpoint infrastructure on UNIX systems. Touchpoint is Tivoli's CIM implementation for UNIX. For the database resources, IBM Tivoli Monitoring for Databases provides several data providers for acquiring the database metrics.

- The reference model is the active part of the monitoring that:
  - Acquires the resource metric.
  - Interprets the quality of a resource against a threshold.
  - Determines the root cause of a quality reduction by a set of logic.
  - Generates the indication of the problem, such as sending an event or executing an action.

All the IBM Tivoli Monitoring for Databases have their own resource models, which are shipped with the IBM Tivoli Monitoring for Databases products. for more information on how IBM Tivoli Monitoring and its resource model works, refer to *IBM Tivoli Monitoring Version 5.1: Advanced Resource Monitoring*, SG24-5519.

3.2 Supplied resource models

This section discusses the specific resource models that are supplied by the IBM Tivoli Monitoring for Databases products. The following sub-sections lists the available resource models by module and provide a brief description of each. For more information on the resource models, refer to the reference manuals of the IBM Tivoli Monitoring for Databases product as listed in "Related publications" on page 177.

We categorized database monitoring resource models into the following groups:

- Process monitoring: This is the basic measurement of the database availability. This monitors whether a specified database server process exists or not.
- Memory usage monitoring: Relational database systems use memory to buffer or cache a lot of information to minimize the need to access the data from the disk. The memory usage needs to be monitored to ensure that there is always enough space and that it has a high hit ratio (the data that is searched is found in memory).
- Activity monitors: Monitors the activity of the database, the number of connections, CPU utilization, I/O rate, and number of applications. These need to be monitored to ensure that there is no sudden change of load and to predict future growth.

- Disk space usage monitors: The actual data of a database is stored in the disk for persistence. Disk space needs to be monitored to ensure availability to insert more data, and disk organization needs to be monitored to ensure performance when accessing data in the disk.
- Locking monitors: Data is locked in memory before transactions are committed. Some of these transactions may lock data for a long time or lock contention may occur. While locks guarantee the data integrity, it may possibly defer access to data and even, in the case of deadlocks, disable access to data.
- Log monitors: All updates to data in memory are logged, and these logs are necessary to ensure that data can be recovered. Some transactions may perform a lot of updates before committing them, causing a large chunk of needed logs.
- Replication monitors: Monitors the status of data replication activities. These resource models only apply to DB2 and Informix.
- Others: Each database servers has its own special monitors for functions that are unique for the database systems.

3.2.1 IBM Tivoli Monitoring for Databases: DB2 resource models

IBM Tivoli Monitoring for Databases: DB2 comes with a number of resource models ready to use with little or no customization.

The available resource models for IBM Tivoli Monitoring for Databases: DB2, grouped by their functions, are shown below:

- Process monitoring resource models:
  - DB2 Agent resource model
  - DB2 Instance Status resource model
- Memory usage monitoring resource models:
  - DB2 Buffer Pools resource model
  - DB2 Buffer Pool/Extended Storage resource model
  - DB2 Catalog Cache resource model
  - DB2 Package Cache resource model
- Activity monitoring resource models:
  - DB2 CPU Utilization resource model
  - DB2 Database Activity resource model
  - DB2 Direct IO resource model
  - DB2 Host Throughput resource model

  - DB2 SQL Cursor Activity resource model
  - DB2 SQL Statement Activity resource model
  - DB2 Table Activity resource model
- Disk space usage resource models:
  - DB2 SAP Tablespace Usage/Status resource model
- Lock monitoring resource models:
  - DB2 Locks and Deadlocks resource model
  - DB2 Lock Waits resource model
- Log monitoring resource models:
  - DB2 Logging resource model
- Replication monitoring resource models:
  - DB2 Apply Replication Status resource model
  - DB2 Replication Capture resource model
  - DB2 Table Apply Replication resource models
- Miscellaneous resource models:
  - DB2 Fast Communication Manager resource model
  - DB2 Sorting resource model

For more information on resource models, refer to the *IBM Tivoli Monitoring for Databases DB2 Reference Guide Version 5.1.0*, SC23-4727.

3.2.2 IBM Tivoli Monitoring for Databases: Oracle resource models

IBM Tivoli Monitoring for Databases: Oracle comes with a number of resource models ready to use with little or no customization. The following lists the available resource models that are shipped with IBM Tivoli Monitoring for Databases: Oracle, grouped by functions:

- Process monitoring resource models:
  - Listener State resource model
  - Process State resource model
  - RDBMS State resource model
- Memory usage monitoring resource models:
  - SGA resource model
- Activity monitoring resource models:
  - Full Table Scan resource model

  - I/O resource model
  - Job Queue resource model
  - Multi-Threaded Server resource model
  - Other performance monitor resource model
  - Recursive Calls resource model
  - Transaction resource model
- Disk space usage monitoring resource models:
  - Dump Space resource model
  - Extents resource model
  - Extents by Tablespace resource model
  - Extents by User resource model
  - Free Space Deficit resource model
  - Free Space Deficit By Tablespace resource model
  - Free Space Deficit by User resource model
  - Free Space Fragmentation resource model
  - Free Space Fragmentation by Tablespace resource model
  - Free Tablespace resource model
  - Free Tablespace by Tablespace resource model
  - Maximum Extents resource model
  - Maximum Extents by Tablespace resource model
  - Maximum Extents by User resource model
  - Temporary Extents resource model (This resource model is not applicable for Oracle 9i)
  - Temporary Extents by Tablespace resource model
  - Temporary Extents by User resource model
  - Other Storage resource model (This resource model is only applicable for Oracle 9i only)
- Lock monitoring resource models:
  - Lock resource model
- Log monitoring resource models:
  - Archive Destination Logs resource model
  - Log Event resource model

  - Redo Log resource model
  - Rollback Segment resource model
  - Undo Space resource model
- Miscellaneous resource models:
  - Advanced Queue resource model
  - Checkpoints resource model
  - Data Warehouse Collector resource model
  - SQL Number resource model
  - SQL String resource model

For more information on the resource models available with IBM Tivoli Monitoring for Databases: Oracle, please refer to the *IBM Tivoli Monitoring for Databases Oracle Reference Guide Version 5.1.0*, SC23-4724.

3.2.3 IBM Tivoli Monitoring for Databases: Informix resource models

IBM Tivoli Monitoring for Databases: Informix comes with a number of resource models ready to use with little or no customization. The following are the available resource models that are shipped with IBM Tivoli Monitoring for Databases: Informix, grouped by their functions:

- Process monitoring resource models:
  - IBMInformix State Monitor resource model
  - IBMInformix Archive Monitor
- Memory usage monitoring resource models:
  - IBMInformix Cache Hit Ratio Monitor resource model
  - IBMInformix LRU Queues Monitor resource model
  - IBMInformix Memory Segment Monitor resource model
- Activity monitoring resource models:
  - IBMInformix Active Transactions Monitor
  - IBMInformix Overflows Monitor resource model
  - IBMInformix Virtual Processors Monitor resource model
  - IBMInformix Waits Monitor resource model
  - IBMInformix Writes Monitor resource model
- Disk space usage monitoring resource models:
  - IBMInformix Dbspace Monitor resource model

  - IBMInformix Free Dbspace Monitor resource model
  - IBMInformix Free Space Deficit Monitor resource model
  - IBMInformix Table Extents Monitor resource model
- Lock monitoring resource models:
  - IBMInformix Deadlocks Monitor resource mode
  - IBMInformix DML Locks Ratio Monitor resource model
- Log monitoring resource models:
  - IBMInformix Logical Log Monitor resource model
  - IBMInformix Logical Log Backup Monitor resource model
  - IBMInformix Physical Log Usage Ratio Monitor resource model
  - IBMInformix Rollback Ratio Monitor resource model
- Replication monitoring resource model
  - IBMInformix HDR Monitor resource model
- Miscellaneous resource models:
  - IBMInformix Checkpoint Monitor resource model
  - IBMInformix Update Statistics resource model

For more information on the resource models available with IBM Tivoli Monitoring for Databases: Informix, please refer to the *IBM Tivoli Monitoring for Databases Informix Reference Guide Version 5.1.0*, SC23-4728.

3.3 Customizing resource models

Although a supplied resource model is sufficient for most cases, in the implementation of the product, we were often faced with a requirement to perform a monitoring that slightly different from what is supplied. In this case, we can customize the existing resource model.

Customizing would only be required if the feature is not provided by the resource model or if you would like to add some new features and functionality. You can always create a resource model from scratch or you can use an existing resource model and modify it. It is a good idea to make a copy of the original file before customizing it, that way you can always revert back to the original if needs be.

In this section, we provide a sample customization for DB2InstanceStatus and Oracle FreeTablespace and add actions for the model. In the examples, we do not change any existing processing logic of the resource model.

3.3.1 DB2InstanceStatus resource model

Here in the ITSO lab, we have customized the DB2InstanceStatus resource model to automatically start the DB2 service if it happens to fall over. In order to customize resource models, you will need to install the IBM Tivoli Monitoring Workbench Version 5.1.1. For more information on how to install the Workbench, refer to the *IBM Tivoli Monitoring Version 5.1: Advanced Resource Monitoring*, SG24-5519.

These are the steps that we performed to customize the resource model:

1. Start the IBM Tivoli Monitoring Workbench using the  icon or select **Programs** -> **IBM Tivoli Monitoring 5.1.1** -> **Workbench** -> **Workbench**.
2. Once the GUI is launched, open the Distributed Monitoring Java Script Workspace (dmjsws) file provided in the IBM Tivoli Monitoring for Databases: DB2 CD under the Workbench directory. Select **File** -> **Open...**. Figure 3-2 shows the Workbench directory of the IBM Tivoli Monitoring for Databases: DB2.

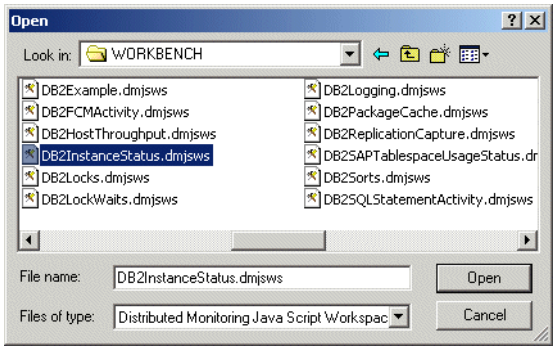


Figure 3-2 Opening the Java script workspace file

3. Once the Java script workspace is opened, you see DB2InstanceStatus in the left window pane. You may want to change the resource model name so that it would not overwrite the existing ones. Right-click on the DB2InstanceStatus and select **Modify**. The resource model general settings window is shown in Figure 3-3 on page 54. We name the new model DB2InstanceStatus1. Click **Apply**.

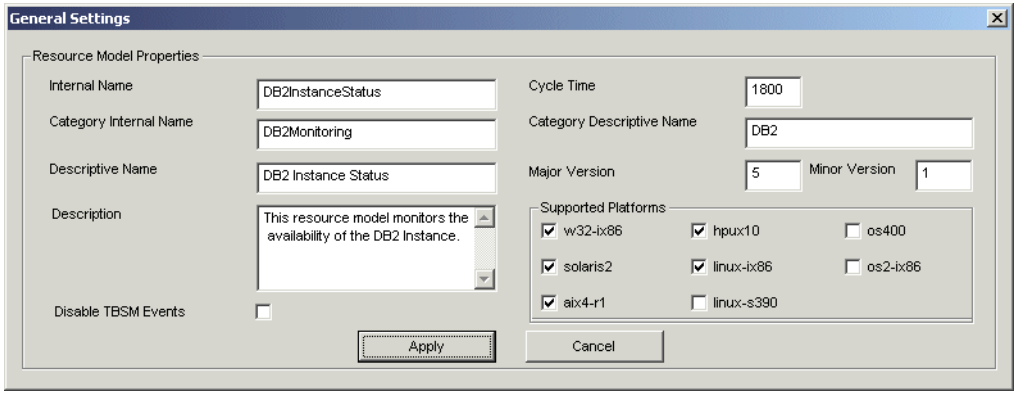


Figure 3-3 General settings window

4. Expand the resource model by clicking on the + sign until you get to the DB2DownStatus, as shown in Figure 3-4.

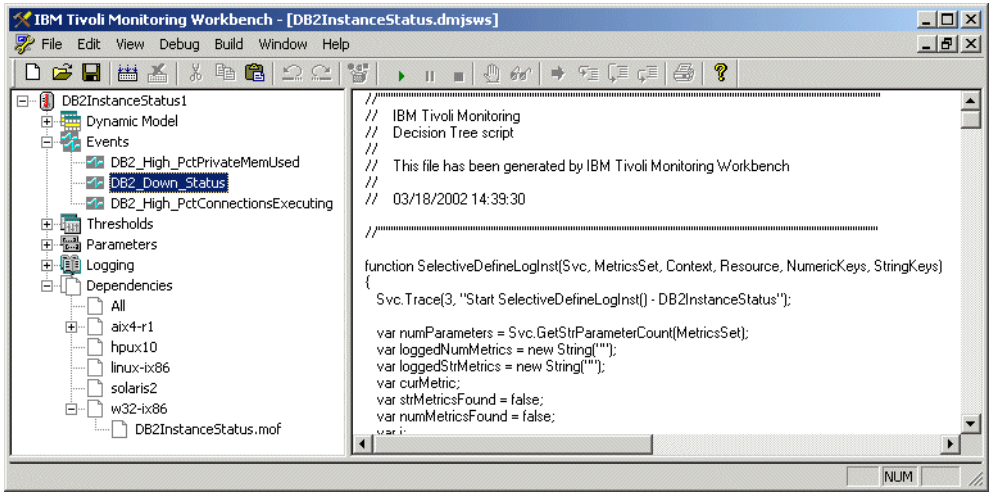


Figure 3-4 DB2InstanceStatus tree

5. Right-click on DB2_Down_Status and select **Modify** from the pop-up menu. The dialog in Figure 3-5 on page 55 is displayed.

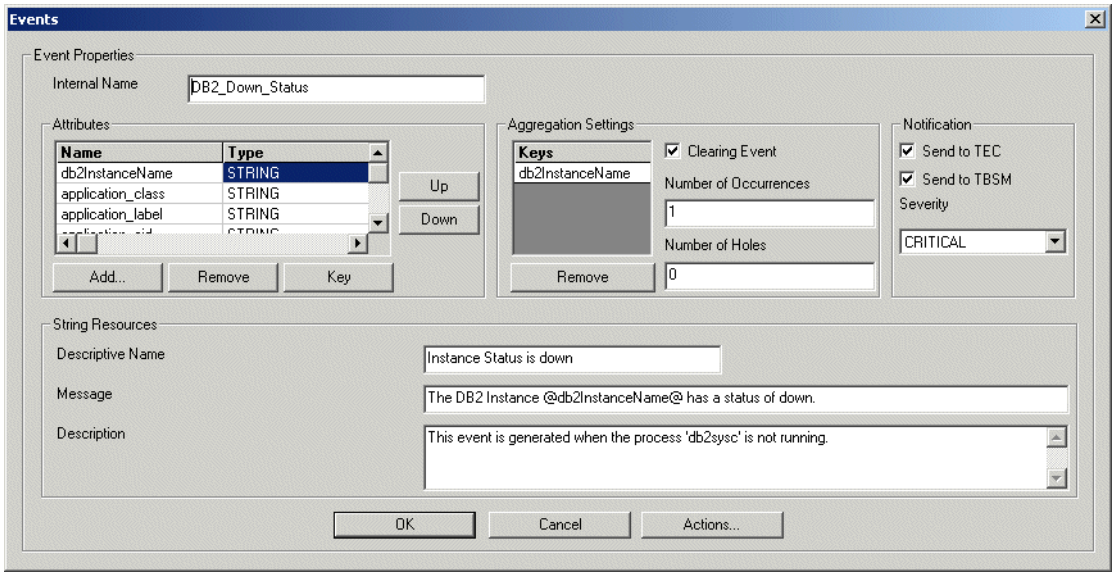


Figure 3-5 Events dialog

6. Click on the **Actions** button and the Actions dialog box will appear, as shown in Figure 3-6.

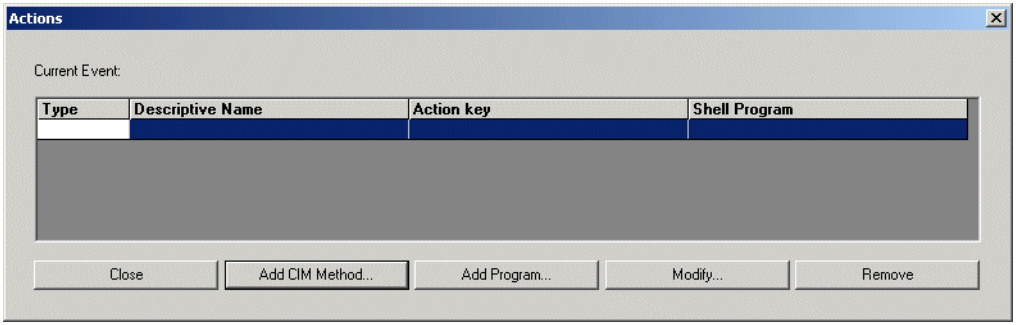


Figure 3-6 Actions dialog

7. Click the **Add Program...** button and fill in all the fields in the Define the Run Program Action dialog box, as shown in Figure 3-7 on page 56.

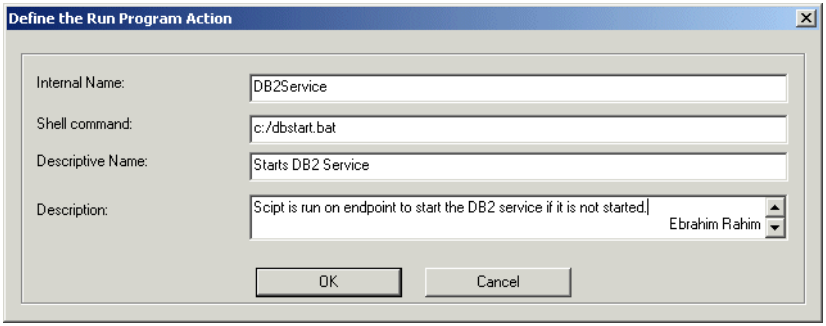


Figure 3-7 Run program action

8. Once you have filled in all the required fields, click **OK**. The Actions dialog box is displayed with the new action you have just added. Click the **Close** button to close this window as well. Click the **OK** button on the Events dialog box as well to close it.
9. In the ITSO environment, we tested this action on a Windows 2000 machine; a similar action can be done for a UNIX machine by using a shell script containing the relevant platform specific commands.

   We created the batch file using notepad. The contents are shown in Example 3-1.

Example 3-1 dbstart.bat

```
@echo off
cmd /c "net start db2 > c:\temp\started.txt"
exit
```

10. We now need to get the dbstart.bat file on the endpoint to run in the event that the DB2 service stops. To do this task, we need to create a dependency. From the main left window pane, similar to Figure 3-4 on page 54, do the following:
    a. Select Dependencies.
    b. Click on the plus sign to expand it.
    c. Select w32-ix86.
    d. Right-click and select **Add** from the pop-up menu.
    e. Select the dbstart.bat file from the directory where you created the file and click **Open**.
11. Once all this is done, you will see the dependency under the w32-ix86 Dependencies, as shown Figure 3-8 on page 57.

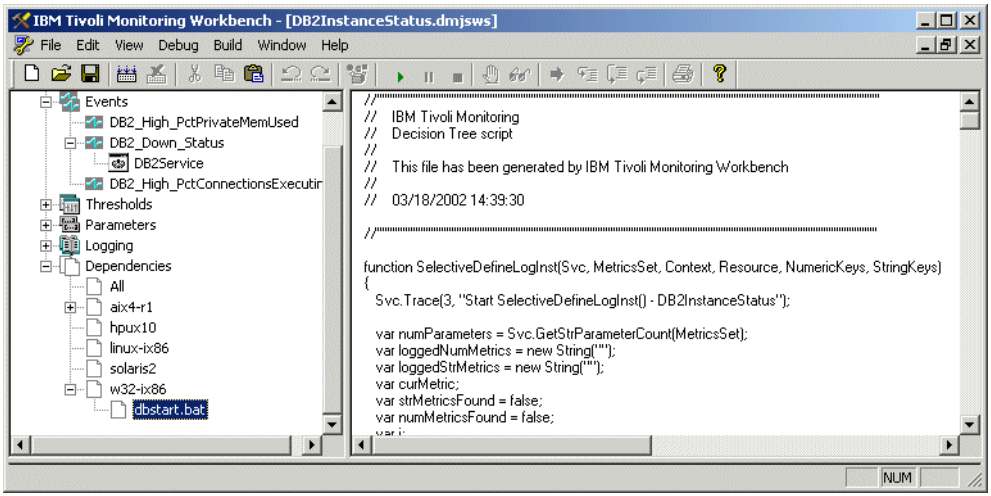


Figure 3-8 Dependencies

12. You will now need to build the package. From the **Build** menu, select **Build Package**. The Save As dialog box opens, as shown in Figure 3-9.

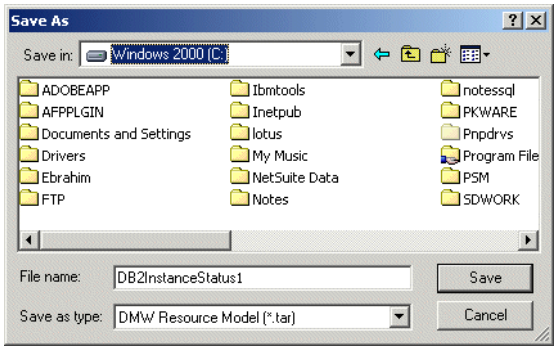


Figure 3-9 Save as dialog

Important: Save the file with a .tar extension

13. You will need to copy the tar file onto your Tivoli Management Region (TMR) server in order to import the resource model.

14. Once the file is copied, issue the **wdmrm** command, and the output should look like Example 3-1.

Example 3-2 Output of wdmrm command

```
# wdmrm -add DB2InstanceStatus1.tar
IBM Tivoli Monitoring - Adding new resource model

Copying DB2InstanceStatus1.cat msgfile ...
Copying DB2InstanceStatus1.cat zipfile ...


IBM Tivoli Monitoring - Resource model utility

Parsing configuration file DB2InstanceStatus1.conf ...
Configuration file successfully parsed.
Checking for event redefinition...
Starting resource DB2InstanceStatus1 registration ...
the resource DB2InstanceStatus1 has been successfully stored.
Registration completed.
Installation completed.
```

15. If you create a new profile now containing the DB2InstanceStatus1 resource model, you should see in the Indications section under the Action List; the DB2 Service action, if the Instance Status is down, is selected from the indications.

When we have the new resource model, we create a Tmw2kProfile with the DB2InstanceStatus1 resource model and distribute it. For testing purposes, we stop the DB2 instance in the jakarta endpoint to see the result of the action.

3.3.2 Free Tablespace by Tablespace resource model

For the Oracle Free Tablespace by Tablespace resource model, we customized it to automatically send an e-mail to the relevant administrator once a Tablespace has exceeded the predefined limit. We use the IBM Tivoli Monitoring Workbench Version 5.1.1. For more information on how to install the Workbench, refer to the *IBM Tivoli Monitoring Version 5.1: Advanced Resource Monitoring*, SG24-5519.

We use a shareware called NetMailBot for sending e-mail from command line, which can be downloaded from:

http://www.exclamationsoft.com/exclamationsoft/NetMailBot/

We perform the customization similar to the DB2InstanceStatus customization in 3.3.1, “DB2InstanceStatus resource model” on page 53. The following is the detailed steps:

1. From the IBM Tivoli Monitoring Workbench mail window, open the FreeTablespaceByTablespace.dmjsws Java script from the IBM Tivoli

Monitoring for Databases: Oracle CD under the WORKBENCH directory. The resulting window looks like Figure 3-10.

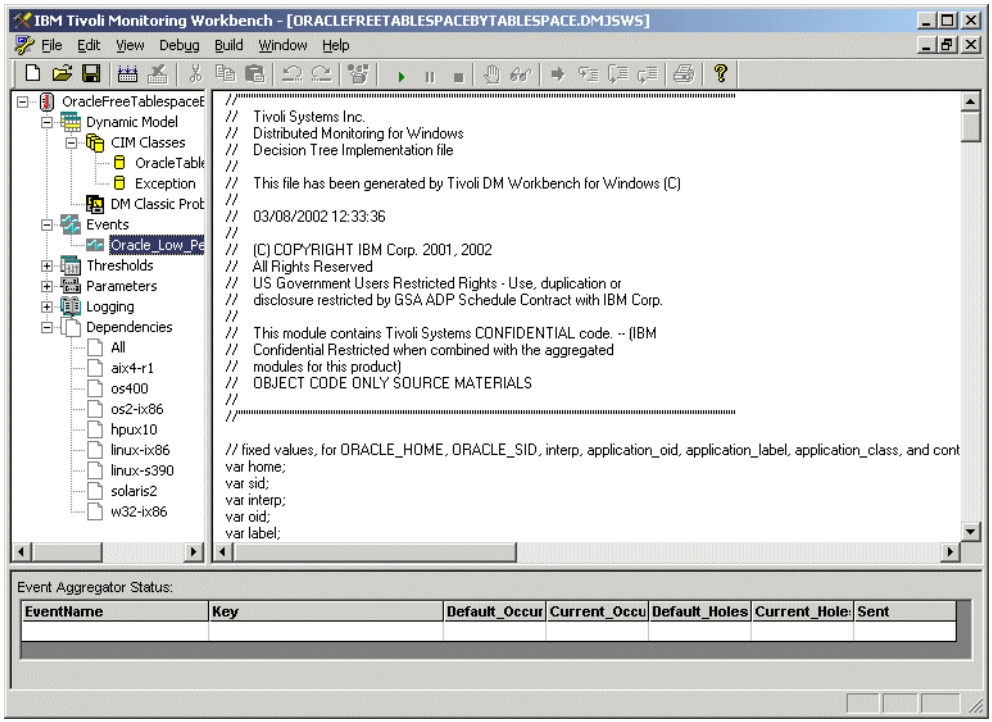


Figure 3-10 Oracle Free Tablespace by Tablespace resource model

2. Once you have opened the Java script resource model, you may want to rename the resource model. Right-click on the name OracleFreeTablespacebyTablespace and select **Modify**. Change the name and then click **Apply**. We use the name OracleFreeTablespacebyTablespace1.
3. Navigate the left pane from the OracleFreeTablespaceByTablespace1 to get to the Oracle_Low_PercentFreeTablespaceByTablespace event.
4. Right-click the event and select **Modify**. You should see the Events dialog box (see Figure 3-5 on page 55). Click **Actions**, and then add a new program using the **Add Program...** command button from this dialog to open the Define the Run Program Action dialog, as shown in Figure 3-11 on page 60.

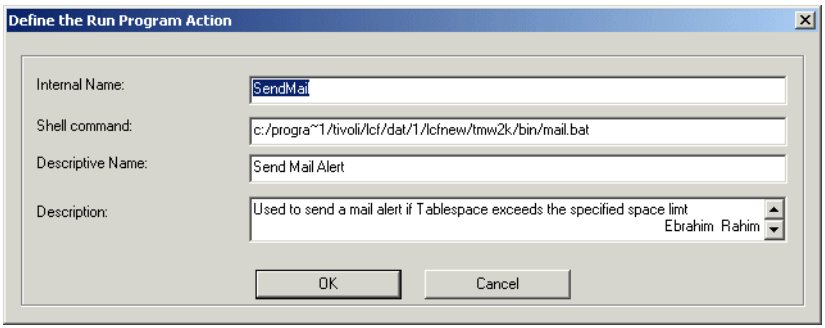


Figure 3-11 Define run action dialog

5. We then need to prepare the batch file script. The mail.bat is shown in Example 3-3.

Example 3-3 mail.bat

```
@echo off
cmd.exe /c C:\NetMailBot.exe NetMailBot -to rahimeb@za.ibm.com -from
root@capecod -subject "TABLESPACE LOW" -server smtp.prserv.com -body "Please
attend to the problem Ebrahim"
exit
```

6. Select **OK** to close the window. Close the Actions dialog box by clicking the **Close** command button. Click the **OK** button in the Events dialog box to save and close.
7. We now have to create a dependency to run the e-mail software once an event triggers. To do this, expand the dependencies section by clicking on the plus sign and select w32-ix86 for windows platforms. Put in the dependency for the following files:
   - NetMailBot.exe
   - BotDlll.dll
   - DBDll.dll
   - MailDll.dll
   - MailListBotDll.dll

   Right-click on w32-ix86 and select **Add**; browse to the location where the files resides and click on **Open**. Once you have specified all the necessary files, your file tree structure should look similar to Figure 3-12 on page 61.

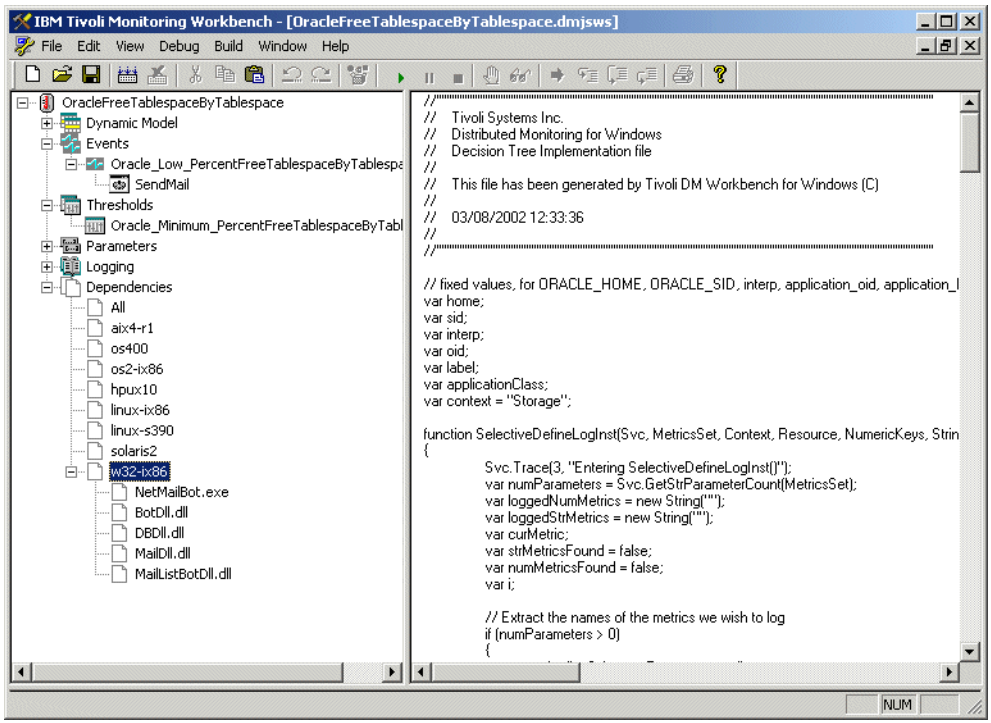


Figure 3-12 Dependencies for sending e-mail

8. Build the package and save it as a tar file. Copy it onto the TMR server and import the resource model. For more information on building and importing the package, refer to 3.3.1, “DB2InstanceStatus resource model” on page 53.

In the ITSO environment, we used the threshold limit of 15%. Once there was less than 15% of space left in the tablespace, an indication would be triggered. The outcome of this resource model can be seen in Figure 3-13 on page 62.

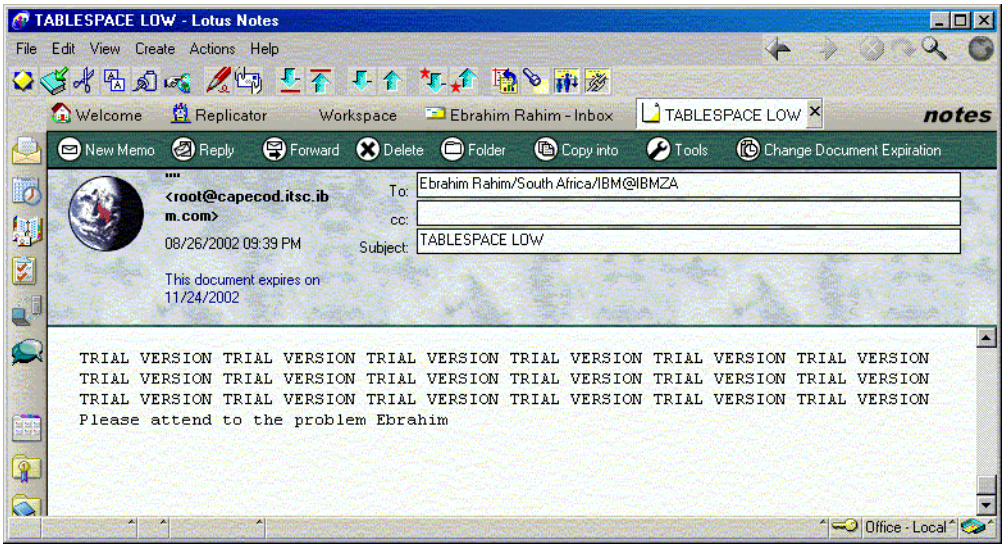


Figure 3-13 Mail output

4

Managing databases scenarios

This chapter discusses different monitoring scenarios along with operational tasks required for those scenarios. The scenarios are discussed for DB2, Oracle, and Informix databases. A simple model is introduced to explain various stages in a system monitoring environment. The resource models and tasks are briefly discussed for their relevance in each stage of the model.

The discussion consists of:

- 4.1, “Database administration” on page 64 describes the general model of managing a database systems; we introduce a model that we call a KIMA model.
- 4.2, “Sample operation scenarios” on page 65 shows the application of this model for several operational scenarios.
- 4.3, “Operational tasks” on page 77 discusses the available operational tasks in IBM Tivoli Monitoring for Databases.

4.1 Database administration

Data is the lifeblood for organizations, and it is stored in various Database Management Systems. The database systems can be from different vendors. Data, which becomes meaningful information after analysis, is crucial to the survival of any business. A Database Administrator (DBA) is responsible for ensuring continuous and efficient operation of the databases.

Typical DBA activities include the following:

- Ensuring that the database is updated accurately and regularly by maintaining data integrity
- Controlling access to databases (security and authorization)
- Ensuring that the users' problems are resolved in a timely manner
- Ensuring that the increasing demand for data access is resolved
- Disaster recovery
- Archiving and backup
- Database monitoring (starting, stopping, health check, and so on)
- Performance monitoring and tuning
- Capacity planning
- Installation, maintenance, and migration/upgrades of databases
- Database availability

The fact that the databases are from different vendors and on different platforms further complicates the job of a DBA. The evolution of DBA into e-DBA by linking the database to the Internet adds a new dimension to the role. This leads us into a clear need of comprehensive tools like IBM Tivoli Monitoring for Databases to manage a DBA’s work more efficiently.

In this chapter, we discuss commonly performed tasks for a DBA in the following areas:

- Installation and configuration
- Administration
- Performance monitoring and tuning
- Backup and restore

The discussion of these areas cover most of the activities a DBA needs to perform day-to-day.

The approach that we took involves a model called the KIMA process; the steps of the process are shown in Figure 4-1.

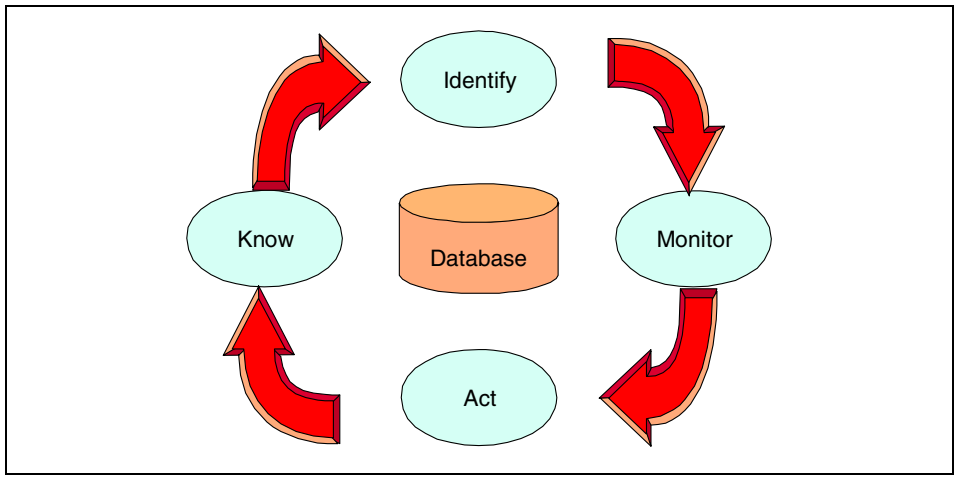


Figure 4-1 KIMA process

The process shown in Figure 4-1 can be broken down into these steps:

Know	Know the database management objectives and tasks
Identify	Identify the critical parameters to be controlled
Monitor	Monitor those parameters
Act	Act proactive

These stages are relevant for all types of databases and in different scenarios that we will discuss. IBM Tivoli Monitoring for Databases comes with rich features to better manage the database resources. The resource models and operational tasks provided with the IBM Tivoli Monitoring for Databases are used to monitor and manage the resources efficiently.

4.2 Sample operation scenarios

The operational scenarios are discussed in light of KIMA process for DB2, Oracle, and Informix databases. Default resource models and operational tasks that can be used are mentioned at the appropriate point. The scenarios are:

- 4.2.1, “Installation and configuration” on page 66
- 4.2.2, “Administration” on page 68
- 4.2.3, “Performance monitoring and tuning” on page 71

► 4.2.4, “Backup and restore” on page 75

4.2.1 Installation and configuration

Depending on the roles defined in an organization, the administrator or any other authorized personnel takes up the job of installing and configuring the database software. It is a daunting task; proper installation and configuration options influence the availability and performance of the database. The following basic tasks are considered for the installation and configuration scenario:

- Installing the database, tools, and patches
- Adding/changing/configuring a database instance
- Adding/changing/configuring a database
- Setting logging and auditing options

We consider the installation and configuration scenario along with our KIMA process and try to understand various issues involved with managing the scenario with available resource models and tasks from IBM Tivoli Monitoring for Databases. DB2, Oracle, and Informix databases are considered.

Know

During this stage, the database management objectives and basic tasks involved, like installing software, patches, configuring databases, logs, and so on are considered.

Identify

Critical parameters are identified, which can be monitored for further action, if required. During installation and configuration, the following parameters at operating system level are important:

- Memory
- Network interface
- Logical disks
- Processor

Monitor

Using the following default resource models or any custom resource model developed for a specific purpose, the resources can be monitored on different databases. The process status monitors for each of the components can be used to determine whether the database are active during the installation or configuration.

For DB2, the following resource models can be considered during the installation and configuration scenario

- DB2 Instance Status resource model: Monitors the availability of the DB2 instance

For Oracle, the following resource models can be considered for the installation and configuration scenario.

- Listener State resource model: Monitors the state of the Oracle listeners. The resource model can only be distributed to database instances.
- Process State resource model: Monitors the Oracle processes returning a state as low or unavailable.
- RDBMS State resource model: monitors the state of an Oracle database instance.

For Informix, the following resource model can be considered for the installation and configuration scenario

- IBMInformix State Monitor resource model: Monitors the state of the IBM Informix server

Act

Various individual tasks are to be carried during installation and configuration of databases. Using IBM Tivoli Monitoring for Databases, it is possible to automate some of the tasks. The tasks can be run manually or scheduled for periodic execution or run as a response to an event.

Typical tasks that can be used in this scenario relate to the configuration of the database engine, startup and shutdown of instances and processes, and various registration tasks. For more detail on the tasks of each category, refer to 4.3, “Operational tasks” on page 77.

For all databases, the IBM Tivoli Monitoring for Databases needs to re-discover the added or removed databases instances and servers. This can be performed by the discovery objects from the Tivoli desktop; refer to 2.3, “Installation steps overview” on page 19 for more information on the discovery process. You may also want to initiate the TBSM discovery tasks for each database systems, which are:

- ECC_TBSM_Discovery for DB2
- OracleTBSMDiscovery for Oracle
- TBSM_Discovery for IBM Informix

4.2.2 Administration

There are different possible roles in an organization for managing databases, depending on the magnitude of operations and internal policies. The roles may be called System Administrator, Data Administrator, and Database Administrator. For simplicity, we assume all the roles are managed by single administrator. In an administration scenario, we consider the following basic tasks:

- User administration (create, change, group, authorize, and so on)
- Scheduling jobs
- Log management
- Data integrity management
- Problem determination
- Managing database resources, such as tablespaces, bufferpools, and index
- Managing data (importing, loading, exporting, moving, and so on.)

The administration scenario explains the database objectives and the task required to accomplish those objectives using the default resource models and tasks. DB2, Oracle, and Informix databases are considered along with the Know, Identify, Monitor, and Act classification.

Know

We considered basic objectives of database administration and various tasks involved in administering databases, such as:

- User administration (create, change, group, authorize, and so on)
- Scheduling jobs
- Log management
- Problem determination
- Managing a system, instance, database, table space, tables, and so on

Identify

The necessary parameters in this scenario relate to the availability of resources and the status or number of usage. Some example for those parameters are:

- Disk space usage
- Number of users
- Average system load
- Size of data

Monitor

DB2, Oracle, and Informix databases are monitored using default/custom resource models, as explained in Chapter 3, “Database monitoring and resource model” on page 45. For the administration scenario, the monitoring requirement is usually more concerned with ensuring the continuity of the database operation. The disk space usage monitors and logging monitors are used extensively. For DB2, administrator also needs to monitor data replication using the replication monitors.

For DB2, the following resource models are relevant during the database administration phase:

- Disk usage resource models:
  - DB2 SAP Tablespace Usage/Status resource model
- Log monitoring resource models:
  - DB2 Logging resource model
- Replication monitoring resource models:
  - DB2 Apply Replication Status resource model
  - DB2 Replication Capture resource model
  - DB2 Table Replication Apply resource model

For Oracle, the following resource models are relevant during the administration stage:

- Data Warehouse Collector resource model: A mandatory resource model for Oracle integration with IBM Tivoli Enterprise Data Warehouse (TEDW); collects attribute data required by TEDW
- Dump Space resource model: Monitors the used space percentage for the dump destination directories
- Free Space Deficit resource model: Checks whether any segments are unable to allocate their next extent because of insufficient free space
- Free Space Deficit By Tablespace resource model: Checks whether any segments within specified tablespaces are unable to allocate their next extent because of insufficient free space
- Free Space Deficit By User resource model: Checks whether any segments for a specified user are unable to allocate their next extent because of insufficient free space
- Free Space Fragmentation resource model: Monitors the free space fragmentation index over all tablespaces within an Oracle instance
- Free Space Fragmentation by Tablespace resource model: Monitors the free space fragmentation index for specific tablespaces within an Oracle instance

- Free Tablespace resource model: Monitors the percentage of free space for each tablespace in a database
- Free Tablespace by Tablespace resource model: Monitors the percentage of free space for a specified tablespace
- Job Queue resource model: Monitors the number of broken, failed, and past due jobs in the job queue
- Lock resource model: Monitors total number of different type of locks by various criteria, such as per instance, per user, and so on
- Log Event resource model: Monitors Oracle background job events for errors
- Other Storage resource model: Monitors various issues related to chained rows, deleted rows, datafiles, and so on
- SQL Number resource model: Checks a resource for a number specified by a user-defined SQL select statement
- SQL String resource model: Checks a resource for an alphanumeric value specified by a user-defined SQL select statement
- Transaction resource model: Monitors transaction related issues, such as long running transactions, locked transactions, active transactions, and so on
- Undo Space resource model: Monitors the No Space Error and the Snapshot Too Old Error

For Informix, the following resource models are considered for administration scenario:

- IBMInformix Dbspace Monitor resource model: Logs information about each Dbspace
- IBMInformix Deadlocks Monitor resource model: Monitors the number of deadlocks in an IBM Informix database system
- IBMInformix DML Locks Ration Monitor resource model: Monitors the DML locks and total locks and calculates the percentage of DML locks to total locks
- IBMInformix Free Dbspace Monitor: Monitors the HDR type and state of the IBM Informix server
- IBMInformix Log Event Monitor resource model: Monitors the log event availability and raises an indication when the resource model logs a message to the IBM Informix message log
- IBMInformix Logical Log Monitor resource model: Monitors the logical log files and raises an indication when the available logical log space is less than the specified threshold value
- IBMInformix Physical Log Usage Ratio Monitor resource model: Monitors the physical log size and the amount of physical log used

- IBMInformix Table Extents Monitor resource model: Monitors the number of extents for every table
- IBMInformix Virtual Processors Monitor: Monitors the virtual processes

Act

Most of the automatic administration tasks that are required for this mode of operation relates to the starting and stopping subsystems regularly, either for scheduled maintenance or other purposes. Other regularly scheduled tasks can relate to auditing and log archival. These tasks are discussed later in 4.3, “Operational tasks” on page 77.

4.2.3 Performance monitoring and tuning

The performance of the system depends on various factors, such as load on the system, software and hardware capabilities, query formation, configuration parameters, and so on. To maintain and improve the performance, fine tuning of the system is required. This particular scenario carries more importance as the efficiency of operations depends on the tasks involved. Some of the tasks during performance monitoring and tuning are listed below.

- Monitoring performance
- Reorganizing a table
- Configuring database parameters
- Buffer pool management
- I/O management

The performance monitoring and tuning scenario consists of critical activities and the same is observed using default resource models and tasks available with the product. The Know, Identify, Monitor, and Act steps are documented for DB2, Oracle, and Informix databases for this scenario.

Know

We have considered the database management objectives related to performance monitoring and tuning along with basic tasks required to achieve the same. Some of the tasks are:

- Monitoring performance
- Configuring database parameters
- Buffer pool management
- I/O management

Identify

The performance parameter varies greatly based on the systems configuration. Some common parameters related to the system load and performance are mainly needed to understand the current performance and predict the future workload. Some necessary parameters to monitor are:

- CPU utilization
- Disk (data) growth
- Logging size, which indicate transaction activity
- Number of active connection
- Memory usage, such as cache utilization and buffer pool hit ratio
- Locking and deadlock information

Monitor

DB2, Oracle, and Informix databases are monitored using default/custom resource models, as explained in Chapter 3, “Database monitoring and resource model” on page 45.

For DB2, the following resource models are relevant for database performance monitoring and tuning:

- DB2 Agent resource model: Monitors the database agents and their related applications
- DB2 Buffer Pools resource model: Monitors buffer pool input, output, and utilization on a database, table space, and application
- DB2 Buffer Pool/Extended Storage resource model: Provides extended storage monitoring of buffer pool extended storage at database and table space level
- DB2 Catalog Cache resource model: Monitors the effectiveness of catalog cache
- DB2 CPU Utilization resource model: Monitors the central processing unit time used by applications database manager agents
- DB2 Direct IO resource model: Monitors non-buffered input and output at the database, table space, and application levels
- DB2 Host Throughput resource model: Monitors the DB2 host throughput
- DB2 SAP Tablespace Usage/Status resource model: Monitors table space usage and the status of DB2 database used by SAP R/3 application (SAP R/3 from SAP AG is a leading ERP application.)
- DB2 Sorting resource model: Monitors the sorting activity of the database manager at the database and application level

- DB2 SQL Cursor Activity resource model: Monitors the total number of local/remote open and blocking cursors that are currently open for the database

For Oracle, the following resource models are considered for performance monitoring and tuning scenario

- Advanced Queue resource model: Monitors for long waiting messages and performance bottlenecks
- Checkpoints resource model: Monitors number of checkpoints by database writer and the number of checkpoints requested by the server
- Extents resource model: Monitors and gives alerts on database objects when the extents exceed the specified limit for them
- Extents by Tablespace resource model: Gives alerts on database objects for the specified table spaces when the extents exceed the specified limit
- Extents by User resource model: For the specified user, gives alerts on database objects whose extents exceed the specified limit
- Full Table Scan resource model: Monitors and reports on percentage of total rows retrieved from full table scans, which gives an idea about the extent to which the performance can be tuned
- IO resource model: Monitors number of physical reads/writes for the datafiles
- Maximum Extents resource model: Gives alerts on segments whose free extents are less than the specified threshold
- Maximum Extents by Tablespace resource model: Gives alerts on segments, in specified tablespaces, whose free extents are less than the specified threshold
- Maximum Extents by User resource model: Gives alerts on segments, for a specified user, whose free extents are less than the specified threshold
- Multi-Threaded Server resource model: Monitors the percentage of time that dispatcher processes for the protocol with the most contention are busy; monitors the time a response waits in the response queue for a dispatcher process to route it, for the protocol with the longest average wait time; monitors the number of shared-server processes as a percentage of the instance limit; reports the average wait time a request waits in the request queue
- Other Performance Monitors resource model: Monitors various performance related metrics of an Oracle database
- PGA resource model: Monitors multipass executions, optimal percentage, work area percentage, and component area in the program global area

- Recursive Calls resource model: Monitors recursive call percentage, user calls, recursive calls, recursive call rate, and so on
- Redo Log resource model: Monitors various parameters related to redo log
- Rollback Segment resource model: Monitors the ratio of rollback segment header waits to rollback header gets, user rollbacks to total transactions, and consistent changes to consistent gets
- SGA resource model: Monitors the current ratio of buffer cache hits to total requests, dictionary cache hits to total requests for the monitoring interval, and the percentage of entries in the library cache that were parsed over the lifetime of the instance (for the current monitoring interval)
- Temporary Extents resource model: Gives alerts on temporary segments whose free extents are less than the specified threshold
- Temporary Extents by Tablespace resource model: Gives alerts on temporary segments, within specified tablespaces, whose free extents are less than specified threshold
- Temporary Extents by User resource model: Gives alerts on temporary segments, for a specified user, whose free extents are less than the specified threshold

For Informix, the following resource models are considered for the performance monitoring and tuning scenario:

- IBMInformix Active Transaction Monitor resource model: Monitors the number of active transactions in the system
- IBMInformix Cache Hit Ratio Monitor resource model: Monitors the number of reads and writes from both buffer cache and disk and calculates a ratio of the amount of buffer hits to disk hits for both reads and writes
- IBMInformix Checkpoint Monitor resource model: Monitors the IBM Informix checkpoint availability
- IBMInformix LRU Queues Monitor resource model: Monitors the free buffers and modified buffers and calculates the percentage modified buffers for each LRU queue
- IBMInformix Memory Segment Monitor resource model: Monitors and logs the IBM Informix Memory segments
- IBMInformix Overflows Monitor resource model: Monitors the user thread overflows, transaction overflows, and lock overflows
- IBMInformix Rollback Ratio Monitor resource model: Monitors the ratio of rollbacks to commits and generates an event if the ratio exceeds the threshold for the indication that belongs to this resource model

- IBMInformix Update Statistics resource model: Monitors table data distribution by comparing the ratio of table modifications to table rows
- IBMInformix Waits Monitor resource model: Monitors buffer waits, lock waits, and latch waits
- IBMInformix Writes Monitor resource model: Monitors the chunk writes, LRU writes, and foreground writes

Act

Some important tasks for ensuring performance can relate to the reconfiguration of database parameters and online tasks to update the systems, such as reorganization and updating statistics.

4.2.4 Backup and restore

The vital organizational data is being stored in databases which makes it imperative to make sure that the data is always available. But, there are possibilities that make the data unavailable or unusable. The reasons may vary from human or application errors to natural disasters. These circumstances lead us to have planned and unplanned backups of the database. It is possible to restore the systems, in case of any unforeseen interruptions, to a particular point using backed up data. Backup and restore are crucial activities for timely availability of data. The scenario encompasses the following tasks:

- Backing up a database/table space
- Checking data in a table space and copying a table space
- Restoring a database/table space
- Rolling forward a database/table space

Back and restore is a regular activity for databases. This scenario is looked at in light of the Know, Identify, Monitor, and Act steps for DB2, Oracle, and Informix databases.

Know

We tried to understand the basic objectives of the backup and restore scenario and realized the following common tasks for all the databases:

- Backing up a database/table space
- Restoring a database/table space

Identify

Backup is the primary task that needs to be performed, along with the restore procedure testing. For this task, most of the parameters that need to be identified are related to the backup size, backup frequency, and success of the backup process.

Monitor

We monitored the database systems for DB2, Oracle, and Informix, along with the backup and restore activities.

For DB2, we used the following resource models to monitor the resources:

- DB2 Apply Replication Status resource model: Monitors the status of the DB2 Apply component from IBM Replication. Replication is a process of maintaining a defined set of data in more than one location. It involves copying designated changes from one location (a source) to another (a target), and synchronizing the data in both locations. The source and target can be in logical servers that are on the same machine or on different machines in a distributed network.
- DB2 Replication Capture resource model: Monitors the status of DB2 Capture component from IBM Replication.

For Oracle, we considered the following resource model for the backup and restore scenario:

- Archive Destination Logs resource model: Monitors the Oracle archive logs and the amount of free space and the percentage of free space in the Oracle archiving log destination

For Informix, the following resource models are relevant for the backup and restore scenario:

- IBMInformix Archive Monitor resource model: Monitors the IBM Informix backup processes at cycle time
- IBMInformix Logical Log Backup Monitor resource model: Monitors the informix online log file for logical log backups

Act

The action that relates to the backup and restore process can be assisted by automated tasks.

4.3 Operational tasks

This section discusses the available operational tasks from IBM Tivoli Monitoring for Databases. The tasks supplied with IBM Tivoli Monitoring for Databases are categorized into:

- Product implementation tasks: These tasks help in installing and configuring the IBM Tivoli Monitoring for Databases software.
- Administration tasks: These tasks start and stop subsystems and instances of the database engine.
- Other tasks: These tasks perform specialized functions for the related database.

Most of the supplied tasks for IBM Tivoli Monitoring for Databases relate to the product implementation and administration. IBM Tivoli Monitoring for Databases: DB2 supplied quite an extensive collection of utilities, which do not exist for the other modules.

4.3.1 IBM Tivoli Monitoring for Databases: DB2

These DB2 tasks are available:

- Product implementation tasks:
  - ECC_Start_DB2SNMP_Agent
  - ECC_Set_Up_Monitoring_Agent
  - ECC_Start_Monitoring_Agent
  - ECC_Stop_DB2SNMP_Agent
  - ECC_Stop_Monitoring_Agent
  - ECC_Send_Files_To_TEC_TMR
  - ECC_TBSM_Discovery
  - ECC_Configure_TEC_Classes
- Notifying other users and operator:
  - ECC_Broadcast_Message: Broadcasts message to all Tivoli desktops
  - ECC_Send_Notice: Sends notice to all Tivoli administrators, which can be retrieved by them using the desktop
- Modifying and changing audit feature:
  - ECC_Configure_Audit: Makes changes that take effect in the DB2 auditing facility

  - ECC_Get_Audit_Configuration: Returns the current audit configuration to audit events
  - ECC_Reset_Audit_Configuration: Resets the DB2 audit configuration to the initial configuration
- DB2 administration server manipulation:
  - ECC_Create_Admin_Server: Creates a DB2 administration server
  - ECC_Drop_Admin_Server: Drops the DB2 administration server
- DB2 sample database manipulation for testing:
  - ECC_Create_Sample_Database: Creates a sample DB2 database
  - ECC_Drop_Sample_Database: Drops the DB2 sample database
- Changing configuration for administration server:
  - ECC_Get_Admin_Configuration: Retrieves the DB2 administration configuration
  - ECC_Reset_Admin_Configuration: Resets the configuration values in the administration server to the system default values
  - ECC_Update_Admin_Configuration: Modifies the values in the database manager configuration file that are relevant for the DB2 administration server
- Changing configuration for a database:
  - ECC_Get_Database_Configuration: Gets the configuration of DB2 database from the configuration file
  - ECC_Reset_Database_Configuration: Resets the values in a specific database configuration file to system defaults
  - ECC_Update_Database_Configuration: Updates the database configuration with specific values
- Changing configuration for a database instance:
  - ECC_Get_Database_Manager_Configuration: Retrieves individual values from the database manager configuration file
  - ECC_Reset_Database_Manager_Configuration: Resets the configuration parameters in a database manager configuration file to system defaults
  - ECC_Update_Database_Manager_Configuration: Updates the database manager configuration values
- Additional utilities for setup and configuration:
  - ECC_Create_Explain_Tables: Creates explain tables
  - ECC_Get_Admin_Server_Instance: Retrieves the instance name on which the DB2 administration server is running

  - ECC_List Node_Directory: Lists the contents of a node directory
- Updating application packages
  - ECC_Rebind_All_Packages: Invokes the **rebind** command on all DB2 packages
  - ECC_Rebind_Package: Rebinds a specific DB2 package
- Auditing
  - ECC_Create_Import_Audit_Log: Extracts audit records
  - ECC_Extract_Audit_Log: Copies the Audit Facility log into a specified file
  - ECC_Flush_Audit_Buffer: Writes pending audit records to the Audit log
  - ECC_Prune_Audit_Log: Deletes records in the Audit Facility log file
  - ECC_Purge_Audit_Log: Deletes Audit records in the Audit Facility log
  - ECC_Start_Audititing: Starts auditing events
  - ECC_Stop_Auditing: Stops auditing events
- Administration tasks
  - ECC_Activate_Database: Activates a DB2 database
  - ECC_Restart_Database: Restarts a DB2 database
  - ECC_Set_Admin_Server_ID: Adds or modifies the user associated with the DB2 administration server instance
  - ECC_Start_Admin_Server: Starts the DB2 administration server
  - ECC_Start_DB2: Starts DB2
  - ECC_Start_DB2_NT_Security: Starts the DB2 security service on a Windows system
  - ECC_Stop_Admin_Server: Stops the DB2 administration server
  - ECC_Stop_DB2: Stops DB2
  - ECC_Stop_DB2_NT_Security: Stops the DB2 security service on a Windows system
- Application management
  - ECC_Force_All_Applications: Forces all DB2 applications off the system
  - ECC_Force_Application: Forces a specific DB2 application off the system
  - ECC_List_Applications: Lists all DB2 applications
  - ECC_List_DCS_Applications: Lists DB2 DCS applications
- Other tasks
  - ECC_Invoke_Stored_Procedure: Invokes a stored procedure

  - ECC_Broadcast_Message: Broadcasts messages to all Tivoli desktops active
  - ECC_Send_Notice: Sends notices about regular or sudden backup and restore activities to all the Tivoli administrators
- Backup and recovery support tasks
  - ECC_Backup_Database: Creates backup of a DB2 database
  - ECC_List_Backup_Recovery_File: Lists the DB2 backup and recovery history file
  - ECC_Prune_Recovery_History_File: Deletes the DB2 recovery history file
- Replication tasks
  - ECC_Apply_Start: Starts the data replication Apply program
  - ECC_Apply_Stop: Stops the data replication Apply program
  - ECC_Capture_Get_Log_Seq: Retrieves the current log sequence number and time stamp
  - ECC_Capture_Prune: Prunes the data replication Capture program
  - ECC_Capture_Reinit: Reinitializes the data replication Capture program
  - ECC_Capture_Resume: Resumes the data replication Capture program
  - ECC_Capture_Start: Starts the data replication Capture program
  - ECC_Capture_Stop: Stops the data replication Capture program
  - ECC_Capture_Suspend: Suspends the data replication Capture program
- Invoking DB2 utility
  - ECC_Reorganize_Table: Reorganizes a DB2 database table
  - ECC_Reorgchk: Runs the DB2 command REORGCHK
  - ECC_Run_Statistics: Runs DB2 statistics

4.3.2 IBM Tivoli Monitoring for Databases: Oracle

The following tasks exist for IBM Tivoli Monitoring for Databases: Oracle management:

- Product implementation tasks:
  - ConfigureTECOracle
  - OracleTBSMDiscovery
  - DisableResourceModels
  - EnableResourceModels

- Administrative tasks:
  - Listener: Starts or stops the listener process
- Configuration tasks:
  - AdvancedNetworkOption: Enables or disables the advanced network option
- Other tasks:
  - CurrentRunningSQL: Shows the current SQL statements for any user connected to Oracle database, allowing you to evaluate current statements for any problems

4.3.3 IBM Tivoli Monitoring for Databases: Informix

The following tasks exists for IBM Tivoli Monitoring for Databases: Informix management:

- Product implementation tasks:
  - Configure_TEC
  - Send_TEC_Files_To_TEC
  - TBSM_Discovery
- Administrative tasks:
  - Start-up_To_On-Line: Starts the IBM Informix server in the multi-user online state
  - Start-up_To_Quiescent: Starts a database server in the administrative state from offline state
  - Stop_To_Off-Line: Shuts down the server from an online or quiescent state and removes the online shared memory
  - Stop_To_Quiescent_(Gracefully): Takes the server to the quiescent state for administrative purposes while allowing running processes to finish before shutting down
  - Stop_To_Quiescent_(Immediately): Takes the server to the quiescent state for administrative purposes immediately

5

Real-time monitoring

In this chapter, we discuss the implementation of real-time monitoring with IBM Tivoli Monitoring for Databases, which can be extended to include event correlation and business impact analysis using Tivoli Enterprise Console and Tivoli Business Systems Manager. The discussion consists of:

- 5.1, “Web Health Console” on page 84 shows an overview of how the Web Health Console can be used in the IBM Tivoli Monitoring for Databases context.
- 5.2, “Tivoli Enterprise Console integration” on page 86 discusses the basic integration points between TEC and IBM Tivoli Monitoring for Databases.
- 5.3, “Supplied events and rules” on page 87 lists the events and rule files that are supplied by IBM Tivoli Monitoring for Databases.
- 5.4, “Configuring Tivoli Enterprise Console rulebase” on page 91 describes the steps that need to be performed for the TEC rulebase to process events from IBM Tivoli Monitoring for Databases.
- 5.5, “Tivoli Business Systems Manager” on page 94 discusses the basic integration between TBSM and IBM Tivoli Monitoring for Databases.
- 5.6, “Setting up TBSM integration” on page 96 describes the tasks needed to set up the integration.
- 5.7, “Using Tivoli Business Systems Manager” on page 98 shows how TBSM monitors IBM Tivoli Monitoring for Databases resources.

5.1 Web Health Console

You can use the Web Health Console to check, display, and analyze the status and health of resources that reside in an endpoint from the profiles and resource models that are deployed. Status reflects the state of the endpoint displayed on the Web Health Console, such as running or stopped. Health is a numeric value determined by resource model settings. The typical settings include required occurrences, cycle times, thresholds, and parameters for indications. These are defined when the resource model is created in the IBM Tivoli Monitoring Workbench. You can also use the Web Health Console to work with real-time or historical data from an endpoint that is logged to the IBM Tivoli Monitoring database.

5.1.1 Configuration

If IBM Tivoli Monitoring for Databases has been installed using rapid deployment (see 2.5, “Rapid deployment installation support” on page 40), you can use the Web Health Console without any additional configuration. If you used any other method to install your IBM Tivoli Monitoring for Databases, then you need to configure the Web Health Console to view events. Perform the following tasks on the machine where Web Health Console is installed:

1. Insert the product CD-ROM in the CD-ROM drive and go to the /HCONSOLE directory, there are a set of Java class files that provide a text translation for the resource and health status names.
2. Copy the class files into the relevant HCONSOLE directory for your IBM Tivoli Monitoring for Databases to the directory on the server where the Web Health Console is installed:

   ```
   <WHC_dir>/installedApps/dm.ear/dm.war/WEB-INF/classes/com/tivoli/DmForNt/
   resources
   ```

   where WHC_dir is the directory where you installed the Web Health Console.
3. Stop and restart the Web Health Console.

   Once the Web Health Console has been configured, you can connect the Web Health Console to any Tivoli management region server or managed node and configure it to monitor any or all of the endpoints that are found in that region. The Web Health Console does not have to be within the region itself, although it could be. To connect to the Web Health Console you need access to the server on which the Web Health Console server is installed and the Tivoli management region on which you want to monitor health. All user management and security is handled through the IBM Tivoli management environment. This includes creating users and passwords, as well as assigning authority.

5.1.2 Usage overview

Use the following steps if you are logging on to the Web Health Console for the first time:

1. To log on to the Web Health Console, use a Web browser. Go to the URL `http://<server_name>/dmwhc/` (specific to the Health Console machine). Input the appropriate user ID, password, and host name for the TMR that you want to use.
2. The first time you log on to the Web Health Console, the Preferences view is displayed. You must populate the Selected Endpoint list before you can access any other Web Health Console views. When you log on subsequently, the endpoint list is automatically loaded.
3. Select the endpoints that you want to see and choose the Endpoint Health view. This is the most detailed view of the health of an endpoint. In this view, the following information is displayed:
   - The health and status of all resource models installed on the endpoint
   - The health of the indications that make up the resource model and historical data

Figure 5-1 on page 86 shows an example of real-time monitoring of a DB2 server.

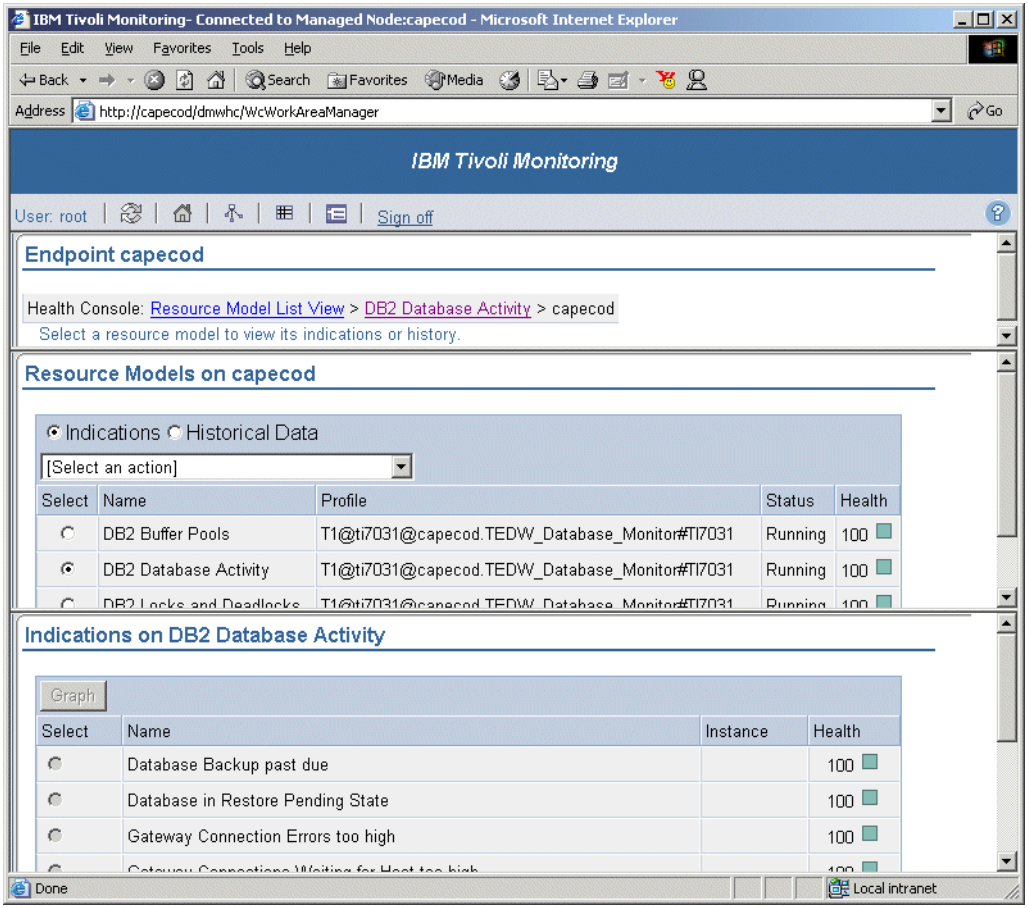


Figure 5-1 Using the Web Health Console

5.2 Tivoli Enterprise Console integration

Tivoli Enterprise Console (TEC) receives events from multiple sources and perform processing on them to correlate and aggregate events. TEC works on the basis of events and rules.

TEC events are defined in object-oriented definition files called baroc files. These events are defined hierarchically according to their type. Each event type is called an event class. When TEC receives an event, it parses the event to determine the event class and then applies the class definition to parse the rest of the event. When the parsing is successful, the event is stored in the TEC database.

When a new event is stored, a timer is expired, or a field (or slot in TEC's term) is changed, TEC evaluates a set of rules to be applied to the event. These rules are stored in ruleset files, which are written in the Prolog language. When a matching rule is found, the action part of the rule gets executed. These rules enable events to be correlated and aggregated. It can also define automatic responses to certain conditions; usually, these are corrective actions.

In IBM Tivoli Monitoring for Databases perspective, events are generated from monitoring profiles by the database monitors. These events are defined in TEC and a set of predefined rule exists to correlate and process these events.

5.3 Supplied events and rules

The IBM Tivoli Monitoring for Databases events are documented in the following baroc files:

- DB2 events: The hierarchy can be seen in Figure 5-2 on page 88.
  - DB2_Event.baroc: The primary baroc file that is a sub-class of the TMW_Event.
  - Other baroc files are sub-classes of the DB2_Event and represent the event that can be generated by the resource models:
    - DB2Agents.baroc
    - DB2ApplyReplication.baroc
    - DB2BufferPool.baroc
    - DB2BufferPoolExtStorage.baroc
    - DB2CatalogCache.baroc
    - DB2CpuUtilization.baroc
    - DB2Cursor.baroc
    - DB2DatabaseStatus.baroc
    - DB2DirectIO.baroc
    - DB2FCMActivity.baroc
    - DB2HostThroughput.baroc
    - DB2InstanceStatus.baroc
    - DB2Locks.baroc
    - DB2LockWaits.baroc
    - DB2Logging.baroc
    - DB2PackageCache.baroc

- DB2ReplicationCapture.baroc
- DB2SAPTablespaceUsageStatus.baroc
- DB2Sorts.baroc
- DB2SQLStatementActivity.baroc
- DB2TableActivity.baroc
- DB2TableApplyReplication.baroc

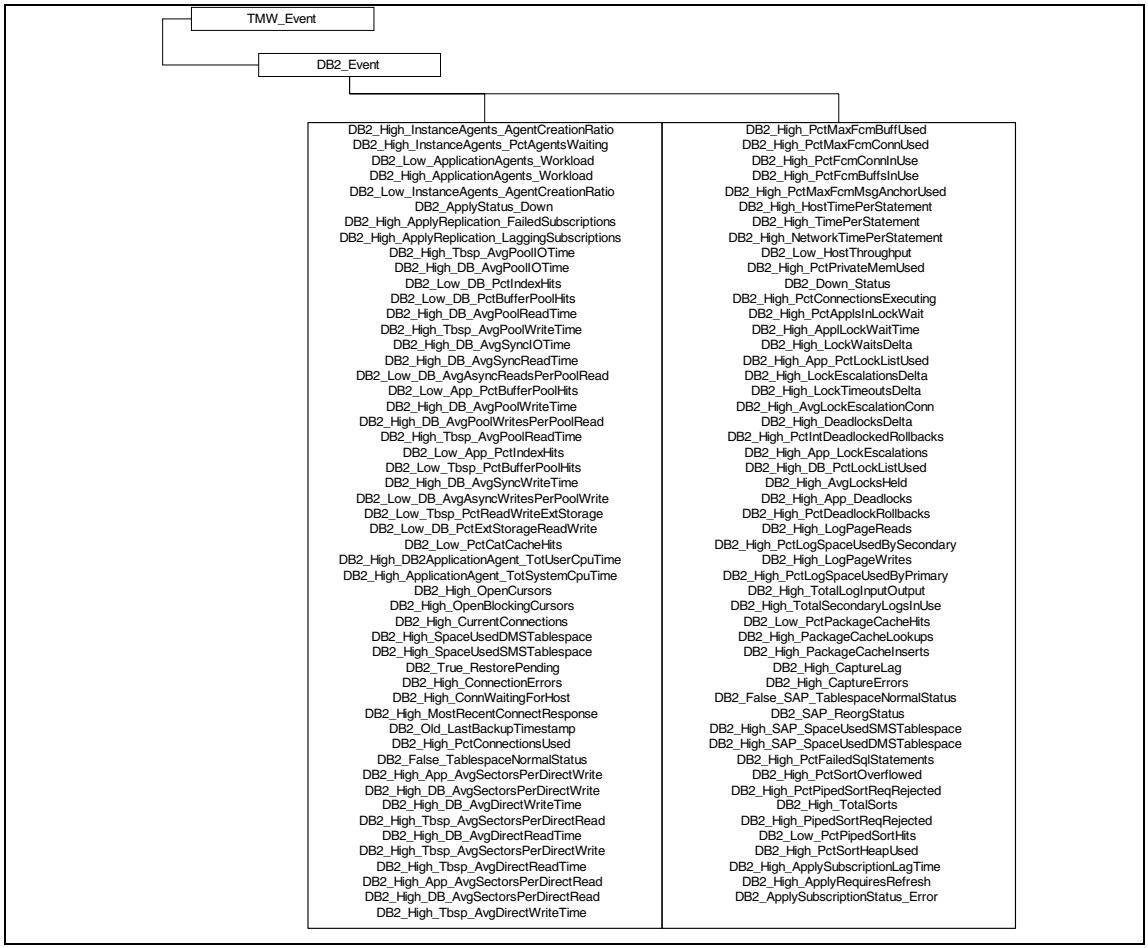


Figure 5-2 DB2 event hierarchy

- Oracle events: The hierarchy can be seen in Figure 5-3 on page 89.
  - ITMApplications.baroc: TMW_Event is a sub-class of the ITMSystem_Base and ITMDatabase_Base classes.

– ITMOracle.baroc: Sub-classes the ITMDatabase_Base into categories for events generated by the instance manager or database manager.

– ITMOracleResourceModels.baroc: Lists all events generated by the resource models; all are a sub-class of either InstanceManager or DatabaseManager events.

– ESMTask.baroc: Sub-class of the root EVENT

– OracleTask.baroc: Defines events whether an Oracle task is successful, failed, or abended.

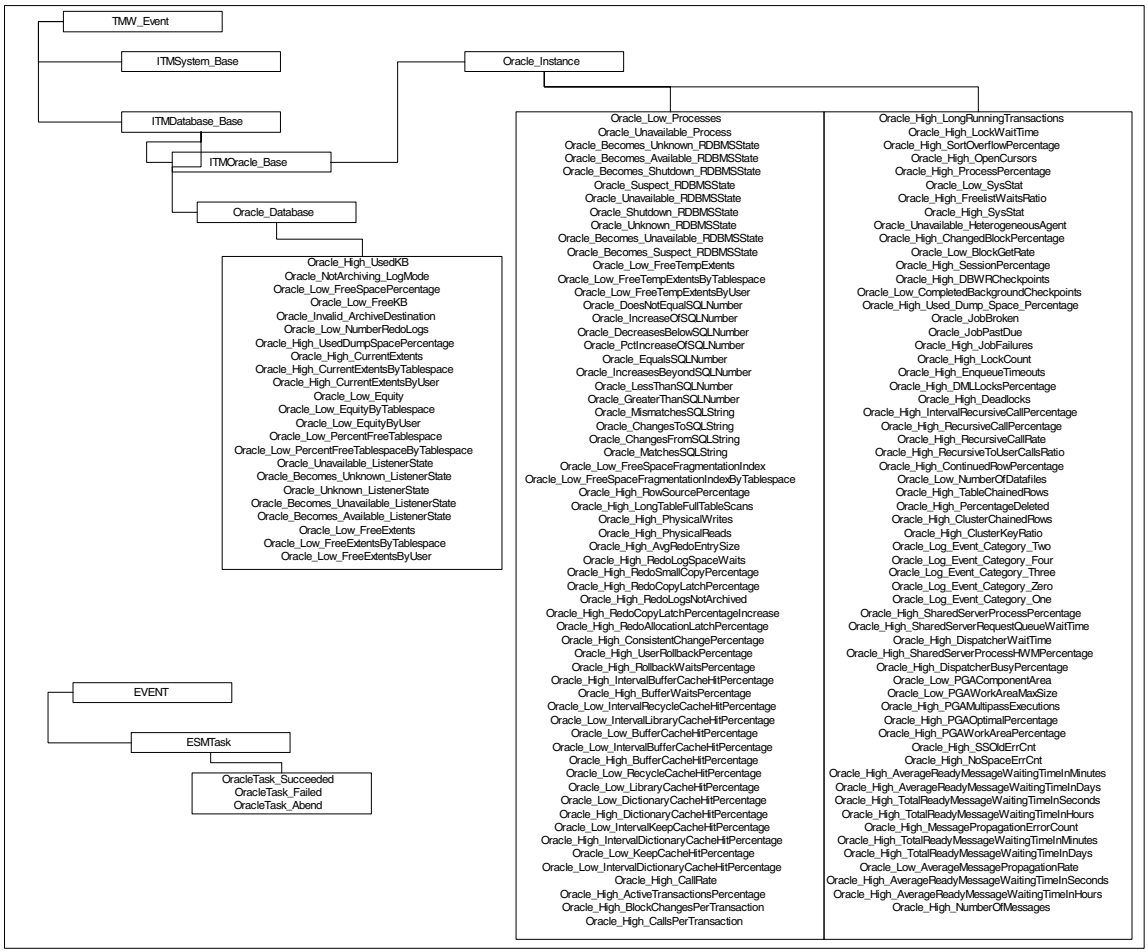


Figure 5-3 Oracle event hierarchy

- IBM Informix events: The hierarchy can be seen in Figure 5-4 on page 91.
  - IFX_Event.baroc: The primary baroc file that is a sub-class of TMW_Event.
  - Other baroc files are sub-classes of the IFX_Event and represent the event that can be generated by the resource models:
    - IBMInformixActiveTransactions.baroc
    - IBMInformixArchive.baroc
    - IBMInformixCacheHitRatio.baroc
    - IBMInformixCheckpoint.baroc
    - IBMInformixDbspace.baroc
    - IBMInformixDeadlocks.baroc
    - IBMInformixDmlLocksRatio.baroc
    - IBMInformixFilesystem.baroc
    - IBMInformixFreeDbspace.baroc
    - IBMInformixFreeSpaceDeficit.baroc
    - IBMInformixHDR.baroc
    - IBMInformixLRUQueues.baroc
    - IBMInformixLogEvent.baroc
    - IBMInformixLogicalLog.baroc
    - IBMInformixLogicalLogBackup.baroc
    - IBMInformixMemorySegment.baroc
    - IBMInformixOverflows.baroc
    - IBMInformixPhysicalLogUsageRatio.baroc
    - IBMInformixRollbackRatio.baroc
    - IBMInformixServerState.baroc
    - IBMInformixTableExtents.baroc
    - IBMInformixUpdateStatistics.baroc
    - IBMInformixVirtualProcessors.baroc
    - IBMInformixWaits.baroc
    - IBMInformixWrites.baroc

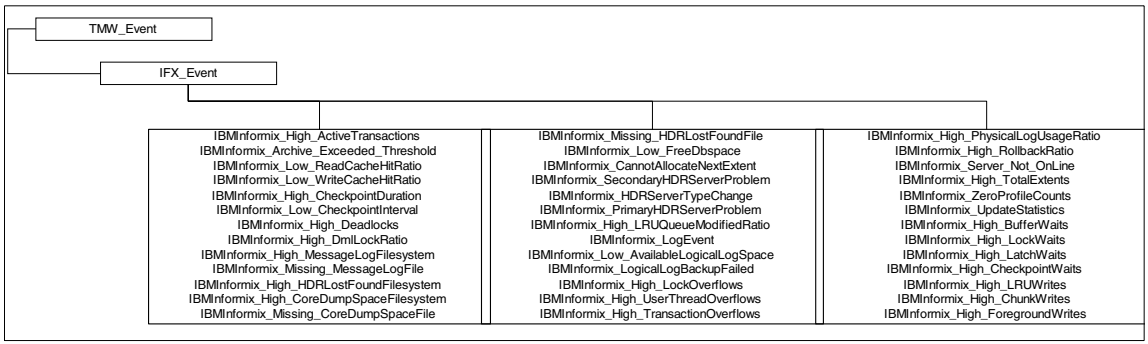


Figure 5-4 IBM Informix event hierarchy

For the IBM Tivoli Monitoring for Databases, only rulesets that send events to Tivoli Business Systems Manager are provided. These rulesets are called:

- DB2_to_TBSM.rls
- Oracle_to_TBSM.rls
- ibminformix_forward_tbsm.rls

5.4 Configuring Tivoli Enterprise Console rulebase

The installed product has already created a task for configuring the TEC server for the IBM Tivoli Monitoring for Databases events. It imports all the necessary class definitions and rulesets according to the options that you choose.

Attention: For IBM Tivoli Monitoring for Databases: Informix, the TEC configuration task must be preceded by the Send_TEC_Files_To_TEC task, as discussed in 2.3.3, “Configuring IBM Tivoli Monitoring for Databases: Informix” on page 34.

The configure event server tasks are shown in Table 5-1.

Table 5-1 Configuring event server task location

	DB2	Oracle	Informix
Policy region	Monitoring for DB2	Monitoring for Oracle	Monitoring for Informix
Task library	DB2ManagerAdminTasks	ITMOracle Tasks	IBM Informix Server Tasks

	DB2	Oracle	Informix
Task name	ECC_Configure_TEC_Classes	ConfigureTECOracle	Configure_TEC

You can run each of these tasks independently of each other. The order of executing these tasks are not important.

- For ECC_Configure_TEC_Classes, the option screen for this task is shown in Figure 5-5.

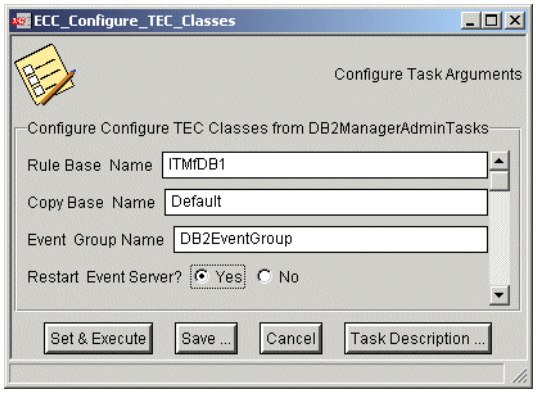


Figure 5-5 Configuring Event Server for DB2

- For Configure_TEC_Oracle, the option screen for this task is shown in Figure 5-6 on page 93.

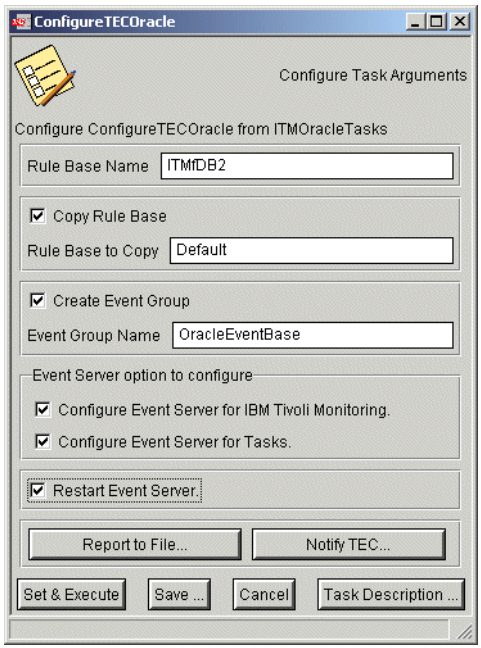


Figure 5-6 Configuring Event Server for Oracle

- For Configure_TEC (IBM Informix), the option screen for this task is shown in Figure 5-7.

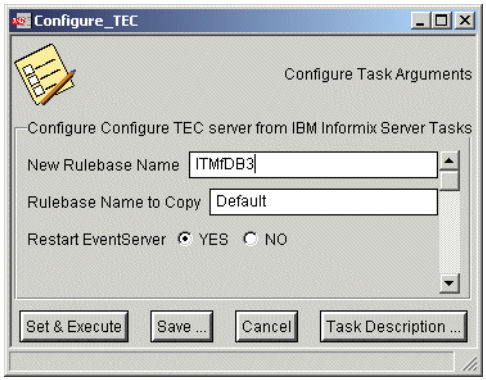


Figure 5-7 Configure event server for IBM Informix events

If you want the TEC rules that forward events to Tivoli Business Systems Manager to be active, you need to activate them. The files are stored in the rulebase, but are not currently activated for the current event server. We have a new rulebase called ITMfDB as the combined rulebase that contains the

customization from all three components. We need to activate the rules using the **wrb -imptgtrules** command. The results are shown in Example 5-1.

Example 5-1 Importing rulebase

```
# wrb -imptgtrule DB2_to_TBSM EventServer ITMfDB
# wrb -imptgtrule Oracle_to_TBSM EventServer ITMfDB
Adding 'DB2_to_TBSM'
# wrb -imptgtrule ibminformix_forward_tbsm EventServer ITMfDB
Adding 'DB2_to_TBSM'
Adding 'Oracle_to_TBSM'
# wrb -lsrbtarget EventServer -detailed ITMfDB
RB Targets
-------------------------------------
EventServer
        rule_set: 'DB2_to_TBSM'
        rule_set: 'Oracle_to_TBSM'
        rule_set: 'ibminformix_forward_tbsm'
```

5.5 Tivoli Business Systems Manager

Tivoli Business Systems Manager gives operation and business executives a graphical interface to quickly see and understand the health of the IT infrastructure they are using or managing. Tivoli Business Systems Manager shows business executives how individual components or resources affect a business function. Tivoli Business Systems Manager also shows operation personnel what business functions are affected by an outage of a single component. In Tivoli Business Systems Manager, the business function is represented by a Line of Business resources.

Tivoli Business Systems Manager collects information of resources' status from various parts of the enterprise. It gets feeds from the mainframe environment, job scheduling subsystem, Tivoli Framework, network management software, or other third party applications. It processes all events from those feeds and shows an integrated view of an enterprise.

Related to IBM Tivoli Monitoring for Databases, Tivoli Business Systems Manager can show the status of DB2, Oracle, and Informix resources as they relate to a business function.

IBM Tivoli Monitoring for Databases generates events through the resource models. These events go through TEC, and specialized TEC rules are employed to forward these events to Tivoli Business Systems Manager. Tivoli Business Systems Manager then process these events as they show the database resources status.

Important: The Tivoli Business Systems Manager system must be Version 1.5 with the following patches installed on the TBSM server:

- 1.5-BSM-0024
- 1.5-BSM-0029
- 1.5-BSM-0035

The Java console must be installed using the 1.5-BSM-0036 patch.

Tivoli Business Systems Manager Distributed Edition Version 1.5 must be installed on the TEC server with the following patches:

- 1.5-BSM-0032
- 1.5-BSM-0038

5.5.1 Tivoli Business Systems Manager event flow

A detailed event flow for Tivoli Business Systems Manager integration is shown in Figure 5-8.

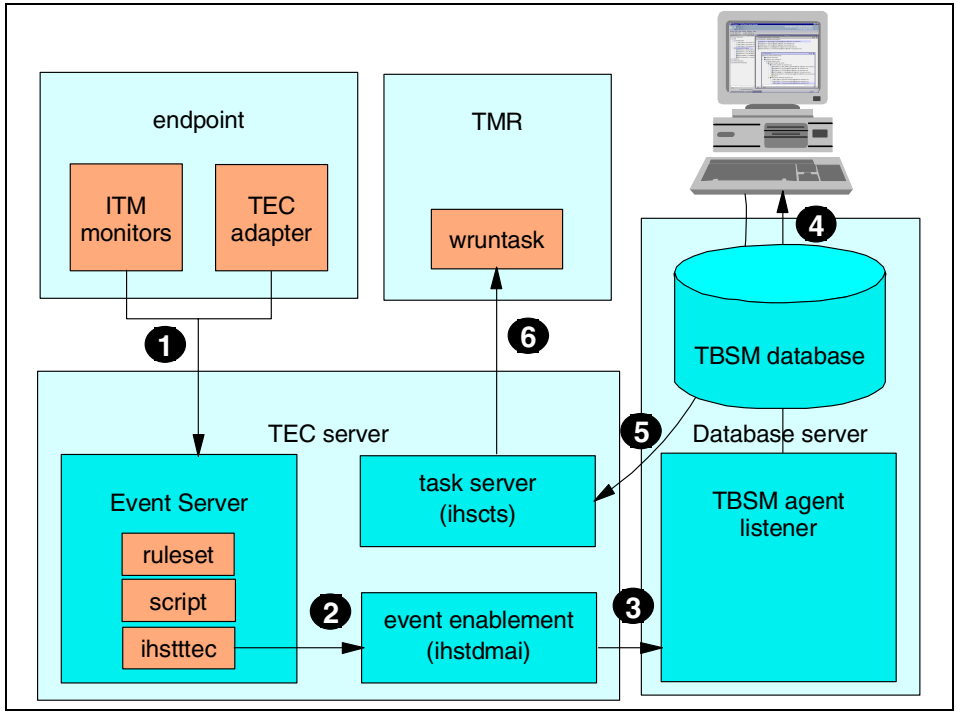


Figure 5-8 Event flow for Tivoli Business Systems Manager integration

As shown in Figure 5-8 on page 95, the following happened:

1. Events are generated by the ITM resource model or TEC adapter. The events are sent to TEC event server for processing.
2. The TEC event server compares the event against criteria on the rules that it has. Specific rulesets for Tivoli Business Systems Manager forwarding for IBM Tivoli Monitoring for Databases are called DB2_to_TBSM.rls, Oracle_to_TBSM.rls, and ibminformix_forward_tbsm.rls. These rulesets invoke forwarding scripts that reside in the $BINDIR/TME/TEC/scripts, which are called DB2_send_to_TBSM.sh and ibminformix_send_to_tbsm.sh, and for Oracle, they reside in $BINDIR/AMS/ORACLE/Oracle_send_to_TBSM.sh. All these scripts invoke ihstttec to forward the event information to the event enablement process.
3. The event enablement sends the formatted event to the agent listener, which resides in the Tivoli Business Systems Manager database server. The agent listener evaluates the event and stores the event in the Tivoli Business Systems Manager database.
4. The Tivoli Business Systems Manager console is informed of the event and the status of monitored objects are changed accordingly.
5. When an operator invokes an operational tasks by clicking the context menu of any IBM Tivoli Monitoring for Databases resources, a request is sent to the task server process.
6. The task server executes the operational task using the **wruntask** command.

5.6 Setting up TBSM integration

These are the things that you need to perform to set up the integration between IBM Tivoli Monitoring for Databases and Tivoli Business Systems Manager:

1. Define IBM Tivoli Monitoring for Databases resources to Tivoli Business Systems Manager. This includes defining the object types, icons, operational tasks, and other necessary attributes. Use the installation program in the IBM Tivoli Monitoring for Databases CD-ROM in the TBSM subdirectory. Run the install.bat on the Tivoli Business Systems Manager database server. The installation will copy the uninstallation program and start the initialization script called <component>_init_tbsm.sh, where component indicates the component to be installed.
2. Install Tivoli Business Systems Manager Distributed Edition to the TEC server and apply the necessary patches. See the Tivoli Business Systems Manager documentation to learn how to do this.

3. Modify the TEC event server rule base to include the forwarding ruleset. This may have been done when you configured the TEC integration. See 5.4, “Configuring Tivoli Enterprise Console rulebase” on page 91.
4. From the TBSM database server, configure the enterprise mapping for the TEC server host. The mapping resides in the GEM_EEHost_to_Enterprise table. Open the SQL Query Analyzer by selecting **Programs** -> **Microsoft SQL Server 7.0** -> **Query Analyzer**, and use the query shown in Figure 5-9.

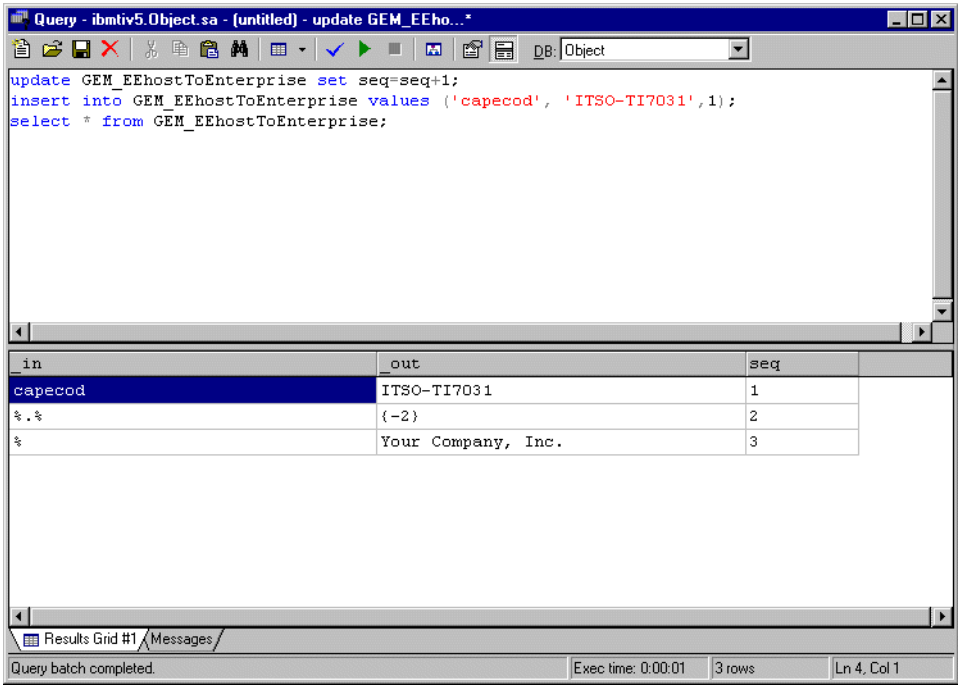


Figure 5-9 Setting the EEhost mapping

5. From the TBSM database server, connect the Agent Listener with the event enablement system, so that events from TEC start flowing into TBSM. Perform this using the **gemeeconfig -a *<eehost>*** command, as shown in Example 5-2. You need to restart the Agent Listener service afterward. You can then check the connection using the **gemeeconfig** command again.

Example 5-2 Running the gemeeconfig command

```
C:\>gemeeconfig -a capecod
Adding Event Enabler capecod
New Event Enablement definition created.

C:\>net stop ASIAgentListenerSvc
The Tivoli BSM Agent Listener service is stopping.
```

```
The Tivoli BSM Agent Listener service was stopped successfully.


C:\>net start ASIAgentListenerSvc
The Tivoli BSM Agent Listener service is starting.
The Tivoli BSM Agent Listener service was started successfully.


C:\>gemeeconfig


Listing configured Event Enablers:

Event Enabler: capecod
        Connection Status: Connected
        Enabled for connection at startup.
        Port: Default Port
        RetryTime: Default
        MaxWaitTime: Default
Done.
```

6. Run the discovery tasks for IBM Tivoli Monitoring for Databases resources. The task will perform a set of **wlookup** commands to get the currently managed resources and send the appropriate DISCOVER events to TEC. The discovery tasks is shown in Table 5-1 on page 91.

Table 5-2 Configuring event server task location

	DB2	Oracle	Informix
Policy region	Monitoring for DB2	Monitoring for Oracle	Monitoring for Informix
Task library	DB2ManagerAdminTasks	ITMOracle Tasks	IBM Informix Server Tasks
Task name	ECC_TBSM_Discovery	OracleTBSMDiscovery	TBSM_Discovery

You can run each of these tasks independently of each other. The order of executing these tasks are not important.

5.7 Using Tivoli Business Systems Manager

Figure 5-10 on page 99 shows a sample Tivoli Business Systems Manager console when some of the IBM Tivoli Monitoring for Databases resources has been discovered.

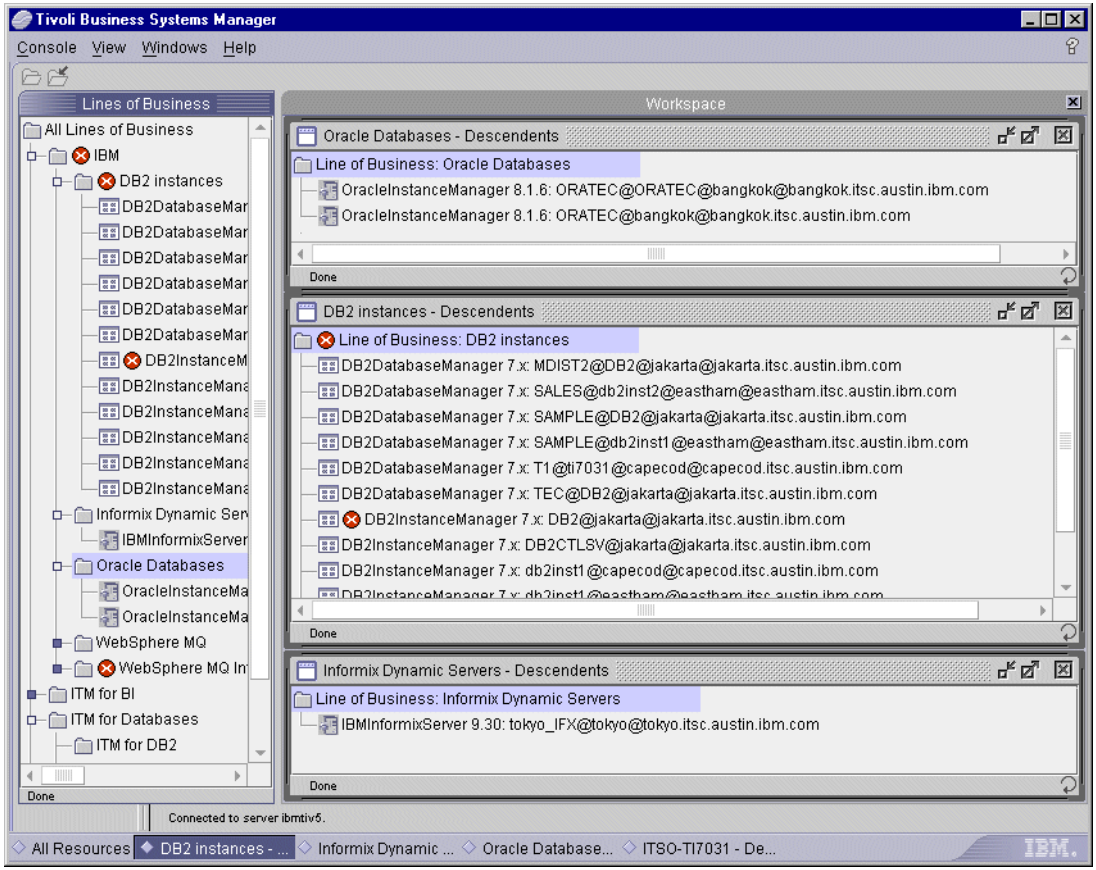


Figure 5-10 TBSM console with IBM Tivoli Monitoring for Databases resources

In Figure 5-10, the Line of Business that is created by default from the installation is called IBM and, by product, it groups all resources from that product, which are DB2, Oracle, and Informix. The objects are discovered under the network region based on their fully qualified endpoint host name, as shown in the All Resources view in Figure 5-11 on page 100.

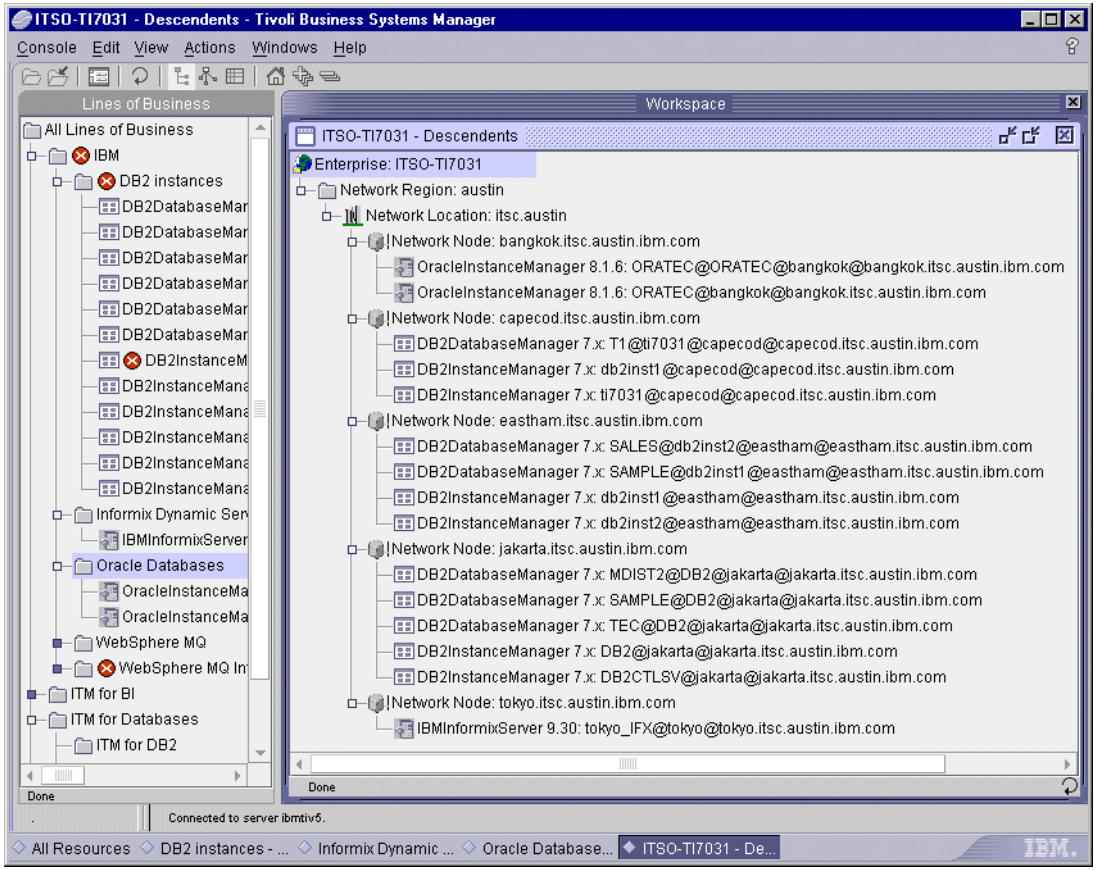


Figure 5-11 Physical view of the enterprise

For more information on Tivoli Business Systems Manager and its usage, refer to the following Redbooks:

- *Tivoli Business Systems Manager A Complete End-to-End Management Solution*, SG24-6202
- *Tivoli Business Systems Manager An Implementation Case Study*, SG24-6032

6

Historical reporting

The Tivoli Enterprise Data Warehouse (TEDW) is an infrastructure used to collect and manage data from various Tivoli and non-Tivoli system management applications. The data is imported into the Tivoli Enterprise Data Warehouse databases through specialized programs called extract, transform, and load (ETL) programs, from the management application data sources, and further processed for historical analysis and evaluation. It is Tivoli's strategy to have most of its products providing ETLs so that the Tivoli Enterprise Data Warehouse databases can be populated with meaningful systems management data. The IBM Tivoli Monitoring for Databases is one of the many products to fully leverage and utilize Tivoli Enterprise Data Warehouse.

This chapter goes into detail on how IBM Tivoli Monitoring for Databases leverages the Tivoli Enterprise Data Warehouse, how to achieve integration, and how to obtain the Tivoli Enterprise Data Warehouse reports. We cover:

6.1 Tivoli Enterprise Data Warehouse overview

Customers can benefit from using Tivoli Enterprise Data Warehouse in various ways, such as:

- Tivoli Enterprise Data Warehouse collects historical data from many applications into one central place.

  Tivoli Enterprise Data Warehouse collects the underlying data about customers' network devices/connections, desktops/servers, applications/software, and the problems and activities that have gone on to manage the infrastructure. This allows the customers to construct an end-to-end view of their enterprise and view the components independent of specific applications used to monitor and control resources.

- Tivoli Enterprise Data Warehouse adds value to raw data.

  Tivoli Enterprise Data Warehouse performs data aggregation in hourly levels in the central data repository and can be restricted to other levels, such as daily and weekly, for data reporting in the data marts. The data is also cleaned and consolidated in order to allow the data model of the central repository to share common dimensions. For example, Tivoli Enterprise Data Warehouse ensures that time, host name, and IP address are the same dimensions across all the applications.

- Tivoli Enterprise Data Warehouse allows the correlation of information from many Tivoli applications.

  Tivoli Enterprise Data Warehouse can also be used to derive added value by correlating data from many Tivoli applications. It allows reports to be written, which correlate cross application data.

- Tivoli Enterprise Data Warehouse uses open, proven interfaces for extracting, storing, and sharing the data.

  Tivoli Enterprise Data Warehouse can extract data from any application (Tivoli and non-Tivoli) and store it in a common central database. Tivoli Enterprise Data Warehouse also provides transparent access for third-party Business Intelligence (BI) solutions (CWM standard), such as IBM DB2 OLAP, Crystal Decisions, Cognos, BusinessObjects, Brio Technology, and Microsoft OLAP Server. CWM stands for Common Warehouse Metadata, an industry standard specification for metadata interchange defined by the Object Management Group (see `http://www.omg.org`). Tivoli Enterprise Data Warehouse provides a Web-based reporting front end, called the Reporting Interface, but the open architecture provided by the Tivoli Enterprise Data Warehouse allows other BI front ends to be used to access the data in the central warehouse. The value here is *flexibility*. Customers can use the reporting application of their choice; they are not limited to any one.

- Tivoli Enterprise Data Warehouse provides a robust security mechanism.

  Tivoli Enterprise Data Warehouse provides a robust security mechanism by allowing data marts to be built with data from subsets of managed resources. By providing database level authorization to access those data marts, Tivoli Enterprise Data Warehouse can address most of the security requirements related to limiting access to specific data to those customers/business units with a need to know.

- Tivoli Enterprise Data Warehouse provides a scalable architecture.

  Since Tivoli Enterprise Data Warehouse depends on the proven and industry standard RDBMS technology, it provides a scalable architecture for storing and retrieving the data.

6.1.1 Tivoli Enterprise Data Warehouse concepts and components

In this section, we describe the key concepts and the various components of Tivoli Enterprise Data Warehouse in the logical order that the measurement data flows: from the monitors collecting raw data to the final detailed report.

Figure 6-1 depicts a typical Tivoli Enterprise Data Warehouse configuration that can be used as an illustration.

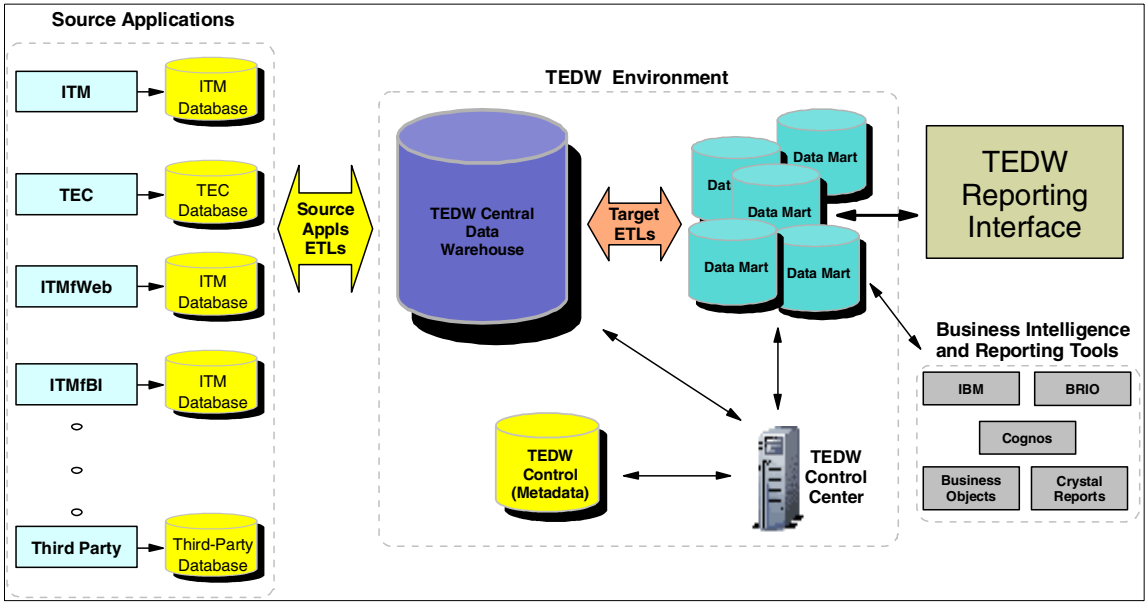


Figure 6-1 A typical Tivoli Enterprise Data Warehouse environment

It is common for enterprises to have various distributed performance and availability monitoring applications deployed that collect some sort of measurement data and provide some type of threshold management, central event management, and other basic monitoring functions. These applications are referred to as source applications.

The first step in obtaining management data is to enable the source applications. This means providing all tools and customizations necessary to import the

that makes the data accessible to many analysis solutions. The database is organized in a very flexible way, which lets you store data from new applications without adding or changing tables.

The Tivoli Enterprise Data Warehouse server is an IBM DB2 Universal Database Enterprise Edition server that hosts the Tivoli Enterprise Data Warehouse Central Data Warehouse databases. These databases are populated with operational data from Tivoli and/or other third-party applications for historical analyses.

A data mart is a subset of the historical data that satisfies the needs of a specific department, team, or customer. A data mart is optimized for interactive reporting and data analysis. The format of a data mart is specific to the reporting or analysis tool you plan to use. Each application that provides a data mart ETL creates its data marts in the appropriate format.

Tivoli Enterprise Data Warehouse provides a Report Interface (RI) that creates static two-dimensional reports of your data using the data marts. The Report Interface is a role-based Web interface that can be accessed with a simple Web browser without any additional software installed on the client. You can also use other tools to perform OLAP analysis, business intelligence reporting, or data mining.

The Tivoli Enterprise Data Warehouse control center is the IBM DB2 Universal Database Enterprise Edition server containing the Tivoli Enterprise Data Warehouse control database that manages your Tivoli Enterprise Data Warehouse environment. From the Tivoli Enterprise Data Warehouse control center, you can see all source applications databases in your environment. The default internal name for the Tivoli Enterprise Data Warehouse Control database is TWH_MD. The Tivoli Enterprise Data Warehouse Control Center also manages the communication between the various components, such as the Tivoli Enterprise Data Warehouse Central Data Warehouse, the data marts, and the Report Interfaces. The Tivoli Enterprise Data Warehouse Control Center uses the DB2 Data Warehouse Center utility to define, maintain, schedule, and monitor the ETL processes

The Tivoli Enterprise Data Warehouse stores cleansed historical data from all Tivoli and third-party application databases in the Tivoli Enterprise Data Warehouse Central Data Warehouse database. The database name of the Tivoli Enterprise Data Warehouse Central Data Warehouse database is TWH_CDW. Once the data has been inserted into the TWH_CDW database, it is available for either the Tivoli Enterprise Data Warehouse ETLs to load to the Tivoli Enterprise Data Warehouse Data Mart database (the database name of the Tivoli Enterprise Data Warehouse Data Mart database is TWH_MART) or to any other application-specific ETL to process the data and load the application-specific Data Mart database.

All Tivoli Enterprise Data Warehouse ETL programs follow a naming convention using a three letter application-specific product code known as *measurement source code*. Some examples of these measurement source codes can be found in the later sections.

6.2 IBM Tivoli Monitoring for Databases integration components

The structure of the IBM Tivoli Monitoring for Databases reporting component can be shown in Figure 6-2.

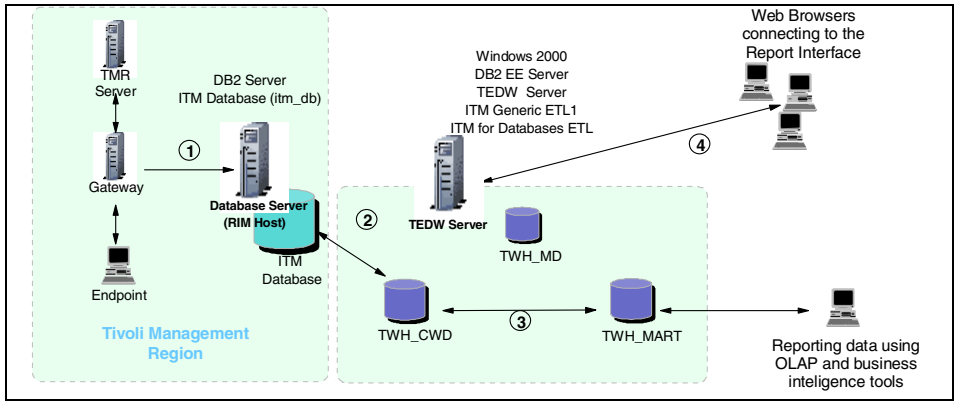


Figure 6-2 IBM Tivoli Monitoring for Databases warehouse component

From Figure 6-2, the warehouse processing goes through the following steps:

1. The resource models for the IBM Tivoli Monitoring for Databases generates data into the IBM Tivoli Monitoring endpoint databases, which is uploaded into the IBM Tivoli Monitoring historical database. The IBM Tivoli Monitoring historical database is a RIM object where the Tivoli Gateway writes the collected measurement for each endpoints that reports to it.
2. Using the IBM Tivoli Monitoring 5.1.1 source ETL, data from the RIM database is loaded to the central data warehouse. The source ETL from IBM Tivoli Monitoring has a measurement code of AMX.
3. From this central data warehouse, each module from the IBM Tivoli Monitoring for Databases can extract the data related to them and store them in the data marts. The target ETLs are provided by each warehouse pack for DB2, Oracle, and IBM Informix. The measurement codes are: CTD for DB2, CTO for Oracle, and CTR for IBM Informix.
4. Data in the data marts can be shown using the predefined reports using the reporting interface through the IBM console application.

Now, we can start installing the components.

6.3 Installation and configuration for data warehouse

In this section, we describe the necessary steps to configure the data gathering process in your environment. The high level steps are as follows:

1. Have the environment with the IBM Tivoli Monitoring for Databases installed and working properly.
2. Install and configure IBM Tivoli Monitoring - Tivoli Enterprise Data Warehouse Support, Version 5.1.1,;see 6.3.1, “Installing and configuring data warehouse support” on page 108.
3. Have the Tivoli Enterprise Data Warehouse server or servers installed properly with the necessary patches; see 6.3.2, “Warehouse integration pre-installation steps” on page 111.
4. Install and configure the IBM Tivoli Monitoring version 5.1.1 Warehouse component; see 6.3.3, “Setting up the source ETL” on page 112.
5. Install and configure the IBM Tivoli Monitoring for Databases Warehouse component; see 6.3.4, “Setting up the target ETLs” on page 127.
6. Define and deploy IBM Tivoli Monitoring for Databases resource models that log data into the historical database and run the ETL processes, both for the source ETL or the target ETL. This step is discussed in 6.4, “Activating collection” on page 135.
7. Display and view the reports using the IBM console; see 6.5, “IBM Tivoli Monitoring for Databases reports” on page 139.

Before going into the details of each step, we present the environment used in the ITSO lab. This can be used as a start point for setting up the data gathering process. We assume no preexisting components will be used and describe the steps of a brand new installation. We install most of the data warehouse components in a single machine; however, for a production environment, you may want to have the following separate machines:

- Tivoli Enterprise Data Warehouse Server machine hosting the Central Warehouse and the Warehouse data mart databases; this needs the largest disk capacity and the fastest processor.
- Tivoli Enterprise Data Warehouse Control Center machine hosting the Warehouse metadata database and handling all the ETLs scheduling.
- Tivoli Enterprise Data Warehouse Reporting Interface machine allowing end users to obtain reports from data stored in the IBM Tivoli Monitoring data marts.

6.3.1 Installing and configuring data warehouse support

In this section, we describe the steps to install and configure the IBM Tivoli Monitoring - Tivoli Enterprise Data Warehouse Support, Version 5.1.1. This component should be installed on the TMR server and all gateways in your Tivoli environment where you want to collect data from endpoints and make the data available for Tivoli Enterprise Data Warehouse. As a requirement, the IBM Tivoli Monitoring 5.1.1 product should be already installed. The installation method described here uses the Tivoli Desktop, as follows:

1. From the Tivoli desktop, select **Desktop** -> **Install** -> **Install Product**. The install product dialog shows the products that are available to install, as shown in Figure 6-3.

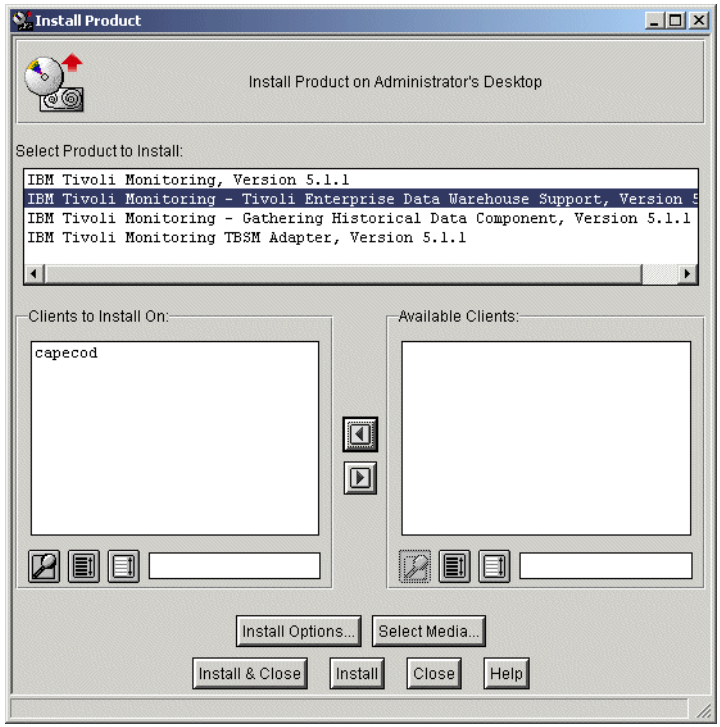


Figure 6-3 Installing warehouse support

2. Select **IBM Tivoli Monitoring - Tivoli Enterprise Data Warehouse Support, Version 5.1.1**, then select your Tivoli Management Region server and the gateways that you want to have it installed on.
3. RIM configuration is required to proceed the installation, as shown in Figure 6-4 on page 109.

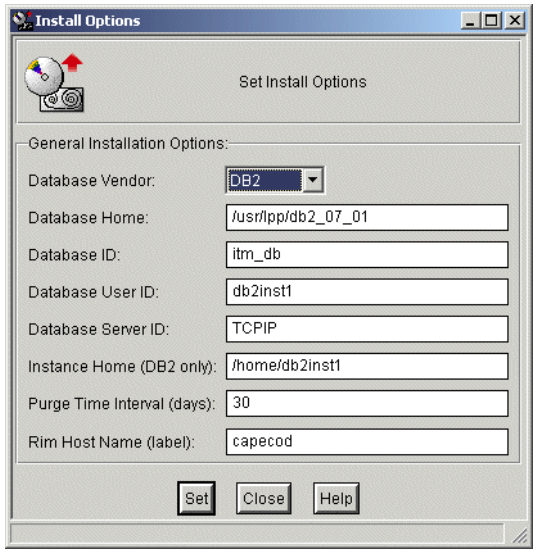


Figure 6-4 RIM setup options

The installation process will create a RIM object named itm_rim_<nodename>, where <nodename> is the RIM host of your Tivoli environment. The RIM object can also be created a later time using, for instance, the following command, assuming a DB2 database server:

```
wcrtrim -v DB2 -h emlab3 -d itm_db -u db2 -H c:/db2/sqllib -s TCPIP -I
c:/db2 itm_rim_emlab3
```

This RIM object has, by default, `itmitm` as the password, which must be changed to match the password of your database instance owner. Use the **`wsetrimpw`** command as follows:

```
wsetrimpw itm_rim_<nodename> itmitm <newpw>
```

where <newpw> is the database instance owner password.

4. Click **Set** and select **Install** and follow the normal installation dialogs.
5. The physical database for the Warehouse Support component name is ITM_DB and needs now to be created. This process can be accomplished by either using a provided shell script or using SQL scripts. If you intend to use the provided shell script, make sure you grant the RDBMS administrator (or database instance owner) user ID with Administrator (root) and Tivoli_Admin_Privileges and run the script logged in as your user ID. The reason for that is the shell script collects information from the previously created RIM object in order to create both the database and its structure. The shell script name is cr_itm_db.sh and it is located in the $BINDIR/TME/Tmw2k/Warehousecfg directory.

As an alternative method, you can use the SQL scripts. These scripts are also located in the $BINDIR/TME/Tmw2k/Warehousecfg directory and have the following naming standard:

– cr_bd.<DBext>

– cr_tbl.<DBext>

where <DBext> is the database vendor designator.

The follow describes the creation process for DB2 using the SQL scripts:

a. On the RIM Host machine, log in as your instance owner, in our case, db2inst1.

b. Only perform this step if the RIM Host machine does not have the Warehouse Support component installed. Copy the cr_db.db2 and cr_tbl.db2 files from the $BINDIR/TME/Tmw2k/Warehousecfg directory from your TMR Server to the RIM Host machine.

c. Move to the directory where the SQL scripts are located and rename cr_db.db2 to cr_db_db2.sql and rename cr_tbl.db2 to cr_tbl_db2.sql.

d. Run the following command to create the itm_db database:

```
db2 -td$ -vf cr_db_db2.sql
```

e. In order to have the itm_db database structure created, run the following commands, where <db2inst1pw> is the database instance owner password:

```
db2 connect to itm_db user db2inst1 using <db2inst1pw>
db2 -td$ -vf cr_tbl_db2.sql
```

6. On the TMR server, test the RIM object connection:

```
wrimtest -l itm_rim_<rimhost>
```

The output should be similar to the following:

Example 6-1 Testing the RIM object

```
c:\>wrimtest -l itm_rim_emlab3
Resource Type  : RIM
Resource Label : itm_rim_emlab3
Hostname       : emlab3
User Name      : db2
Vendor         : DB2
Database       : itm_db
Database Home  : c:/db2/sqllib
Server ID      : TCPIP
Instance Home  : c:/db2
Opening Regular Session...Session Opened
RIM : Enter Option >
```

7. The data collection process of the Warehouse Support component needs to be configured. The configuration file is named .config and it is located in the $DBDIR/dmml directory. The Warehouse Support entries in the.config file have the prefix datacollector. Such entries should be added/modified using the **wdmconfig** command and it is important to notice that this file must not be modified manually. For details on the **wdmconfig** command, refer to the *IBM Tivoli Monitoring User's Guide Version 5.1*, SH19-4569. Issue the following command:

```
wdmconfig -m <nodename> -D datacollector.rim_name=itm_rim_<rimhost> \
-D datacollector.db_purge_interval=30 \
-D datacollector.db_purge_time=0 \
-D datacollector.delay=30 \
-D datacollector.sleep_time=1 \
-D datacollector.max_retry_time=6
```

You can check if the entries were correctly set by issuing:

```
wdmconfig -m <nodename> -G datacollector*
```

The output should be similar to Example 6-2.

Example 6-2 Datacollector configuration

```
# wdmconfig -m capecod -G datacollector*
Managed Node: 'capecod':
========================
        datacollector.delay = 30
        datacollector.sleep_time = 1
        datacollector.max_retry_time = 6
        datacollector.rim_name = itm_rim_capecod
        datacollector.db_purge_time = 0
        datacollector.db_purge_interval = 30
```

6.3.2 Warehouse integration pre-installation steps

In this section, it is assumed that your Tivoli Enterprise Data Warehouse Environment Version 1.1 is already installed and up and running. Details for that can be found in the redbook *Introduction to Tivoli Enterprise Data Warehouse*, SG24-6607.

Prior to the installation of the Warehouse packs programs, it is required to perform the following tasks:

- Upgrade IBM DB2 Universal Database Enterprise Edition Version 7.2 to at least FixPack 6 level on your Tivoli Enterprise Data Warehouse environment

FixPack6 for IBM DB2 Universal Database Enterprise Edition can be downloaded from the official IBM DB2 technical support web site at:

http://www-3.ibm.com/cgi-bin/db2www/data/db2/udb/winos2unix/support/v7fphist.d2w/report

- Apply the following fixes for the Tivoli Enterprise Data Warehouse:
  - 1.1-TDW-0002
  - 1.1-TDW-0005E
  - 1.1-TDW-FP01a

  These fixes for Tivoli Enterprise Data Warehouse can be downloaded from the IBM Tivoli Software support Web site, under the Tivoli Enterprise Data Warehouse category at:

  http://www.ibm.com/software/sysmgmt/products/support/

The documentation that accompanies the FixPacks details the steps for installation in great details.

Important: The redbook *Introduction to Tivoli Enterprise Data Warehouse,* SG24-6607, mentions that the Windows Services “Warehouse Server” and “Warehouse Logger” must be reconfigured to run as the db2admin user. If you have not made this change, you will see failures when trying import data into the TWH_CDW database. Please confirm you have reconfigured these services and restarted them.

6.3.3 Setting up the source ETL

This section describes the installation procedures needed to install and configure the source ETL, which is provided from IBM Tivoli Monitoring Version 5.1.1. IBM Tivoli Monitoring Version 5.1.1 warehouse ETL extracts data from the IBM Tivoli Monitoring for Databases (ITM_DB*)* database and loads it into the Tivoli Enterprise Data Warehouse Central Data Warehouse database (TWH_CDW), therefore acting as a source ETL. The IBM Tivoli Monitoring Version 5.1.1 warehouse ETL is often referred as IBM Tivoli Monitoring Generic ETL1. The measurement source code for the IBM Tivoli Monitoring Generic ETL1 is AMX.

The installation can be done using the Tivoli Enterprise Data Warehouse CLI or the GUI installation program. Here we describe the process using the GUI method. The following steps should be performed in the Tivoli Enterprise Data Warehouse Control Center server.

You need both the Tivoli Enterprise Data Warehouse and the IBM Tivoli Monitoring Version 5.1.1 products installation media.

1. Run the setup.exe from the Tivoli Enterprise Data Warehouse CD-ROM and click **OK** to start the installation.
2. When the InstallShield Wizard dialogue window for Tivoli Enterprise Data Warehouse Installation appears, click **Next**.
3. The dialogue window for the type of installation appears. Select **Application installation only** and the directory name where the Tivoli Enterprise Data Warehouse components are installed. We used C:\TWH. Click **Next** to continue.
4. The host name dialogue window appears. Verify that this is the correct host name for the Tivoli Enterprise Data Warehouse Control Center server. Click **Next**.
5. The local system DB2 configuration dialogue window appears. The installation process asks for a valid DB2 user ID. Enter the valid DB2 user ID and password that were created during the DB2 installation on your local system. In our case, we used db2admin. Click **Next**.
6. The path to the installation media for the application packages dialogue window appears, as shown in Figure 6-5 on page 114. You should provide the location of the IBM Tivoli Monitoring Version 5.1.1 Generic ETL1 program. Change out the Tivoli Enterprise Data Warehouse CD in the CD-ROM drive with the IBM Tivoli Monitoring Version 5.1.1 installation CD. Specify the path to the twh_app_install_list.cfg file (by default, in the *CD_drive*:\tedw_apps\amx\ directory) on the IBM Tivoli Monitoring Version 5.1.1 CD in the directory name field. Leave the **Now (prevents typing errors)** option checked to verify that the source directory is immediately accessible and that it contains the correct files. Click **Next**.

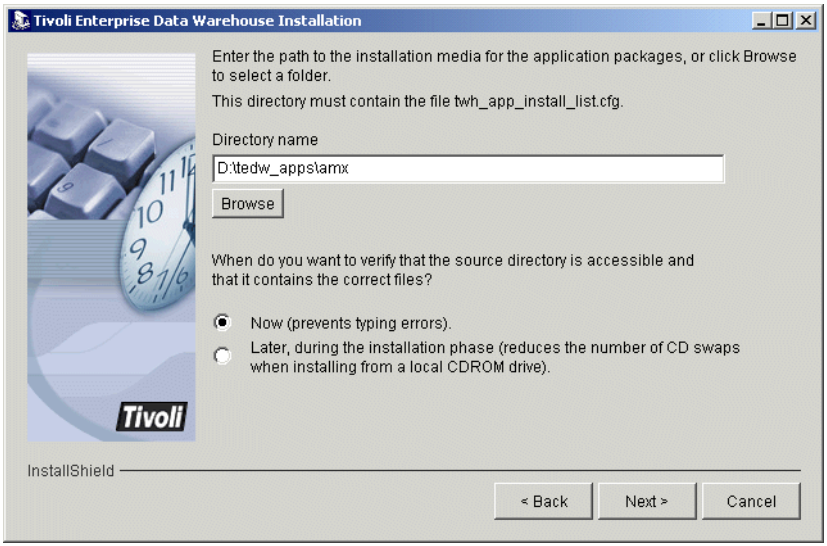


Figure 6-5 Path to the installation media for the AMX

7. The overview of selected features dialogue window appears, as shown in Figure 6-6. Click **Install** to start the installation.

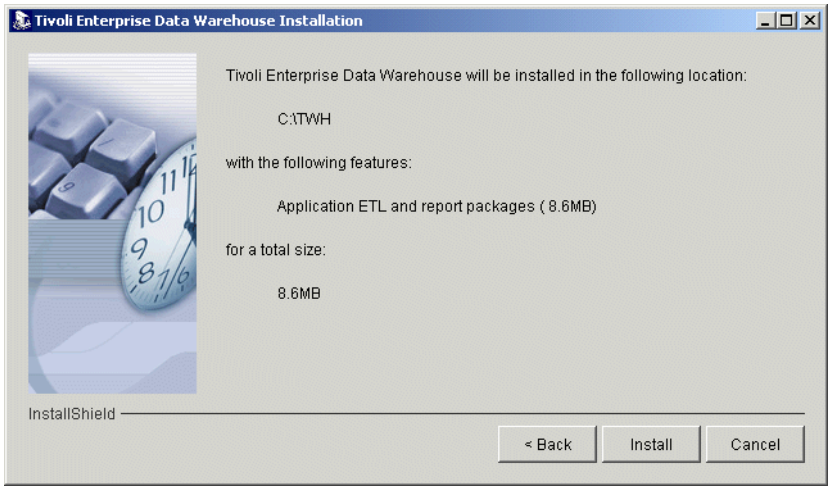


Figure 6-6 IBM Tivoli Monitoring Generic ETL1 program installation

8. Once the installation is finished, the Installation summary window appears, as shown in Figure 6-7 on page 115. If the installation was not successful, check the TWHApp.log file for any errors. This log file is located in

<TWH_inst_dir>\apps\AMX\, where <TWH_inst_dir> is the Tivoli Enterprise Data Warehouse installation directory.

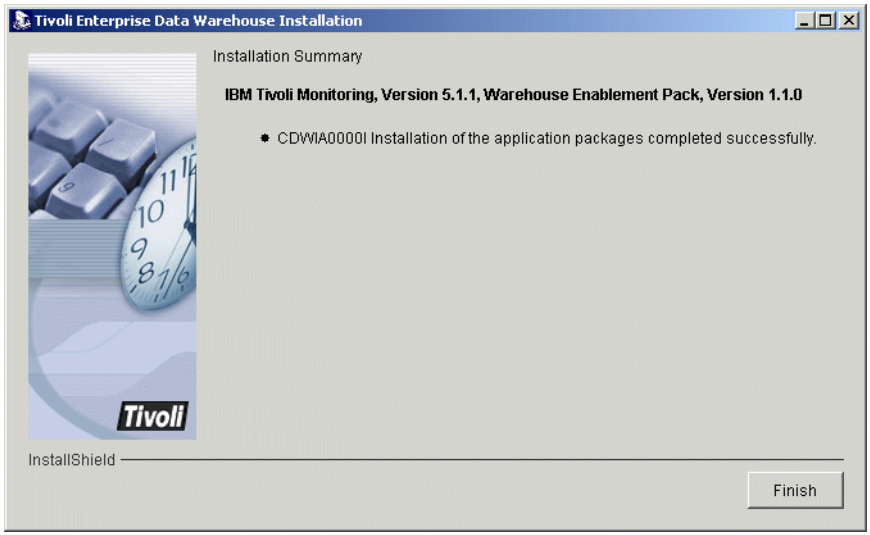


Figure 6-7 Installation summary window - IBM Tivoli Monitoring Generic ETL1

9. After the pack installation conclusion, you must reboot the Tivoli Enterprise Data Warehouse Control Center server.

There is a need to change some configuration settings for the IBM Tivoli Monitoring Version 5.1.1 Warehouse Pack to function properly. They are:

- Change the control heap size in the TWH_CDW database
- Create a new ODBC connection to the ITM_DB database
- Define user authority to the Warehouse Sources and Targets
- Schedule the ETL
- Change the ETL status to Production

Changes on the TWH_CDW database

The applications control heap size on the TWH_CDW database needs to be set to at least 512, as follows:

1. Logged in as your DB2 administrator user ID on your Tivoli Enterprise Data Warehouse Server machine, in our case db2admin, connect to the TWH_CDW database:

   ```
   db2 connect to TWH_CDW user db2admin using <db2pw>
   ```

   where <db2pw> is the database administrator password.

2. In order to determine the actual heap size, issue:

```
db2 get db cfg for TWH_CDW | grep CTL_HEAP
```

   The output should be similar to this:

```
Max appl. control heap size (4KB)     (APP_CTL_HEAP_SZ) = 128
```

3. If the heap size is less that 512, change it by performing:

```
db2 update db cfg for TWH_CDW using APP_CTL_HEAP_SZ 512
```

   The output should be similar to this:

```
DB20000I  The UPDATE DATABASE CONFIGURATION command completed successfully.
DB21026I  For most configuration parameters, all applications must
disconnect
from this database before the changes become effective.
```

4. You should now restart DB2 by issuing:

```
db2 disconnect THW_CDW
db2 force application all
db2 terminate
db2stop
db2admin stop
db2admin start
db2start
```

Creating an ODBC connection to the ITM_DB database

The Tivoli Enterprise Data Warehouse Control Center server hosts all the ETL, and it needs to have access to the various databases that the SQL scripts deal with. In the case of the IBM Tivoli Monitoring Version 5.1.1 Generic ETL1, a connection to the ITM_DB database should be defined. On the Tivoli Enterprise Data Warehouse Control Center server, using a DB2 command line window, issue the following commands:

```
db2 catalog tcpip node CAT_ODBC remote <TWH_SVR> server <DB2_PORT>
db2 catalog database ITM_DB at node CAT_ODBC
db2 catalog system odbc data source ITM_DB
```

Where <TWH_SVR> is the Tivoli Enterprise Data Warehouse Server host name, and <DB2_PORT> is the TCP/IP port used by DB2 (the default is 50000).

Tip: You can also perform this task using the DB2 Client Configuration Assistant.

Defining user authority to the Warehouse Sources and Targets

You should inform the Tivoli Enterprise Data Warehouse Control Center server of user access information for every Source and Target ETL process installed by the IBM Tivoli Monitoring Version 5.1.1 Generic ETL1. The following steps should be taken:

1. Start the IBM DB2 Control Center utility by selecting **Start -> Programs -> IBM DB2 -> Control Center**.
2. On the IBM DB2 Control Center utility, start the IBM DB2 Data Warehouse Center utility by selecting **Tools -> Data Warehouse Center**. The Data Warehouse Center logon windows appears.
3. Log into the IBM DB2 Data Warehouse Center utility using the local DB2 administrator user ID, in our case, db2admin.
4. In the Data Warehouse Center window, expand the Warehouse Sources folder. As shown in Figure 6-8, there are two entries for the IBM Tivoli Monitoring Version 5.1.1 Generic ETL1 programs that need to be configured, as follows:
   - AMX_ITM_RIM_Source
   - AMX_TWH_CDW_Source

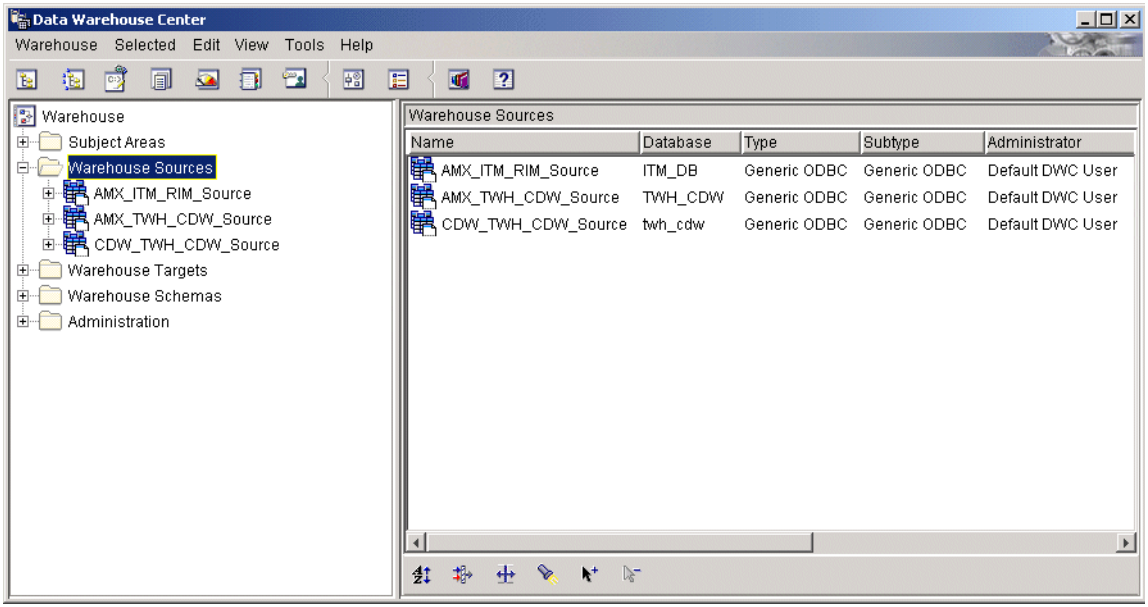


Figure 6-8 IBM Tivoli Monitoring Version 5.1.1 Generic ETL1 Sources

You should edit the properties of each one of the above entries. In order to do that, right-click it and select **Properties** and then select the **Database** tab. Fill

in the database instance owner user ID information. For our environment, the values are shown in Figure 6-9, using the AMX_ITM_RIM_Source as an example.

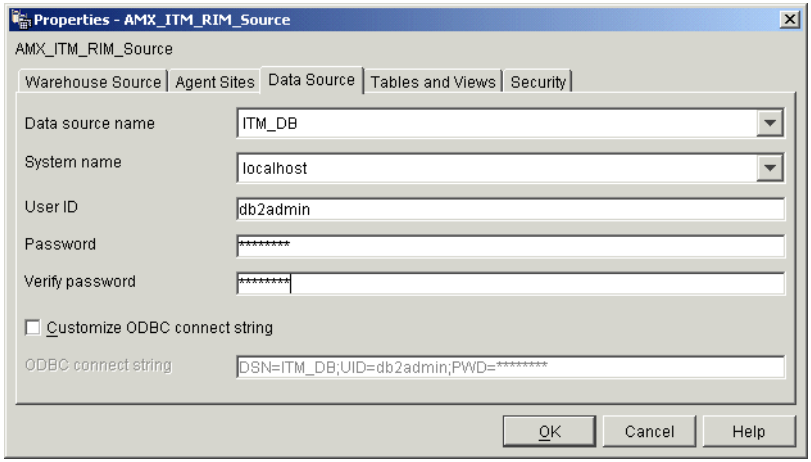


Figure 6-9 AMX_ITM_RIM_Source user ID information

5. For the IBM Tivoli Monitoring Version 5.1.1 Generic ETL1 Targets, shown in Figure 6-10 on page 119, expand the Warehouse Target folder, right-click the AMX_TWH_CDW_Target, select **Properties**, and then select the **Database** tab. Fill in the user ID information.

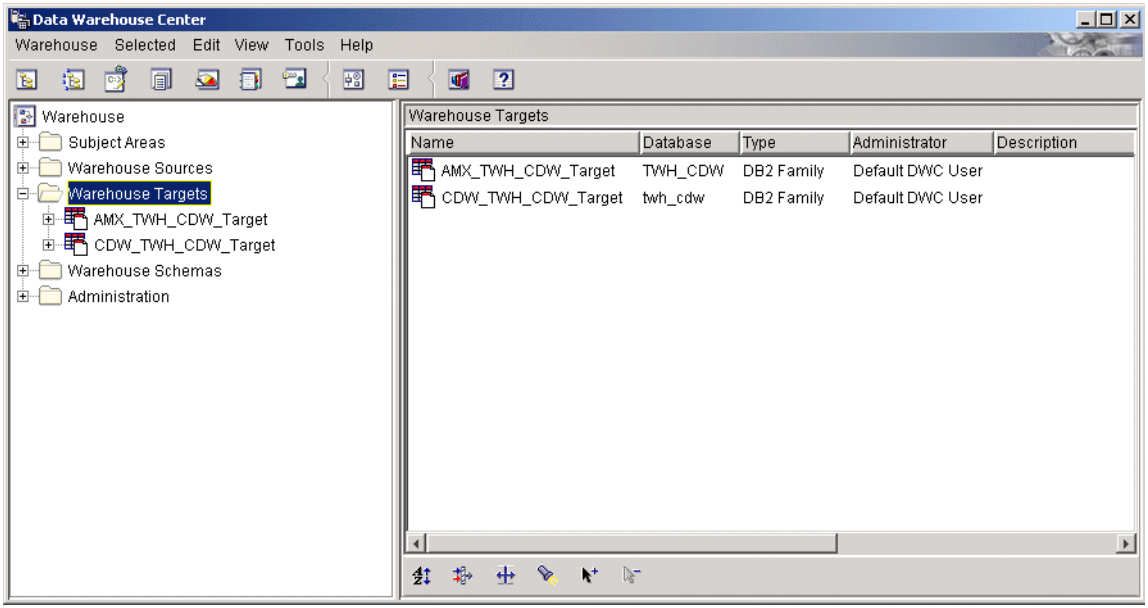


Figure 6-10 IBM Tivoli Monitoring Version 5.1.1 Generic ETL1 Target

For our environment, the values are shown in Figure 6-11 on page 120. If your database for IBM Tivoli Monitoring is not called ITM_DB, you may change the database name.

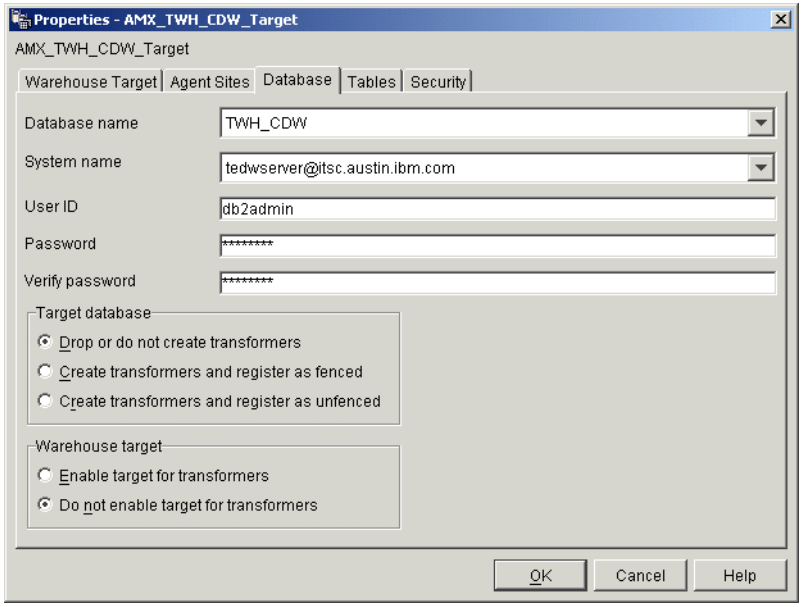


Figure 6-11 AMX_TWH_CDW_Target user ID information

Modifying the ETL for source table name

For a DB2 RIM object, the RIM user must be the instance name. In the Windows platform, the typical instance name is DB2, while in a UNIX platform, the typical instance name is db2inst1. For other databases, there may be a different user ID that is used as the RIM user.

The AMX component, when it is loaded, assumes that the RIM user, and therefore the tables creator names, is db2admin. We need to change this to the appropriate user ID from our RIM setup. We have a DB2 RIM and we run the RIM database in Windows. We use DB2 as the user ID and instance name. The following steps needs to be performed:

1. From the Data warehouse center, from the warehouse sources list, select the AMX_ITM_RIM_Source and open the property page. Go to the Tables and Views tab, as shown in Figure 6-12 on page 121.

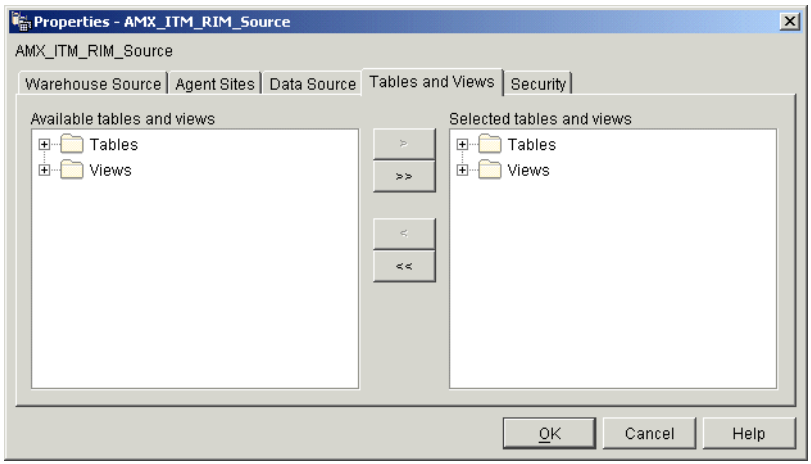


Figure 6-12 Tables and views of AMX_ITM_TIM_Source

2. Expand the Tables folder and you will get a dialog asking for the name filter, such as shown in Figure 6-13. We only need to get the table called ENDPOINTS. The schema name is the RIM user ID.

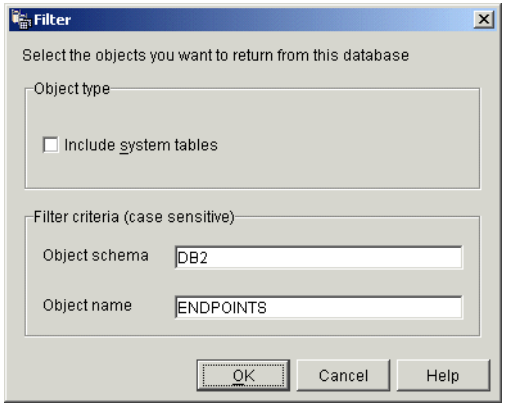


Figure 6-13 Table name filter specification

3. When the DB2.ENDPOINTS has been found, move it from the available tables and views to the selected tables and views box by clicking the **>** button. You will have two ENDPOINTS tables, as shown in Figure 6-14 on page 122. Click **OK**.

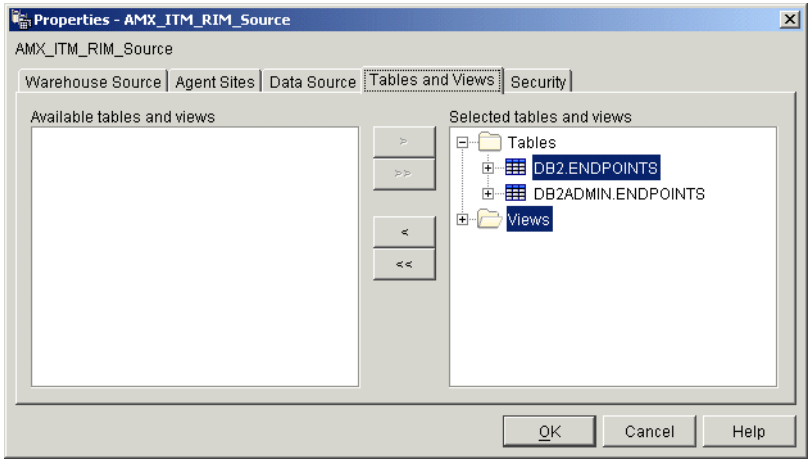


Figure 6-14 Endpoint tables

4. Now from the data warehouse center, expand the Subject Area and find the process called AMX_c05_ETL1_Process. Right click on it and select **Open**. The Process Modeler window is shown in Figure 6-15 on page 123.

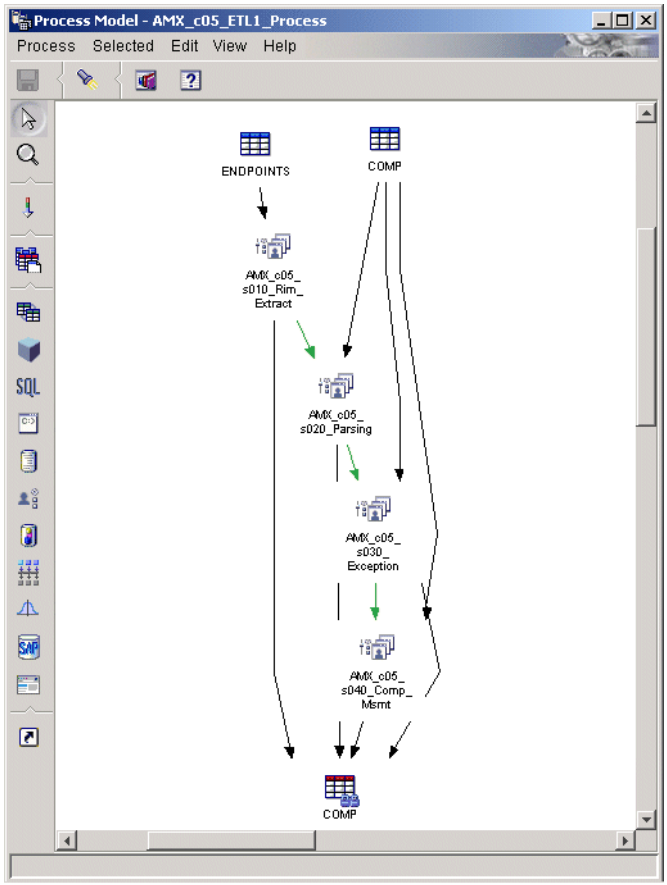


Figure 6-15 AMX_c05_ETL1 process

5. Click on the tables icon and click on the work area; a dialog box will show up, as shown in Figure 6-16 on page 124. Select the DB2.ENDPOINTS table and click the **>** button. Then click **OK**.

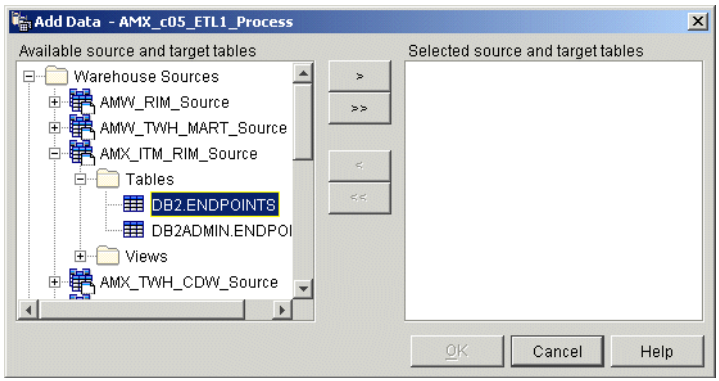


Figure 6-16 Selecting new table

6. The new table is now shown in the process modeler window; now we need to connect the tables to the first step. Use the link icon and select data links. Drag the cursor from the ENDPOINTS table to the AMX_c05_s010_RIM_Extract step and a new link is created.
7. Remove the old link by selecting the link, right-click, and select **Remove**. Remove also the old DB2ADMIN.ENDPOINTS table by selecting it, right-click and select **Remove**.
8. Save the process model by selecting **Process** -> **Save** and close the window.

Scheduling the source ETL

There are two processes that need to be scheduled for the IBM Tivoli Monitoring Version 5.1.1 Generic ETL1 to run. They are:

- AMX_c05_ETL1_Process
- AMX_c10_Rim_Prune_Process

The following steps are similar for both processes and we will use AMX_c05_ETL1_Process to describe them:

1. On the Tivoli Enterprise Data Warehouse Control Center server, using the Data Warehouse Center window, expand **Subject Areas**. Select **AMX_IBM_TIVOLI_Monitoring_v5.1.1_Subject_Area -> Processes** and right-click on **AMX_c05_ETL1_Process.** Choose **Schedule**, as shown in Figure 6-17 on page 125.

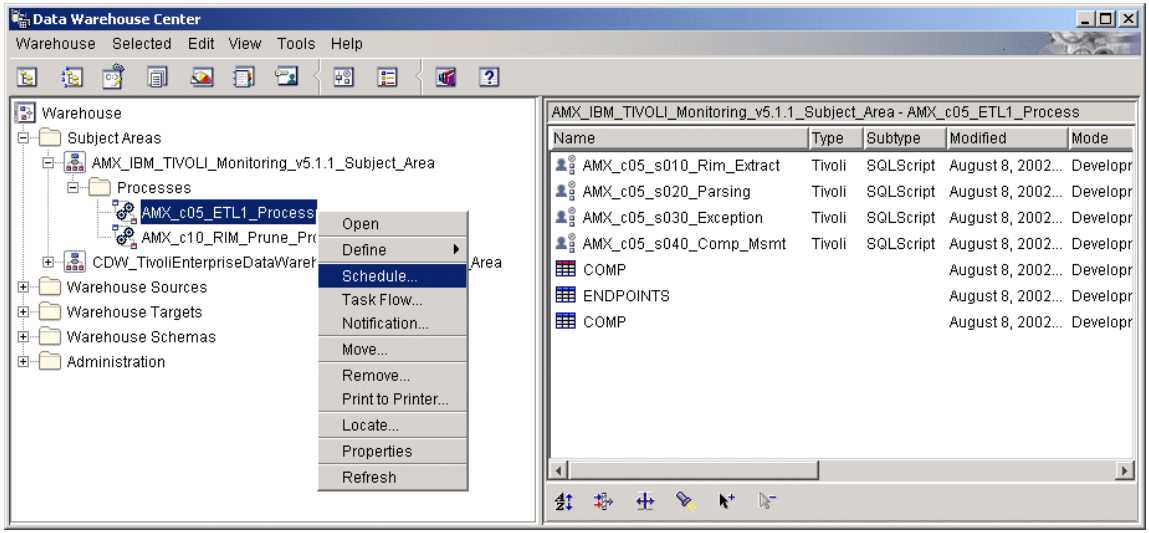


Figure 6-17 Schedule AMX_c05_ETL1_Process

2. Selecting Schedule will open up a dialog box, as shown in Figure 6-18.

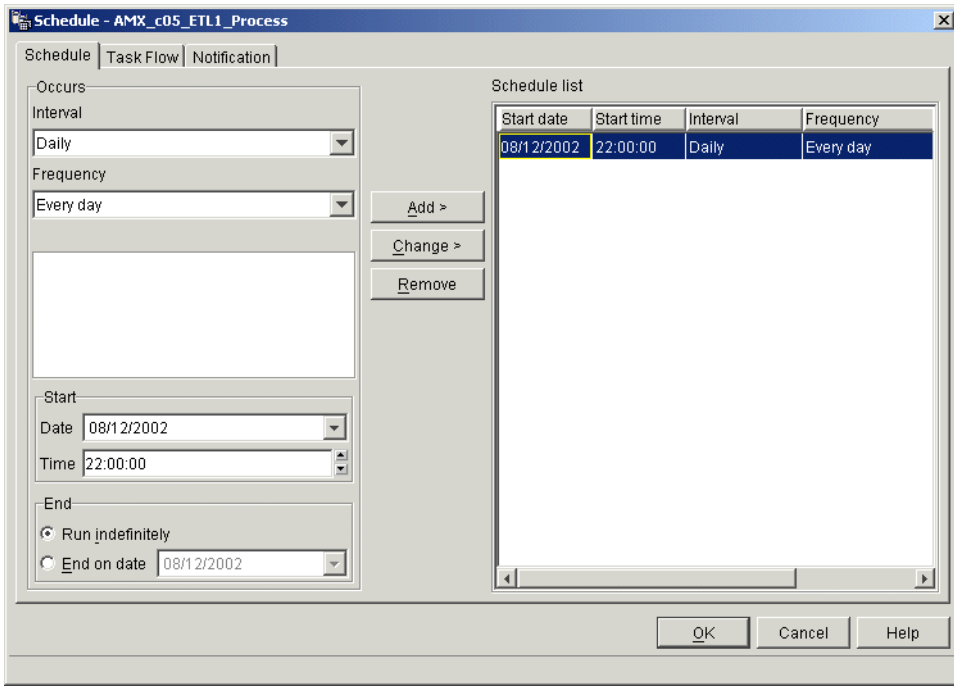


Figure 6-18 Schedule configuration for AMX_c05_ETL1_Process

You should schedule the ETL to run daily during off-hours.

Changing the source ETL status to Production

Both the AMX_c05_ETL1_Process and AMX_c10_Rim_Prune_Process are composed by process steps that have the Development status set as the default. In order for them to run, their status need to be changed from Development to Production. They are:

- AMX_c05_ETL1_Process
  - AMX_c05_s010_Rim_Extract
  - AMX_c05_s020_Parsing
  - AMX_c05_s030_Exception
  - AMX_c05_s040_Comp_Msmt
- AMX_c10_Rim_Prune_Process
  - AMX_c10_s010_Rim_Prune

The following step must be performed for all processes described above. Here we use AMX_c05_ETL1_Process to describe it. On the Tivoli Enterprise Data Warehouse Control Center server, using the Data Warehouse Center window, select the above processes and right-click on them. Select **Mode -> Production**, as shown in Figure 6-19.

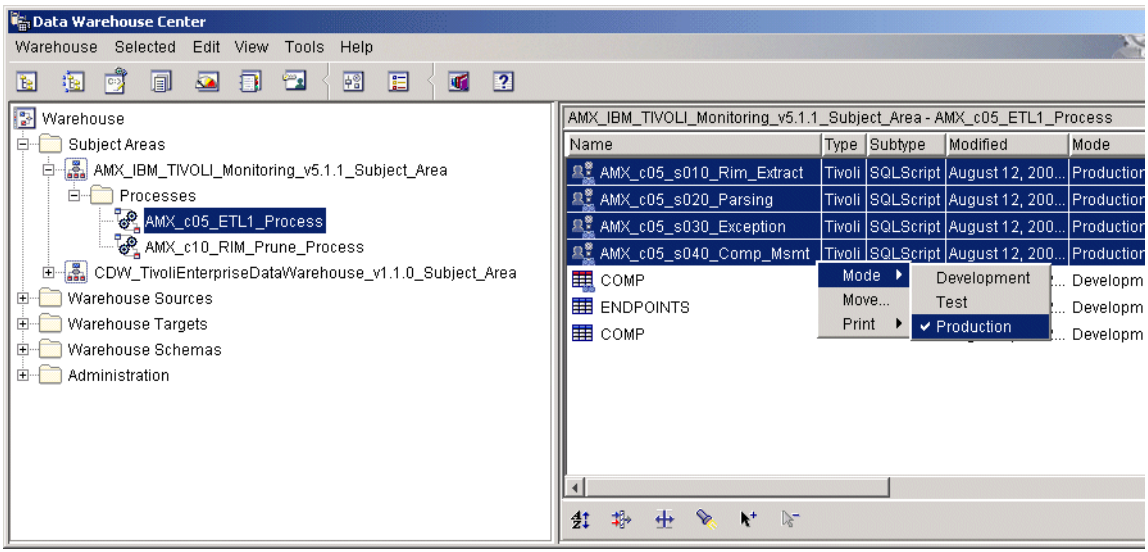


Figure 6-19 Promoting scheduled processes to Production status

6.3.4 Setting up the target ETLs

In this section, we describe the installation procedures of IBM Tivoli Monitoring for Databases warehouse components, also known as IBM Tivoli Monitoring for Databases ETLs. We have three sets of ETLs to load, respectively, for DB2, Oracle, and IBM Informix.

All the IBM Tivoli Monitoring for Databases ETLs provide a set of metadata to collect data from the IBM Tivoli Monitoring Generic ETL1 information that has already been stored in the CDW. They retrieve data collected by the IBM Tivoli Monitoring Version 5.1.1 resource models, create a data mart, and provide some pre-defined reports.

The measurement code for the IBM Tivoli Monitoring for Databases: DB2 is CTD, for IBM Tivoli Monitoring for Databases: Oracle it is CTO and for IBM Tivoli Monitoring for Databases: Informix it is CTR.

The installation can be done similarly to the AMX installation in 6.3.3, “Setting up the source ETL” on page 112 using the Tivoli Enterprise Data Warehouse CLI or the GUI installation program. Here we describe the process using the GUI method. The following steps should be performed in the Tivoli Enterprise Data Warehouse Control Center server.

You need both the Tivoli Enterprise Data Warehouse and the IBM Tivoli Monitoring for Databases products installation media. The installation is similar for all ETLs.

1. Run the setup.exe from the Tivoli Enterprise Data Warehouse CD-ROM and click **OK** to start the installation.
2. When the InstallShield Wizard dialogue window for Tivoli Enterprise Data Warehouse Installation appears, click **Next**.
3. The dialogue window for the type of installation appears. Select **Application installation only** and the directory name where the Tivoli Enterprise Data Warehouse components are installed. We used C:\TWH. Click **Next** to continue.
4. The host name dialogue window appears. Verify that this is the correct host name for the Tivoli Enterprise Data Warehouse Control Center server. Click **Next**.
5. The local system DB2 configuration dialogue window appears. The installation process asks for a valid DB2 user ID. Enter the valid DB2 user ID and password that were created during the DB2 installation on your local system. In our case, we used db2admin. Click **Next**.
6. The path to the installation media for the application packages dialogue window appears. You should provide the location of the ETL installation

directory; it is typically on the IBM Tivoli Monitoring for Databases CD under the sub-directory of tedw_apps_etl. You need to select the sub-directory where the file twh_app_install_list.cfg resides, such as:

- CD_drive:\tedw_apps_etl\ctd\ for DB2
- CD_drive:\tedw_apps_etl\cto\ for Oracle
- CD_drive:\tedw_apps_etl\ctr\ for IBM Informix

Leave the **Now (prevents typing errors)** option checked to verify that the source directory is immediately accessible and that it contains the correct files. Click **Next**. Figure 6-20 shows the IBM Tivoli Monitoring for Databases: DB2 ETL as an example.

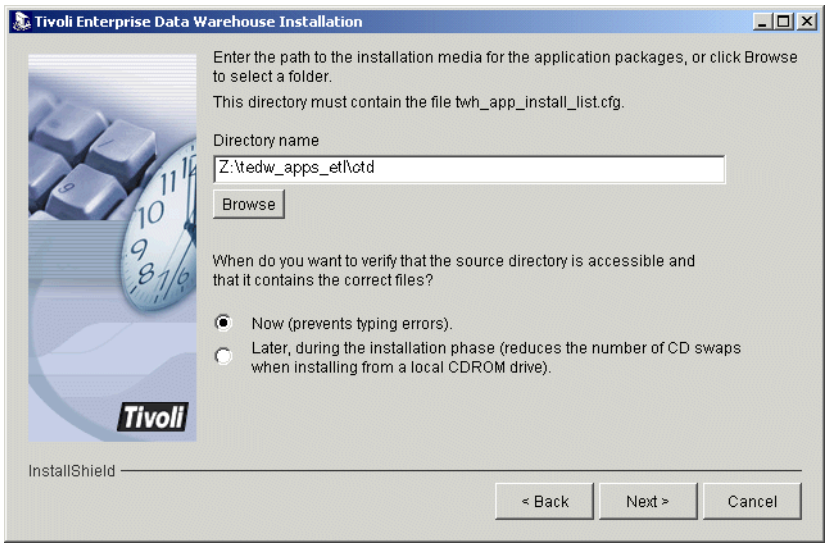


Figure 6-20 Installation media for the IBM Tivoli Monitoring for Databases ETL

7. The overview of selected features dialogue window appears, as shown in Figure 6-21 on page 129. Click **Install** to start the installation.

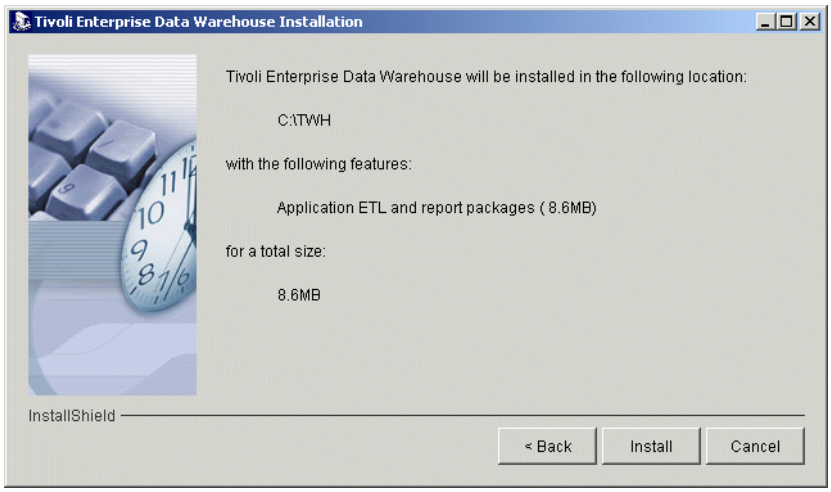


Figure 6-21 IBM Tivoli Monitoring for Databases ETL programs installation

8. Once the installation is finished, the Installation summary window appears. Figure 6-22 shows a successful installation of the IBM Tivoli Monitoring for Databases: DB2. If the installation was not successful, check the TWHApp.log file for any errors. This log file is located at the <TWH_inst_dir>\apps\<meascode>\, where <TWH_inst_dir> is the Tivoli Enterprise Data Warehouse installation directory and <meascode> is the component specific measurement code.

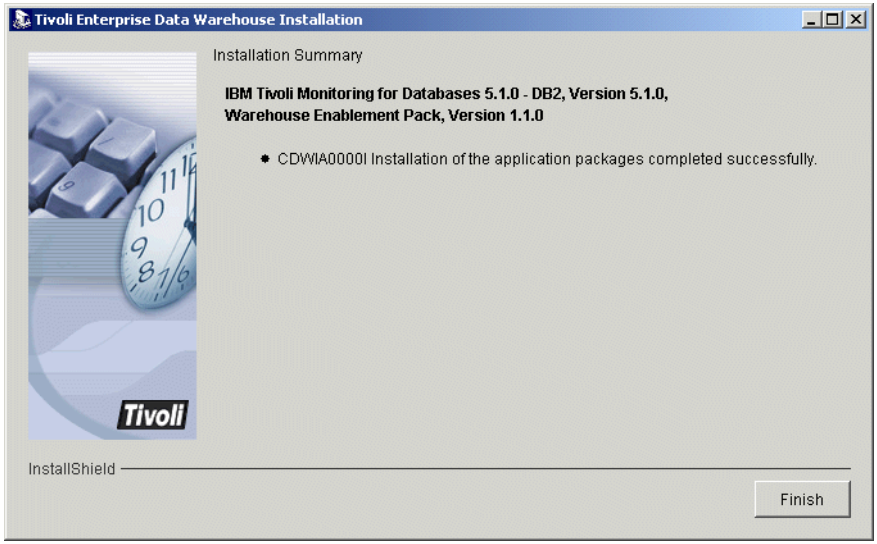


Figure 6-22 IBM Tivoli Monitoring for Databases ETL - installation summary

9. After the component installation is completed, you must reboot the Tivoli Enterprise Data Warehouse Control Center server (and also other servers for the Tivoli Presentation Services, if it is on a different server).

There is a need to change some configuration settings for the IBM Tivoli Monitoring for Databases warehouse components to function properly. The following steps should be performed for all IBM Tivoli Monitoring for Databases ETLs:

1. Define user authority to the Warehouse Sources and Targets
2. Schedule the ETL
3. Change the ETL status to Production

Refer to Table 6-1 for the item names in the procedures in the sub-sections.

Table 6-1 IBM Tivoli Monitoring for Databases warehouse objects name

	DB2	Oracle	IBM Informix
Warehouse sources	CTD_TWH_CDW_Source CTD_TWH_MART_Source	CTO_TWH_CDW_Source CTO_TWH_MART_Source	CTR_TWH_CDW_Source
Warehouse targets	CTD_TWH_CDW_Target CTD_TWH_MART_Target CTD_TWH_MD_Target	CTO_TWH_CDW_Target CTO_TWH_MART_Target CTO_TWH_MD_Target	CTR_TWH_MART_Target CTR_TWH_MD_Target
Subject areas	CTD_m05_ETL2_Process	CTO_m05_ETL2_Process	CTR_m05_CreateMarts_ Process
Steps	CTD_m05_s010_Init_Attr CTD_m05_s020_Dimension CTD_m05_s030_Fact CTD_m05_s040_RollUp CTD_m05_s050_Prune	CTO_m05_s010_Init_Attr CTO_m05_s020_Dimension CTO_m05_s030_Fact CTO_m05_s040_RollUp CTO_m05_s050_Prune CTO_m05_s060_Schedule	CTR_m05_s010_ Buildmart CTR_m05_s020_Rollup

Defining user authority to the Warehouse Sources and Targets

You should inform the Tivoli Enterprise Data Warehouse Control Center server of user access information for every Source and Target ETL process installed by the IBM Tivoli Monitoring for Databases ETL. The following steps should be followed:

1. Start the IBM DB2 Control Center utility by selecting **Start -> Programs -> IBM DB2 -> Control Center**.
2. On the IBM DB2 Control Center utility, start the IBM DB2 Data Warehouse Center utility by selecting **Tools -> Data Warehouse Center**. The Data Warehouse Center logon windows appears.

3. Log into the IBM DB2 Data Warehouse Center utility using the local DB2 administrator user ID, in our case, db2admin.
4. In the Data Warehouse Center window, expand the Warehouse Sources folder. Update the database sources that relates to the application that you want to configure, as provided in Table 6-1 on page 130.

   You should edit the properties of each one of those warehouse sources. In order to do that, right-click and select **Properties** and then select the **Database** tab. Fill in the database user ID and password information. For our environment, the values are shown in Figure 6-23, using CTD_TWH_CDW_Source as an example.

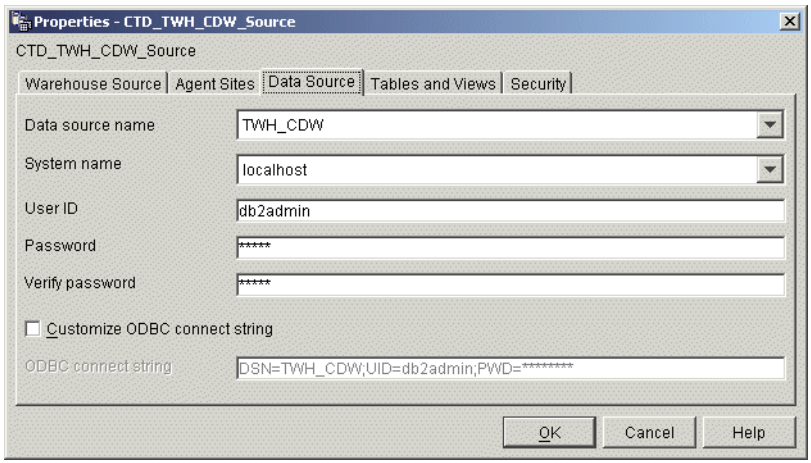


Figure 6-23 CTD_TWH_CDW_Source user ID information

5. Again, for the warehouse targets specified in Table 6-1 on page 130, you need to modify the user ID information from the property pages. You should edit the properties of each one of those warehouse targets. In order to do that, right-click it and select **Properties** and then select the **Database** tab. Fill in the database user ID and password information. For our environment, the values are shown in Figure 6-24 on page 132, using CTD_TWH_MART_Target as an example.

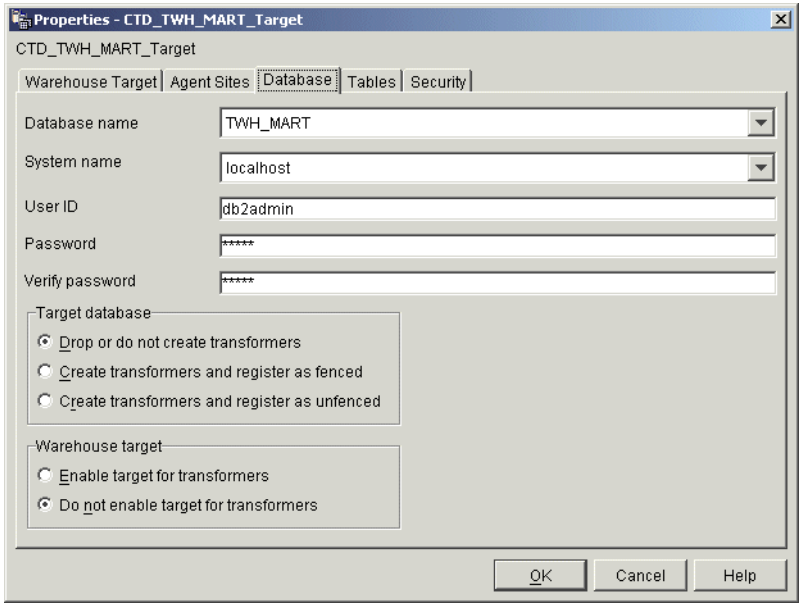


Figure 6-24 CTD_TWH_MART_Target user ID information

Modification for CTO_m05_s060_Schedule_Step

For the IBM Tivoli Monitoring for Databases: Oracle, the step CTO_m05_s060_Schedule_Step need a special processing. It is defined to run on the agent site called REE_AgentSite, which is the local host for our Tivoli Enterprise Data Warehouse server. We need to enable this process at the Default DWC Agent Site, as this also refers to the local host.

1. Modify the Default DWC Agent Site to add the runReport process:
   a. From the data warehouse center, select **Administration** -> **Agent Site** -> **Default DWC Agent Site**, right-click, and select **Properties**.
   b. Select the **Programs** tab of the Property window
   c. Select the runReport program and move it to the selected programs area.
   d. Click **OK**.
2. Specify the CTO_m05_s060_Schedule_Step to run on the Default DWC Agent Site:
   a. From the data warehouse center, select **Subject Areas** -> **CTO IBM Tivoli Monitoring for Databases: Oracle Version 5.1.1** -> **Processes** -> **CTO_m05_ETL2_Process**.
   b. Right-click on the CTO_m05_s060_Schedule_Step and select **Properties**.

c. Go to the last tab and set the Agent site to be the Default DWC Agent Site.

d. Click **OK**.

Scheduling the target ETL

The processes that need to be scheduled depend on the warehouse component, as shown in Table 6-1 on page 130. There is one process to schedule for each database type. The following steps are similar for all the processes mentioned in Table 6-1 on page 130, and we will use CTD_m05_ETL2_Process to describe them.

1. Using the Data Warehouse Center, expand Subject Areas and select the appropriate process name. Right-click on it and select **Schedule**, as shown in Figure 6-25.

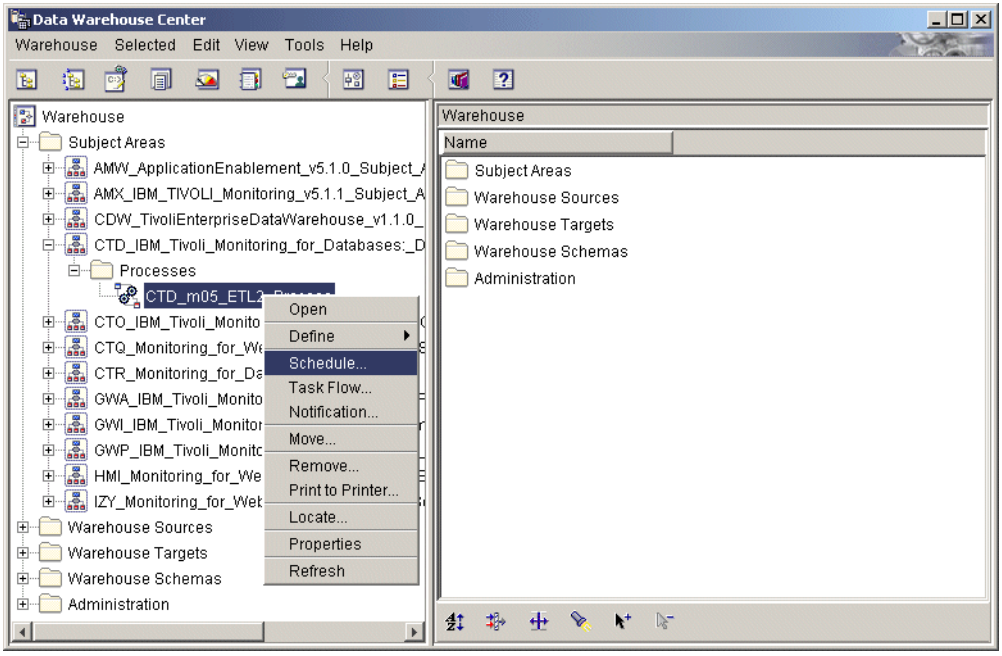


Figure 6-25 Scheduling CTD_m05_ETL2_Process

2. Selecting Schedule will open up a dialog box, as shown in Figure 6-26 on page 134. In that dialog box, you can specify the time and frequency specification of when you want the process to be run. Remember that these are target ETL processes that needs to be run after the source ETL has been completed.

Tip: If you are running multiple target ETLs for different components, it is better to have these ETLs running in serial; do not run them in parallel, as they may use same tables. Also, these target ETLs must run after the source ETL is scheduled, as in “Scheduling the source ETL” on page 124. You may also create short cuts to link the ETLs together, so you only need to schedule the first ETL. For more information on creating short cuts, refer to Chapter 7 of *Tivoli Enterprise Data Warehouse Installing and Configuring Version 1.1*, GC32-0744.

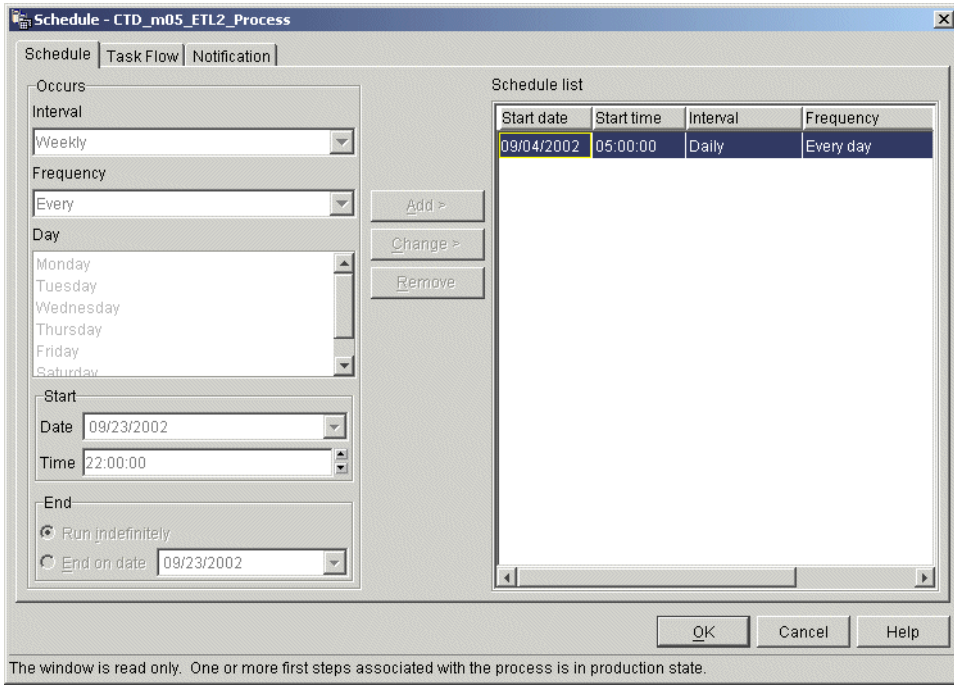


Figure 6-26 Schedule configuration for CTQ_m15_Load_Fact_Data_Process

Changing the target ETL status to Production

All IBM Tivoli Monitoring for Databases processes are composed by steps that have their mode as Development. In order for them to run, their status need to be changed from Development to Production. See the lists of steps for each application in Table 6-1 on page 130. The following step must be performed for all processes described above. Here we use CTD_m05_ETL2_Process to describe it.

On the Tivoli Enterprise Data Warehouse Control Center server, using the Data Warehouse Center window, select the steps and right-click on them. Choose **Mode -> Production**, as shown in Figure 6-27.

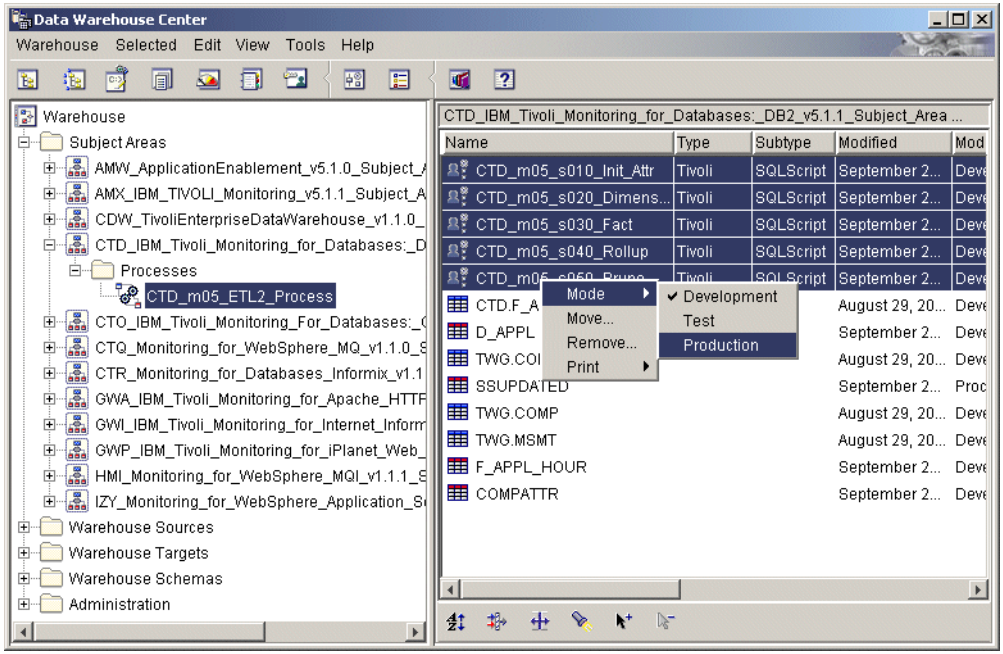


Figure 6-27 Promoting IBM Tivoli Monitoring for Databases ETL to production

6.4 Activating collection

This section discusses how to set up the data itself so that reporting can be performed. The discussion consists of the following:

- Setting up resource models to run on the endpoint that will generate the appropriate information for the data marts and the collection for IBM Tivoli Monitoring so the resource model data is logged into the RIM database
- Running the ETLs to collect data into the data marts

6.4.1 Running resource models

All resource models generate measurement data into the database. Specific data is needed to fill up the predefined reports. Refer to 6.5.3, “Working with data mart and reports” on page 147 for detailed resource model indicators needed for the report.

We need to create the profile and set up ETL collection. The following steps need to be performed:

1. Create a profile manager for storing the profiles for the data warehouse collection. Create the Tmw2kProfile objects in the profile manager and add the appropriate resource models.
2. For each resource model, open it by clicking **Edit**. In the Edit resource model window, click **Logging**. The logging window will be shown, as in Figure 6-28. You need to check the **Enable Data Logging** and **TEDW data** check boxes. Click on **Apply Changes and Close** when done. Click on **Modify & Close** from the Edit resource model window.

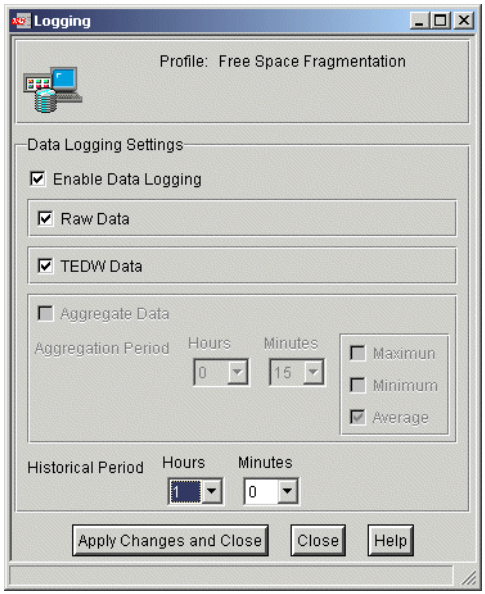


Figure 6-28 Logging option

3. Now you can specify the subscribers. Each resource model can only have a specific subscriber types. You may want to create multiple profiles where each profiles contain resource models for a specific object types.
4. Distribute the profile; you can either do it from the Tivoli desktop or by using the **`wdmdistrib`** command.
5. Check the distribution of the profile and the execution of it using the **`wdmlseng`** command. Ensure that all the profiles have the state of Running.

6. You need to tell the gateways, to which the endpoint reports to, to start collecting historical information to be put into the RIM object. Use the **wdmcollect** command. The syntax of the command is:

   ```
   wdmcollect -e <endpoint> -s <time>
   ```

   where <endpoint> is the endpoint name and <time> is the collection interval in hours. After you run this command for all the endpoints, you can check the result using the **wdmcollect -q** command. The managed node will pull the data from the endpoint every interval and store it under $DBDIR/dmml/tedw.

7. Overnight, you can start checking on the RIM database whether the data has been collected. We check the collection by verifying that the ENDPOINTS table has been populated and by checking the timekey_dttm in the metricsdata table, as shown in Example 6-3.

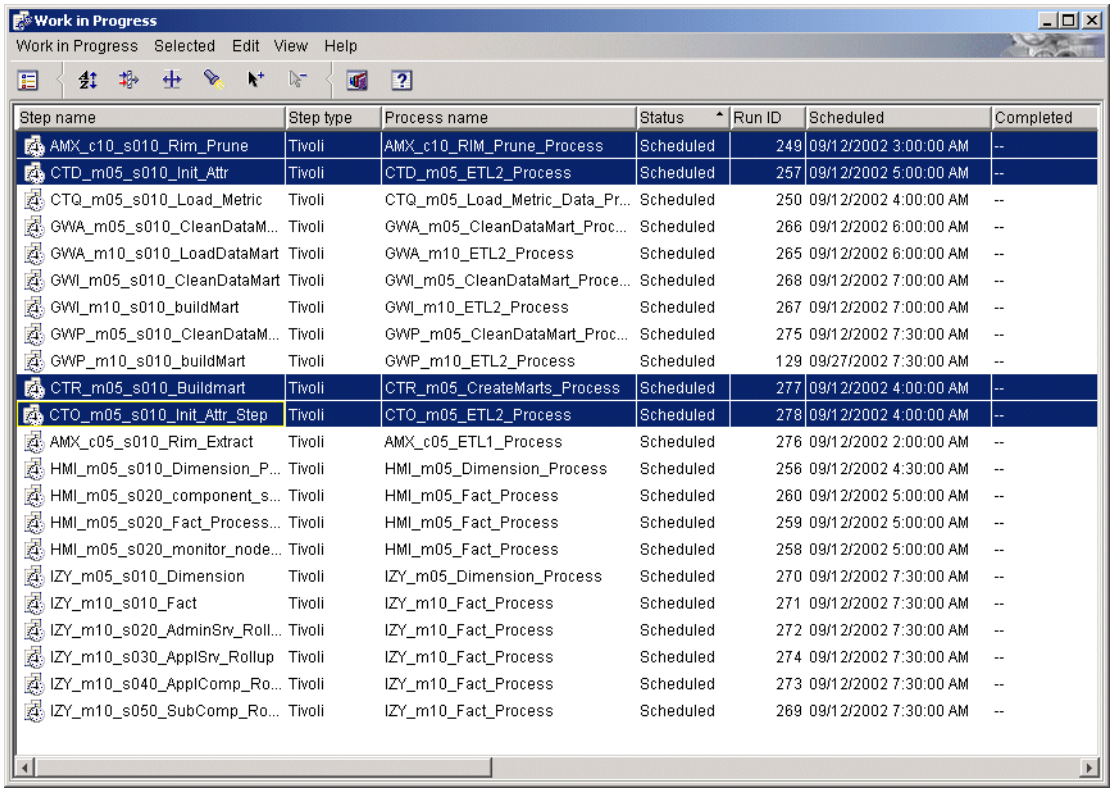
Example 6-3 Sample SQL that check the collection

```
db2 => connect to itm_db

   Database Connection Information

 Database server        = DB2/6000 7.2.0
 SQL authorization ID   = DB2INST1
 Local database alias   = ITM_DB

db2 => select host_name from endpoints

HOST_NAME
--------------------------------------------------------------------------------
tokyo
JAKARTA.itsc.austin.ibm.com
paris
capecod.itsc.austin.ibm.com
bangkok

  5 record(s) selected.

db2 => select max(timekey_dttm) from metricsdata

1
--------------------------
2002-09-30-15.00.55.000000

  1 record(s) selected.
```

6.4.2 Running ETLs

The ETLs have been scheduled to run at a given time in “Scheduling the source ETL” on page 124 and “Scheduling the target ETL” on page 133. You may also want to run the ETL manually. The following steps show how to run the ETLs manually:

1. From the data warehouse center, select **Warehouse** -> **Work In Progress**. This will bring up the Work in Progress window, as shown in Figure 6-29. The figure shows all scheduled processes with the IBM Tivoli Monitoring for Databases processes highlighted. You may also see some other items with the status of either Successful, Failed, or In progress.



Step name	Step type	Process name	Status	Run ID	Scheduled	Completed
AMX_c10_s010_Rim_Prune	Tivoli	AMX_c10_RIM_Prune_Process	Scheduled	249	09/12/2002 3:00:00 AM	--
CTD_m05_s010_Init_Attr	Tivoli	CTD_m05_ETL2_Process	Scheduled	257	09/12/2002 5:00:00 AM	--
CTQ_m05_s010_Load_Metric	Tivoli	CTQ_m05_Load_Metric_Data_Pr...	Scheduled	250	09/12/2002 4:00:00 AM	--
GWA_m05_s010_CleanDataM...	Tivoli	GWA_m05_CleanDataMart_Proc...	Scheduled	266	09/12/2002 6:00:00 AM	--
GWA_m10_s010_LoadDataMart	Tivoli	GWA_m10_ETL2_Process	Scheduled	265	09/12/2002 6:00:00 AM	--
GWI_m05_s010_CleanDataMart	Tivoli	GWI_m05_CleanDataMart_Proce...	Scheduled	268	09/12/2002 7:00:00 AM	--
GWI_m10_s010_buildMart	Tivoli	GWI_m10_ETL2_Process	Scheduled	267	09/12/2002 7:00:00 AM	--
GWP_m05_s010_CleanDataM...	Tivoli	GWP_m05_CleanDataMart_Proc...	Scheduled	275	09/12/2002 7:30:00 AM	--
GWP_m10_s010_buildMart	Tivoli	GWP_m10_ETL2_Process	Scheduled	129	09/27/2002 7:30:00 AM	--
CTR_m05_s010_Buildmart	Tivoli	CTR_m05_CreateMarts_Process	Scheduled	277	09/12/2002 4:00:00 AM	--
CTO_m05_s010_Init_Attr_Step	Tivoli	CTO_m05_ETL2_Process	Scheduled	278	09/12/2002 4:00:00 AM	--
AMX_c05_s010_Rim_Extract	Tivoli	AMX_c05_ETL1_Process	Scheduled	276	09/12/2002 2:00:00 AM	--
HMI_m05_s010_Dimension_P...	Tivoli	HMI_m05_Dimension_Process	Scheduled	256	09/12/2002 4:30:00 AM	--
HMI_m05_s020_component_s...	Tivoli	HMI_m05_Fact_Process	Scheduled	260	09/12/2002 5:00:00 AM	--
HMI_m05_s020_Fact_Process...	Tivoli	HMI_m05_Fact_Process	Scheduled	259	09/12/2002 5:00:00 AM	--
HMI_m05_s020_monitor_node...	Tivoli	HMI_m05_Fact_Process	Scheduled	258	09/12/2002 5:00:00 AM	--
IZY_m05_s010_Dimension	Tivoli	IZY_m05_Dimension_Process	Scheduled	270	09/12/2002 7:30:00 AM	--
IZY_m10_s010_Fact	Tivoli	IZY_m10_Fact_Process	Scheduled	271	09/12/2002 7:30:00 AM	--
IZY_m10_s020_AdminSrv_Roll...	Tivoli	IZY_m10_Fact_Process	Scheduled	272	09/12/2002 7:30:00 AM	--
IZY_m10_s030_ApplSrv_Rollup	Tivoli	IZY_m10_Fact_Process	Scheduled	274	09/12/2002 7:30:00 AM	--
IZY_m10_s040_ApplComp_Ro...	Tivoli	IZY_m10_Fact_Process	Scheduled	273	09/12/2002 7:30:00 AM	--
IZY_m10_s050_SubComp_Ro...	Tivoli	IZY_m10_Fact_Process	Scheduled	269	09/12/2002 7:30:00 AM	--

Figure 6-29 Work in Progress window

2. Right-click on the process that you want to run and select **Run now**. The steps within the process will be run sequentially, based on the dependency in the process.

6.5 IBM Tivoli Monitoring for Databases reports

IBM Tivoli Monitoring for Databases provides a set of data marts and predefined reports that allow you to use the IBM console for viewing and creating reports. The Report Interface is not meant to replace OLAP or business intelligence tools. If you have multidimensional reporting requirements or need to create a more sophisticated analysis of your data, Tivoli Enterprise Data Warehouse's open structure provides you an easy interface to plug into OLAP or other business intelligence tools. Nevertheless, for two dimensional reporting requirements, Tivoli Enterprise Data Warehouse Reporting Interface provides you a powerful tool.

The Reporting Interface of Tivoli Enterprise Data Warehouse provides three types of reports:

- Summary reports

  The summary report is typically used to display a many measurements versus a many components relation. The result is a table where the rows show components or groups of components and the columns show typically the measurements. Additionally, summary values for all components and all groups of components are shown. This kind of report can be used if, for example, you want to create an overview of the workload of servers or server groups.

- Extreme case reports

  The extreme case report is a one measurement versus many components report. With this type of report, you can find the components or component groups with the highest or lowest values of a certain metric. The result will be a graph with the worst or best components in the x-axis and the corresponding metric values in the y-axis.

- Health reports

  The health report is typically used to display many measurements against many components versus time relation. The result is a graph where the x-axis shows the time and the y-axis shows typically the measurements. This kind of report is used to show the time-development of a metric. This allows you to recognize a trend and to predict possible health problems of a component in the future.

To understand what reporting facilities are provided for the IBM Tivoli Monitoring for Databases, we give the details on the data marts and reports provided with the products in the following sections.

6.5.1 Data mart schemas

A data mart is a set of tables that fulfil a specialized set of reporting requirements. The data mart is usually organized as star schema or a set of star schemas. A star schema get its name from the shape of its diagram with a single table in the middle, which is called a fact table, and a set of table related to the fact table around it, which are called dimension tables.

The fact table contains the specific measurement value that references the attributes that are stored in the dimension tables. The easiest way is to imagine a sales data mart with the revenue information as the fact table, while the dimensions can be location (country, state, city, and zipcode), time (year, month, and day), product (group, item, and size), and so on. So each entry in the fact table represents a single revenue from a specific product in a specific zipcode at a specific day. The data mart for IBM Tivoli Monitoring for Databases relates to the measurement metric from the resource models that were deployed in the TMR.

This section lists all the default schemas available with the IBM Tivoli Monitoring for Databases products. The IBM Tivoli Monitoring for Databases have a naming convention that uniquely identifies to which product they belong, as can be seen in Table 6-2. Each schema has three characters as the schema name to identify which product they belong, such as DB2, Oracle, or IBM Informix.

Table 6-2 Database identifiers

Schema name	Component
CTD	IBM Tivoli Monitoring for Databases: DB2
CTO	IBM Tivoli Monitoring for Databases: Oracle
CTR	IBM Tivoli Monitoring for Databases: Informix

IBM Tivoli Monitoring for Databases: DB2 data marts

The IBM Tivoli Monitoring for Databases: DB2 data mart is called TWH_MART in the RPI.DATAMART table in the TWH_MD database. There are several star schemas that belong to the data mart. The star schema naming convention is:

```
CTD <period> <resourcetype > Star Schema
```

where:

period The time detail of the data in the fact table, this can be Hourly, Daily, Weekly, or Monthly.

resourcetype The resource type that the star schema applies to, which can be:

- Database
- Node
- Application
- Tablespace
- Table
- Apply
- Table Apply
- Instance
- Gateway

Each star schema contains fact tables and dimension tables. The fact tables defines the information in the star schema. The table names are:

- CTD.D_<resource>: Dimension tables that contain the list of objects for the specific resource type. These tables provide primary keys for most of the fact tables.
- CTD.D_<resource>_METRIC: Dimension tables that contain the list of metrics for a specific resource type. Each of these dimension tables are linked to a specific fact table by the key of METRIC_ID.
- CTD.F_<type>_<period>: Fact tables that contain the measurement information (maximum, minimum, average, and count) for a specific type on a specific frequency.

where:

period The time detail of the data in the fact table, this can be HOUR, DAY, WEEK, or MONTH.

type The abbreviated type that the fact table apples to, which can be:

- DB
- INST
- NODE
- APPL

– TBSP
– TABLE
– APPLY
– TAPPLY
– GTWY

IBM Tivoli Monitoring for Databases: Oracle data marts

The IBM Tivoli Monitoring for Databases: Oracle data mart is called CTO Oracle Data Mart in the RPI.DATAMART table in the TWH_MD database. There are several star schemas that belong to the data mart. The star schema naming convention is:

```
CTO <resourcetype> <period> Star Schema
```

where:

resourcetype The resource type that the star schema apply to, which can be:

– Archive
– Class Ping
– Cluster
– Database Cache Advanced
– Database
– Datafile
– Dispatcher
– Dispatcher Protocol
– Dump Space
– Heterogeneous Agent
– Index
– Instance
– Job
– Listener
– PQ Slave Process
– Process
– Queue
– Queue Scheduled

– Redo Log Group
– Segment
– Shared Server
– System Event
– Table
– Tablespace
– Undo Statistics

period The time detail of the data in the fact table, this can be Hourly, Daily, Weekly, or Monthly.

Each star schema contains fact tables and dimension tables. The fact tables defines the information in the star schema. The table names are:

- CTO.D_<resource>: Dimension tables that contain the list of objects for the specific resource type. These tables provide primary keys for most of the fact tables.
- CTO.D_<resource>_METRIC: Dimension tables that contain the list of metrics for a specific resource type. Each of these dimension tables are linked to a specific fact table by the key of METRIC_ID.
- CTO.F_<type>_<period>: Fact tables that contain the measurement information (maximum, minimum, average, and count) for a specific type on a specific frequency.

where:

period The time detail of the data in the fact table, this can be HOUR, DAY, WEEK, or MONTH.

type The abbreviated type that the fact table applies to, which can be:

– ARCHIVE
– CLSPING
– CLUSTER
– DBCACHAD
– DATABASE
– DATAFILE
– DISPATCH
– DISP_PROT
– DUMPSPACE

- HS_AGENT
- INDEX
- INSTANCE
- JOB
- LISTENER
- PQ_SLAVE
- PROCESS
- QUEUE
- QUESCHED
- REDOLOGRP
- SEGMENT
- SHARESERV
- SYS_EVENT
- TABLE
- TBSP
- UNDO_STAT

IBM Tivoli Monitoring for Databases: Informix data marts

The IBM Tivoli Monitoring for Databases: Informix data mart is called CTR Informix Daily Data Mart in the RPI.DATAMART table in the TWH_MD database. There are several star schemas that belong to the data mart. The star schema naming convention is:

```
CTR Informix <resourcetype> <period> Star Schema
```

where:

resourcetype The resource type that the star schema apply to, which can be:

- Database space
- Instance
- Least Recently Used Queue
- Logical Log
- Memory Segment
- Storage Chunk
- Virtual Processor

period The time detail of the data in the fact table, this can be Hourly, Daily, Weekly, or Monthly.

Each star schema contains fact tables and dimension tables. The fact tables defines the information in the star schema. The table names are:

- CTR.D_<resource>: Dimension tables that contain the list of objects for the specific resource type. These tables provide primary keys for most of the fact tables.
- CTR.D_<resource>_METRIC: Dimension tables that contain the list of metrics for a specific resource type. Each of these dimension tables are linked to a specific fact table by the key of METRIC_ID.
- CTR.F_<type>_<period>: Fact tables that contain the measurement information (maximum, minimum, average and count) for a specific type on a specific frequency.
- Additional dimension tables, such as:
  - CTR.D_VER: Version of Informix Dynamic Server
  - CTR.D_INTERP: Operating system hosting the Informix Dynamic Server

where:

period The time detail of the data in the fact table, this can be HOUR, DAY, WEEK, or MONTH.

type The abbreviated type that the fact table applies to, which can be:

– DBSPACE

– INSTANCE

– LRUQ

– LLOG

– MEMSEG

– CHUNK

– VP_ID

6.5.2 Reports available with IBM Tivoli Monitoring for Databases

There are several ready made reports provided with IBM Tivoli Monitoring for Databases. The following sub-sections lists the reports by components.

IBM Tivoli Monitoring for Databases: DB2 reports

Table 6-3 on page 146 lists the available reports for IBM Tivoli Monitoring for Databases: DB2.

Table 6-3 IBM Tivoli Monitoring for Databases: DB2 reports

Report name	Description	Type	Measurement
CTD Hourly Percent Connections Used	Percent connections used on a hourly basis.	Health Check	PctConnectionsUsed
CTD Hourly Deadlocks Delta Health Report	The hourly deadlocks found between monitoring intervals.	Health Check	DeadlocksDelta
CTD Hourly Applications Connected	The average number of connections to a database on a hourly basis.	Health Check	NumConnectionsToLocalDBases
CTD Hourly Minimum Buffer Pool Hit Ratio	The minimum buffer pool hit ratio per hour.	Health Check	PctBufferPoolHits
CTD Hourly Maximum Percentage Used of Primary Log	The maximum utilization of the primary log space per hour.	Health Check	PctLogSpaceUsedByPrimary

IBM Tivoli Monitoring for Databases: Oracle reports

Table 6-4 lists the available reports for IBM Tivoli Monitoring for Databases: Oracle.

Table 6-4 IBM Tivoli Monitoring for Databases: Oracle reports

Report name	Description	Type	Measurement
Buffer Cache Hit Ratio (Daily) - Extreme Case	Buffer Cache Hit Ratio (Daily) - Extreme Case	Extreme Case	BufferCacheHitPercentage
Deadlocks (Daily) - Health Check	Deadlocks (Daily) - Health Check	Health Check	Deadlocks
Dispatcher Contention (Daily) - Summary	Dispatcher Contention (Daily) - Summary	Summary	AvgWaitTime BusyPercentage
Oracle RDBMS Availability (Daily) - Extreme Case	Oracle RDBMS Availability (Daily) - Extreme Case	Extreme Case	-
Tablespace Usage (Daily) - Extreme Case	Tablespace Usage (Daily) - Extreme Case	Extreme Case	MBUsed

IBM Tivoli Monitoring for Databases: Informix reports

Table 6-5 lists the available reports for IBM Tivoli Monitoring for Databases: Informix.

Table 6-5 IBM Tivoli Monitoring for Databases: Informix reports

Report name	Description	Type	Measurement
Informix Health Check - 7 Days	Informix Health Check - 7 Days	Health Check	Active_Transactions Number_of_Commits Number_of_Rollbacks Percent_State_On-Line Transaction_Overflows
Informix Thread Activity - 7 Days	Informix Thread Activity - 7 Days	Health Check	Number_of_Deadlocks User_Thread_Overflows System_CPU User_CPU
Informix Disk Utilization - 7 Days	Informix Disk Utilization - 7 Days	Health Check	Free_Pages Pages Number_Chunks Percent_Free_Space
Informix Logical Log - 7 Days	Informix Logical Log - 7 Days	Health Check	Logical_Log_Percent_ Available Pages_Used

6.5.3 Working with data mart and reports

The following simple procedure gives a simplified method of going to the IBM Console to view reports. For more detail on working with IBM Console, refer to *Introduction to Tivoli Enterprise Data Warehouse*, SG24-6607.

1. Open a Web browser and go to the IBM Console URL. In our case, it is installed in the Tivoli Enterprise Data Warehouse server, which is ibmtiv12, hence the URL is `http://ibmtiv12/IBMConsole`.
2. Input your user ID and password and you will get to the IBM Console.
3. Expand the Work with Reports in the left window pane. Click the **Manage Reports and Report Output** link.
4. Click on the square expansion box at the left of Root to expand it.
5. Select **Reports** and you should see a window with the lists of reports, as shown in Figure 6-30 on page 148.

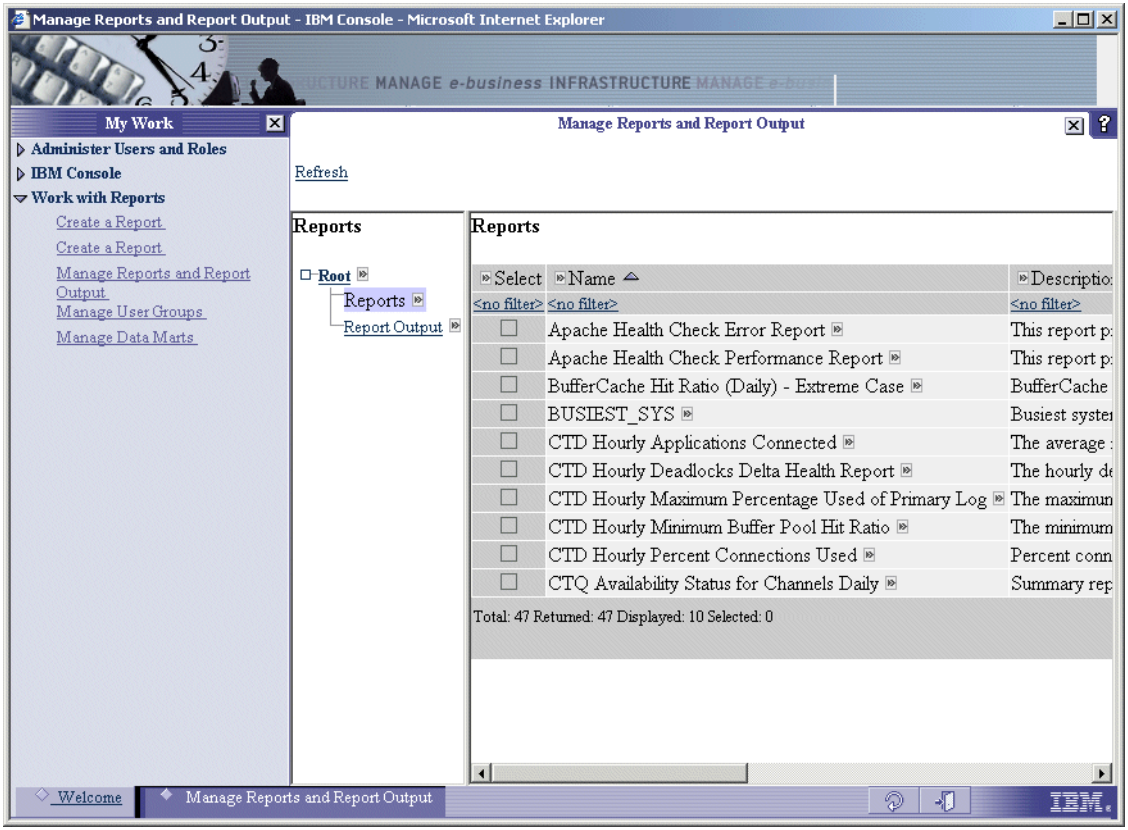


Figure 6-30 Manage Reports and Report Output dialog box

6. Click on the small context menu link (>>) and select **Run**. The specified report will be executed and displayed. An example report is shown in Figure 6-31 on page 149.

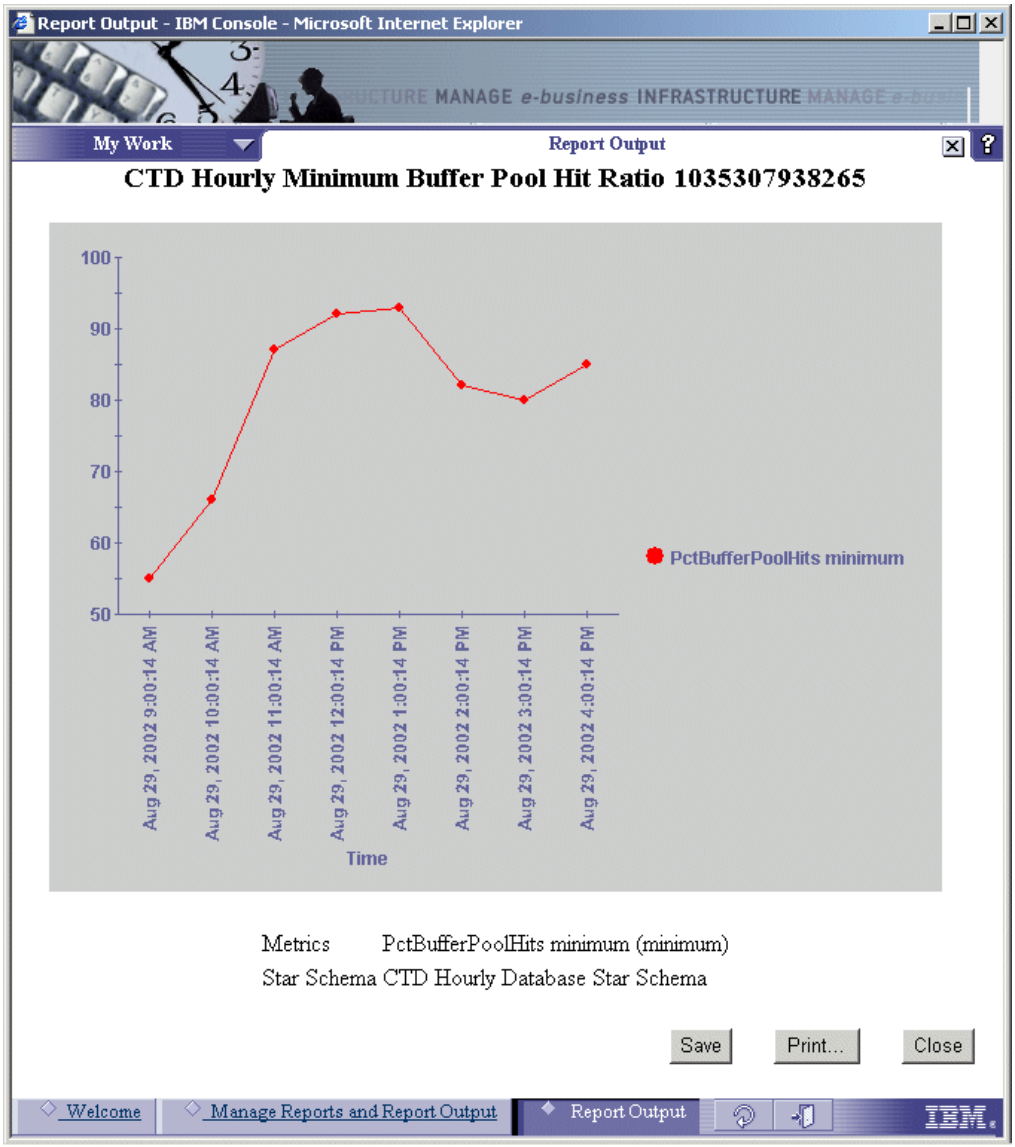


Figure 6-31 Sample report output

7

Log files and troubleshooting

This chapter provides an in-depth view of the IBM Tivoli Monitoring for Databases product structure as well as providing some tips on troubleshooting and problem determination. The discussion in this chapter is provided in the following sections:

- 7.1, “In-depth product structure” on page 152 shows a conceptual structure of the product.
- 7.2, “Debugging procedures and information” on page 156 gives some sample procedures on solving a certain problem with IBM Tivoli Monitoring for Databases.

7.1 In-depth product structure

In this section, we will explain the product structure to help you perform problem determination. Understanding of the structure is critical in finding where things are broken and possibly fixing them. The IBM Tivoli Monitoring for Databases is based largely on the IBM Tivoli Monitoring product structure.

The structure of the product at the endpoint is shown in Figure 7-1.

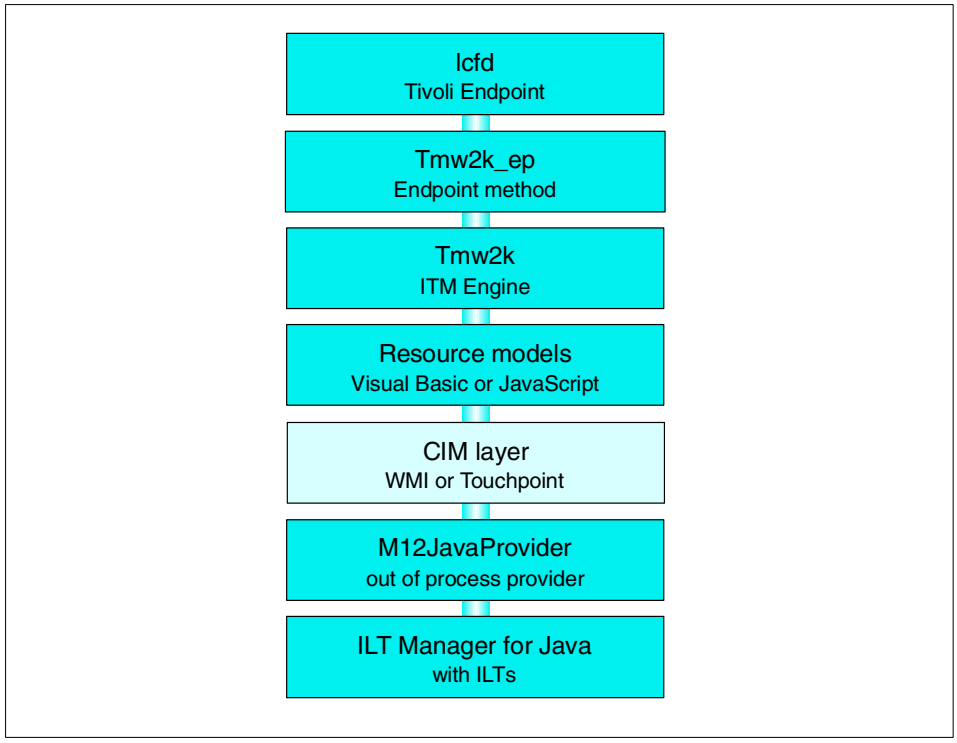


Figure 7-1 End-point architecture for IBM Tivoli Monitoring for Databases: DB2

From Figure 7-1, we can see:

- The core structure of the IBM Tivoli Monitoring product uses the Common Information Model (CIM) management structure. In the Windows platform, this is implemented using Windows Management Interface (WMI), which is installed by default in Windows 2000 and above. In other platforms, Tivoli creates a CIM-like layer that is written in pure Java that provides the CIM services; this layer is called Touchpoint.
- The monitoring engine is called tmw2k (based on the Tivoli Manager for Windows 2000), which can be started and stopped by the Tivoli end-point

code. The tmw2k engine uses resource models that are downloaded from the gateway when a Tmw2kProfile is distributed.

- The resource model is generally consists of:
  - mof files
  - Code written in Java scripts for the IBM Tivoli Monitoring for Databases
  - jar files for the Java monitors
- The resource model extracts data using an out-of-process data provider to the Windows management infrastructure. For a Java based provider, Tivoli uses a program called M12JavaProvider. This process will interact with the actual resources to provide data for specific metric that the resource model requires. Some of this data is acquired directly from the Java program; some runs a command line shell program or other mechanism.
- The Instrumentation Library Types (ILTs) are processes that actually access the resources for monitoring metrics. There are several types of ILTs that Tivoli provides.

7.1.1 IBM Tivoli Monitoring for Databases: DB2

This section shows detailed structure of the IBM Tivoli Monitoring for Databases: DB2 product at the endpoint. The IBM Tivoli Monitoring for Databases: DB2 structure is shown in Figure 7-2.

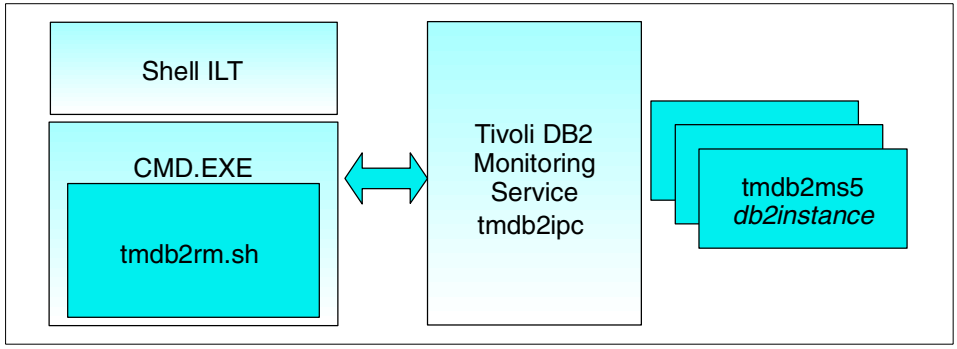


Figure 7-2 IBM Tivoli Monitoring for Databases: DB2 end-point processes

The ILT shell periodically invokes a command shell to launch tmdb2rm.sh. This program executes tmdb2mc to talk with the Tivoli DB2 Monitoring Service (which is the program called tmdb2ipc). The monitoring itself is performed by the tmdb2ms5 programs. There is a tmdb2ms5 running for each DB2 database instance.

We can actually used the tmdb2mc to manually query a specific information about the database. The following is the procedure to do that:

1. Establish the Tivoli endpoint environment by executing the **lcf_env** command. In AIX, the command is in /opt/Tivoli/lcf/dat/1/lcf_env.sh; In Windows, it is in Program Files\Tivoli\lcf\dat\1\lcf_env.cmd.
2. Define an environment variable called AMS_INSTANCE_CLASS, which represents the DB2 object class name, for example, DB2DatabaseManager
3. Define an environment variable called AMS_INSTANCE_OID, which is the object ID of the DB2 object that you want to monitor. This can be queried using the **wlookup** command, similar to Example 7-1. As we will monitor the database called SAMPLE, we set AMS_INSTANCE_OID as 2035659666.1.1097.

Example 7-1 Getting object ID for DB2 resources

```
# wlookup -ar DB2DatabaseManager
MDIST2@DB2@jakarta      2035659666.1.1095#DB2DatabaseManager#
SAMPLE@DB2@jakarta      2035659666.1.1097#DB2DatabaseManager#
T1@ti7031@capecod       2035659666.1.1098#DB2DatabaseManager#
TEC@DB2@jakarta 2035659666.1.1096#DB2DatabaseManager#
```

4. From the $LCF_BINDIR/../AMS/DB2ECC/monitoring, we can run the **tmdb2mc** command. It has the following syntax:

   tmdb2mc *key_list value1 value2 ...*

   where:

   key_list A list of parameters to pass
   value1 value2 Values for the parameters that are passed

   For getting a monitor value, the key must contain at least MonName, MonType, and MonForDB2Version. MonType and MonForDB2Version are 2. As an example, we want to get the value of number of deadlocks, so we issue the following command:

   ```
   tmdb2mc MonName, MonType and MonForDB2Version db_deadlocks 2 2
   ```

   For more information on the list of monitor names, refer to “DB2 metrics for tmdb2mc command” on page 162.

7.1.2 IBM Tivoli Monitoring for Databases: Oracle

The IBM Tivoli Monitoring for Databases: Oracle uses a configuration that is called DBMX Simple Method Framework (DSMF). DSMF governs the long running processes in the endpoints. The monitoring part of the IBM Tivoli Monitoring for Databases: Oracle uses a set of Java classes that executes SQL commands through the JDBC connector for the Oracle server.

The configuration of an IBM Tivoli Monitoring for Databases: Oracle endpoint is shown in Figure 7-3.

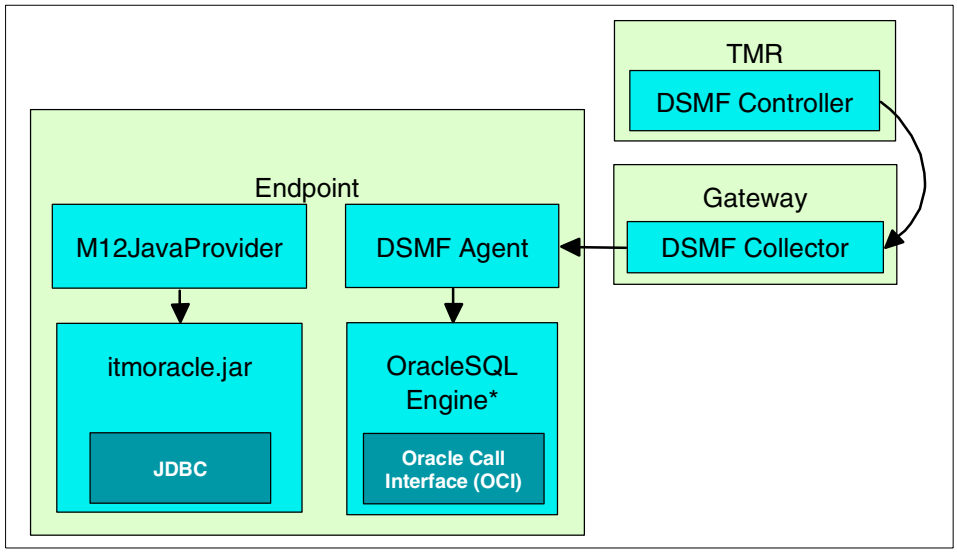


Figure 7-3 DSMF configuration

The supplied Java classes are called by the M12JavaProvider to gather specific Oracle metrics. They were stored in $LCF_DATDIR/LCFNEW/ITM/PACS/Oracle/itmoracle.jar. Other functions go through the DSMF, such as task execution. The DSMF has four components:

- DSMF Controller, which resides at the TMR server.
- DSMF Collector, which resides at the Gateway and acts as the intermediary between the controller and agent.
- DSMF Agent, which resides at the endpoint and perform actions requested from the controller; the agent also monitors DSMF clients.
- DSMF clients are interface specification, which are implemented by the OracleSQLEngine process. The Oracle SQL Engine connects to the Oracle database using the Oracle Call Interface (OCI).

7.1.3 IBM Tivoli Monitoring for Databases: Informix

IBM Tivoli Monitoring for Databases: Informix structure at the endpoint is shown in Figure 7-4 on page 156. It is simpler than its DB2 and Oracle counterparts.

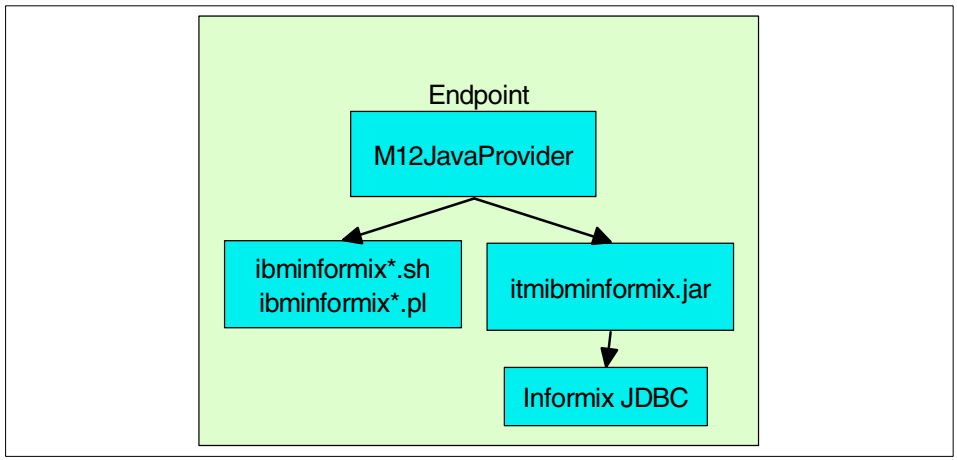


Figure 7-4 IBM Tivoli Monitoring for Databases: Informix end-point structure

The monitoring functions of the IBM Tivoli Monitoring for Databases: Informix is performed under the M12JavaProvider with the assistance of a set of scripts that are stored at $LCF_BINDIR/generic/AMS/IBMINFORMIX and the Java classes collected in $LCF_DATDIR/LCFNEW/ITM/PACS/IBMInformix/itmibminformix.jar.

7.2 Debugging procedures and information

There are several log files that you may need to look at in the case of a problem. The discussion is covered in:

- 7.2.1, “Distribution and connection problems” on page 156
- 7.2.2, “Problems on the resource models and monitoring engine” on page 157
- 7.2.3, “Problems on the monitoring value and information” on page 159
- 7.2.4, “Problems with Tivoli Enterprise Data Warehouse collection” on page 160

7.2.1 Distribution and connection problems

IBM Tivoli Monitoring for Databases profiles are IBM Tivoli Monitoring profiles, which are also known as Tmw2kProfile. In IBM Tivoli Monitoring, profiles are distributed using the MDist2 capability. For a complete discussion on MDist2 debugging, refer to *Tivoli Software Distribution 4.1: New Features and Scenarios*, SG24-6045. We will cover some basic procedures and logs here.

You can perform problem determination, either using a command line or from the MDist2 GUI. For a command line, the **wmdist** command will provide all the necessary distribution information, such as:

- Showing the distributions and their status:

```
wmdist -l
```

- Showing the repeater hierarchy:

```
wmdist -q <dist-id>
```

- Showing the endpoint status:

```
wmdist -e
```

- Showing and controlling the debugging level:

```
wmdist -D
```

When you see that something has gone wrong, you can refer to the following log files:

- On the gateway: $DBDIR/gatelog
- On the repeater: $DBDIR/rpt2log
- On the TMR server: $DBDIR/distmgr.log

7.2.2 Problems on the resource models and monitoring engine

Resource models are run on the endpoint based on the IBM Tivoli Monitoring engine. For a complete discussion of debugging and logging for IBM Tivoli Monitoring, refer to *IBM Tivoli Monitoring Version 5.1: Advanced Resource Monitoring*, SG24-5519.

The resource model is distributed as a zip file, which actually has the format of tar.gz. It contains several files:

- A JavaScript (.js) file that contains the logic of the initialization and operation of the model.
- One or more mof file(s) that contains the CIM definition for the resource model.
- A configuration (.conf) file that contains resource model information.
- A catalog (.cat) file that contains the textual part of the resource model, such as messages or help.

The resource model and most of the IBM Tivoli Monitoring components are located in the $LCF_DATDIR/LCFNEW/Tmw2k. A sample directory tree for a Windows endpoint is shown in Figure 7-5 on page 158.

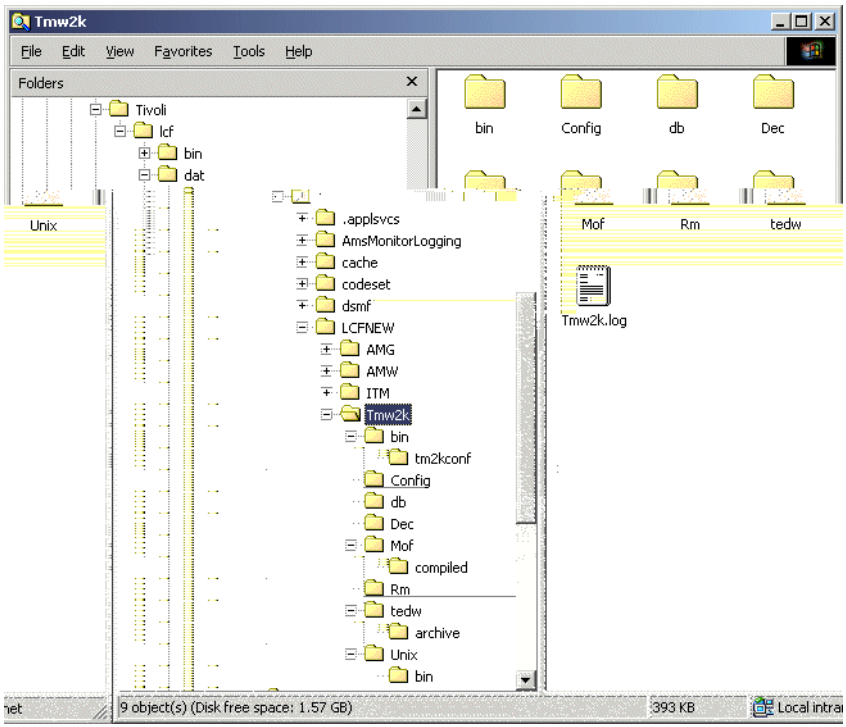


Figure 7-5 Tmw2k directory structure

As the sub-directory names indicates, the mof files are stored under the mof directory and they are compiled before they are loaded. The JavaScript files are stored under the dec directory. The Rm directory contains the complete zip files (which is in the format of tar.gz).

The IBM Tivoli Monitoring at the endpoint has the following important log files:

- For Windows platform, most information exists in the Tmw2k.log file, which is stored in $LCF_DATDIR/LCFNEW/Tmw2k, as shown in Figure 7-5. Other log files are in $LCF_DATDIR/LCFNEW/AMW/logs, which contains specific trace and messages from the ILT Manager.
- For the UNIX platform, the CIM implementation is provided by Tivoli using a pure Java architecture. This affects the logging structure. These log files are:
  - $LCF_DATDIR/LCFNEW/AMW/logs/msg_dmxengine.log
  - $LCF_DATDIR/LCFNEW/AMW/logs/trace_dmxengine.log
  - $LCF_DATDIR/LCFNEW/AMW/logs/trace_dmxeu.log
  - $LCF_DATDIR/LCFNEW/AMW/logs/trace_dmxntv.log

– $LCF_DATDIR/LCFNEW/Tmw2k/Unix/Tmx4j_1.log

Logging on the endpoint is determined by the **wdmtrceng** command. You can specify the log file name, logging level, and the log file size. The syntax of the command is:

```
wdmtrceng -e <endpoint> <logfilename> <loglevel> <size>
```

For example:

```
wdmtrceng -e jakarta C:/jakarta.log 3 10000000
```

The example command modifies the logging in the jakarta endpoint. The new log will be called jakarta.log in the C drive, the logging level will be 3, which is most detail, and the log file size is 10 MB (approximately).

Note: The filename option only works for the Tmw2k.log file for a Windows endpoint. It has no effect on a UNIX endpoint.

The IBM Tivoli Monitoring Component Services has its own set of logs and traces that are located in $LCF_DATDIR/LCFNEW/AMG/logs. The default setting is that you have three message log files and three trace files. These are called:

- msg_itmcs1.log, msg_itmcs2.log, and msg_itmcs3.log
- trc_itmcs1.log, trc_itmcs2.log, and trc_itmcs3.log

The message and trace logging are controlled by the file logging.properties in $LCF_DATDIR/LCFNEW/AMG/logs.

7.2.3 Problems on the monitoring value and information

The monitoring value and information are generated by the ILT and other processes below the CIM layer, as shown in Figure 7-1 on page 152. This processes are different for each modules. The discussion in this section will be separated by modules.

IBM Tivoli Monitoring for Databases: DB2

There are no specific logs provided by the Shell ILT interface. However, as explained in 7.1.1, “IBM Tivoli Monitoring for Databases: DB2” on page 153, you can use the **tmdb2mc** commands to inquire about specific monitoring metrics from the product. Some metrics may require the basic information, such as the monitoring name and monitored object, while others may require additional information, such as table name or tablespace name.

IBM Tivoli Monitoring for Databases: Oracle

The DSMF facility is traced in the dsmf.log file, which resides in $LCF_DATDIR/dsmf. The logging and other behavior of DSMF is set in the dsmf.cfg file. More detail on the dsmf setting in the dsmf.cfg file is provided in *IBM Tivoli Monitoring for Databases Oracle User's Guide Version 5.1.0*, SC23-4723.

IBM Tivoli Monitoring for Databases: Informix

The tasks that the IBM Tivoli Monitoring for Databases: Informix runs provides some information logged under the $LCF_DATDIR/CTR/logs directory. Other information is located in the ILT logging in $LCF_DATDIR/LCFNEW/AMW/logs and Tmw2k.log file.

7.2.4 Problems with Tivoli Enterprise Data Warehouse collection

The TEDW collection is performed from the endpoint by the data collector on a managed node. The data collector gets the monitoring value metric from the endpoint at every interval you specified in the **wdmcollect** command.

The collected metrics are then temporarily cached in the managed node in the $DBDIR/dmml/tedw/<endpoint> directory. The file is a zip file that contains the metrics information in XML format. Once you issue the **wdmcollect** command, after the interval expired, you can check whether the file collected existed. The data is loaded to the RIM object every day at midnight.

There are several trace and log files that relate to the data collector process, which exists under $DBDIR/AMW/logs/. They are:

- msg_DataCollector.log
- trace_tmnt_datacollector_eng1.log
- trace_tmnt_hb_eng1.log
- trace_tmnt_profile_core1.log
- trace_tmnt_rimh_eng1.log
- trace_tmnt_rm_eng1.log
- trace_tmnt_task_eng1.log

The message file contains operational messages, while the trace files contain error messages. Each function of the data collector has its own trace file. These functions are the main data collector engine, heartbeat engine, rim interface, resource model engine, and task execution engine. For TEDW interface, the important files are trace_tmnt_rimh_eng1.log and msg_DataCollector.log.

A

Reference materials

This appendix contains reference materials related to this redbook. This appendix contains the following:

- “DB2 metrics for tmdb2mc command” on page 162

DB2 metrics for `tmdb2mc` command

These are the monitoring metrics available with the **`tmdb2mc`** command:

```
dbms_agents_reg                 dbms_agents_waiting             dbms_hwm_agents_reg
dbms_hwm_agents_waiting         dbms_pct_agents_wait            dbms_comm_private_mem
dbms_con_local_dbases           dbms_db2start_time              dbms_idle_agents
dbms_last_reset                 dbms_local_cons                 dbms_local_cons_in_exec
dbms_piped_sorts_accepted       dbms_piped_sorts_requested      dbms_rem_cons_in
dbms_rem_cons_in_exec           dbms_sort_heap_allocated        dbms_piped_sort_hit_ratio
dbms_pct_sort_heap_alloc        dbms_post_thresh_sorts          dbms_pct_pip_srts_rej
dbms_pip_srts_rej               dbms_agentpri                   dbms_aslheapsz
dbms_max_agents                 dbms_maxcagents                 dbms_mon_heap_sz
dbms_rqrioblk                   dbms_sheapthres                 dbms_sqlstmtsz
dbmsPctPrivMemUsed              dbmsMinCommPrivMem              dbmsTotalIdleAgents
dbmsAgentCreationRatio          dbmsPctConnectionsExec          dbmsStolenAgents
dbmsFcmNumAnchors               dbmsFcmNumBuffers               dbmsFcmNumConnect
dbmsFcmNumRqb                   dbmsMaxcoordagents              dbmsMaxcagents
dbmsMaxIdlePoolAgents           dbmsNumPoolAgents               dbmsQueryHeapSz
dbpTotFcmNodes                  dbpConnStatus                   dbpFreeFcmBuf
dbpFcmConnFree                  dbpFcmReqBlkFree                dbpMinFcmBufFree
dbpMinFcmConnEntries            dbpMinFcmMsgAnchors             dbpMinFcmReqBlks
dbpPctFcmBufUsed                dbpPctFcmConnEntriesUsed        dbpPctFcmReqBlkUsed
dbpPctMaxFcmBufUsed             dbpPctMaxFcmConnEntriesUsed     dbpPctMaxFcmMsgAnchorsUsed
dbpPctMaxFcmReqBlksUsed         dbpTotBufReceived               dbpTotBufSent
dbpg_percent_diff_row_dist      dbpg_redist_status              dbpg_nodegrp_redist_status
dbpg_nodegrps_redist_status     dbpg_node_status                db_list_indoubt_trans_num
db_appls_in_db2                 db_appls_in_lkwt                db_pool_async_data_reads
db_pool_async_dat_rd_rq         db_pool_async_data_writes       db_pool_async_index_writes
db_pool_async_read_time         db_pool_async_write_time        db_pool_data_writes
db_pl_drty_pg_steal_cln         db_pl_drty_pg_thrsh_cln         db_pool_index_writes
db_pool_data_l_reads            db_pool_index_l_reads           db_pool_lsn_gap_clns
db_pool_data_p_reads            db_pool_index_p_reads           db_cat_cache_heap_full
db_cat_cache_inserts            db_cat_cache_lookups            db_cat_cache_overflows
db_deadlocks                    db_deadlocks_delta              db_direct_read_time
db_direct_write_time            db_files_closed                 db_last_backup
db_last_reset                   db_lock_escalations             db_lock_escalations_delta
db_locks_held                   db_lock_timeouts                db_lock_timeouts_delta
db_log_reads                    db_log_writes                   db_heap_top
db_connections_top              db_sec_log_used_top             db_tot_log_used_top
db_appls_cur_cons               db_int_auto_rebinds             db_binds_precompiles
db_commit_sql_stmts             db_ddl_sql_stmts                db_direct_read_reqs
db_direct_write_reqs            db_dynamic_sql_stmts            db_failed_sql_stmts
db_int_commits                  db_int_rollbacks                db_int_rows_deleted
db_int_rows_inserted            db_int_rows_updated             db_int_deadlock_rollbacks
db_rollback_sql_stmts           db_rows_deleted                 db_rows_inserted
db_rows_selected                db_rows_updated                 db_sec_logs_allocated
db_select_sql_stmts             db_sort_overflows               db_active_sorts
db_st_sql_stmts                 db_uid_sql_stmts                db_pkg_cache_inserts
db_pkg_cache_lookups            db_direct_reads                 db_direct_writes
db_sort_heap_allocated          db_lock_list_in_use             db_total_sorts
db_total_sort_time              db_x_lock_escals                db_lock_wait_time
db_total_cons                   db_lock_waits                   db_total_read_time
```

```
db_total_write_time           db_lock_waits_delta           db_pool_hit_ratio
db_avg_lock_wait_time         db_locks_held_per_appl        db_pct_appls_in_lck_wt
db_pct_dlck_rollbacks         db_pct_failed_sql_stmts       db_pct_select_sql_stmts
db_pct_uid_sql_stmts          db_pct_ddl_sql_stmts          db_pool_io_rate
db_commit_rate                db_rollback_rate              db_sql_stmt_rate
db_log_io                     db_pkg_hit_ratio              db_avg_lock_escal_per_conn
db_avg_pool_read_time         db_avg_pool_write_time        db_avg_pool_io_time
db_avg_direct_read_time       db_avg_direct_write_time      db_avg_sync_read_time
db_avg_sync_io_time           db_avg_write_time             db_avg_sync_write_time
db_cat_cache_hit_ratio        db_indx_pl_hit_ratio          db_page_cleans
db_pages_per_clean            db_pages_per_prefetch         db_sync_data_reads
db_sync_data_writes           db_sync_read_time             db_sync_write_time
db_total_pool_io_time         db_total_sync_ios             db_total_sync_io_time
db_total_direct_io_time       db_total_pool_reads           db_total_pool_writes
db_avg_sort_time              db_pct_sort_ovflowed          dbAvgAsyncReadsPerTotReads
dbAvgAsyncWritesPerTotWrites  dbAvgSectorsReadPerDirectRead dbAvgSectorsWritePerDirectWri
                                                            te
dbAvgPlWritesPerPlReads       dbPctConnUsed                 dbPctHeapSpaceUsed
dbPctIntRbPerIntDeadlk        dbPctUsedInPriLog             dbPctUsedInSecLog
dbPriLogSpaceAlloc            dbPctLocklistUsed             dbMaxPriLogSpaceUsed
dbAgentsAssoWithAppls         dbMaxAgentsAssoWithAppls      dbMaxCoordAgent
dbSyncIndexReads              dbSyncIndexWrites             dbDataPagesCopiedfromExtended
                                                            Storage
dbDataPagesCopiedtoExtendedSt dbIndexPagesCopiedfromExtende dbIndexPagesCopiedtoExtendedS
orage                         dStorage                      torage
dbPoolAsyncIndexReads         dbTimeWaitedforPreFetch       dbExtendedStorageReadWriteRat
                                                            io
db_avg_appls                  db_buffpage                   db_catalogcache_sz
db_chngpgs_thresh             db_dbheap                     db_locklist
db_logbufsz                   db_logprimary                 db_maxappls
db_maxlocks                   db_mincommit                  db_newlogpath
db_num_iocleaners             db_num_ioservers              db_pckcachesz
db_seqdetect                  db_sortheap                   dbAppCtlHeapSz
dbApplHeapSz                  dbRestorePending              db_total_db_tbsp
db_total_event_monitor        db_total_inv_trigger          db_total_system_tbsp
db_total_table                db_total_tablespace           db_total_tbsp_long
db_total_trigger              db_total_user_idx             db_total_view
db_dpropr_cap_err             db_dpropr_cap_lag             db_dpropr_cap_pruning
db_dpropr_app_sub_err         db_dpropr_app_sub_status      db_dpropr_app_sub_lagtime
db_dpropr_app_sub_totlag      db_dpropr_app_sub_fullref     db_dpropr_app_active
db_dpropr_heartbeat           conn_direct_reads             conn_direct_writes
conn_indx_pl_hit_ratio        conn_pkg_cache_hit_ratio      conn_pool_index_p_reads
conn_pool_index_writes        conn_pool_data_l_reads        conn_pool_data_writes
conn_pool_read_time           conn_pool_write_time          conn_pool_hit_ratio
conn_total_pool_io_time       conn_deadlocks_delta          conn_locks_held
conn_lock_escals_delta        conn_lock_waits               conn_lock_wait_time_delta
conn_uow_lock_wait_time       conn_sorts                    conn_sort_time
conn_stmt_sorts               conn_commits                  conn_commit_sql_stmts
conn_ddl_sql_stmts            conn_dynamic_sql_stmts        conn_failed_sql_stmts
conn_open_cursors             conn_open_blk_cursors         conn_rollbacks
conn_rollback_sql_stmts       conn_rows_deleted             conn_rows_inserted
conn_rows_selected            conn_rows_updated             conn_select_sql_stmts
conn_static_sql_stmts         conn_uid_sql_stmts            connAvgSectorsReadPerDirectRe
                                                            ad
```

```
connAvgSectorsWritePerDirectW connPctLocklistUsed          connPctDdlSql
rite
connPctUidSql                 connApplWorkLoad             connMaxAssoAgents
connStolenAgents              connSystemCPUTime            connUserCPUTime
connApplSectionInserts        connApplSectionLookups       connBindsPrecompAttempted
connSQLReqSinceLastCommit     connTimeWaitedforPreFetch    tbspHeader_last_reset
tbsp_avg_pool_read_time       tbsp_avg_pool_write_time     tbsp_avg_pool_io_time
tbsp_avg_direct_read_time     tbsp_avg_direct_write_time   tbsp_avg_io_time
tbsp_avg_sync_io_time         tbsp_avg_sync_read_time      tbsp_avg_sync_write_time
tbsp_direct_reads             tbsp_direct_read_reqs        tbsp_direct_read_time
tbsp_direct_writes            tbsp_direct_write_reqs       tbsp_direct_write_time
tbsp_files_closed             tbsp_indx_pl_hit_ratio       tbsp_pct_total_io
tbsp_pool_async_data_reads    tbsp_pool_async_read_time    tbsp_pool_async_data_writes
tbsp_pool_async_write_time    tbsp_pool_async_index_writes tbsp_pool_data_l_reads
tbsp_pool_data_p_reads        tbsp_pool_data_writes        tbsp_pool_hit_ratio
tbsp_pool_index_l_reads       tbsp_pool_index_p_reads      tbsp_pool_index_writes
tbsp_pool_io_rate             tbsp_pl_async_dat_rd_rq      tbsp_pct_asy_dat_rd_rq_s
tbsp_sync_data_reads          tbsp_sync_data_writes        tbsp_sync_read_time
tbsp_sync_write_time          tbsp_total_sync_ios          tbsp_total_sync_io_time
tbsp_total_read_time          tbsp_total_write_time        tbsp_total_pool_io_time
tbsp_total_direct_io_time     tbsp_total_pool_reads        tbsp_total_pool_writes
tbspAvgSectorsReadPerDirectRe tbspAvgSectorsWritePerDirectW tbspSyncIndexReads
ad                            rite
tbspSyncIndexWrites           tbspDataPagesCopiedfromExtend tbspDataPagesCopiedtoExtended
                              edStorage                    Storage
tbspIndexPagesCopiedfromExten tbspIndexPagesCopiedtoExtende tbspPoolAsyncIndexReads
dedStorage                    dStorage
tbspExtendedStorageReadWriteR tbsp_total_table             table_reorgchk_one
atio
table_overflow_acc            table_rows_read              table_rows_written
tableHeader_last_reset        table_total_row              user_total_table
user_defined_sql              db_invalid_pkg_all           db_invalid_pkg_user
db_invalid_pkg_system         tbspPctSpaceUsedDMS          tbspSpaceUsedSMS
tbspStatus                    dbmsConnectAttemptedConnectio dbConnectAttemptedConnections
                              ns
dbCurrentGatewayConnections   dbmsConnectionsWaitingForHost dbConnectionsWaitingForHostRe
                              Reply                        ply
dbmsConnectionsWaitingForClie dbConnectionsWaitingForClient dbSQLStatmentsAttempted
ntReq                         Req
dbOutboundBytesSent           dbOutboundBytesReceived      dbHostThroughput
dbElapsedTimePerStatement     dbNetworkTimePerStatement    dbHostResponsePerStatement
dbMostRecentConnectTime       dbConnectionElapsedTime      dbNumConnectionErrors
dbStatementElapsedTime
```

B

Patterns for e-business

This appendix discusses the patterns for e-business and their applicability to systems management. The discussion consists of:

Patterns for e-business and patterns approaches

The IBM Patterns for e-business are a collective set of proven architectures that have been compiled from more than 20,000 successful Internet-based engagements. This repository of assets can be used by companies to facilitate the development of Web-based applications. They help an organization understand and analyze complex business problems and break them down into smaller, more manageable functions that can then be implemented using low-level design patterns.

Introduction to Patterns for e-business

As companies compete in the e-business marketplace, they find that they must re-evaluate their business processes and applications so that their technology is not limited by time, space, organizational boundaries, or territorial borders. They must consider the time it takes to implement the solution, as well as the resources (people, money, and time) they have at their disposal to successfully execute the solution. These challenges, coupled with the integration issues of existing legacy systems and the pressure to deliver consistent high-quality service, present a significant undertaking when developing an e-business solution.

In an effort to alleviate the tasks involved in defining an e-business solution, IBM has built a repository of “patterns” to simplify the effort. In simple terms, a “pattern” can be defined as a model or plan used as a guide in making things. As such, patterns serve to facilitate the development and production of things. Patterns codify the repeatable experience and knowledge of people who have performed similar tasks before. Patterns not only document solutions to common problems, but also point out pitfalls that should be avoided. IBM’s Patterns for e-business consists of documented architectural best practices. They define a comprehensive framework of guidelines and techniques that were actually used in creating architectures for customer engagements. The Patterns for e-business bridge the business and IT gap by defining architectural patterns at various levels, from Business patterns to Application patterns to Runtime patterns, enabling easy navigation from one level to the next. Each of the patterns (Business, Integration, Application, and Runtime) help companies understand the true scope of their development project and provide the necessary tools to facilitate the application development process, thereby allowing companies to shorten time to market, reduce risk, and more importantly, realize a more significant return on investment.

The core types of Patterns for e-business are:

- Business patterns
- Integration patterns
- Composite patterns
- Application patterns
- Runtime patterns and Product matching mappings

When a company takes advantage of these documented assets, they are able to reduce the time and risk involved in completing a project.

For example, a line-of-business (LOB) executive who understands the business aspects and requirements of a solution can use Business patterns to develop a high-level structure for a solution. Business patterns represent common business problems. A LOB executive can match their requirements (IT and business drivers) to Business patterns that have already been documented. The patterns provide tangible solutions to the most frequently encountered business challenges by identifying common interactions among users, business, and data.

Senior technical executives can utilize Application patterns to make critical decisions related to the structure and architecture of the proposed solution. Application patterns help refine Business patterns so that they can be implemented as computer-based solutions. Technical executives can use these patterns to identify and describe the high-level logical components that are needed to implement the key functions identified in a Business pattern. Each Application pattern would describe the structure (tiers of the application), placement of the data, and the integration (loosely or tightly coupled) of the systems involved.

Finally, solution architects and systems designers can develop a technical architecture by using Runtime patterns to realize the Application patterns. Runtime patterns describe the logical architecture that is required to implement an Application pattern. Solution architects can match Runtime patterns to existing environmental and business needs. The Runtime pattern they implement, establishes the components needed to support the chosen Application pattern. It defines the logical middleware nodes, their roles, and the interfaces among these nodes in order to meet business requirements. The Runtime pattern documents what must be in place to complete the application but does not specify product brands. Determination of actual products is made in the Product mapping phase of the patterns.

In summary, Patterns for e-business captures e-business approaches that have been tested and proven. By making these approaches available and classifying them into useful categories, LOB executives, planners, architects, and developers can further refine them into useful, tangible guidelines. The patterns and their associated guidelines allow the individual to start with a problem and a

vision, find a conceptual pattern that fits this vision, define the necessary functional pieces that the application will need to succeed, and then actually build the application. Furthermore, the Patterns for e-business provides common terminology from a project's onset and ensures that the application supports business objectives, significantly reducing cost and risk.

The Patterns for e-business layered asset model

The Patterns for e-business approach enables architects to implement successful e-business solutions through the reuse of components and solution elements from proven successful experiences. The Patterns approach is based on a set of layered assets that can be exploited by any existing development methodology. These layered assets are structured in a way that each level of detail builds on the last. These assets include:

- Business patterns that identify the interaction between users, businesses, and data.
- Integration patterns that tie multiple Business patterns together when a solution cannot be provided based on a single Business pattern.
- Composite patterns that represent commonly occurring combinations of Business patterns and Integration patterns.
- Application patterns that provide a conceptual layout describing how the application components and data within a Business pattern or Integration pattern interact.
- Runtime patterns that define the logical middleware structure supporting an Application pattern. Runtime patterns depict the major middleware nodes, their roles, and the interfaces between these nodes.
- Product mappings that identify proven and tested software implementations for each Runtime pattern.
- Best-practice guidelines for design, development, deployment, and management of e-business applications.

These assets and their relation to each other are shown in Figure B-1 on page 169.

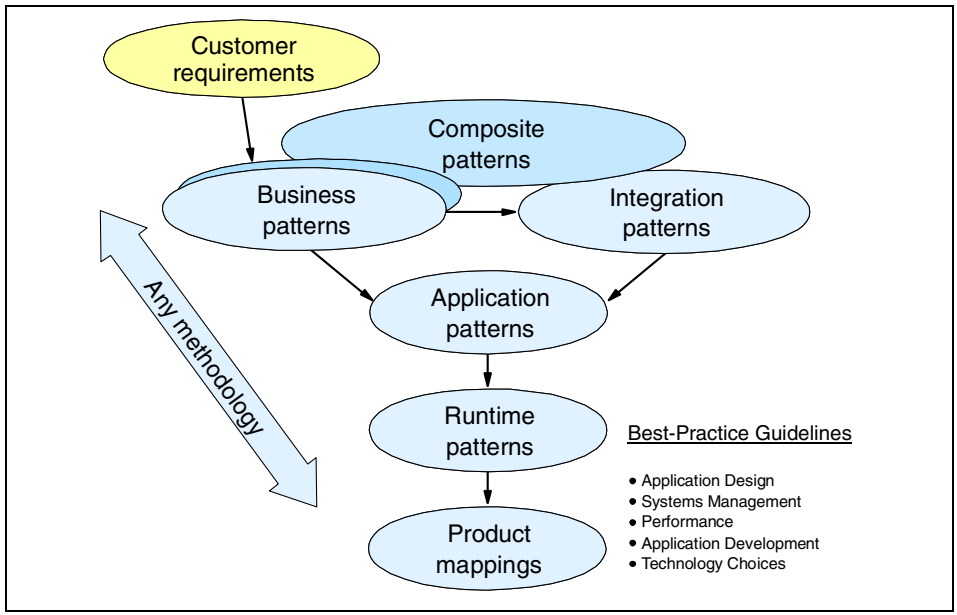


Figure B-1 The Patterns layered asset model

Patterns for e-business Web site

The Patterns Web site provides an easy way of navigating top down through the layered Patterns' assets in order to determine the preferred reusable assets for an engagement.

For easy reference to Patterns for e-business refer to the Patterns for e-business Web site at `http://www.ibm.com/developerWorks/patterns/`.

How to use the Patterns for e-business

As described in the last section, the Patterns for e-business are structured in a way that each level of detail builds on the last. At the highest level are Business patterns that describe the entities involved in the e-business solution. A Business pattern describes the relationship between the users, the business organization or applications, and the data to be accessed.

Composite patterns appear in the hierarchy (shown in Figure B-1) above the Business patterns. However, Composite patterns are made up of a number of individual Business patterns, and at least one Integration pattern. In this section, we will discuss how to use the layered structure of the Patterns for e-business assets.

There are four primary Business patterns, as shown in Table B-1.

Table B-1 Business patterns

Business patterns	Description	Examples
Self-Service (User-to-Business)	Applications where users interact with a business via the Internet	Simple Web site applications
Information Aggregation (User-to-Data)	Applications where users can extract useful information from large volumes of data, text, images, and so on.	Business intelligence, knowledge management, and Web crawlers
Collaboration (User-to-User)	Applications where the Internet supports collaborative work between users.	E-mail, community, chat, video conferencing, and so on
Extended Enterprise (Business-to-Business)	Applications that link two or more business processes across separate enterprises.	EDI, supply chain management, and so on

It would be very convenient if all problems fitted nicely into the four Business patterns above, but reality says that things will often be more complicated. The patterns assume that all problems, when broken down into their most basic components, will fit more than one of these patterns. When a problem describes multiple objectives that fit into multiple Business patterns, the Patterns for e-business provide the solution in the form of Integration patterns.

Integration patterns allow us to tie together multiple Business patterns to solve a problem. The Integration patterns are listed in Table B-2.

Table B-2 Integration patterns

Integration patterns	Description	Examples
Access Integration	Integration of a number of services through a common entry point	Portals
Application Integration	Integration of multiple applications and data sources without the user directly invoking them	Message brokers and workflow managers

These Business and Integration patterns can be combined to implement installation-specific business solutions. We call this a Custom design.

We can represent the use of a Custom design to address a business problem through an iconic representation, as shown in Figure B-2.

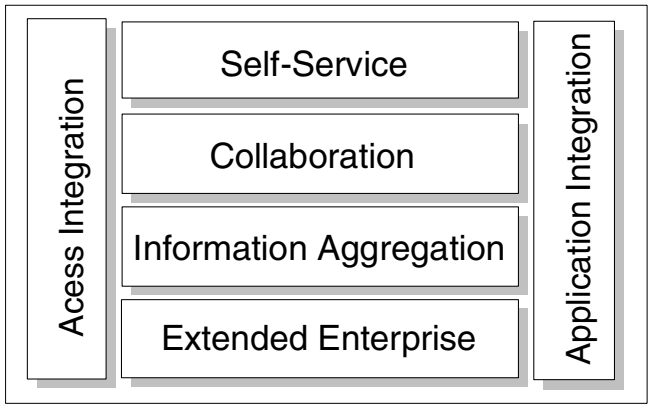


Figure B-2 Pattern Representation of a Custom design

If any of the Business or Integration patterns are not used in a Custom design, we can show that with the blocks lighter than the other blocks. For example, Figure B-3 shows a Custom design that does not have a mandatory Collaboration business pattern or an Extended Enterprise business pattern for a business problem.

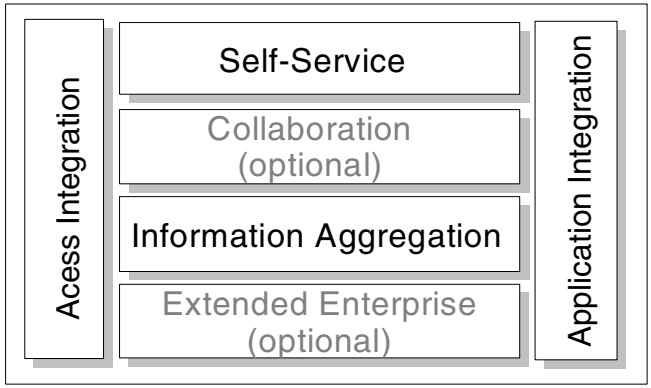


Figure B-3 Custom design with several patterns components

A Custom design may also be a Composite pattern if it recurs many times across domains with similar business problems. For example, the iconic view of a Custom design in Figure B-3 can also describe a Sell-Side Hub composite pattern.

Several common uses of Business and Integration patterns have been identified and formalized into Composite patterns. The identified Composite patterns are as follows shown in Table B-3.

Table B-3 Composite patterns

Composite patterns	Description	Examples
Electronic commerce	User-to-Online-Buying	► `http://www.macys.com` ► `http://www.amazon.com`
Portal	Typically designed to aggregate multiple information sources and applications to provide uniform, seamless, and personalized access for its users.	► Enterprise intranet portal providing self-service functions, such as payroll, benefits, and travel expenses. ► Collaboration providers who provide services such as e-mail or instant messaging.
Account Access	Provide customers with around-the-clock account access to their account information.	► Online brokerage trading applications. ► Telephone company account manager functions. ► Bank, credit card, and insurance company online applications.
Trading Exchange	Allows buyers and sellers to trade goods and services on a public site.	► Buyer's side: Interaction between buyer's procurement system and commerce functions of e-Marketplace. ► Seller's side: Interaction between the procurement functions of the e-Marketplace and its suppliers.
Sell-Side Hub (Supplier)	The seller owns the e-Marketplace and uses it as a vehicle to sell goods and services on the Web.	► `http://www.carmax.com` (car purchase)

Composite patterns	Description	Examples
Buy-Side Hub (Purchaser)	The buyer of the goods owns the e-Marketplace and uses it as a vehicle to leverage the buying or procurement budget in soliciting the best deals for goods and services from prospective sellers across the Web.	► http://www.wre.org (WorldWide Retail Exchange)

The makeup of these patterns is variable in that there will be basic patterns present for each type, but the Composite can easily be extended to meet additional criteria. For more information on Composite patterns, refer to *Patterns for e-business: A Strategy for Reuse by* Jonathan Adams, et al.

Selecting Patterns and product mapping

Once the appropriate Business pattern is identified, the next step is to define the high-level logical components that make up the solution and how these components interact. This is known as the Application pattern. A Business pattern will usually have multiple Application patterns identified that describe the possible logical components and their interactions. For example, an Application pattern may have logical components that describe a presentation tier for interacting with users, a Web application tier, and a back-end application tier.

The Application pattern requires an underpinning of middleware that is expressed as one or more Runtime patterns. Runtime patterns define functional nodes that represent middleware functions that must be performed.

Once a Runtime pattern has been identified, the next logical step is to determine the actual product and platform to use for each node. The Patterns for e-business have Product mappings that correlate to the Runtime patterns, describing actual products that have been used to build an e-business solution for this situation.

Finally, guidelines assist you in creating the application using best practices that have been identified through experience.

For more information on determining how to select each of the layered assets, refer to the Patterns for e-business Web site at:

http://www.ibm.com/developerWorks/patterns/

Abbreviations and acronyms

ACF	Adapter Configuration Facility
AIX	Advanced Interactive Executive
CD-ROM	Compact Disc Read Only Memory
CIM	Common Information Model
CLI	Command Line Interface
CPU	Central Processing Unit
CWM	Common Warehouse Metadata
DB2	Database 2
DBA	Database Administrator
DCS	Distributed Connection Services
DML	Data Manipulation Language
DSMF	DBMX Simple Method Framework
DWC	Data Warehouse Center
EEE	Extended-Enterprise Edition
ERP	Enterprise Resource Planning
ETL	Extract, Transform, and Load
GUI	Graphical User Interface
IBM	International Business Machine Corporation
ILT	Implemenatation Library Task
ITM	IBM Tivoli Monitoring
ITSO	International Technical Support Organization
JDBC	Java Database Connectivity
JRE	Java Runtime Environment
KB	Kilobytes
KIMA	Know Identify Measure Act
LOB	Line of Business
LRU	Least Recently Used
MB	Megabytes
OCI	Oracle Call Interface
ODBC	Open Database Connectivity
OLAP	Online Analytical Processing
PGA	Program Global Area
RDBMS	Relational Database Management Systems
RIM	RDBMS Interface Module
SGA	System Global Area
SIS	Software Installation Services
SQL	Structured Query Language
TBSM	Tivoli Business Systems Manager
TCPIP	Transmission Control Protocol/Internet Protocol
TEC	Tivoli Enterprise Console
TEDW	Tivoli Enterprise Data Warehouse
TMR	Tivoli Management Region
URL	Universal Resource Locator
WMI	Windows Management Interface

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 179.

- *IBM Tivoli Monitoring Version 5.1: Advanced Resource Monitoring*, SG24-5519
- *Introduction to Tivoli Enterprise Data Warehouse*, SG24-6607
- *Tivoli Business Systems Manager A Complete End-to-End Management Solution*, SG24-6202
- *Tivoli Business Systems Manager An Implementation Case Study*, SG24-6032
- *Tivoli Software Distribution 4.1: New Features and Scenarios*, SG24-6045

Other resources

The following are the IBM Tivoli Monitoring for Databases product manuals:

- *IBM Tivoli Monitoring for Databases DB2 Limitations and Workarounds Supplement Version 5.1.0*, SC23-4786
- *IBM Tivoli Monitoring for Databases DB2 Reference Guide Version 5.1.0*, SC23-4727
- *IBM Tivoli Monitoring for Databases DB2 User’s Guide Version 5.1.0*, SC23-4726
- *IBM Tivoli Monitoring for Databases Informix Limitations and Workarounds Supplement Version 5.1.0*, SC23-4787
- *IBM Tivoli Monitoring for Databases Informix Reference Guide Version 5.1.0*, SC23-4728
- *IBM Tivoli Monitoring for Databases Informix User’s Guide Version 5.1.0*, SC23-4729
- *IBM Tivoli Monitoring for Databases Installation and Setup Guide Version 5.1.0*, GC23-4730

- *IBM Tivoli Monitoring for Databases Oracle Reference Guide Version 5.1.0*, SC23-4724
- *IBM Tivoli Monitoring for Databases Oracle User's Guide Version 5.1.0*, SC23-4723
- *IBM Tivoli Monitoring for Databases Oracle User Management Guide Version 5.1.0*, GC23-4731
- *IBM Tivoli Monitoring for Databases Oracle Warehouse Enablement Pack*, SC09-7779
- *IBM Tivoli Monitoring for Databases Release Notes Version 5.1.0*, GI11-0933

These publications are also relevant as further information sources:

- *IBM Tivoli Monitoring User's Guide Version 5.1*, SH19-4569
- *Tivoli Enterprise Data Warehouse Installing and Configuring Version 1.1*, GC32-0744
- *Tivoli Framework 3.7.1 User's Guide,* GC31-8433

Referenced Web sites

These Web sites are also relevant as further information sources:

- Java download pages

  `http://www6.software.ibm.com/dl/dka/priv/dka-h?S_PKG=dka130ww`
  `http://www6.software.ibm.com/dl/wspt/priv/wspt-h?S_PKG=pretechww`
- NetMailBot home page

  `http://www.exclamationsoft.com/exclamationsoft/NetMailBot`
- Object Management Group home page

  `http://www.omg.org`
- DB2 support site

  `http://www-3.ibm.com/software/data/db2/udb/support.html`
- Tivoli patch factory

  `http://www.ibm.com/software/sysmgmt/products/support`
- Pattern Web site

  `http://www.ibm.com/developerWorks/patterns`

How to get IBM Redbooks

You can order hardcopy Redbooks, as well as view, download, or search for Redbooks at the following Web site:

ibm.com/redbooks

You can also download additional materials (code samples or diskette/CD-ROM images) from that site.

IBM Redbooks collections

Redbooks are also available on CD-ROMs. Click the CD-ROMs button on the Redbooks Web site for information about all the CD-ROMs offered, as well as updates and formats.

Index

E

G

H

I

J

K

L

W

IBM Tivoli Monitoring for Databases

Database Management Made Simple

Redbooks

Extensive installation and migration examples

In-depth coverage of resource models and tasks

Integration to TBSM and TEDW

This IBM Redbook will help you install, tailor, and configure the new IBM Tivoli Monitoring for Databases Version 5.1.0 (5724-B96). This product actually consists of the following three modules:

- IBM Tivoli Monitoring for DB2
- IBM Tivoli Monitoring for Informix
- IBM Tivoli Monitoring for Oracle

In this redbook, we show the installation and migration steps for the product, listing and customizing the monitoring feature, and discussing some operational scenarios. We also provide integration of the IBM Tivoli Monitoring for Databases with other Tivoli product to provide higher level of management, such as real time business impact management with Tivoli Business Systems Manager and Tivoli Enterprise Console, and predictive management with Tivoli Enterprise Data Warehouse.

This redbook also discuss some detailed product structure with tips on performing troubleshooting for the IBM Tivoli Monitoring for Databases.

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

SG24-6613-00 ISBN 0738427071