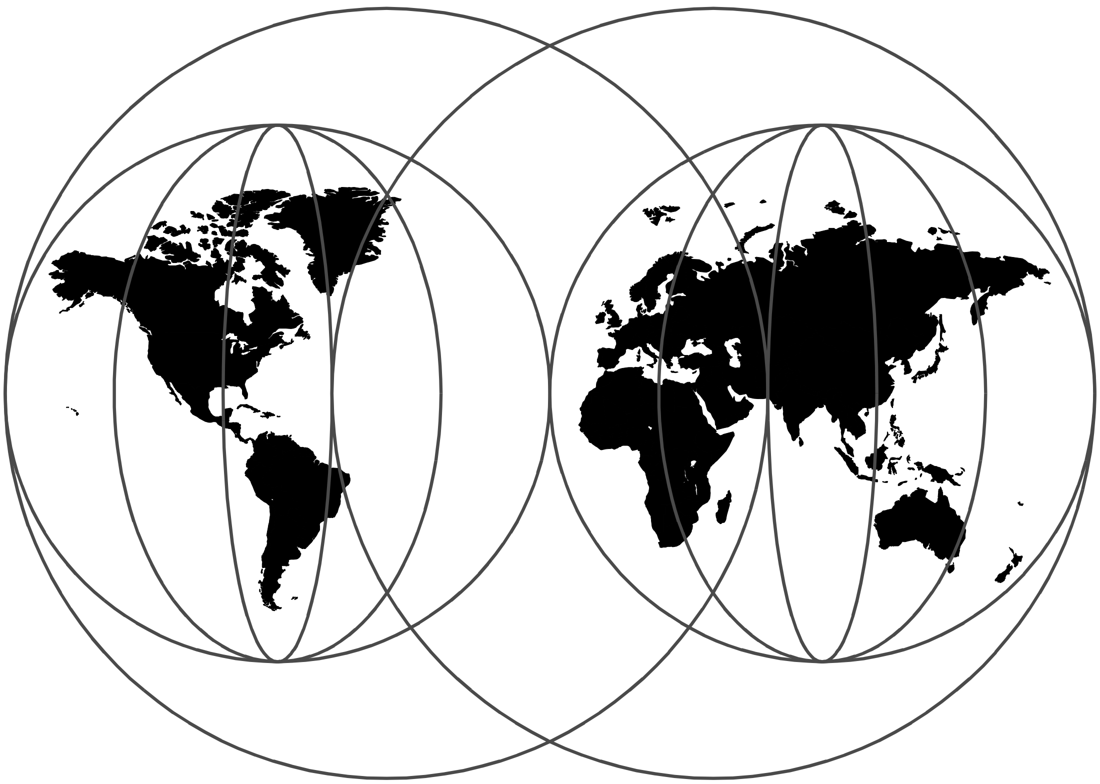
Problem Management Using Tivoli Service Desk and the TEC

Paul Fearn, Stefan Uelpenich, Niklas Haggstrom, Predrag Ludic
Raffaele Pullo, Renata Rossi, Michel Rubira, Juergen Schaefer



International Technical Support Organization

http://www.redbooks.ibm.com

SG24-5301-00

International Technical Support Organization

SG24-5301-00

Problem Management Using Tivoli Service Desk and the TEC

November 1998

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix B, "Special Notices" on page 367.

First Edition (November 1998)

This edition applies to Tivoli Framework Version 3.6, Tivoli Enterprise Console Version 3.6, The Tivoli Service Desk Suite Version 5.0.1 for use with the AIX Version 4.2 and Windows NT Version 4.0 operating systems.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. HZ8 Building 678
P.O. Box 12195
Research Triangle Park, NC 27709-2195

Contents

Figures

Tables

Preface

This redbook introduces the Tivoli Service Desk application suite, the new Tivoli product resulting from the integration of Software Artistry into Tivoli systems. It helps you position Tivoli Service Desk and Tivoli as solutions for enterprise problem management and enterprise systems management.

This book uses practical examples to illustrate the installation and customization tasks required to implement an end-to-end problem management solution using Tivoli Service Desk in combination with other Tivoli products, such as Tivoli Enterprise Console, Tivoli Distributed Monitoring and Tivoli NetView.

This redbook also provides advice on planning to assist professionals who will be designing and implementing the Tivoli Service Desk applications. The book will provide a valuable addition to the product documentation when implementing a solution and a good reference for I/T architects designing problem management solutions.

Knowledge of the Tivoli Framework is assumed.

The Team That Wrote This Redbook

This redbook was produced by a team of specialists from around the world working at the Systems Management and Networking ITSO Center, Raleigh.

Paul Fearn is an Advisory ITSO Representative working as a project leader at the Systems Management and Networking ITSO Center, Raleigh. He writes extensively and teaches IBM classes worldwide on all areas of Tivoli and network management applications. Before joining the ITSO two years ago, Paul worked in the global services division in the UK, working on large customer engagements providing consulting for systems and network management.

Stefan Uelpenich is an Advisory ITSO Representative working as a project leader at the Systems Management and Networking ITSO Center, Raleigh. He applies his extensive field experience as an I/T architect and project leader to his work at the ITSO, where he writes extensively and consults worldwide on all areas of systems management. Before joining the ITSO, Stefan worked in IBM Germany's Professional Services organization as an Advisory I/T Architect for Systems Management, consulting major IBM customers. In this role, he architected the configuration management solution for one of Germany's largest client/server networks.

Niklas Häggström is an I/T Specialist working for IBM's Product Support Services in Stockholm, Sweden. He has been with IBM since 1998 and has two years experience in system management and help desk implementations. Niklas is currently working on a number of Tivoli Management Environment implementations in Sweden. His areas of expertise include availability products of the Tivoli application suite, help desk applications and the integration between these products.

Michel Rubira is an I/T Specialist working for IBM Global Services, Service Delivery in La Gaude, France. He has been with IBM since April 1995 and has three years experience with network and system management. Michel is currently working for strategic outsourcing in France implementing Tivoli solutions. His areas of expertise are the availability products of the Tivoli application suite, Distributed Monitoring and Tivoli Enterprise Console.

Predrag Lucic is an I/T Specialist at IBM Global Services in Slovenia. He has been with IBM since February 1995 and has held various positions as an AIX support professional and product specialist. Predrag is currently a leader of the Tivoli support team in Software Technical Support Center for Central Europe and Russia. His area of experience include CAD/CAM, Tivoli and ADSM.

Renata Rossi is an I/T Specialist at IBM SW Technical Support in Italy. She has been working for IBM for 13 years, worked as second level support in AIX Division, and currently is on the pre-sales team for Tivoli solutions in Italy and the South Europe region as a Tivoli Availability Specialist.

Raffaele Pullo is an I/T Specialist Advisor at IBM Global Services in Italy. He joined IBM in 1990 and worked for six years in the Rome Tivoli Laboratory as a software developer and team leader. Since 1997 he has been working for IBM Global Services in the systems management area. He has several years experience in the problem management field. He holds a degree in Physics from Rome University. Raffaele currently works in defining architecture and implementing Tivoli solutions in the South Europe region.

Juergen Schaefer is an I/T Architect in IBM Global Services Germany. He joined IBM in 1993 and has worked in the systems management area. Since 1997 he has been is part of the Tivoli Core Team. He currently works for the Tivoli Solution Center in Germany providing Tivoli solution designs in EMEA’s Central Region. His areas of expertise are the availability products of the Tivoli application suite, distributed monitoring and the Tivoli Enterprise Console.

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

The Editing Team
International Technical Support Organization, Raleigh Center

Joe Steinfeld and the Tivoli Indianapolis Team

Christine Deweese
Tivoli

Jeff Brantley
Tivoli Austin

Dave Thoenen
IBM Raleigh

Mary Kircher
IBM Raleigh

Comments Welcome

Your comments are important to us!

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

- Fax the evaluation form found in “ITSO Redbook Evaluation” on page 379 to the fax number shown on the form.
- Use the electronic evaluation form found on the Redbooks Web sites:

For Internet users	`http://www.redbooks.ibm.com`
For IBM Intranet users	`http://w3.itso.ibm.com`

- Send us a note at the following address:

  `redbook@us.ibm.com`

Chapter 1. Overview

In this chapter we provide an overview of the Tivoli Service Desk applications, the Tivoli enterprise problem management strategy and the project objectives covered in this redbook.

1.1 Introduction to Tivoli Service Desk

Tivoli Service Desk is the new Tivoli offering for enterprise problem management that is resulting from Tivoli’s acquisition of Software Artistry.

The Tivoli Service Desk products in conjunction with the Tivoli management applications offer a comprehensive enterprise management solution for a support operation and will manage both planned and unplanned events. In the case of a planned event, the Tivoli Service Desk suite enables change policies and procedures to be executed efficiently. In the case of an unplanned event such as a network or application failure, the Tivoli Service Desk products ensure the proper recording, notification and resolution of the event. This is accomplished with the direct integration with many of the other Tivoli framework products such as the NetView product and the Tivoli Enterprise Console.

The version we used for this project was 5.01. The application naming will be changed when Tivoli Service Desk Version 5.02 is released. Throughout this redbook we refer to both the new and old names for the applications.

Tivoli Service Desk comprises of the following applications:

Tivoli Problem Management: Enables an organization to service the IT-related problems of its employees. This application uses problem resolution technologies that enable lower skilled analysts to handle more difficult problems. This reduces the need for highly skilled personnel, thereby reducing support costs.

Tivoli Problem Management automatically and efficiently tracks, logs, and escalates users' interactions or requests and also includes a sophisticated knowledge engine that enables analysts with different levels of experience to quickly resolve many types of problems.

Tivoli Change Management: Automates and enforces policies and best practices for rolling out IT-related technology changes. Users can run an impact analysis of planned changes to determine potential glitches with an IT change. Change management is the process through which alterations to the IT environment are introduced, executed and measured.

Tivoli Asset Management: Manages IT assets from acquisition to disposition. Knowing the history of assets from a financial and historical perspective Tivoli Asset Management helps service desk personnel make informed decisions regarding the support of IT assets.

Tivoli Decision Support: Pro-actively planning future IT changes and determining their impact on the organization Tivoli Decision Support translates transactional data into meaningful knowledge about your IT environment, your service desk, your users, and your relationship with them. Designed to provide the best possible environment for facilitating the decision-making process, Tivoli Decision Support offers a unique, dynamic, and interactive environment through which analysis tools and business intelligence models work together senselessly to shed new light on your data. It is an essential tool that allows you to filter data on a wide variety of criteria to help you make better decisions faster and manage your support operation in the most efficient way.

Tivoli Decision Support Guides: Tivoli Decision Support guides are templates that assist users in selecting which questions to ask and locating the data that will answer these questions. With different guides as front end templates, Tivoli Decision Support becomes specifically tailored to your organization to help you easily explore information.

Knowledge-Pak Desktop Suite: This product provides the most comprehensive collection of knowledge content for today's most popular desktop products with more than 30,000 complete descriptions, causes and solutions to thousands of desktop-related problems, designed specifically for first-level help desk analysts and end users. The Desktop Suite includes products such as Lotus Notes.

Knowledge-Pak Network Suite: This product provides packaged knowledge bases containing solutions to thousands of common network-related problems, covering the most widely used network operating system titles and versions.

Systems and Networking Management Applications: Using the integration with the Tivoli Framework products this solution can manage everything from a user of an application and that user's data, something critical to the user's effectiveness in his or her role. The systems and networking management applications also manages solutions across the application life-cycle so that whether an application was internally developed or purchased off the shelf, the management infrastructure in place is equally as effective.

The management applications covered include:

- Tivoli Framework
- Tivoli Enterprise Console
- Tivoli NetView
- Tivoli Distributed Monitoring
- Tivoli Inventory

1.2 Integration of Tivoli Service Desk with Other Tivoli Applications

Tivoli Service Desk fits in naturally with the Tivoli Management Environment. We show detailed examples in this book of how Tivoli Service Desk and the Tivoli Framework and applications can be integrated to build an enterprise system and problem management solution.

Integration between Tivoli Service Desk and other Tivoli applications takes place on different levels, depending on the part of Tivoli Service Desk integrated.

Figure 1 on page 3 gives an overview of the application integration.

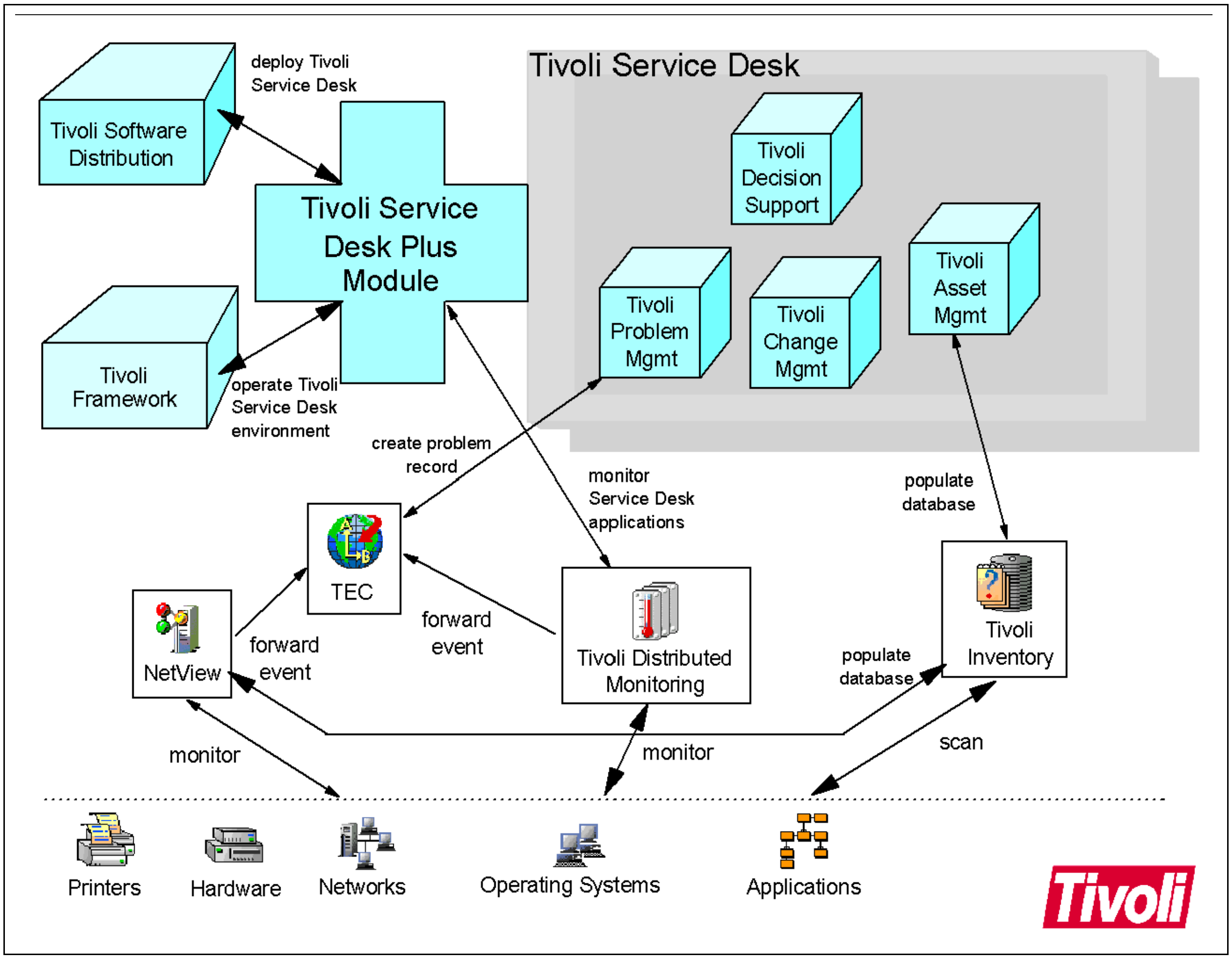


Figure 1. Integration of Tivoli Service Desk Application

For example, Tivoli NetView, Tivoli Distributed Monitoring and Tivoli Enterprise Console typically act as event sources that can feed problems into the Tivoli Service Desk where the problems are acted on by help desk personnel. As Tivoli NetView, Tivoli Enterprise Console and Tivoli Distributed Monitoring are also tightly integrated themselves, the integration of Tivoli Service Desk further strengthens the ability of the user to create a totally integrated problem management solution.

One example is that Tivoli NetView detects a network problem in its typical role as the enterprise SNMP manager. Through the seamless event forwarding integration with TEC the event is then forwarded to TEC where it can be correlated with events from other sources, such as Tivoli Distributed Monitoring.

Events from either Tivoli NetView or Tivoli Enterprise Console can then be used to create and problem record in Tivoli Service Desk. This allows for an end-to-end problem resolution where specific problems can be resolved directly at the source but complex user problems can be forwarded for resolution to the Tivoli Service Desk. For example, the network administrator typically resolves network problems that are reported by NetView, but some of these events can also be useful for resolving more complex problems. A typical impact could be that the network problem caused an application to fail.

The application failure problem can be resolved at the next level of correlation, the TEC, where events from NetView and the applications are correlated. Using TEC correlation rules and automated responses, typical application problems can be resolved more efficiently.

The next level of problem resolution is an end user who calls the help desk to report a problem he or she has when working with the application, which he or she expects a resolution for. The help desk personnel can use the data forwarded from TEC or NetView to help resolve the problem.

Having the ability to correlate on many different levels gives the system designer almost unlimited flexibility in designing the enterprise problem management solution.

The Tivoli Plus module for Tivoli Service Desk that is also described in this book adds further integration between the help desk solution and the Tivoli Framework applications. For example, it allows you to use Tivoli Software Distribution to deploy components of Tivoli Service Desk in the enterprise and monitor the management environment.

Other components of Tivoli Service Desk are integrated with Tivoli Framework applications as well. For example, Tivoli Asset Management is integrated with Tivoli Inventory, so that inventory data discovered by the Tivoli Inventory scanners can be used in Tivoli Asset Management.

1.3 Project Objectives

Our main objective is to show how to install and customize the management applications to address a set of customer requirements and not to provide information on generic problem management methodologies and processes, although we do address some of these processes when demonstrating the functionality of the Tivoli Service Desks applications.

This serves two purposes:

- Introduces the reader to the new Tivoli Service Desk application suite and shows how it can be integrated with other Tivoli applications.
- Shows how to create an enterprise problem management solution using the integrated set of products.

We show an example of end-to-end problem management solutions and document the stages involved in implementation. The remaining parts of this book document our approach, which involved the following steps:

Planning Obtain all the information that is required to provide input to the overall design.

Installation Using the planning information we deploy the management applications and document each specific stage of the installation.

Configuration Setting up and customizing the Tivoli Service Desk applications, including how to populate the databases with the required information relevant to our environment, such as location information.

Integration Shows how to integrate the Tivoli Service Desk applications with TEC and other Tivoli applications to provide the ability to automatically generate problem management calls.

Usage Once we have set up our integrated environment we show examples of how to use the Tivoli Service Desk applications. We show how a help desk user can receive and resolve problem calls using the suite of applications installed.

The next stage is to look at the planning and what steps we took.

Chapter 2. Planning for a Problem Management Solution

This chapter is aimed at providing guidelines to show what type of information is typically required in order to provide a problem management solution.

It is import to perform good planning stages before deploying the management applications. We have highlighted the types of questions that need to be asked and data that needs to be gathered before initiating the implementation stages.

This type of information could for instance be passed from the architects to the implementation professionals. The design and planning steps are divided up into the following:

- Understand the managed environment.
- List informational prerequisites for a problem management implementation and event information for the events potentially generated in the environment.
- Clarify and define terms and concepts such as the mapping of data in order to provide true integration.
- Show the general steps that have to be taken in planning for a problem management implementation.

Note

You should notice that the term Tivoli Service Desk is the new name used for the former Software Artistry products we deal with in this redbook. In the screen captures and descriptions you will often find that we still refer to the original Software Artistry names for the products. We point out the correct matches between old and new names where appropriate.

2.1 An Introduction to the Management Environment

As part of this project we used information from real-life customer scenarios to build our test environment. The overall view of both the management environment and management solution is shown in Figure 2 on page 8.

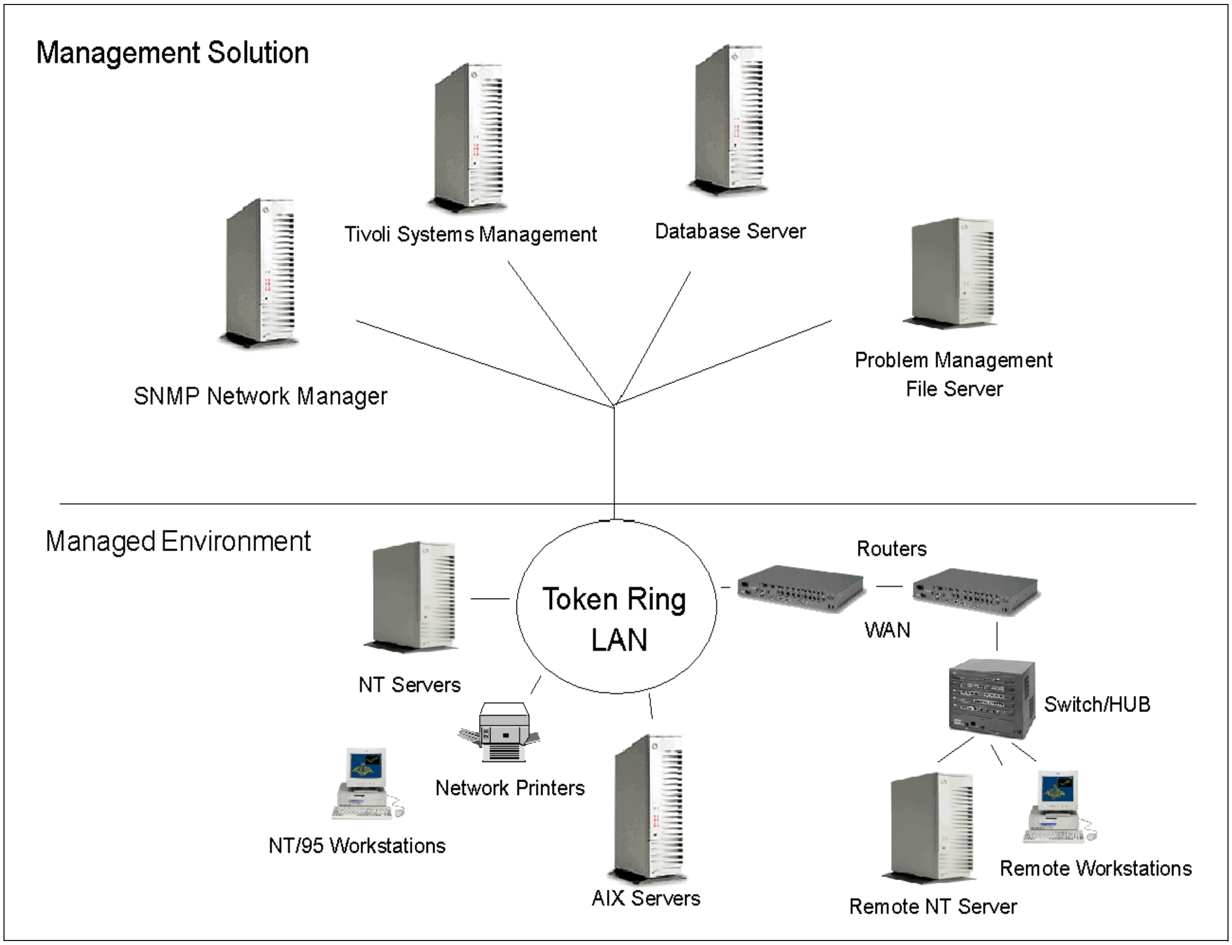


Figure 2. Customer Scenario

The customer environment is comprised of a number of managed elements. These managed elements can be broken down into resources that will be monitored by the Tivoli applications, such as adapters. Both the network and system elements will be managed. The type of information that could feed into a problem management system are:

- Network problems
- Server problems
- Printer problems
- Application problems

The environment includes servers, network devices such as routers, hubs, switches and networked printers. By breaking down these managed elements into resources you can build a table similar to the one shown in Table 1 on page 8.

Table 1. Managed Elements- Example

Elements	Managed Resources	Events	Adapter
Router	- Network Traffic - Network Errors	Yes	NetView

Elements	Managed Resources	Events	Adapter
Switches	- Network Traffic - Throughput	Yes	- NetView - Network Management Tools
NT Servers	-Disk Capacity -Memory -Applications	Yes	- NT Logfile - Distributed Monitoring
AIX Servers	- Disk - Memory.Paging - Applications	Yes	- AIX Logfile - Distributed Monitoring
Network Printers	Availability	Yes	-NT Adapter (print queues)

This table can be extended to include more specific data for example, a particular file system on the AIX server or a type of connection on a switch. These managed resources will be the resources that will send events either from monitors or directly from the devices. For instance, the switches will generate network problems encountered when a port is down.

In order to provide this management functionality the following servers are required:

- SNMP network manager
- Tivoli systems manager
- Database server
- Problem management server

The section below outlines our approach to gathering the data we need to continue.

2.2 Informational Prerequisites

The following list outlines the different kinds of information that you may need to collect and document. Such information could include:

- Project-specific information (what is to be accomplished?)
  - Customers requirements in the form of a proposal
  - Physical network topology such as a network topology diagram
  - Policies such as problem, change and asset management
  - Service level agreements

In addition to the customer requirements we perform a simple event methodology or build a list of the potential events that will be generated. With this information we can perform the event analysis as follows:

  - Build event relationship diagrams for all event sources to show what events can be correlated, therefore reducing the number of critical events that will be forwarded to the problem management applications.
  - Develop flowcharts showing the event relationships to assist with the development of the TEC rules.

- Document the event policies. This includes what events will become trouble tickets and how long we wait before we create a trouble ticket.

For each of the managed resources we document the following:

- What are the managed resources?
- What management tools are required in order to manage the resources?
- What adapters and distributed monitoring agents need to be deployed to provide the monitoring functions?
- What network management tools do we need to monitor the network devices?
- What applications need to be monitored, for instance, daemons and operating systems?

The process design for problem management is based on the customer requirements. Some of these requirements will include:

- Definition of the support structure if this currently exists
- Escalation and notification policies
- Specific problem management and help desk requirements

The next section details more specific requirements and considerations

2.2.1 Problem Management

As the pace of change of informational systems accelerates and new technologies are implemented, problem management is becoming more complex and more important. Companies are becoming increasingly dependent on their IT services, so IT must be able to quickly identify, bypass, resolve and prevent problems.

Problem management includes:

- Taking calls from users with issues or problems
- Handling alerts sent from software systems
- Tracking the problems
- Report on the overall amount of work that has been done to solve the reported problems

It also has to provide various interfaces to the outside world such as human interfaces as well as the non-human interfaces such as APIs or CLIs.

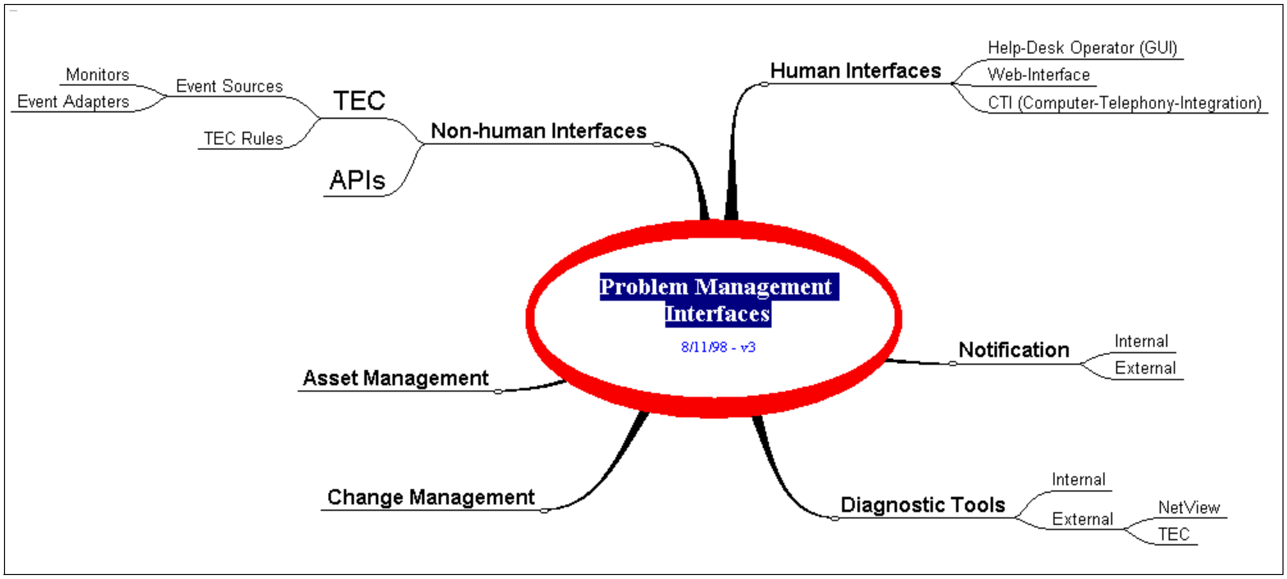


Figure 3. Problem Management Interfaces

Therefore a successful implementation of a problem management solution includes a great deal of planning, discussing and process defining, independent on tools chosen to build a problem management solution.

The diagram shown in Figure 3 on page 11 also shows the integration with asset and change management services.

As the scope of this book is to show how to integrate Tivoli Service Desk and TEC in end-to-end management, we do not define the specific problem, change and asset management processes within a company. However, we do use ideas from the IBM System Management Framework Design (SMFD) as well as the Event Management and Correlation Methodology (EMCD).

From a functionality point of view, what we would expect the following from a problem management solution:

- Data tracking and data availability

  Save problem-related information and have it available for interrogation. This implies the possibility of performing guided data registration and an efficient query capability to have immediate access to data when required.

- Event management interfaces

  There are two major entry points for the creation of trouble tickets: human and non-human.

  A non-human interface could be implemented as a two-way interface, allowing a bidirectional exchange of information between the problem management application and the event management application. Depending on the defined problem management process policies, it may be necessary to restrict the abilities of this interface. Therefore, a practicable way of restricting these should be provided.

- Workflow flexibility

The solution model has to fit into any business model or process for problem management, and at the same time it has to be flexible enough to be adaptable to process change.

- Categorization and correlation

  Most problems start with a telephone call from an end user. Until this call has been categorized or until it has been checked to see if the call correlates to any other existing calls (or even information sent from other sources), it may be inefficient to start work on the call. The problem may just be a subsequent problem that is already being worked on.

  The problem management solution has to be able to assist in quickly identifying and categorizing the problem.

  An intelligent categorization is a prerequisite to successfully correlating calls and is a key point to efficiently performing the function of assignment and resolving problems.

- Define priorities

  It must be possible to assign different priorities to a problem, which are related to its impact on the critical resources in an environment.

  The priority will influence the policy of actions performed on that specific problem.

- Assignment

  This is the capability of a problem management solution to identify who is responsible for taking care of a problem at any time, and the capability to transfer this responsibility from one user or group of users to another depending on the skills that are required. The problem should always be assigned to the right person, meaning this person has the responsibility of the problem and the skills needed, therefore, leading to a faster solution.

  Of course, a good categorization of the problem helps to identify what skills are necessary to resolve a problem and then to assign the appropriate person to the problem.

- Notification

  Throughout the existence of a problem record certain people or systems may need to be notified depending on notification policies. As it is often the case during a problem resolution process, we may need to forward information internally or externally using different mechanisms. Therefore the notification interface should be flexible enough to be integrated with different notification methods such as e-mail and paging.

  Additional reasons for notification are:

  - Notify help desk when a problem has been assigned to them either by manual dispatch process or automatic problem generation through integration with the event management facility.
  - Notify management when a service level agreement (SLA) has been violated.
  - Inform an end user when a problem has been resolved.

- Escalation

  Escalation is a way to define and increase the awareness of a certain problem to the help desk staff. There has to be a mechanism to automatically escalate a problem if it has not been solved or even addressed within a defined amount of time. This amount of time generally depends on factors such as problem severity. The primary drivers for escalation are the SLAs.

- Diagnostic aids

  These aids assist the operation staff in resolving problems, automating solutions or delegating the more simple activities to different support personnel.

  It is very important to have the ability to save approved solutions or workarounds that have been found suitable to resolve a problem and to quickly point to these solutions if needed.

  The capability of organizing and retrieving solutions depends very much on a good categorization of the incoming problems at the time they are created inside the problem management system. Some of these aids include:

  - A knowledge repository that contains the knowledge that in the past was only known to a few highly skilled engineers or professionals.
  - Help desk call resolution thus reducing the IT costs.
  - Depending on the skill level there should be different methods for accessing diagnostic information. This allows for efficient use of the product, which in turn will speed up resolution times while reducing the IT costs.
  - Systems should learn on the fly. As a system is used to record and resolve problems the solution database should become more refined thus returning more accurate solutions which will again reduce the resolution time.

- Web interface

  A Web interface can provide multiple functions to the customer such as giving him or her access to open calls via an Internet browser or search for possible solutions for his or her problem in the available knowledge databases. This may help customers to work through problems themselves or verify what the help desk has documented about their problem.

- Reporting

  In order to efficiently take the right decisions about actions to perform on IT systems or changes that have to be done, it is critical to have the specific information about the problem records managed. We may want to know static information such as statistics of systems that repeatedly fail, the resources mostly impacted or responsiveness of the support organization. Or we may want to know dynamic information such as the number of open problem records, number of problems solved, number of assigned problems per support person to estimate the workload.

  The key areas addressed by the reporting facilities are:

  - Measuring service desk efficiency
  - Trend analysis
  - Forecasting

The three main categories for reporting are:

Batch Predefined schedule reports, for example, number of problems per week.

Real Time A management console that would display exactly how many severity one calls are currently open.

Multi-Dimensional Analysis allowing information to be analyzed along different dimensions thus providing the user with views of his or her information. By showing different views of the data you can perform true trend analysis and forecasting.

With the problem management system receiving information from external sources such as the TEC, this is also important to perform an event correlation exercise to eliminate the unwanted events that become trouble tickets so that only genuine problem management calls are generated. To do this we must perform an event correlation process. This is detailed in the following section.

2.2.2 Event Management and Correlation Methodology

One of the major entry points to a problem management solution is the interface to the event management.

In the context of a production IT environment many events are generated. They can, for example, represent status changes of a resource. Generally these events are originated by many sources and are presented in various data formats, for example, SNMP.

To be able to handle this issue, we need a way of managing the potentially large volumes of different types of events that can be generated. We need to determine which of the events are relevant and which can be filtered out.

The main questions that have to be answered are:

- What event sources should be considered?
- What is the filtering criteria for these events?
- What severities have to be assigned to the events?
- How does the event management process link into the problem management process?

The Event Management and Correlation Design Methodology (EMCD) is an IBM methodology that addresses these needs and delivers the solution to the questions above.

We can not cover the entire methodology in this book, but we do summarize the main steps involved. Essentially, it is based on four activities:

- **Activity 1**: Select the event sources

  Select a list of managed elements for the event analysis.

- **Activity 2**: Inventory managed elements event repertoires

  Create client-specific workbooks consisting of event repertoire worksheet templates used in the events analysis.

- **Activity 3**: Document event policy and processing decisions

  Definition and application of event processing policies to decisions on filtering, throttling and forwarding actions to be taken for events generated by event sources.

- **Activity 4**: Conduct event correlation analysis

  Analysis of enterprise-significant events to identify and document correlation relationships between these events.

At the end of these activities the output is a list of event correlations that need to be performed. By correlating the events we can then resolve which events will be forwarded to the trouble ticketing system and which events can be filtered out.

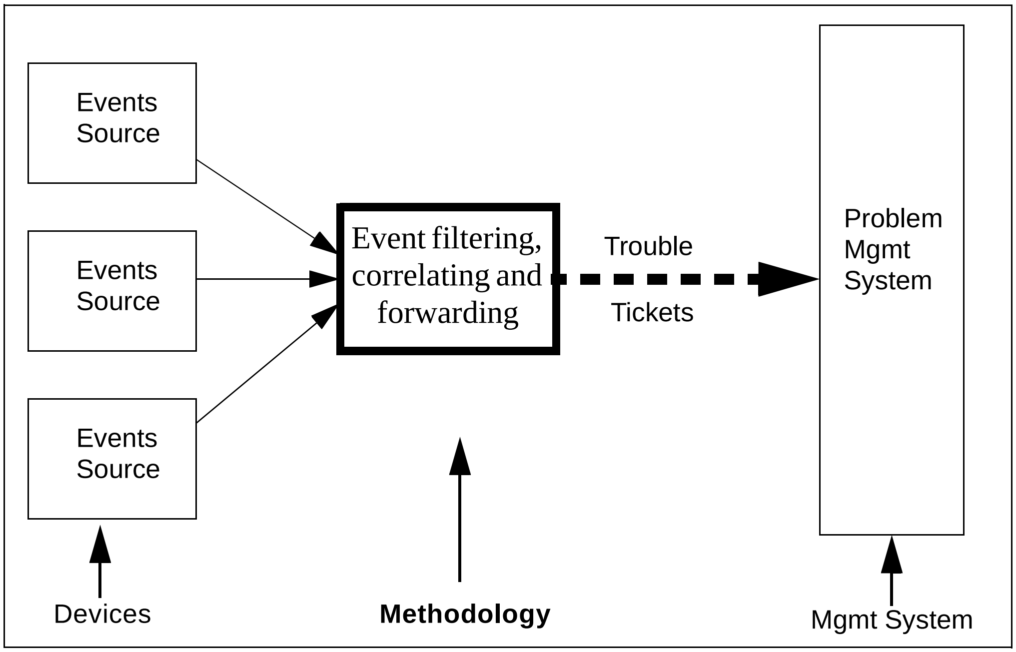


Figure 4. Event Methodology and Problem Management Relationship

Instead of discussing the various policies used throughout the EMCD to help to define customer-specific policies, in Chapter 6, “Problem Management Scenario Using the TEC” on page 241, we assume the policies and focus on the list of events that after having run through the EMCD process are selected to become trouble tickets.

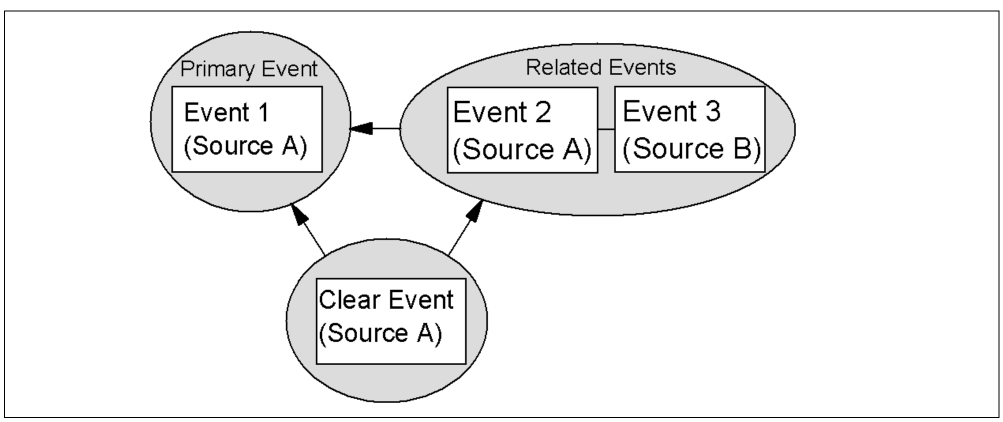


Figure 5. Event Relationship Diagram

Figure 5 on page 15 shows the type of relationship diagram that can be created from the methodology. Here we can see that if the events 1, 2 and 3 are all generated, then events 2 and 3 will be correlated with the primary event 1 and only one trouble ticket will be created instead of potentially three. The information contained in the related events will update the primary trouble ticket when they arrive.

A clear event will close the primary event and also any related events.

Now that we know what events are to be sent to the problem management system from the TEC console we now need to map the data from the various events to the problem management applications. This process is outlined in the following section.

2.3 Data Mapping

The next step after the EMCD process is that we have to provide valid and meaningful information in the events to be able to pass the correct information to the problem management records that will eventually become a trouble ticket. This process we call *data mapping*.

Definition

Data mapping is the integration between the event management and the problem management application.

It consists of mapping information in a correct and convenient way from an event coming from an event management application, to a problem record inside a problem management application.

In contrast to EMCD being a generic methodology independent from specific applications, the data mapping actually is a process that is generally needed. No matter what problem management solution has been selected, it is highly dependent on the specific application involved in the solution for the event and problem management.

Although the process of data mapping is dependent on specific applications, we want to identify general considerations and policies that must be applied to achieve an efficient data mapping process.

In our case we assume that the event management and problem management application has been customized and the policies for the problem management process are already defined. This way, we can concentrate on how to integrate the event management and the problem management solution. Typically, the questions that we need to ask in order to understand the problem management application are outlined below:

- Where is the information stored and how does it make use of this information?
- What information is mandatory?
- What is the relationship to the inventory data? How is this data divided up into specific fields in the problem management records?
- How does the problem management solution communicate with the event management application?

- What notifications have to be transmitted back and forward between the event and problem management application when changes happen on the event?
- Identify the critical information for the problem management application
  - What information is used for specific functions such as categorization, notification or assignment? What information is used by diagnostic tools and if the problem management application provides any?
- Identify corresponding data elements of the event/problem pairs
  - Identify each field containing critical information of the problem record and map these information elements to the corresponding problem record fields. This can mean combining multiple data elements of an event to fit into one problem field or to split a single data element into multiple problem fields.
- Evaluation
  - Identify if there is any information missing in the event that is either necessary or could be useful for the problem application. So, return to the event source (dependent on the implemented system management solution) or event source application (dependent on the way the application issues information about its state), and try to add this information wherever possible. This can mean that you have to customize the system management solution.

You may find relevant event information that cannot be copied into the problem management record because the corresponding field is missing or is incompatible. In this case you may need to customize the problem management application to contain the event information required.

Except for the first two items, the general process described above should be performed on a per event basis. This means to get the most value out of the problem management solution, it is necessary to analyze each event and to figure out what the best way is to do the data mapping.

The described process is only valid for those events that during the EMCD were considered to be significant enough to become trouble tickets. Of course finding a generic data mapping, which is common to many events, will be less time-intensive and easier to implement, but the effectiveness of the problem management solution will be significantly reduced.

2.3.1 Operator Requirements

In addition to the problem management policies the help desk or operator requirements are summarized below.

The requirement is to have a level one help desk for one user, and two level two support operators.

The level one support can only access the problem management software.

The level two support operators require access to the network and system management tools respectively.

Additional functions required for the operators are:

- Ability to change the severity

• Ability to escalate specific trouble tickets
• Ability to open, update and close all trouble tickets

2.3.2 Software

The management tools we selected to provide the end-to-end management system are listed in Table 2 on page 18.

Table 2. Software Tools

Software	Version	Comments
Tivoli Framework	3.6	Provides single TMR environment.
Tivoli Distribution Monitoring	3.6	Provides monitoring for Servers and management applications.
TEC	3.6	Provide the event correlation and deployment of: - NT Logfile Adapter - AIX Logfile Adapter
Tivoli Inventory	3.6	Provides additional inventory information to the problem/asset management tools.
Tivoli Problem Management	5.01	- Provides the help desk interface to the trouble tickets.
Tivoli Change Management	5.01	- Provides the change management functions.
Tivoli Asset Manager	5.01	- Provides asset management functions. - Integrate with Tivoli Inventory.
Tivoli Service Desk Plus Module	5.01	- Provides Management functions for the Service Desk environment.
Tivoli NetView	5.1	Provides SNMP management and integration with the TEC.
Tivoli Decision Support	2.0	Provides management reporting.

For reference the new applications are named as shown in Table 3 on page 19.

Table 3. Naming for the Software Application

Old Product Name	New Product Name
Application Software Expert(ASE)	Developer’s Toolkit
Expert Advisor(EA)	Problem Management
Distributed Data Manager(DDM)	<No Change>
SA-Script	Tivoli Script
ExpertView(EV)	Network Systems Management Gateway or NSM Gateway
EMA	Mail Gateway
E-WEB	End-User Web Interface
Expert Administrator	System Administration
Expert Evolution(EE)	Tivoli Change Management
Expert Foundation Manager(EFM)	Tivoli Asset Management
Expert Discovery	Tivoli Decision Support

2.3.3 End-to-End Management Diagram

Figure 6 on page 20 shows the end-to-end management proposal that outlines a solution.

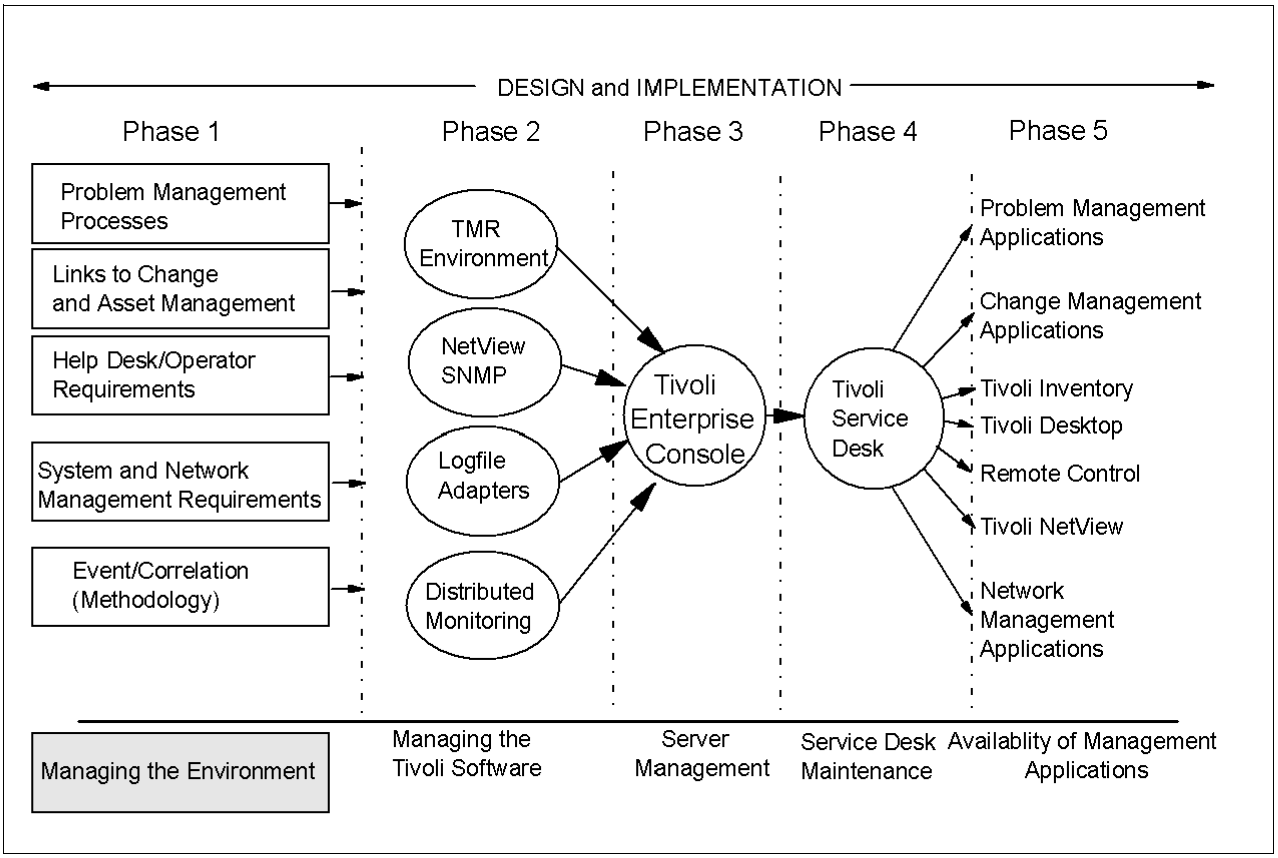


Figure 6. End-to-End Design and Implementation

The design and implementation phases are outlined below:

- Phase 1 - Inputs to the project
  - Problem management process definitions
  - Change and asset management integration
  - Help desk/operator requirements
  - Systems and network management requirements
  - Event methodology output
- Phase 2 - Implementing the monitors
  - Deployment of the Tivoli Framework and environment
- Phase 3 - Event correlation
  - TEC rule development
  - Forwarding mechanism for problem management interface
  - Automation tasks
- Phase 4 - Tivoli Service Desk installation
  - Deploy the service desk applications
  - Build and customize the help desk environment
  - Integrate with the other management applications

- Phase 5 - Install and customize help desk tools
  - Providing the tools to the help desk level 1 and level 2 support

This list of actions provides us with a basis for our project plan.

Due to the complexity of the management environment we added some functions to manage the applications. This involved using the tools provided to manage the management environment to notify systems administrators of a problem with the management applications.

2.4 Deployment of the Management Applications

After the software tools have been selected we can now begin to plan for the deployment of the management applications.

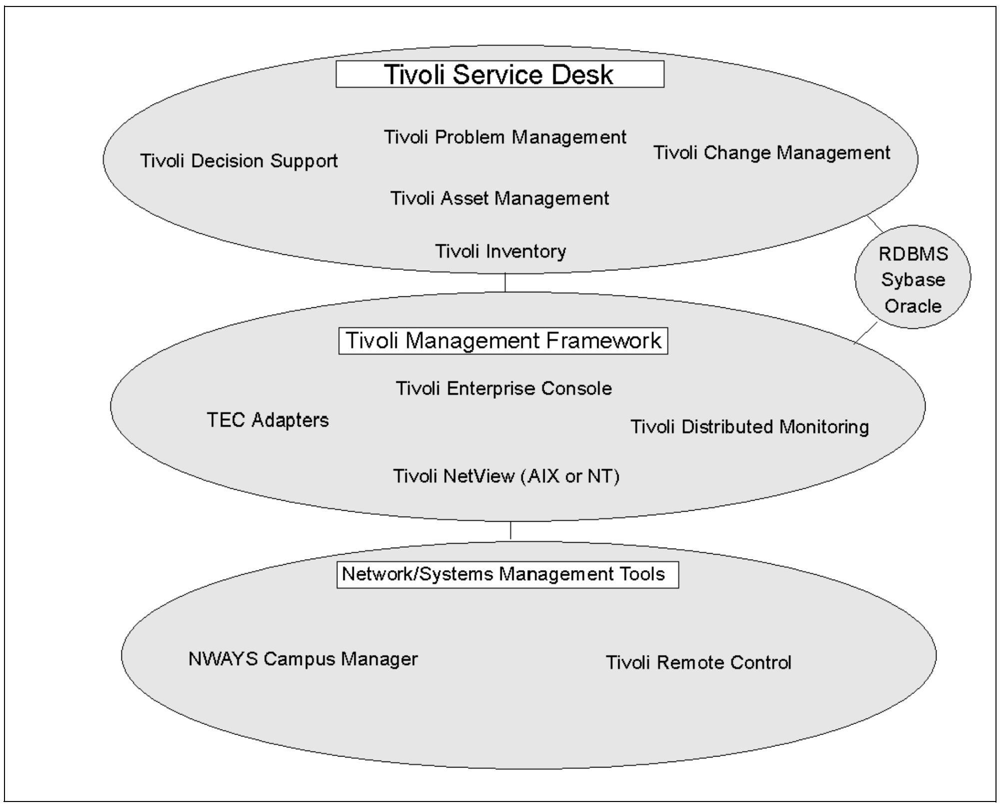


Figure 7. Software Integration

Figure 7 on page 21 shows the view of the software design and the number of tools required. These are grouped as follows:

- Tivoli Service Desk
- Tivoli Management Framework
- Network and Systems Management Tools

2.4.1 Service Desk Applications

Figure 9 on page 23 show the service desk integration with the TEC.

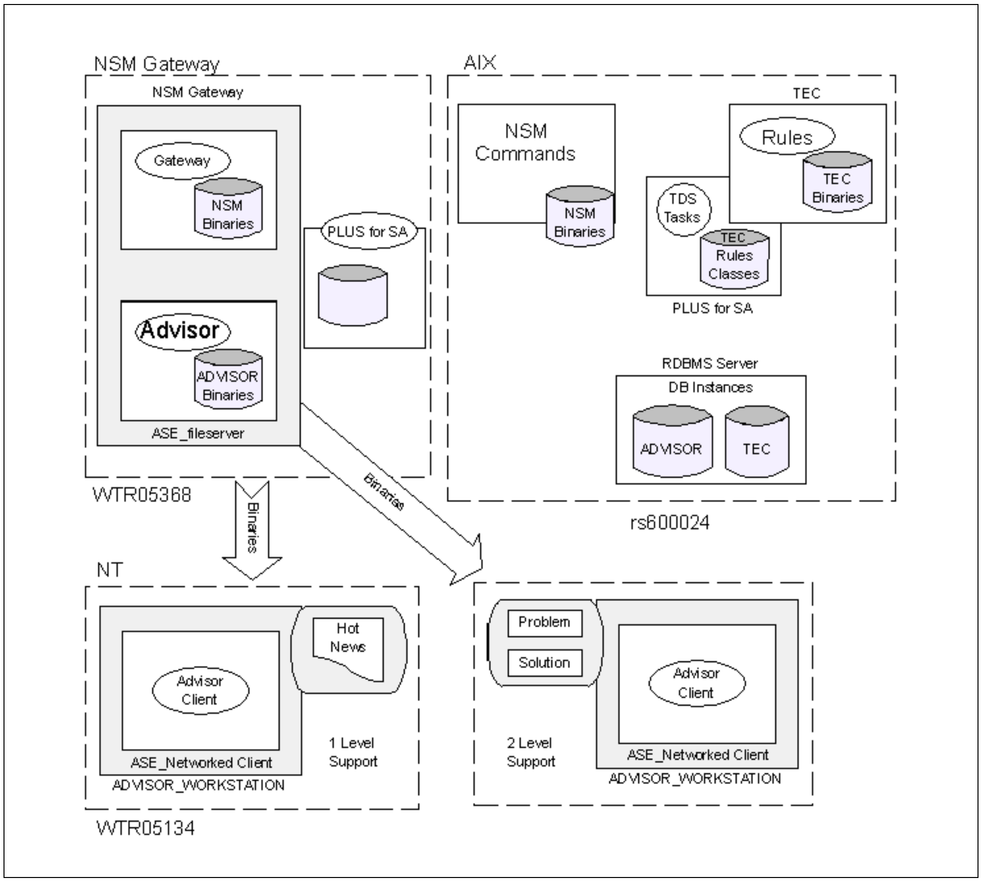


Figure 8. TIvoli Service Desk Environment

Figure 8 on page 22 shows how we grouped the applications and where these applications will be installed. This initial installation of the software components provided us with the base installation. In later chapters we discuss how to add additional applications such as decision support. The initial installation consists of the following.

On AIX server rs600024 we installed:

- The TEC
- NSM commands
- Sybase RDBMS

On NT server WTR05368 we installed:

- ASE code for the file server
- NSM gateway
- Tivoli Plus Module for Service Desk
- All Tivoli Service Desk binaries

On the client workstations we installed:

- ASE networked client

- Client installation for problem, change and asset management

The installation is covered in more detail in Chapter 4, “Implementing the Management Software” on page 51.

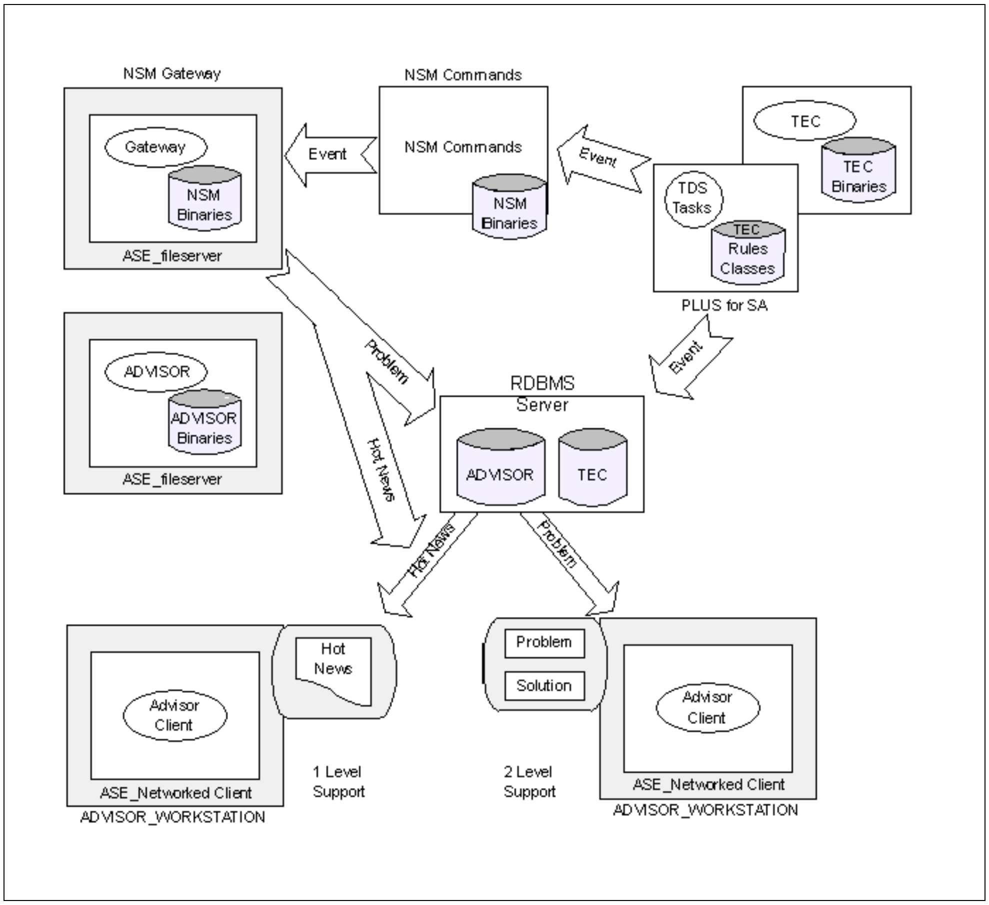


Figure 9. TIvoli Service Desk Interaction

Figure 9 on page 23 shows the event flow and what applications are required to provide the end-to-end event management and automation for the creation of the trouble tickets. The process is outlined below:

- TEC events will be sent using the NSM commands to the NSM gateway.
- The NSM gateway will forward the event to the Advisor as a trouble ticket.
- The problem will be sent to level 2 support directly.
- Hot news will inform level 1 support that the event has occurred.

The following chapter introduces the management tools.

Chapter 3. Product Overview

This chapter provides an overview of all the management tools used to provide the problem management solution. For reference we have also included a list of new features for the latest release of the Tivoli Version 3.6 applications.

The products for the Tivoli Service Desk include:

- Tivoli Problem Management
- Tivoli Change Management
- Tivoli Asset Manager
- Tivoli Decision Support
- Tivoli Service Desk Plus Module

The products for the Tivoli Management Framework include:

- TME Framework Version 3.6
- TEC Version 3.6 new features
- Tivoli Distributed Monitoring Version 3.6
- Tivoli Inventory Version 3.6
- Tivoli NetView Version 5.1

Additional third-party products include:

- Knowledge Pak for Desktop

We also describe the functions used in the solution for each of the product areas. The following section discusses the positioning and functionality of each of the management tools.

3.1 Tivoli Service Desk

Tivoli Problem Management automatically and efficiently tracks, logs, and escalates interactions and requests and is unique in its ability to track them separately. Tivoli Problem Management includes a sophisticated knowledge engine that enables analysts and operators with different skill levels to resolve problems.

The service desk is notified about a network, system, or application event. And the network group has access to problem history data in the organization's knowledge base to better understand trends and possible solutions.

Software Artistry became a Tivoli business unit in January 1998. Their product suite SA-Expertise 5 for Enterprise Support Management (SA-Expertise) has been renamed and integrated with the existing Tivoli products. For a mapping of the various applications within SA-Expertise, see the following list of names of the existing products and their new names within Tivoli.

Tivoli Problem Management consists of the following software components:

- Application Software Expert
- SA-Expert Advisor

- ESM Build
- ESM Script
- Expert Mail Agent
- Expert Web
- Expert View
- Expert Distributed Data Manager

Tivoli Asset Management has been renamed from SA-Expert Foundation Manager.

Tivoli Change Management has been renamed from SA-Expert Evolution.

The combination of these three products will be known as the *Tivoli Service Desk Suite.*

Tivoli Decision Support has been renamed from SA-Expert Discovery.

Tivoli Decision Support Guides has been renamed from SA-Expert Discovery Guides.

The following sections provide an overview of the Tivoli Service Desk applications.

3.1.1 Tivoli Application Software Expert (ASE)

Tivoli Application Software Expert, or ASE, is the base component of all Tivoli Service Desk applications. All the other applications rely on ASE to provide a set of functions. It is also the development environment for creating new Tivoli Service Desk applications. The different parts of ASE are the run-time and developer’s toolkit. The toolkit is comprised of:

- ESM Build

  ESM Build is a utility used to create and maintain the database structure for the Tivoli Service Desk database. It provides the functions to build and view tables, migrate data from an existing environment and perform actions on the database.

- Tivoli Script

  Tivoli Script is the programming language that is used to create an application. It can run over multiple platforms and RDBMS systems. To create a Tivoli Script application, you can write a module which is called a *knowledgebase*. This knowledgebase file is then parsed. This is a kind of compiling, which is performed by the Tivoli Script Parser application. When it has been parsed, the knowledgebase file is ready to be used.

- Interface Designer

  The Interface Designer is used to build the user interface of custom Tivoli Script applications. It can create forms, menus, tool bars and string tables. When you use the interface designer the output will be a file with the suffix .df. To be able to use this file you must first generate the interface with the Interface Generator. When you use the Interface Generator, it creates a file with the ending .dfc. The.dfc file is then used by your Tivoli Script application.

- Fulcrum Search Server

This application is used to create indexes in a very large text file or in a database if you have a large number of database records. This indexing provides faster text searches.

- SQL Configuration Editor

  The SQL Configuration Editor is the application that is used to configure the clients SQL connection to the database. Here you specify what type of database server you have in your environment, the name of the database to be used and any other database-specific options.

3.1.2 Tivoli Advisor

Tivoli Advisor is the help desk application of the Tivoli Service Desk Suite. In Tivoli Advisor, calls are entered as problems and solutions are presented to the problem analyst. All of the other applications have links to this application, which makes it the focal point of the whole environment. Figure 10 on page 27 shows the main screen where a call is entered and is the gateway to other management functions.

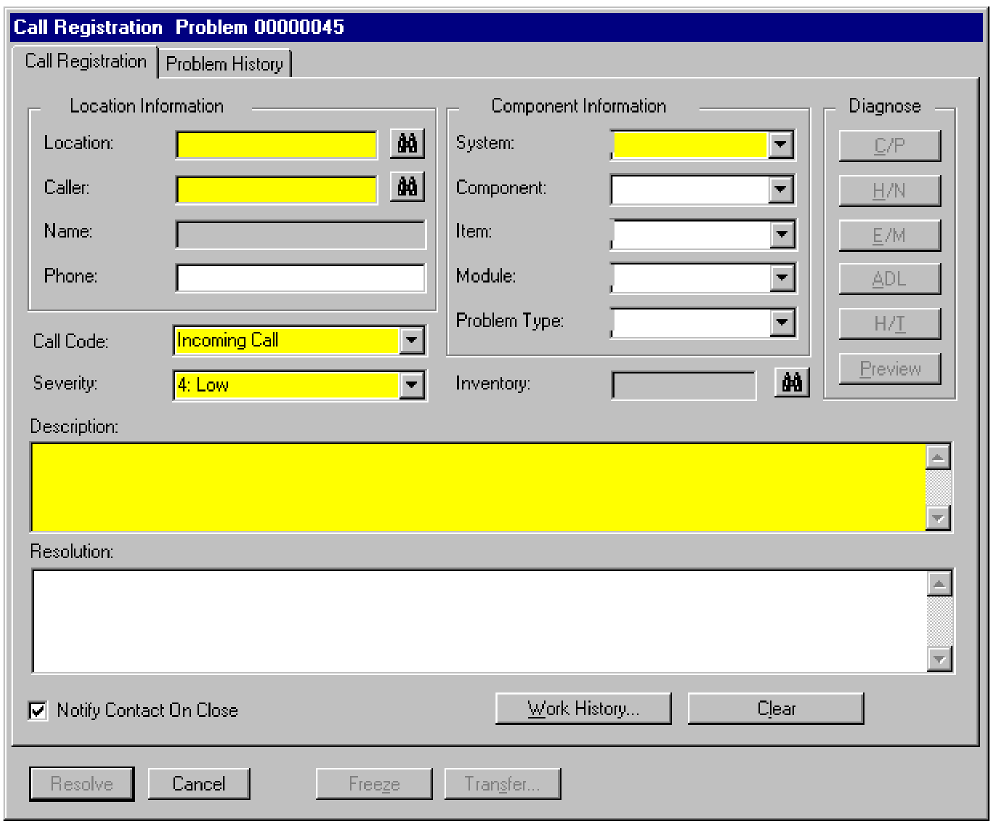


Figure 10. Main Advisor Call Registration Window

3.1.3 Notifications, Alarms and Escalations

Notifications are used to notify a user or caller about the status of a problem. They are created inside Tivoli Advisor, but deliver the notification via an external notification device, such as e-mail or fax. An example of a notification is to inform a caller that his or her problem has been relayed to a level 2 technician, or that the problem has been closed.

Alarms are notifications with a higher priority level. They are created and delivered within Tivoli Advisor, for example, if an escalation occurs, or if an application misbehaves.

Escalations are used to increase the level of awareness of a call or a problem. The escalations are created based on time, severity and other conditions of the problem. When an escalation occurs, notifications are generated.

3.1.4 Archiving and Purging of Data

To maintain an efficient database there are functions for archiving and purging data that does not need to be kept online in the database. The archive function lets you remove data and store it somewhere else to keep the size of the database small, therefore more efficient. The purge function lets you remove redundant solutions that are still in the database.

3.1.5 Setting Problem and Call Options

With workflow tools, you can define what should happen during call registration and problem registration. The behavior of the advisor screen will change. For instance when a call is being entered, the application will ask you to confirm each of your actions.

You can also set the default value for some fields to make the call entering more efficient, for example, set the default severity to medium.

The option of altering the call code gives you the opportunity to define what source the call comes from, such as e-mail, phone calls and the Web interface.

If you alter or add problem codes, you can set a status on a problem for example, closed, open or transferred.

By creating different problem types, you can choose what types of problems there might be in your environment. Some examples of this are hardware, software and communications.

3.1.6 System, Component, Interface and Model (SCIM)

Tivoli Advisor features the principle of SCIM. This is a way of organizing the different products that exists in a help desk environment. The SCIM is used to help narrow down the number of possible solutions to a call or problem. SCIM is hierarchical; this means that you always have to define a system in order to be able to define a component of that system. You must also have a defined component to be able to define an item of that component.

3.1.7 Diagnostic Aids

In Tivoli Advisor there are a number of tools to help perform the diagnostics and find the solution to a problem. The aids included in Advisor are:

- Adaptive Learning (ADL)

  When you use Adaptive Learning, you can enter a text description of your problem and it searches the solutions database for a solution that matches the description. The more you use Adaptive Learning, the better it will become in finding a solution to the problem. Adaptive Learning’s speciality is that it finds solutions to a problem even if the callers describe the problems in different ways.

- Common Problems (C/P)

  With Common Problems, you specify solutions for the most common problems in your environment. An example of a common problem could be how to change the password for an application.

- Error Messages (E/M)

  The Error Messages aid contains solutions and explanations directly related to error messages in your environment.

- Hot News (H/N)

  The Hot News aid is a way to inform all the Advisor users that something is happening right now. For example, if a system goes down, a Hot News record is created so that the help desk analysts are aware of this when the users start calling. The analysts can also be informed when the system is back online.

  Hot News provides an up-to-date bulletin board containing time-sensitive information.

- Hyper Trees (H/T)

  Hyper Tree is a branch structured collection of knowledge information. It consists of questions that a problem analyst can ask the caller about his or her problem. Depending on the answer the caller gives the problem analyst, the Hyper Tree will determine what the next question should be. This is helpful to determine the real problem of a call. You begin with a generic question and then drill down to the problem by answering more and more detailed questions in the tree structure.

  An example of this could be that a caller’s screen has gone blank. The problem analyst asks the caller the initial question provided by the Hyper Tree, this is "Is the power light on?". If the response is no, then the next question is: "Is the power cord attached to the outlet?". This continues until the problem analyst reaches a solution.

- Solutions

  The Solutions diagnostic aid is used to search through the entire problem database, as opposed to the other diagnostic tools, which will only search through parts of the database. The Solutions aid is often used if none of the other tools can provide a solution.

3.1.8 Tivoli Distributed Data Manager (DDM)

The Distributed Data Manager (DDM) is an application that can only be installed if Tivoli Advisor is installed. The purpose of DDM is to transfer data between different Tivoli Advisor databases. Data, in this case, means a call or a problem. This is very useful if you have several Advisor databases. If you have a central service desk with a Tivoli Advisor database on that location, it is not efficient for all the analysts around the world to access this database. The traffic would run over slow WAN links. With the DDM you can transfer the call or problem to a local database and make it more accessible to the analyst in that location.

The DDM can also be used to transfer all calls or problems to a different database. This provides a capability to move the whole service desk function to another location.

DDM can even transfer data between different types of databases. You can, for example, transfer data between a Sybase or Oracle database.

3.1.9 Tivoli Network and Systems Management (NSM) Gateway

NSM stands for network and systems management. The NSM Gateway is the bridge between the Advisor and the management platform. It consists of three parts:

- Tivoli NSM Commands
- Tivoli NSM Gateway
- Tivoli NSM Gateway extensions

The objective of the NSM commands is to supply the gateway with information about network and system events.

The objective of the gateway is to interpret this information and pass it on to the Tivoli Advisor.

The objective of the Tivoli NSM Gateway extension to Tivoli Advisor is to display the data from the gateway to the Tivoli Advisor users.

- Tivoli NSM Gateway and Network Management Systems

  The NSM commands for the network management platforms are implemented by listening and intercepting the events sent to the management application. For instance, the event could be a node or interface down event. When an event comes in the NSM commands report this to the gateway.

  When the gateway receives information about an event from a network management platform, it looks in a list of managed objects to see if the object where the event came from is contained in that list. If it is, the gateway forwards the event to Tivoli Advisor.

  For the network management platform there is also a function to display the status of network objects from the Tivoli Advisor. This is achieved by calling APIs on the gateway from the Tivoli Advisor. The gateway then forwards the request to a daemon on the network management platform.

- Tivoli NSM Gateway and System Management Systems

  The system management application can be configured to call the gateway when an event is forwarded and the Tivoli Advisor is configured to do a callback when the problem is solved.

  This means that there is more flexibility to define which events are going to be problems.

3.1.10 Tivoli Asset Management

Tivoli Asset Management is the link between service desk analysts and the physical inventory they support. It provides the collaborative asset repository view that enables the service desk to better support corporate assets from start to finish. Tivoli Asset Management enables the service desk to deliver efficient support reducing the number of calls. The data from leading asset discovery tools such as Tivoli Inventory, Tally NetCensus, and Microsoft SMS can be viewed or extracted. A flexible data schema gives any service desk the power to define and organize asset attributes to meet specific needs and tailor the system to unique environments.

IT-related problems usually require from help desk personal access to data about corporate IT assets. Tivoli Asset Management acts as a repository to collect all of the organization's IT asset information. Tight integration with Tivoli Problem Management provides the help desk personal with a quick inside view of the asset connected to the open problem.

Besides the inventory view, Tivoli Asset Management also gives a business view of the company assets by providing the data about contracts, maintenance fees and leases associated with the assets, as well as acquisition dates. With these features this product can be used for managing all company's assets, not only IT-related.

Tivoli Asset Management also takes into consideration companies that already have their assets documented in digital form. For this purpose Tivoli Asset Management provides an external utility to populate and manage its database. The Migration and Batch update utilities are designed to move data from existing inventory systems to the Tivoli asset management database. The Export and Import utilities are designed to copy information out and back into the database.

3.1.11 Tivoli Inventory Integration Module

The Tivoli Inventory Integration Module presents the help desk user with a vast amount of data stored in the Tivoli Inventory repository for any given device. The interface links two inventory systems with pointers in Tivoli Asset Management, which points to Tivoli Inventory source records. No data is copied or moved from the Tivoli Inventory database to Tivoli Asset Management database. This eliminates data redundancies between the two systems.

Pointers in the Tivoli Inventory Integration Module are created by running the migration utility. This utility migrates only selected Tivoli Inventory records. The user who selects records for migration can also control migration of Person and Location records associated with each Inventory record.

When a user selects an item that was migrated from Tivoli Inventory, Tivoli Asset Management automatically controls which types of data should be presented for viewing based on the type of record being retrieved from Tivoli, as shown in Figure 11 on page 32.

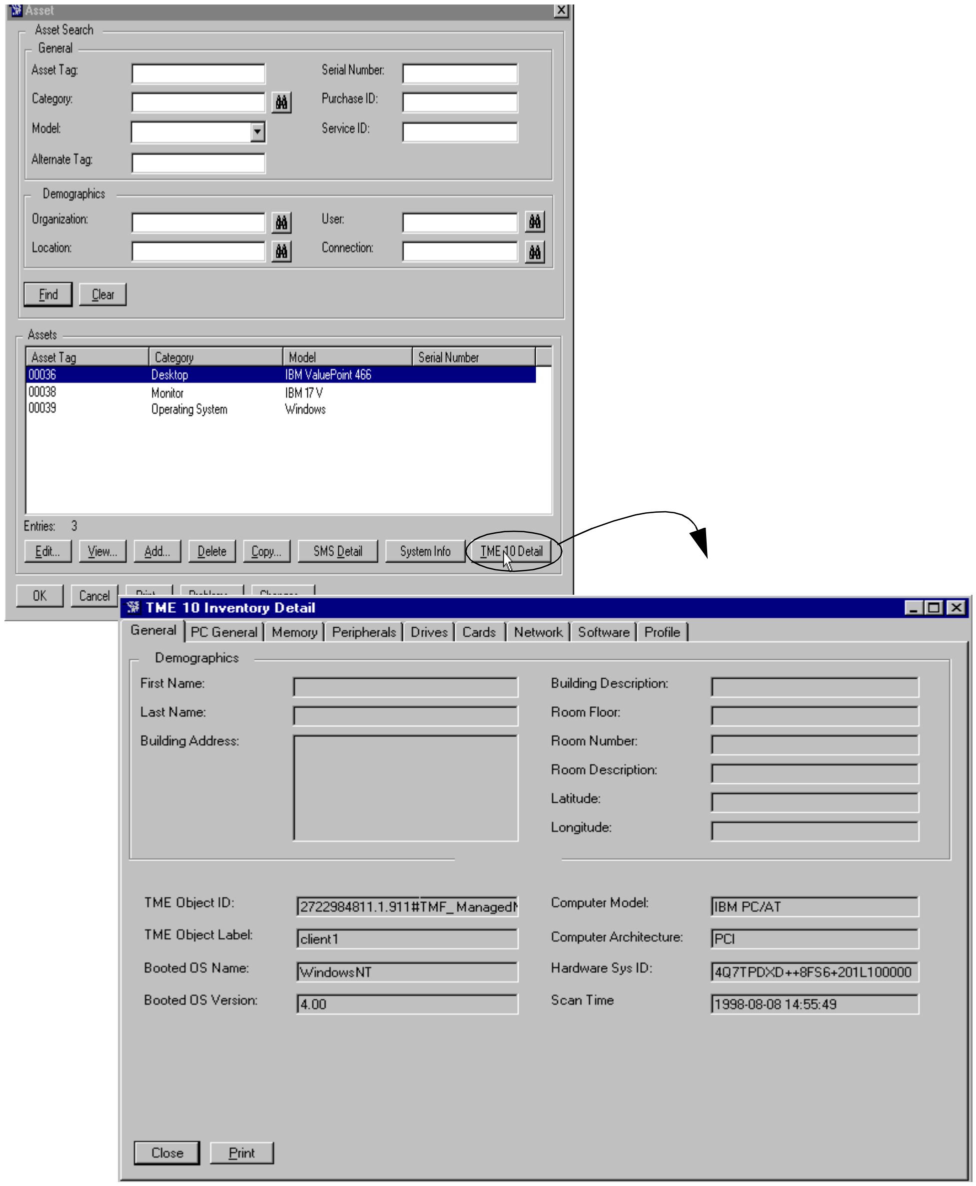


Figure 11. Viewing Tivoli Inventory Records for a Chosen Asset

A prerequisite for the Tivoli Asset Management installation is that the Tivoli Advisor application must be installed. The Tivoli Asset Management application can be installed on a file server, client workstation or stand-alone workstation.

3.1.12 Tivoli Change Management

Tivoli Change Management gives the service desk control over this process from start to finish. By allowing business rules to be embedded, Tivoli Change Management ensures that policies unique to an organization, or best practices, are made part of the change process. With this system, you will have critical information regarding changes that are underway, so it is easy to quickly evaluate their impact.

By analyzing past changes and any associated problems or assets, the service desk is better able to leverage past experience and estimate who or what may be affected by a proposed change. Tivoli Change Management also enables the organization to highlight critical path activities, delayed tasks and outstanding approvals to keep the change process moving.

The change management software can provide a comprehensive link to the asset management application due to the fact that most changes that occur involve some modifications to one or more assets; and therefore a main requirement would be to have the function to automatically update the affected asset or assets.

Open problems require solutions and these solutions can involve a request for change. Change is defined as any modification to a production environment that might affect business performance or operations, for example, software or hardware upgrades, new assets and location changes.

The Tivoli Change Management application manages all kinds of changes in the organization, whether they involve changes in business or technology. This product with its tight integration with Tivoli Problem Management automates the change process from the initial change request to implementation with procedures and policies to:

- Completely manage every step in the change process.
- Notify change participants automatically.
- Create templates for repetitive changes.
- Proactively interpret an impact of change.
- Embed existing business policies and rules into change management.
- Interface with Microsoft Project for project planning purposes.
- Continuously analyze and evaluate the change process.

The change process involves a number of stages, people, resources and processes. Figure 12 on page 34 illustrates change flow and the roles of the person responsible for the change.

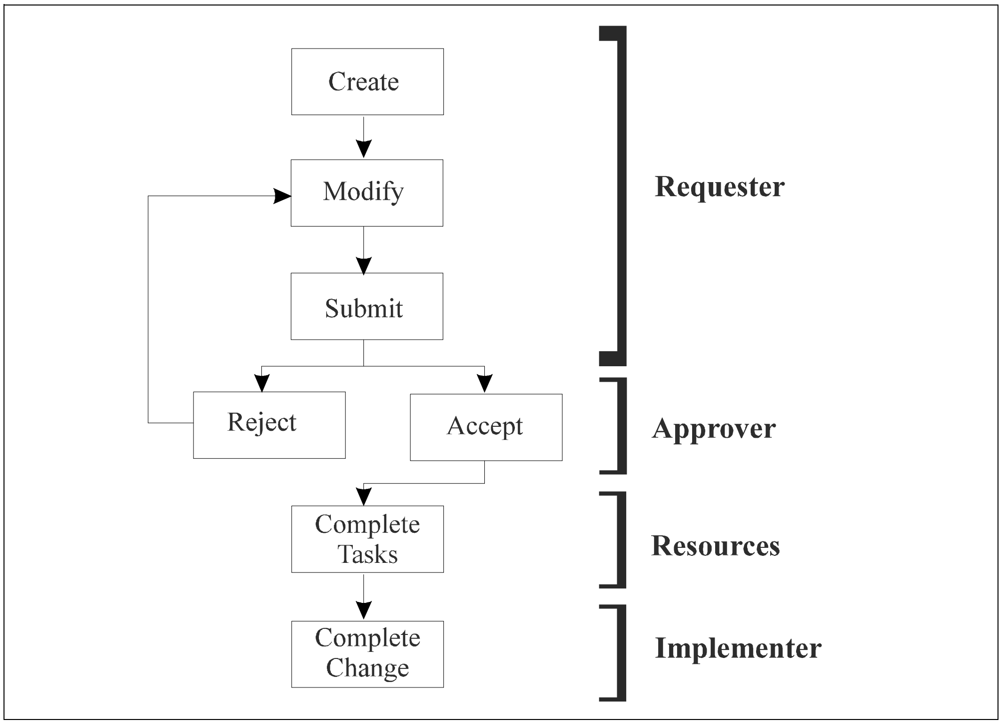


Figure 12. Change Flow and Roles in the Change Process

There are two ways to control access to actions, or in other words the role in the change process, in Tivoli Change Management:

- Choose the appropriate installation type.
- Assign the appropriate security rights to users.

- Installation types

  During the installation of Tivoli Change Management the installer has to choose one of the following installation types:

  - Tivoli Change Management Requester, which contains functions for requesting the change
  - Tivoli Change Management Approver, which contains Requester functionality as well as functions for approving the change
  - Tivoli Change Management Administrator, which includes a system administration functions in addition to the requester and approver's functionality

- Security rights

  Security right controls access to Tivoli Change Management application actions. Users do not have the security rights assigned by default; they should be assigned to them individually or to groups. The Tivoli Change Management application contains security rights, which correspond to the roles in the change process.

  The installation type should correspond to the user's role and vice versa. For example, if the user will participate in the change process as a requester, he or she should have the requester installation type and his or her assigned security rights should allow actions connected to his or her role in the change process. In both cases the user first has to be defined in Tivoli Advisor. After creation of the user, security rights for Tivoli Change Management actions can be granted to him or her.

The requirement for installing any of these three installation types is already having the Tivoli Advisor installed but the full strength of the change process automation can be achieved together with the Tivoli Asset Management application. The Tivoli Change Management can be installed on a file server, stand-alone workstation or client decision support.

3.2 Tivoli Decision Support (TDS)

Tivoli Decision Support is decision support software (DSS) that aids analysis and decision making in an organization. Decision Support enables you to quickly find and use the data in your enterprise's databases in a number of key business areas, including:

- Support center
- Asset management
- Telephony performance
- Change management

For example, your support center database contains information captured as analysts resolve customer requests and manage the support center's daily functions. Some examples of this data are:

- Request trends
- Interaction trends
- Resolution times and rates
- Product defects

Decision Support enables businesses to extract information from the customer data that is gathered daily. It builds content automatically, and is ready to be used by various users. Finally, Decision Support handles the delivery of and access to data. Decision Support facilitates knowledge discovery and user access to information. Data collected can be shared with others in the enterprise using several delivery mechanisms, including hard copy printouts, files, and push content. In the latter case, content that has been collected by one user can be sent to a central repository on a company's intranet or set up on a PointCast server, from which other users can gain knowledge.

Users who have the PointCast client installed can view these articles or data that has been pushed to the ticker without having the TDS discovery interface installed.

By using Decision Support to tap into your operational data, you can:

- Measure the effectiveness of your operation
- Determine how assets are distributed across the enterprise
- Gain insights into the potential satisfaction level of customers
- Gain insight into the value of your customer relationships
- View which products are supported by specific dealers
- Gain insight into the decision tree architecture, its supporting entities and the instance information

- Identify areas of weakness, to convert from reactive support activities to pro-active support planning
- Further leverage your investment in technology and automation

Suppose, for example, you want to determine why customer satisfaction rates fluctuate over a period of time or in relation to a certain product. Using Decision Support, you can examine the categories of data that measure satisfaction from the customer's perspective, including:

- Problematic products - Which products generate the most requests from customers?
- Open requests - Are requests being resolved?
- Average resolution rates such as, how long must customers wait until their requests are resolved?

By examining the relationships between these pieces of data, you may discover patterns that influence your decision making and future planning. The data may reveal, for example, that more support analysts are required, that workloads should be shifted, or that more development is required on a certain product.

3.2.1 More on TDS

Tivoli Decision Support is the sum of a multidimensional query and tool and a simple query and reporting software. Both of them are packed with templates that will help technologist and user to get best results from the tools.

TDS includes both PowerPlay technology by Cognos for the multidimensional views and report, and Seagate's Crystal Reports for simple query and detailed reports.

In addition to the query and reporting tools, we are provided with Decision Support Guides, templates that will guide us through the amount of data we are going to analyze.

3.2.2 Cubes

A cube is a data container used by PowerPlay, a multidimensional reporting and analysis tool packaged with Tivoli Decision Support.

As opposed to a flat, two-dimensional table of rows and columns, a multidimensional array can be visualized as a cube with many sides, with each dimension forming a side. The cube is arranged so that every data item is located and accessed based on the intersection of the dimension members that define it.

Cubes are built by the TDS Discovery Administrator which runs a query against the database and executes the Cognos Transformer. Transformer takes the input queries and summarizes the data (by counting, averaging, calculating percentage, etc.) and packages this information into a compressed cube file (*.MDC). The TDS Discovery Interface executes PowerPlay reports that look at these MDC cube files rather than live data.

3.2.3 Models

A model contains definitions of queries, dimensions, and measures, as well as objects for one or more PowerCubes that Transformer creates for viewing and reporting in PowerPlay.

Models define the layout of cubes as well as define where to get the raw data (queries). These files are the link between the multidimensional snapshot (cube) and the real data (database against which the queries are run).

3.2.4 Dimension, Layer and Category

A dimension is when and by whom (analyst or organization), did what (interaction type). We may want to look at a numbers of calls by any of these dimensions or even a combination of these dimensions. A layer is a drill-down layer within a dimension.

Each dimension normally consists of a single drill-down path containing one or more drill-down levels that present detailed information in hierarchies. For example, a locations dimension could include the regions, states, and cities in which sales were made.

Slicing and dicing, and drilling down into different dimensions, or combination of, is really the definition of MultiDimensional Analysis.

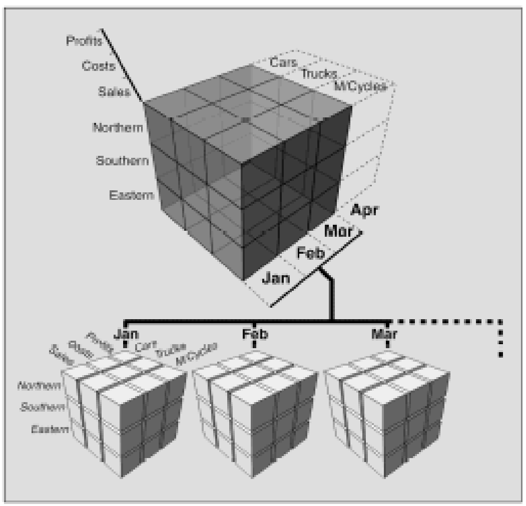


Figure 13. Multidimensional Cube

In PowerPlay, the defined dimensions are shown in the dimension line. The dimension line shows PowerPlay users how to navigate through a cube by indicating the type of information that is available, and the level of detail currently shown in each dimension.

- Measure

  A measure is the summarized counts, average, etc. Some examples include number of calls, the number of problems opened, elapsed business hours, and so on.

- Cognos Transformer

The look of a cube (what dimensions, measures, layers) is defined in Cognos Transformer.

- Discovery Guides

  Discovery Guides organize and group common topics with similar concepts and allow the user to immediately gain insight from the transactional data. They contain algorithms, queries, reports, views and business models as well as contextual informations associated with the views.

By installing the Call Center Management Discovery Guide component we are installing the many pre-built cubes, for example:

- Closed problems, all the problems with status CLOSED, measured over an average of elapsed time or total time spent on a problem.
- Open problems, Diagnostic Aids, for instance the number of problem solved using diagnostic tools.
- Solutions, information on solutions, number of solutions not used on the database, not used in amount of time.

3.2.5 Tivoli Decision Support Architecture

Data used in the Tivoli Decision Support graphing is actually static data, extracted and compiled from real-live databases. These grouping, tracking and calculating capabilities imbedded in the architecture of the product, allow us to effective slice and drill down level by level, getting fast and significant representations of collected data, no matter the amount of them.

Although not the main focus of this redbook we explain in some detail this architecture, with the simple objective of better understanding the product. The process of defining the scope or our reports, is actually by drawing a Transformer model. Creating a new model involves:

- Specifying the query or queries for the model
- Using the dimension map to define the dimensions and levels in the model
- Defining the model measures
- Defining cube objects that use the content of the model to create PowerCubes

Each Transformer model retrieves data from one or more queries that you create to run against your production data. When we run a query, the results are exported to a text file. Using the tools, provided with the product such as a query definition for running queries against databases, we can even cross information from more than one database.

The database fields selected by the query can then be referenced in defining measured values, or in grouping dimensions. In other words, they can be used as the criteria for viewing data.

This process is not as complex as it can look at first glance, the Transformer provides built-in tools for helping an administrator in building models, such as the CheckModel or the AutoDesign feature. The AutoDesign will for example, define calculated fields upon date and time.

Figure 14 on page 39 resolves the contents of a cube definition file, giving us the sensation of what it is like. You can see on the lower left part of the panel the many queries for example, in the Dimension fields.

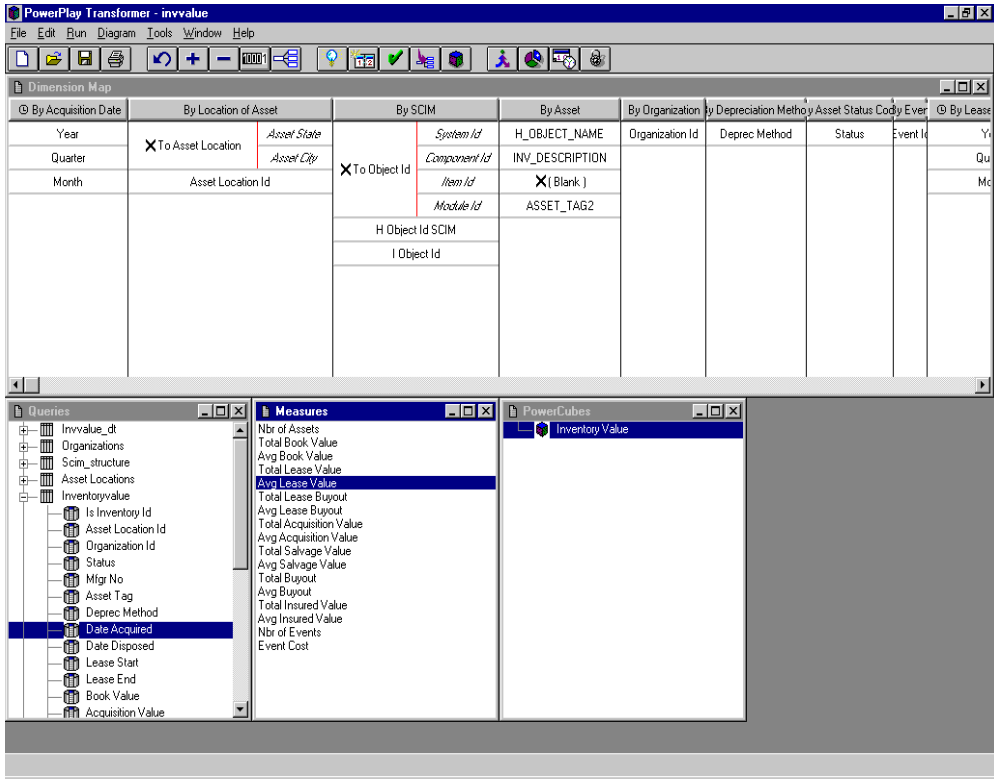


Figure 14. Opened Model

The first compilation step is to run the queries against the database; data is exported in the named text file. Values are extracted from this fields and calculated according to the measure definitions, dimensions are used for creating links and pointers and finally the named cube is compressed and produced.

From now on all the PowerPlay reports will be referred to this cube.

3.3 Tivoli Framework Version 3.6

Tivoli Framework Version 3.6, as well as Version 3.2, can be implemented in a three_tier architecture, with the use of Tivoli Management Agent (TMA), formerly referred as the Lightweight Client Framework (LCF) endpoint.

LCF is an extension to the classic Tivoli Framework that increases scaleability of TMRs while reducing the hardware and software requirements on managed systems.

LCF introduces three new object types that represent system roles in a TMR:

- Tivoli management agent (formerly LCF)
- Tivoli management gateway (formerly LCF gateway)
- Tivoli management server (formerly LCF server)

Though each of the above logically represents a different system’s role in the Tivoli environment, it should be noted that a single physical system can contain more than one of the above object types.

3.3.1 Tivoli Management Agent (TMA)

TMAs are typically installed on systems that are considered *managed only* systems. That is, like most end-user workstations, these systems will be managed, but they will not be involved in the management of other nodes. More specifically, you will not typically be running the Tivoli Desktop or running Tivoli commands from a TMA to manage other resources in the network.

The TMA function resides in the node to be managed. It runs as a small daemon, or background task. This daemon is called the spawner and is also often referred to by the name of its executable module called lcfd**.** This daemon is responsible for executing methods at the request of a managing system. Its only connection to and knowledge of the rest of the TME world is through an endpoint gateway.

When a TMA is installed, a minimal number of files are installed on the managed system. Functionally, the only thing installed is the spawner itself. When an application invokes a method to be executed on the managed system (TMA), the method is automatically downloaded to the endpoint and executed by the spawner. Methods that are downloaded to an endpoint are cached at the endpoint. As long as that method stays in the cache, it doesn’t need to be downloaded again upon a second invocation of the same method. The cache on the endpoint is a disk cache, and therefore is persistent across IPLs of the managed system.

3.3.2 Tivoli Management Gateway (TMG)

The TMG is the software that runs on a full managed node, enabling the managed node to operate as a gateway between a cluster of endpoints and the rest of the TMR. Each TMR can have multiple endpoint gateways. The number will depend on factors such as the available system resources, the number of endpoints, network topology and so on. Currently one TMR server can handle up to 200 endpoint gateways. One endpoint gateway can theoretically handle large amounts of endpoints.

The Tivoli management gateway performs the following functions:

- Listens for endpoint login requests

  The endpoint gateway maintains (with help from the endpoint manager) a list of the endpoints that it is responsible for. As endpoints come online, they will either attempt to log in to a specific endpoint gateway or broadcast a message searching for an endpoint gateway. Endpoint gateways will receive these transmissions, and if responsible for the given endpoint, will proceed with the login process. If the endpoint is not in the gateway’s endpoint list, the endpoint gateway will forward the request to the endpoint manager so that an endpoint gateway can be assigned to the endpoint. The endpoint gateway's list of endpoints for which it is responsible is stored and maintained by the endpoint manager.

- Downcall method requests

  The gateway listens for downcall method requests from other nodes that are targeted for one of the endpoints it is responsible for, and acts as a gateway for method invocations. All operations to or from an endpoint pass through an endpoint gateway. For downcalls, the gateway is transparent. When it receives a method invocation targeted for an endpoint for which it is a gateway, it will pass the method invocation (along with the method and any dependencies, if necessary) on to the endpoint. It will then wait for any method results and pass them back to the original caller.

• Listen for endpoint upcall requests

If an endpoint needs to invoke an operation on another system, it must invoke a method on its own endpoint gateway. The endpoint gateway portion of the appropriate application will supply the method. This method will then take advantage of the full function of the managed node on which it resides to resolve the location of the target object and invoke the appropriate method(s) upon it.

• MDIST repeater activities

Endpoint gateways are automatically defined as MDIST repeaters for all of the endpoints they serve. This gives you the benefit of an intelligent distribution mechanism with little or no administrative overhead.

The repeaters provide a fan-out facility for the distribution of files and data in a Tivoli environment. Therefore, if the same file is being distributed to a set of endpoints using the same endpoint gateway, the file need only be sent once to the TMG and the endpoint gateway will then handle distributing the file to the individual endpoints. For example, endpoint gateways can be installed in various branch offices that are connected to a main site via a relatively slow link. The distributed data may only need to pass once along the slower link, and can then be distributed to the individual nodes on the local network at higher speeds.

Please refer to the *Tivoli Framework Planning and Installation Guide* for a more detailed description about MDIST repeaters.

3.3.3 Tivoli Endpoint Manager

The endpoint manager stores the association between endpoint gateways and endpoints.

The endpoint manager maintains an endpoint list, which keeps track of every endpoint in a TMR. This list tracks which are the associated TMG endpoints. Based on site-specific settings, the endpoint manager reassigns endpoints if an endpoint gateway is unavailable and dynamically adds new endpoints as they appear on the network. The endpoint list contains the information necessary to uniquely identify and manage endpoints. This includes:

- The name of the TMA entered in the Tivoli name registry.
- The TMA's interpreter type which is a string denoting the platform and operating system of the TMA, such as NT
- A unique object dispatcher identifier referred to as odnum which is the system identifier for the TMA
- The name of the TMG that is responsible for communications with the TMA

The endpoint manager plays a role in enforcing site-specific system policies. For example, policies can be put in place that specify which endpoint gateway (TMG) will be assigned to new endpoints joining the network. These policies could base their decisions on a variety of information about the TMA, which is included in the TMA's broadcast message when looking for a new endpoint gateway (TMG).

3.4 The TEC Version 3.6

The TEC Version 3.6 has the following enhancements from the previous release of Version 3.1.

3.4.1 Architecture

With Version 3.6 the TEC exploits the new three-tier architecture of framework in getting events from an TMA and events provided by the LCF_adapters. The LCF_adapter is the logfile adapter and is available for all the pre-existing platforms with new platforms supported as well.

We still need to use a full managed node adapter in our environment. First for compatibility reasons all the TIvoli Plus Modules are not yet ported to the three-tier architecture. Another point is getting events from a network management workstation. We still need a full managed node for installing the NetView, Openview or SunnetManager adapters.

3.4.2 Availability and Reliability Improvements

An auto-reconnect feature has been added for the Event Console when the EventServer goes down. Each active console is automatically reconnected to the EventServer when it comes up again.

The same has been added for logfile adapters, in the way that any logfile_adapter polls the EventServer for restoring a lost connection. With the previous version when the EventServer went down, we had to manually restart the adapters for resuming the connections.

The NT event log adapter now passes the actual hostname in the sub_origin slot rather than the adapter hostname, and uses the COMPUTER field in the NT environment. Multiple pre-filtering statements were added for saving processing time at the TEC engine level.

3.4.3 Expanded Adapter Support

The OS/2 endpoint support has been added via a logfile adapter with both a secure (via the LCF) and a nonsecure connection over TCP/IP. The logfile picks up the FFST events from the system error log. OS/2 has a powerful mechanism for monitoring systems events, that is, First Failure Support Technology (FFST). An application can request to log events and FFST will write them to the OS/2 system error log. Each error log record contains:

- The first 32 bytes of the message
- Up to 512 bytes of additional text
- A symptom string including component, release, subcomponent, line number and message number

For OS/390 event integration the TEC now includes class definition files (.baroc) and rules for integrating the NetView OS/390 events. The adapter itself will be shipped as part of the Event and Automation Service with NV/OS390.

A new adapter for Tandem NSK (tecad_TAdapter) is provided for monitoring Tandem systems, made by ESQ as a system integrator. ESQ, as system integrator, will provide both implementation and technical support. It is implemented via a proxy mechanism that collects information coming from multiple Tandem Guardian (NSK) workstations.

3.4.4 Console Enhancements

We are provided with a new facility to modify the screen appearance, column widths and headings can be modified via an idcall command.

When you select from the event panel **Update OFF** for managing events better, whenever any change command is issued such as close event, the automatic Update ON becomes active. This means an operator has the time to work with and analyze events without the display flickering and with no risk of loosing new incoming events.

With previous versions, the console would slow down when the number of open events became large. In TEC 3.6 console performance is dramatically improved.

3.4.5 Performance Improvements

The secure connection between any adapter and the event server is done by opening a connection at a session level over TCP/IP. For optimizing enqueueing of telnet frames the Nagle algorithm has been introduced. This is a delay mechanism. For the TEC Server this algorithm limits the throughput to a maximum of five events per second.

When the Nagle Algorithm is switched off the TEC 3.6, performance tests showed a throughput of 11.1 events per second, in fact more than twice the ratio than TEC 3.1.

3.4.6 Interoperability Scenarios

Although this is not the main issue of this book, we take into consideration some migration problems that can arise. For compatibility with a previous environment, for keeping the Plus modules up and running we might encourage our customer to let different levels of the TEC modules coexist (see Figure 15 on page 44).

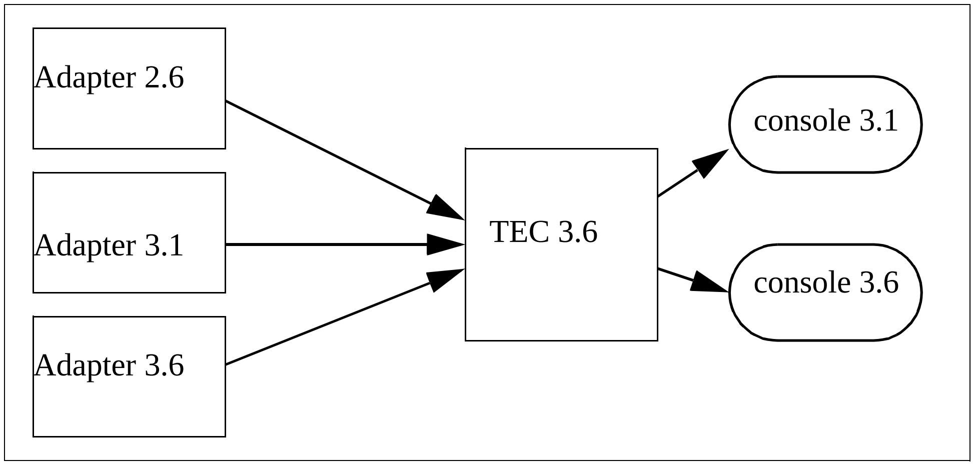


Figure 15. Simple Interoperability Scenario

Generally, any adapter of a previous level will send an event to a 3.6 version of EventServer. The 3.1 console can be connected to a 3.6 TEC engine as well.

3.5 Tivoli Inventory

Tivoli Inventory is Tivoli's application for enterprise inventory management. Tivoli Inventory is able to scan computer systems for a number of attributes of these systems, such as processor model, memory, hard disks and a lot of other information.

This inventory information is then stored in the configuration repository, a relational database in which Tivoli Inventory stores the information it has found scanning the systems and other configuration-relevant data. Other applications use the Tivoli Inventory configuration repository as well.

For example, Tivoli Software Distribution can store information about file packages it has distributed in the configuration repository. Tivoli NetView 5.1, using the TIPN package can extract data from its own network database and store it in the Tivoli Inventory repository as well.

Other applications storing and retrieving information from Tivoli Inventory allows for tight integration between systems management applications, as the applications can share data and exchange and utilize information about the components that make up the IT system.

Tivoli Asset Management can use information gathered by Tivoli Inventory to collect and track asset information.

Detailed information about Tivoli Inventory can be found in the following redbooks:

- *TME 10 Deployment Cookbook: Inventory and Company*, SG24-2120
- *TME 10 Inventory 3.2: New Features and Database Support*, SG24-2135

3.6 Tivoli NetView Version 5.1

This section provides a brief overview of Tivoli NetView 5.1. More detailed information can be found in the redbook *Using Tivoli NetView 5.1*, SG24-5285 (available soon).

For all the SNMP management of our network, we will use the Tivoli NetView application to monitor the SNMP devices such as routers, switches and hubs. The latest release of the Tivoli NetView application provides more integration with the Tivoli framework. We have documented the new features relevant to this project and have highlighted these below:

- Submap sorting capability

  This function allows you to sort all the nodes contained in a submap by certain criteria, taken from the NetView database attribute fields, for example the subnet mask or IP address of the device.

- Status filtering

  A new option had been added to the pull-down menus to allow the hiding or displaying of objects in a submap that have a particular status, for example, do not show the objects with the status of up.

- NetView database enhancements

  The NetView database which is comprised of the managed objects, topology and map information is a collection of several databases. In this release two more files have been defined for each database: a config file that contains parameters for the database and an overflow holds additional data.

New tools for improving database performance have been added for minimize space and maximizing speed.

- Web interface enhancement

  Web client security has been added. Each Web interface requires a NetView user ID and a Web GID. In addition a Web client can be issued both from NetScape 4.0 and Internet Explorer 4.0. The communication between NetView Web Server and a Web client has been enhanced to be faster. Some new utilities have been added for integrating NetView Web Server with an existing httpd.

- New NetView commands

  Some new commands have been included. These are listed below:

nvsniffer For discovering and gathering nodes with defined socket services active. Nodes will be added to a NetView collection.

nvwakeup Used for implementing the Wake_on_LAN capability in NetView.

nvdbformat Used to for exporting selected fields for the NetView database collections based on a set of selection rules.

nvdbimport Used to import object field values targeted at importing multiple field values for a large number of objects using a batch methodology.

ovwexit Used to exit the NetView GUI from the command line.

chmod_web_ovw Used to define the security for the Web interface for the NetView maps.

3.7 Tivoli Environment Discovery from NetView

This new feature allows the NetView map to display a node that is a Tivoli managed object. The NetView map will build TMR environments by interrogating each node and creating a new object field in the NetView database.

NetView performs a query on the name registry or will monitor certain TCP/IP ports.

The new command nvsniffer uses the attributes such as the socket port number used by a defined service and the name of the NetView collection it will use, for example:

```
# /usr/OV/bin/nvsniffer -f /usr/OV/conf/nvsniffer_server.conf
```

where the configuration file contains the following lines:

```
isFTP|21|FTP_Server||*
isDOMINO|1352|Domino_Server||*
```

Once you have discovered your Tivoli environment, you can locate and display the various Tivoli resources within the context of your IP network. You can view the following resources:

- Tivoli Servers
- Tivoli Managed Nodes
- Tivoli PC Managed Nodes
- TMA Gateways
- TMA Endpoints

Then all symbols that either contain the Tivoli resource or that are the Tivoli resource will be highlighted in your NetView submap. An example of this is shown in Figure 16 on page 47.

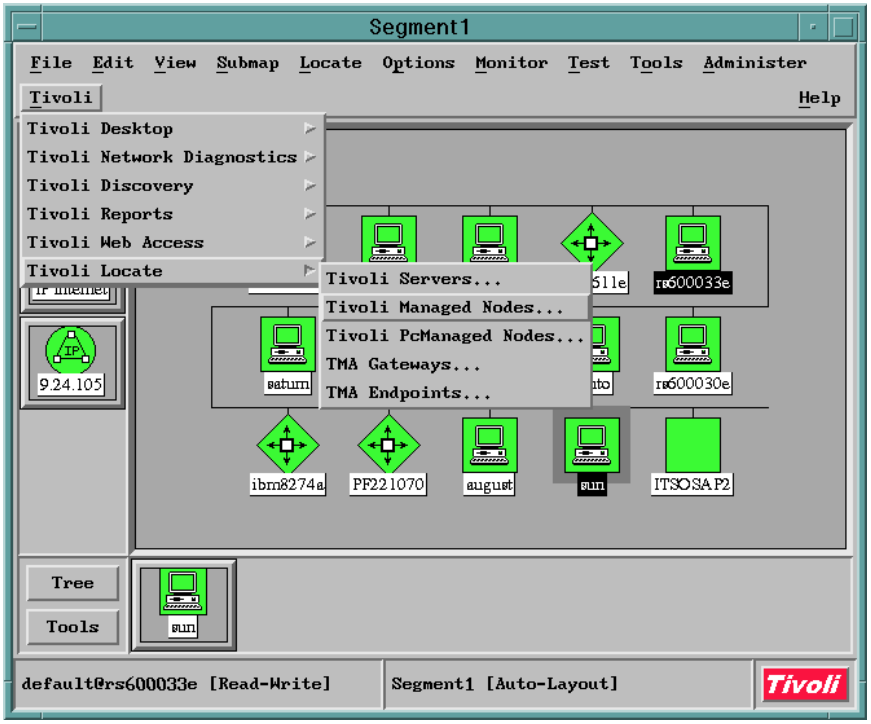


Figure 16. Tivoli Managed Environment from NetView

This feature can be used to monitor the TMR environment.

3.8 NetView for NT New Features

This section briefly describes the new features for the NetView release on NT. These are as follows:

- Submap explorer

  Allows you to navigate the submap hierarchy in an explorer style window. It displays all the objects in a submap and their properties. The explorer window allows you to drill down from the hierarchy to the managed object using attributes such as IP address and events received.

- Service status monitoring using smartsets

  NetView NT Version 5.1 has improved smartset support providing a new command called nvsniffer. The nvsniffer facility dynamically creates smartsets.

- Event correlation using rulesets

  Event browsing can now be filtered by rulesets. By default a set of rulesets are delivered with the application. However the rulesets must be created and modified using the UNIX ruleset editor, then ported to the NT environment. This feature can be used to provide basic event correlation before forwarding events to the TEC.

- NetView NT as an attended MLM

  The Middle-Level Manager (MLM) can now be installed on the same platform as the NetView application. This provides additional filtering and thresholding functionality.

- Installation support for SMS

  The NetView kit includes a Package Definition File (PFF) for NetView. This implies that SMS can create a package for installing and distributing NetView.

3.9 Tivoli Integration Pack (TIPN)

The Tivoli Integration Pack for NetView is intended for integrating NetView and Tivoli applications providing tools common in both system and network management environments.

- Tivoli network diagnostic

  Consists of two modules: a NetView server part and a framework part. Installing both of them you will be able to perform the following functions:

  - Network Diagnostic

    Some new commands are provided these are: wrping, wrtraceroute and wrnetstat. These can be executed from either the Tivoli Desktop or the command line interface (CLI).

  - Tivoli Discovery from NetView

    NetView maps now show the Tivoli managed devices. After the discovery process is complete we can locate the TMRs, managed nodes and endpoints for single or multiple TMRs. This can be useful for endpoint diagnostics.

    NetView can query the Name Registry or monitor TCP/IP ports and will add a new field in NetView object database.

- Tivoli Wake on LAN (WOL)

  This tool can be used to wake a machine up that is compliant with the Magic Packet specification for wake up nodes. This function is useful for software distribution environments.

- Tivoli Inventory integration adapter

  With the installation of the adapter module on the NetView server and the profile module on every Tivoli managed node, we can distribute an inventory profile that scans the NetView object database for every subscriber’s entry. A new query library is created and similar to the Inventory Query Library, we can perform a query before distributing software. Hardware, software and network information is collected in the same Inventory database.

- Tivoli TEC Integration adapter for NetView

  The network operator can now select all events that the TEC receives for a specific node, viewing all the slots as in TEC format. We can use this function to build rulesets for event forwarding to the TEC. The TEC operator might filter only events coming from a NetView collection using the EventGroup definition.

  TEC operator can launch a NetView submap on a selected event, as well as issuing some NetView tasks. The connection between TEC and NetView is achieved by the command dispsub.

3.9.1 Tivoli Reports

By populating the NetView database with the TMR attributes we can now retrieve statistics on how our Tivoli environment is deployed via the reporting functionality. Web-based reports can be run for the following:

- Tivoli servers by segment
- Managed nodes by Tivoli server
- Unmanaged nodes by segment
- PC managed nodes by proxy
- Gateways by segment
- Endpoints by segment
- Endpoints by gateway
- Endpoints by interpreter type

Chapter 4. Implementing the Management Software

This chapter provides a step-by-step guide on how to install each of the management applications using examples.

At this point we are ready to install the applications. We have now established what information we need to perform the installation and customization of the selected tools.

For specific architectural decisions such as sizing, please refer to the manual *SA Expertise for ESM Infrastructure Planning Guide.* However, we have included some of the information contained in the product announcement.

4.1 Software

The proposed installation for the software applications is shown in Figure 17 on page 51. This diagram shows the management platforms and identifies what software is to be installed on each of the servers.

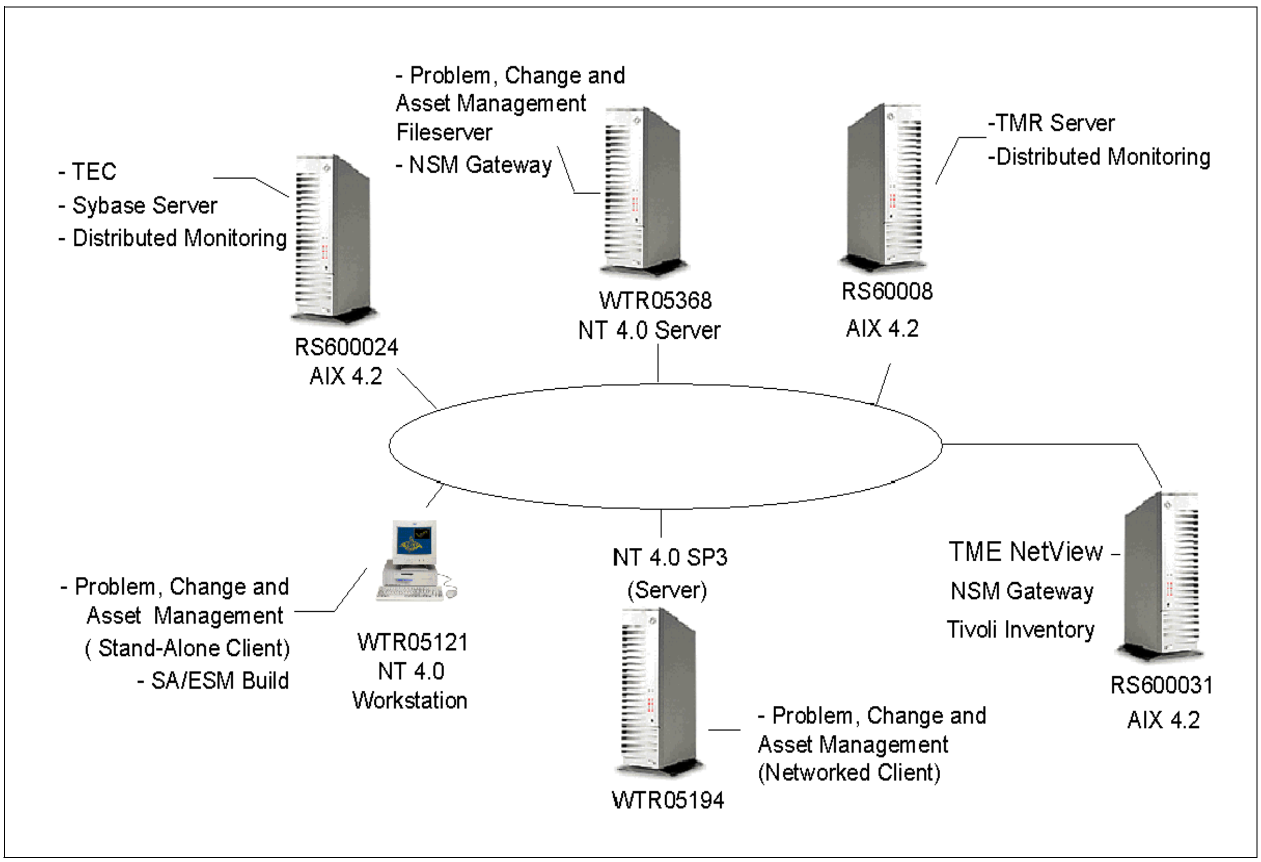


Figure 17. The Management Environment

We tested both the network and stand-alone client installations for access to the problem, change and asset management functions.

In addition to the problem, change and asset management software we also installed Tivoi Decision Support on the file server. The Tivoli Inventory configuration used an Oracle database defined on rs600031.

The implementation of these applications is shown in the next section.

4.2 Implementation Plan

The tasks we performed are detailed in Table 4 on page 52.

Table 4. Implementation Plan

Installation Tasks	Server
TMR	RS60008
TEC	RS600024
NetView	RS600031
Sybase Server	RS600024
Sybase Client	All Machines
Problem Management Server	WTR05368
Asset Management Server	WTR05368
Change Management Server	WTR05368
Clients	All client machines
Tivoli Decision Support	WTR05368
Tivoli Inventory	RS600031

Not all the installation steps are documented in this redbook; however, we do refer to the correct manual for installation instructions where appropriate. Each of the tasks are detailed in the following sections.

4.3 Hardware and Software Prerequisites

This section describes the hardware and software prerequisites.

Hardware Requirements:

The Tivoli Service Desk hardware requirements fall in one of the categories below, based on the type of installation and environment for your organization.

The requirements fall into the following categories:

- Client workstation
- Low-end or high-end client workstations
- File server guidelines
- Application server
- Low, medium, or high-end application servers

Table 5. Prerequisites

Operating System	Low-End Workstation	High-End Workstation
Client Workstations		
Windows 95, Windows NT 3.5.1 or Higher, OS/2 Warp or Higher	Intel Pentium (TM) 120 32 MB RAM SVGA monitor 800x600	Intel Pentium II 266 64 MB RAM SVGA monitor 1024x768
AIX 4.2 of Higher	Power 2 at 67 Mhz PowerPC at 120MHz	PowerPC at 333 MHz
Solaris 2.5.1 or higher	SPARC HyperSPACR at 150 MHz	SPARC UltraSPARC 11 at 250 MHz
HP/UX 10.20 or higher	PA-RISC ar 125 MHz	PA-RISC at 180 MHz

Use of a file server in the Tivoli Service Desk architecture depends on how the client is configured. For example, if the Tivoli Service Desk client software resides on the file server as a shared resource, and each client loads the application from the file server, then a highly fault-tolerant configuration is recommended (see Note). However, if the Tivoli Service Desk client software resides on each client workstation, then configuration of the file server is not needed.

Note: Tivoli Service Desk requires 100 MB available space on the file server and an additional 25 MB for each file server installed.

All application servers require TCP/IP protocols. The following guidelines can be used for the low-end, medium, and high-end application servers in your organization.

Table 6. Application Server Requirements

Operating System	**Low**	**Medium**	**High**
Windows NT	Intel Pentium P5 166 Mhz 64 MB RAM	2 x Intel Pentium P5 166 MHz 128 MB RAM	4 x Intel Pentium Pro 200 256 MB Ram
AIX 4.2 or Higher		Equivalent UNIX performer	Equivalent UNIX performer
Solaris 2.5.1 of Higher		Equivalent UNIX performer	Equivalent UNIX performer
HP/UX 10.20 or Higher		Equivalent UNIX performer	Equivalent UNIX performer

The following are application server platform guidelines your organization can use, based on the number of clients and number of calls per hour.

Table 7. Application Server Requirements

Number of Clients Per Server	Number of Calls Per Hour	Tivoli Service Desk Application Server
01-20 Clients	Less than 50	Low End or Medium
21-75 Clients	51-200	Medium or High
76-200 Clients	201-700	High End
201-1000 Clients	More than 700	More than one application server

Note: Each high-end Tivoli Service Desk Application Server can handle as many as 200 concurrent clients. Application servers should be scaled by one server to each block of 200 concurrent clients.

4.4 Software Prerequisites

The software requirements are as follows:

- Tivoli Change Management (5697-TCM) requires Tivoli Problem Management (5697-TPM).
- Tivoli Asset Management (5697-TAM) requires Tivoli Problem Management (5697-TPM).
- Tivoli Service Desk (5697-TSD) and Tivoli Problem Management (5697-TPM) do not have any software prerequisites.

The Tivoli Service Desk supports the following operating systems:

OS/2 OS/2 Warp 3 with FixPak 26 or OS/2 Warp 4

Windows 95 Windows 95 running as a native 32-bit Windows application

Windows NT Windows NT 3.5.1 with Service Pack 5 or higher running as a native 32-bit Windows application

AIX AIX 4.2 or higher

Solaris Solaris 2.5 or higher

HP/UX HP/UX 10.20 or higher

The database management systems (DBMSs) supported are:

DB2 2.11 or higher

Informix Online Dynamic Server Version 7.1 or higher

SQL Server 6.5 (with Service Pak 3 installed) or higher

Oracle 7.2.x/7.3.x or higher

Sybase System 10 10.0.3 or higher

The Tivoli Service Desk products require 32-bit database client drivers for all platforms. The following are the additional standards for database client driver versions. Major releases that are not listed have not been tested.

Table 8. SQL Client Requirements

Database	OS/2	Windows	UNIX
DB2/2	2.11 or higher	2.11 or higher	2.11 or higher
Informix	Informix Runtime for ESQL/C for OS/2 Version 5.0.1 SD22	7.2	7.2
SQL Server	Not supported	DB-Library 6.5	ODBC
Oracle	Required support files 7.2.x	Required support files 7.2.x/7/3/x	Required support files 7.2.x/7.3.x
Sybase	Open Client/C DB Lib 4.6 NetLibrary 10.0.3	Open Client/C DB Library 10.0.3	Open Client/C DB Library 10.0.3

Additional information:

- SQL*Net products should be from the 2.x series. Also, SQL*Plus for Windows (for diagnostics during installation).
- SQL*Net products should be from the 2.x series. Also, SQLDBA for OS/2 (for diagnostics during installation).
- Open client 10.0.3 recommended with Engineering Bug Fix (EBF).

4.5 Installing the Tivoli Management Environment

The Tivoli environment was built as documented in the *Tivoli Framework Installation Guide*. We used Version 3.6 for the framework. We added a number of administrators and modified their priorities.

The TEC installation was performed from the Tivoli desktop. The TEC customization is documented in Chapter 6, “Problem Management Scenario Using the TEC” on page 241 of this book. In short, the modifications we made to the TEC are as follows:

- Create/defined the Sybase RIM object
- Created a specific TEC administrator
- Created a new rulebase for our event correlation customization
- Created new event groups
- Assigned these event groups to the administrator’s consoles

NetView was installed on rs600031. The procedure we followed is documented in the redbook *Examples of Using NetView Version 5.1,* SG24-5285*.*

We installed the distributed monitoring code on all the machines in our environment. The monitors distributed where necessary.

The TMR wlsinst output is below:

```
*----------------------------------------------------------------------------*
                              Product List

TME 10 Framework
    rs600024      aix4-r1
    rs60008       aix4-r1
    rs600031e     aix4-r1
    wtr05368      w32-ix86
    wtr05219      w32-ix86

TME 10 Enterprise Console Adapter Configuration Facility 3.6
    wtr05219      w32-ix86
    wtr05368      w32-ix86
     rs600031e aix4-r1

TME 10 Inventory, Version 3.6
    rs600031e     aix4-r1
    rs600024      aix4-r1
    rs60008       aix4-r1
    wtr05368      w32-ix86

TME 10 Inventory Gateway, Version 3.6
    rs600031e     aix4-r1
    wtr05368      w32-ix86

TME 10 Distributed Monitoring NT Monitors
    rs60008       aix4-r1

SAExpertisePlus for Tivoli, Version 5.0, Revision 1
    rs600024      aix4-r1
    rs60008       aix4-r1
    wtr05368      w32-ix86

TME 10 Distributed Monitoring 3.6
    rs600031e     aix4-r1
    rs600024      aix4-r1
    rs60008       aix4-r1
    wtr05219      w32-ix86
    wtr05368      w32-ix86

TME 10 Distributed Monitoring Sybase Monitors
    rs600024      aix4-r1

TME 10 Enterprise Console Console 3.6
    rs600024      aix4-r1
    rs60008       aix4-r1

TME 10 Enterprise Console Server 3.6
    rs600024      aix4-r1
    rs60008       aix4-r1

Tivoli NetView/Inventory Integration Adapter for NetView Server
    wtr05219      w32-ix86

Tivoli NetView/Inventory Integration Profile
    rs600031e     aix4-r1
    wtr05219      w32-ix86

TME 10 Distributed Monitoring Unix Monitors
    rs60008       aix4-r1
```

4.6 Installing the Sybase Server

The sybase version we used was Version 11. The installation instructions we followed are contained in the redbook *TEC Implementation Examples*, SG24-2510. We have documented the specific sybase creation of database tables.

To enable the Tivoli Service Desk applications to work properly you have to set up the sybase database server correctly. This is described in the *Tivoli Service Desk Installation Guide*. We used the same database for both the TEC and Tivoli Service Desk.

4.6.1 Creating the Device for the TEMPDB Database

We create a new device for the TEMPDB. Since the default size of the temporary database is only 2 MB, we extended this to 10 MB.

Figure 18. Creating the Device for the TEMPDB Database

Line 3 shows the parameter physname. This is the physical name of the device. We chose not to create a file device instead of a real device since it is more easy to configure. The parameter vdevno is the virtual device number. You can choose a number between 4 and 9. 1, 2 and 3 are reserved for the system devices.

Also note that the size here is specified in 2k blocks, which means that size=5000 gives the following equation: 2000 bytes x 5000 = 10 000 000 bytes.

After typing `go` and press Enter to continue.

4.6.2 Creating the Database TEMPDB

After creating the ADVISOR1_TEMP device we have to create the TEMPDB database on top of that device. This is done by using the command shown in Figure 19 on page 58.

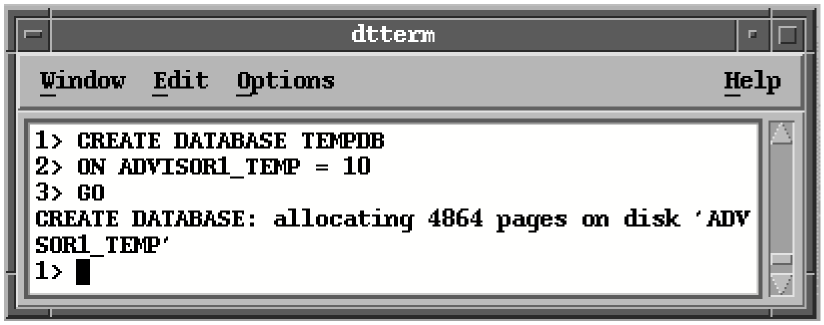


Figure 19. Creating the Database TEMPDB

Here we found a problem in the Tivoli Service Desk documentation. The information tells you to execute the command ALTER DATABASE, which is not correct. It should read CREATE DATABASE.

4.6.3 Creating Devices for the ADVISOR Database

After the creation of the ADVISOR1_TEMP device and the TEMPDB database, we create some other devices that are necessary for the ADVISOR database:

Table 9. Devices in the ADVISOR Sybase Server

Device Name	Physical Name	Vdevno	Size (2K blocks)
ADVISOR_DATA	$SYBASE/ADVISOR.DAT	5	50000
ADVISOR1_LOG	$SYBASE/ADVISOR1.LOG	6	10000
ADVISOR1_NDX	$SYBASE/ADVISOR1.NDX	7	20000

The SYBASE variable is defined when you install the sybase server. In our environment it is set to /usr/local/sybase.

The device ADVISOR1_LOG, which is the transaction log device for the ADVISOR database, should be on a separate physical disk. It is important to separate the logs from the data. The same is true for the ADVISOR1_NDX device.

4.6.4 Creating the ADVISOR Database

Now we can create the ADVISOR database.

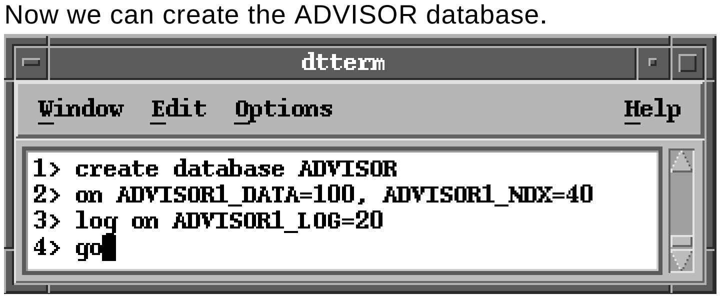


Figure 20. Creating the ADVISOR Database

Here you can see how we can use the data device, the index device and the log device to create the database ADVISOR.

Since we created regular files instead of AIX logical volume we can easily check that the files were created and their size by issuing the command ls -la $SYBASE.

4.6.5 Creating the Segments

The next step was to create segments on the ADVISOR database to improve performance of indexing. The segments are used by the ESMBuild application to place objects on different physical drives. To select the database we used the SQL command use ADVISOR (see Figure 21 on page 59).

Figure 21. Using the Database ADVISOR

Next we added the segment as shown in Figure 22 on page 59.

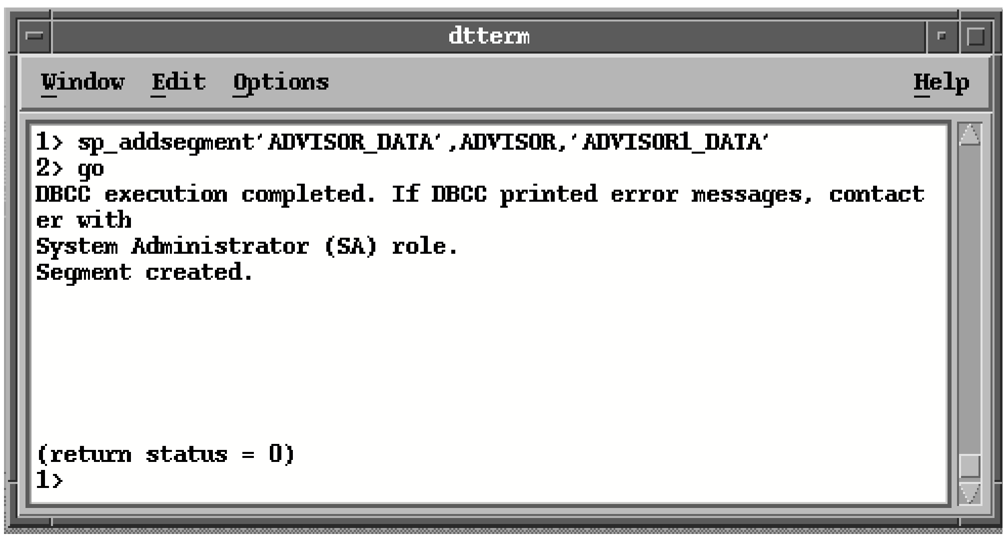


Figure 22. Adding a Segment for the ADVISOR Database

We repeated the command, but this time we replaced DATA with NDX to create the index device segment.

4.6.6 Changing the Owner of the ADVISOR Database and Creating Users

Next we added the administrative user and assigned it the ownership of the ADVISOR database.

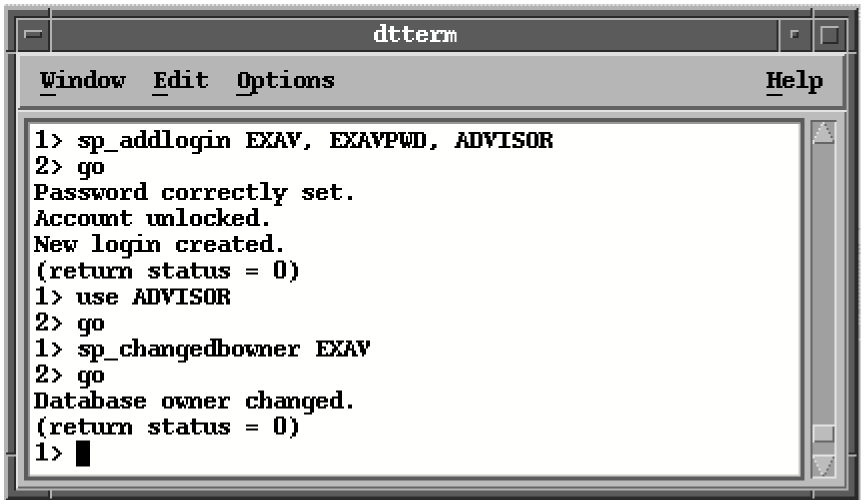


Figure 23. Creating Administrative User for ADVISOR Database

When you are going to add a new Expert Advisor user you have to complete the steps shown in Figure 24 on page 60. Otherwise the user will not be able to create and update the database records.

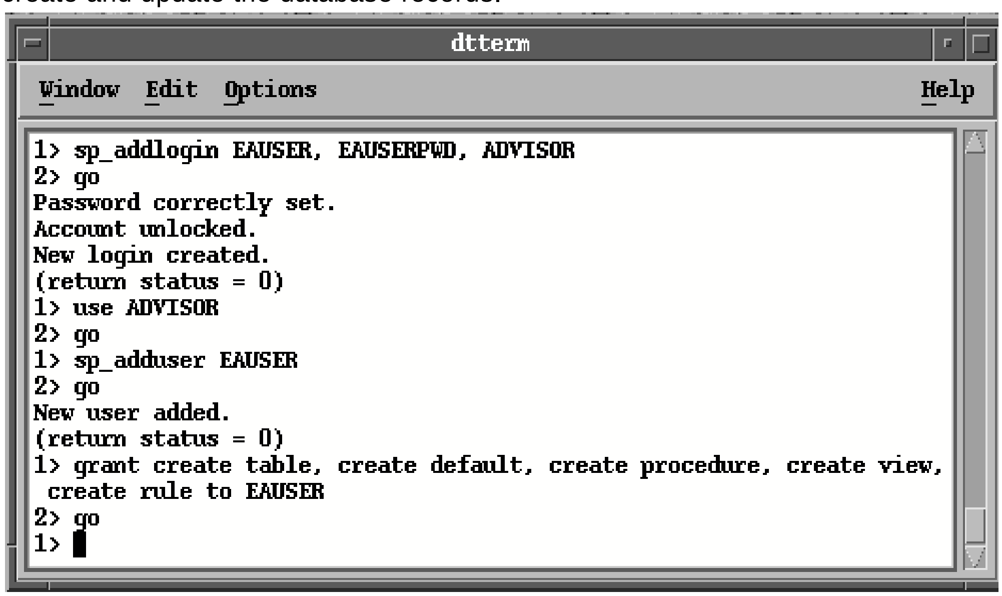


Figure 24. Creating Sybase User

To create another user, just replace EAUSER with the user name and EAUSERPWD with the password of the user to be created.

4.7 Installation of the Sybase Open Client

The first thing that you have to do on the clients before installing any of the products in the Tivoli Service Desk suite is verify that the client has an SQL connection to the database that is going to be used.

In our case we installed the sybase server on the machine rs600024, which means that it automatically gets the SQL connection to the database.

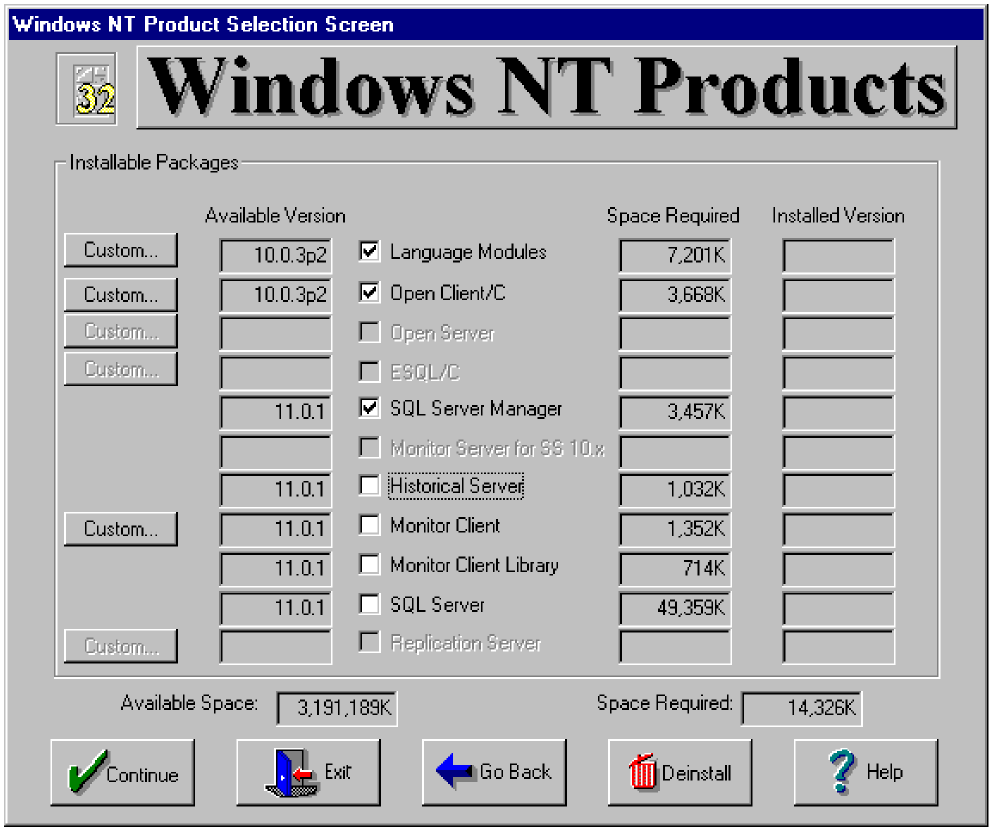


Figure 25. Installing the Sybase Open Client

We selected the **Open Client** option as shown in Figure 25 on page 61.

On an NT machine we installed the open client from a sybase for NT CD-ROM. By installing the client we get two programs to set up and test the SQL connection to our SQL server. These programs are sqledit.exe and sybping.exe. sqledit is used for setting up the connection and sybping is used to test the connection. The SQL screen is shown in Figure 26 on page 62.

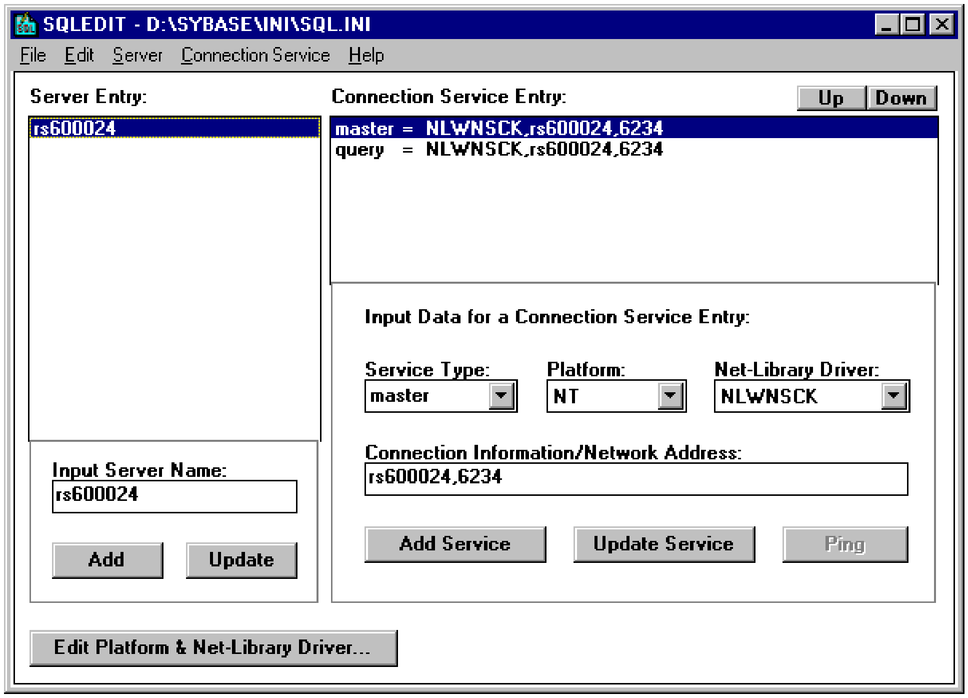


Figure 26. Sybase SQL Client Setup

The Server Name shown in Figure 26 on page 62 is the name of the sybase server instance. It is important to remember that this name is not necessarily the same as the hostname of the machine. We decided to give the sybase server the same name as the hostname of the machine. The Connection information/network address on the other hand is the hostname of the AIX server followed by a comma and the port name of which the SQL server listens.

The port that should be used can be found in the $SYBASE/interfaces file on the SQL server host. In this file you will also find the name of the sybase server instance. When you have set up your SQL connection you can test it with the sybping program.

4.8 Installation of Tivoli Service Desk

This section describes how we installed the Tivoli Service Desk product suite and the Tivoli Service Desk PLUS module. The two primary applications are the Expert Advisor and ExpertView.

4.8.1 Installation of Application Software Expert(ASE)

The Application Software Expert needs to be installed on all the components of Tivoli Service Desk. This is the foundation of the Tivoli Service Desk. All the applications, such as Expert Advisor, rely on ASE for the setup of the SQL connection and the script parser to parse the knowledge base files.

There are three different kinds of installation for ASE. These are:

1. File server
2. Workstation (run from the network)

3. Workstation (stand-alone)

When you perform a file server installation the machine you install on will act as a file server for all the networked clients of the same operating systems type as the file server. For example, a file server that is installed on a Windows NT machine can act as a file server for NT and Windows 95, but not as a file server for a UNIX client.

The only way to have a UNIX machine as a file server for an NT workstation is to install an NFS applications on the NT machine, which enables it to connect to NFS file systems. This is due to the fact that the NT file server uses NT shared file systems for the clients to access and the UNIX file server uses NFS for the clients to access.

4.8.2 Installing ASE As a File Server

As we are about to install ASE on our AIX machine rs60024 as a file server to serve our UNIX client with binaries, we encountered a problem. The installation script install.ase failed because it did not accept AIX 4.3 as a supported operating system, so we had to make some changes to the install.ase script.

In the function TestOS we changed to the following:

```
case ‘uname -r‘in
        2)
                OS=aix
                ;;
        3)
                OS=aix
                ;;
        *)
                OS=NotSupported
                ;;
esac
```

Because we had to move the install.ase script from the CD-ROM to be able to edit it, we have to correct some of the search paths in the script.

In the local variable section:

- Add the variable `CD_DIR=/infocd`.
- Change the variable `AIX_DIR=AIX` to `AIX_DIR=$CD_DIR/AIX`.
- Change `aix_dir=aix` to `aix_dir=$CD_DIR/aix`.

In the check for install.dfc file we did the following:

- Change `[ if -f INSTALL.DFC ]` to `[ if -f $CD_DIR/INSTALL.DFC ]`.
- Change `InstallDfc=./INSTALL.DFC` to `InstallDfc=$CD_DIR/INSTALL.DFC`.
- Change `[ if -f install.dfc ]` to `[ if -f $CD_DIR/install.dfc ]`.
- Changed `InstallDfc=./install.dfc` to `InstallDfc=$CD_DIR/install.dfc`.

You can also perform a workaround. See Chapter 4.11, “Installation of ExpertView NSM Commands” on page 82 for instructions.

The next thing we did was to install the ASE as a file server on rs600024. To initialize the installation, we first ran the modified install.ase script. The output is shown in Figure 27 on page 64.

```
dtterm
Window  Edit  Options                                                    Help
------------------------------------------------------------------------------
 SA-ASE 5.0.1
------------------------------------------------------------------------------


                   Select Installation Type:
                   -------------------------
        1 - Install on File Server
        2 - Install on Client Workstation (Run from Network)
        3 - Install on Client Workstation (Standalone)
        4 - Cancel

Enter one of the above options. (1-4) [2] 1


           Select Components to Install:
           -----------------------------
        1 - ASE Runtime
        2 - Update Workstation Configuration
        3 - Install Example Code
        4 - Developer's Toolkit
        5 - Cancel

Enter one or more of the above options, separated by spaces. (1-5) [1 3] 1

Please enter the root directory for SAI products. [/usr/lpp/sai]
/usr/lpp/sai already exists. Do you want to overwrite ? [no] y

Ready to install SA-ASE.  Continue? [yes]

Creating Directories...
Expanding installation files...
Done with installation. aix




Be sure to run the SQL Configuration Editor to configure ASE for use
with your database.
Press the <ENTER> key to continue.

Please review the readme file READ_501.TXT before using this new version of ASE.
This file contains important installation information that is not available in the
 written documentation.  Would you like to view the readme file now? [yes] n

Done with installation.
```

Figure 27. Installation of ASE As a File Server

4.8.3 Installing ASE As a Networked Workstation

Installing a networked workstation means that you do not place any binaries on the workstation; all binaries are read from the file server.

When installing a Windows networked client, you have to map a network drive to the directory on the file server where the saiapp.ini file is located. This directory is also called the SAI_ROOT directory. If you remove the mapping of the drive and try to start a Tivoli Service Desk application, the application will automatically map the network drive. It is important that you always have the same drive letter; therefore it is recommended that you set the option Reconnect at logon the first time the drive is mapped.

If you are installing a UNIX networked client, make sure that the SAI installation directory on the file server is an NFS file system.

The strength of this concept is that when you make a change in an application, for example, a new button or a new field, you will only have to parse the application once on each file server, and the clients will automatically pick up the changes the next time they download the binaries. The binaries are downloaded when you exit and restart the application.

The binaries are not stored on the local machine but remain on the file server disks. In our case we create a shared drive on the server called SAI.

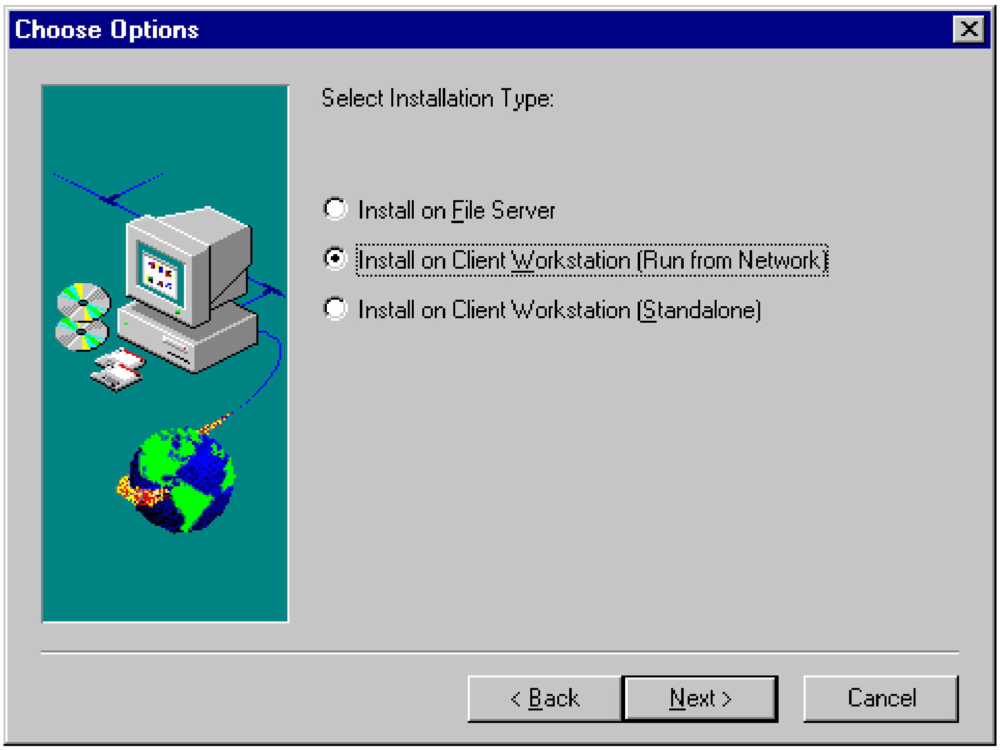


Figure 28. Client Installation from the Network

Next select the options shown in Figure 29 on page 66 and click on **Next**.

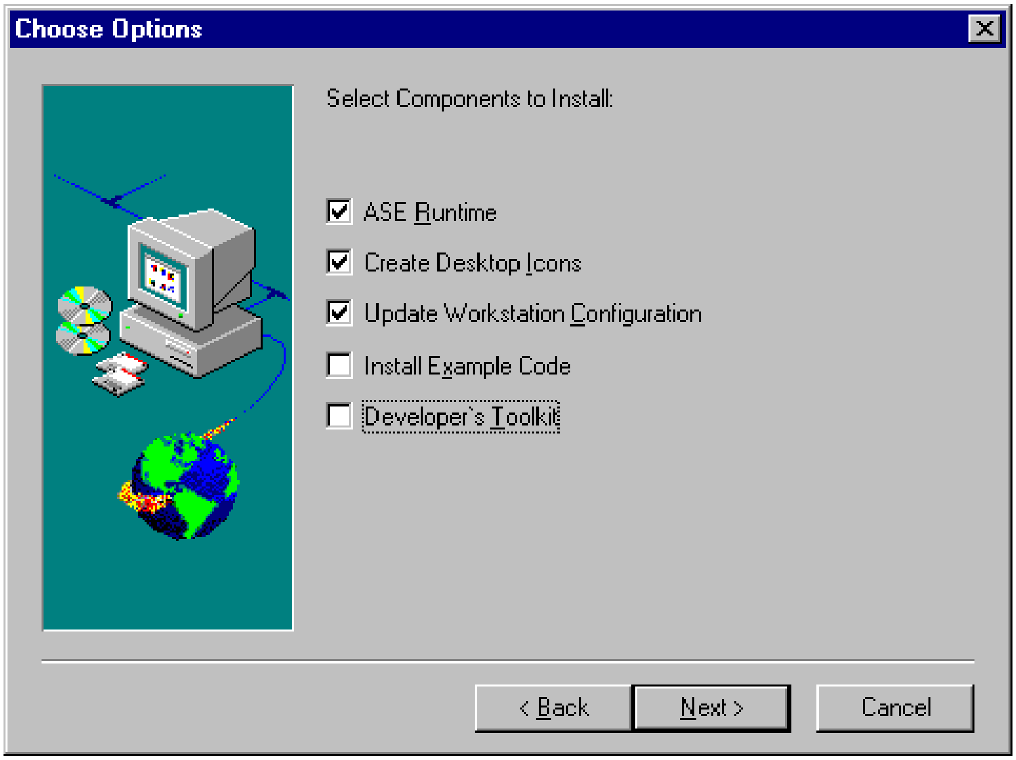


Figure 29. Installing Network Client

We selected **ASE Runtime**, **Create Desktop Icons** and to **Update Workstation Configuration**. Click on **Next** to see Figure 30 on page 67.

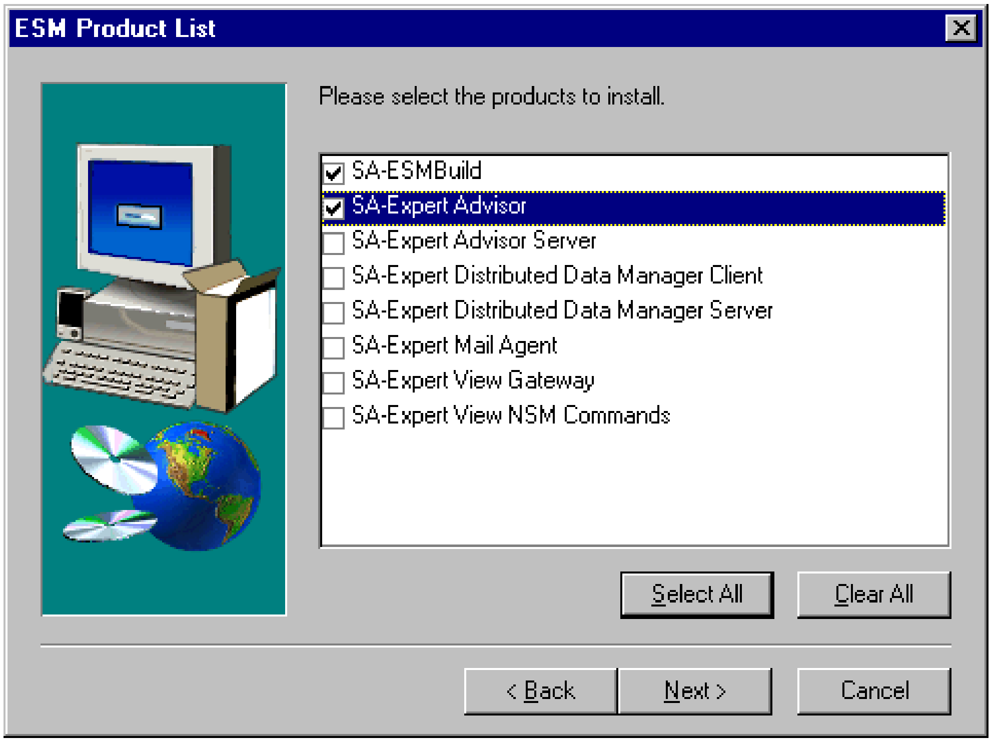


Figure 30. Selecting the Products to Install

Here we selected the SA-ESMBuild and SA-Expert Advisor components. Once selected click on **Next**.

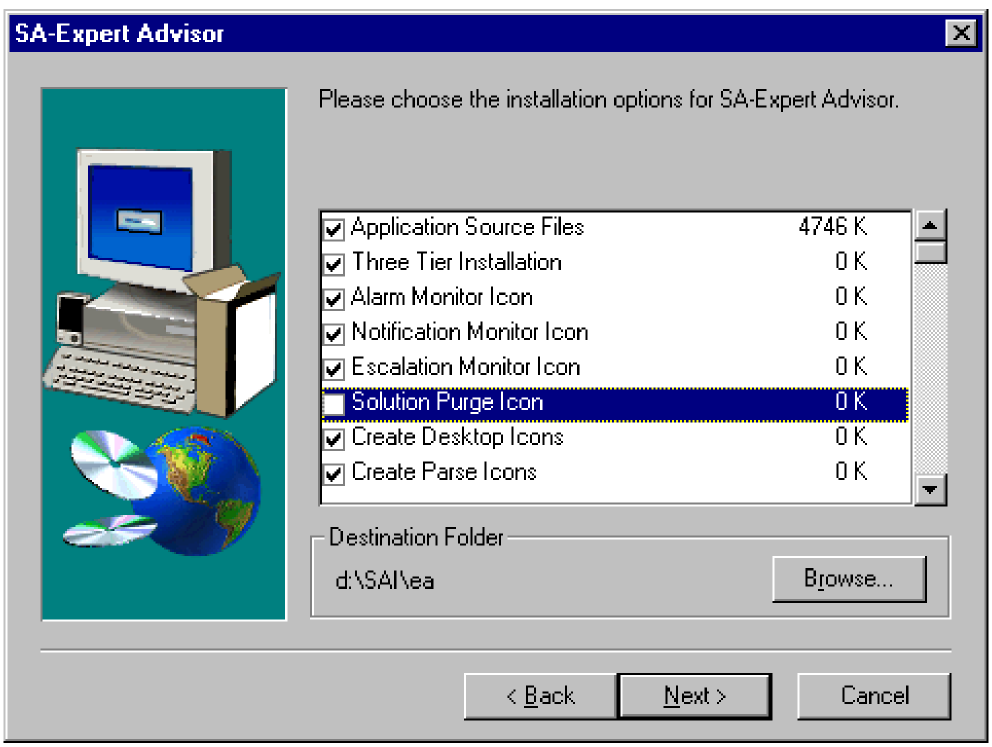


Figure 31. Installation Options for SA-Expeprt Advisor

Figure 31 on page 67 shows the Advisor options we selected. To continue click on **Next**.

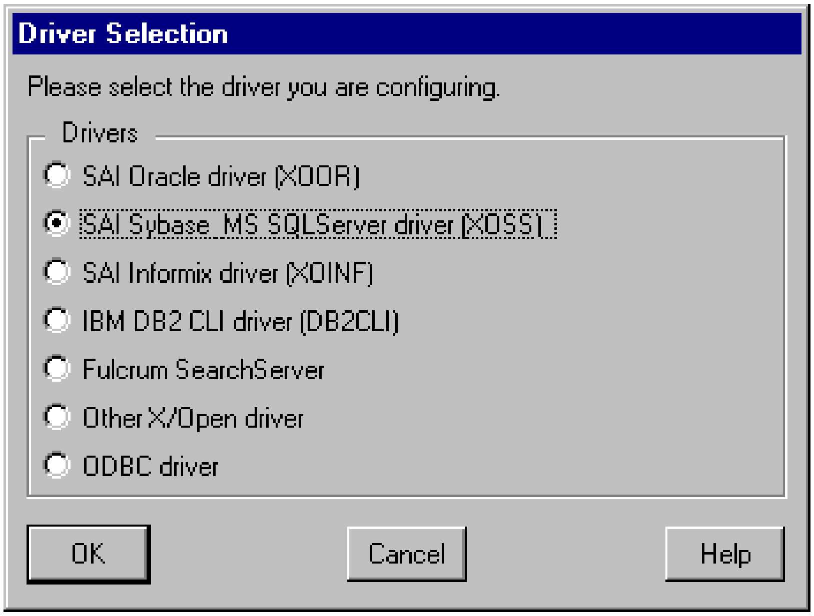


Figure 32. Selecting the Driver for SQL

Select the SAI Sybase selection as shown in Figure 32 on page 68. When complete click on **OK**.

We encountered a problem when starting the application. This was due to the parsing not working correctly. We re-parsed the application and this was ok.

Figure 33. Error When Starting Client

To re-parse the application run the script parser with the options shown in Figure 34 on page 69.

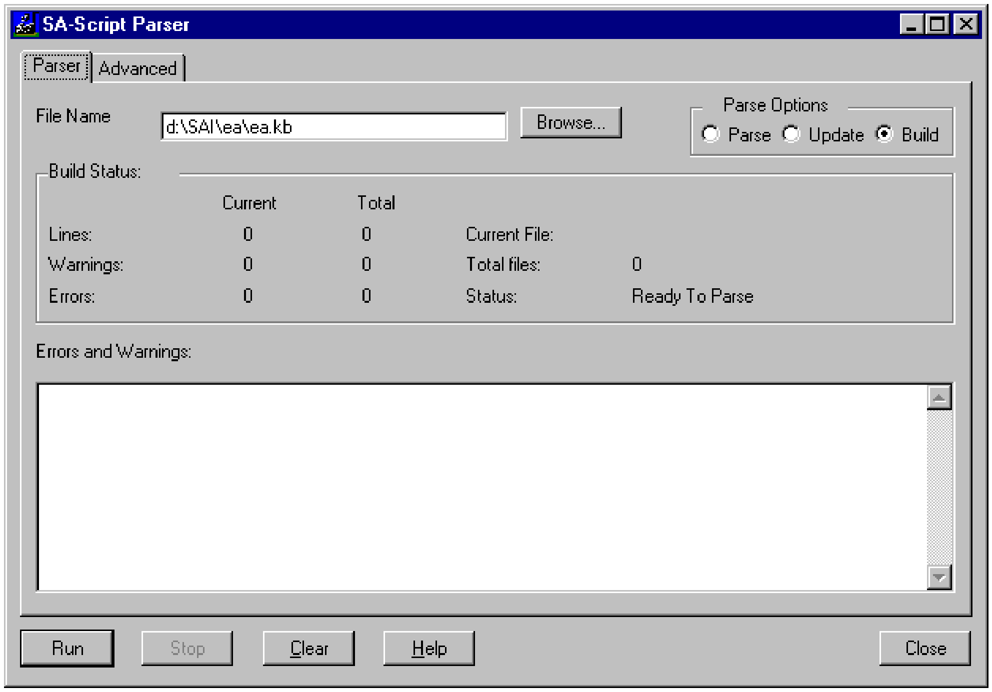


Figure 34. Parsing the ea.kbc File

4.8.4 Installing ASE As a Stand-Alone Workstation

To install ASE on a Windows machine follow the instructions on page 50 in the *Tivoli Service Desk Installation Guide.*

The stand-alone workstation has all the binaries it needs on the local hard drive; therefore it does not need a file server. This can be useful if you have a workstation that acts as a notification manager or escalation manager and has to be able to stop and start the applications without having to access a file server over the network. If the workstation is a stand-alone installation, you would have eliminated the file server as a point of failure.The stand-alone client installation screen is shown in Figure 35 on page 70.

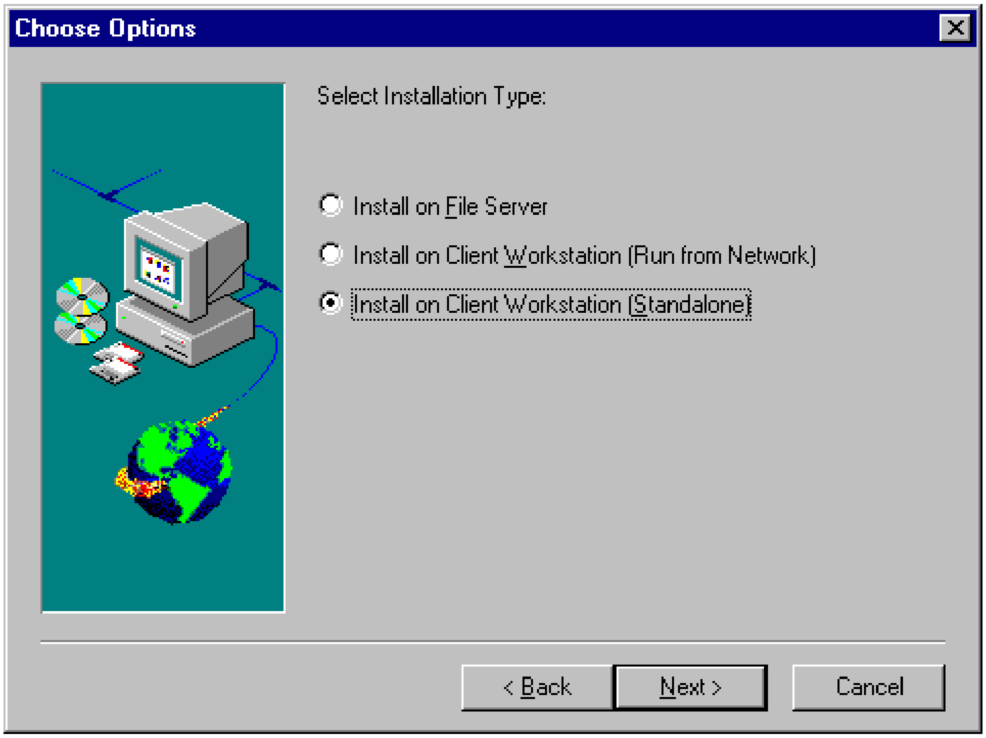


Figure 35. Installing the Stand-Alone Client

The applications must be parsed in order for the applications to function correctly. The parsing screen is shown in Figure 36 on page 70.

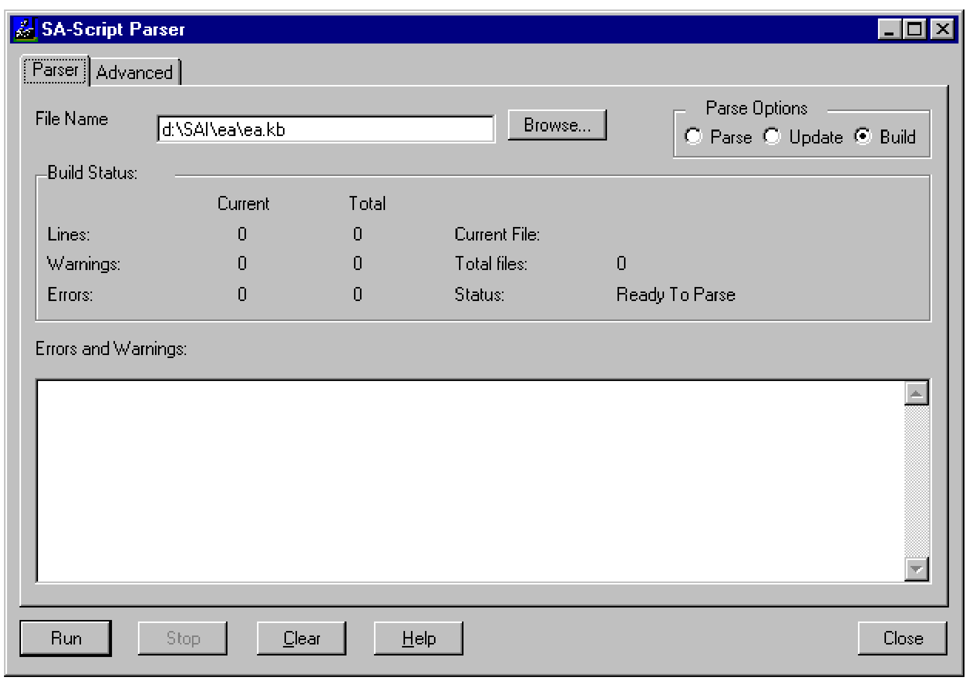


Figure 36. SA-Script Parser

When running the SA application verify when parsing that the variable is set as shown in Figure 37 on page 71.

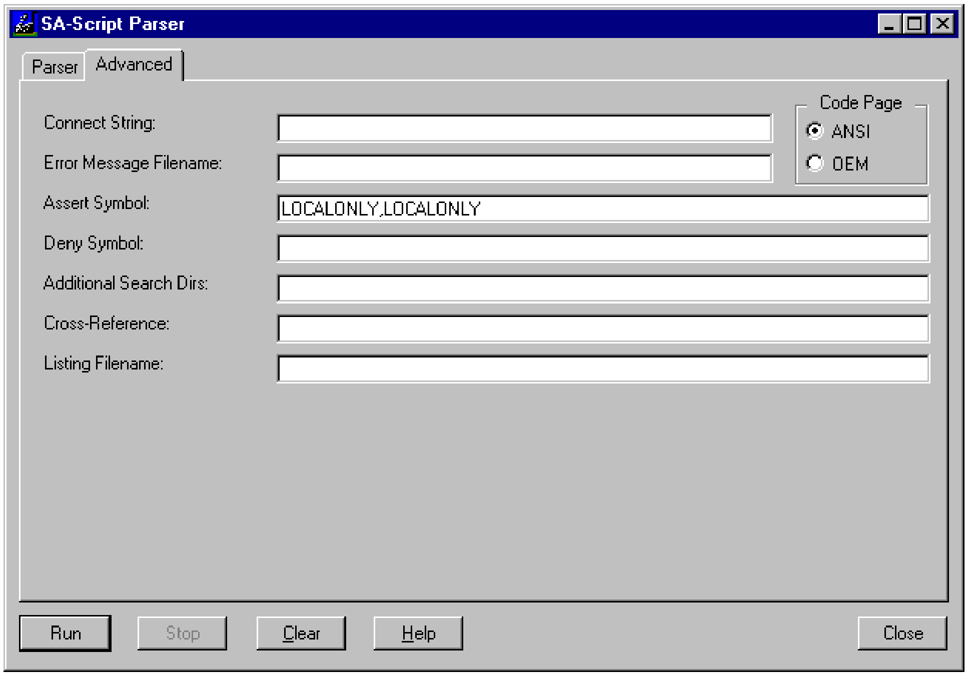


Figure 37. SA-Script Variable Settings

One point to remember is to verify that the Asset Symbol field is set to:

`LOCAL ONLY, LOCAL ONLY.`

4.8.5 Using the SQL Configuration Editor

When you have installed ASE, there is an application called SQL Configuration Editor installed. This application is used to set up the SQL connection to your SQL server. In Windows an icon for this application is created when you install ASE, while in UNIX the command to start the SQL Configuration Editor is:

`kml ce.kbc`

The is SQL Configuration Editor the first time it is started is shown in Figure 38 on page 72.

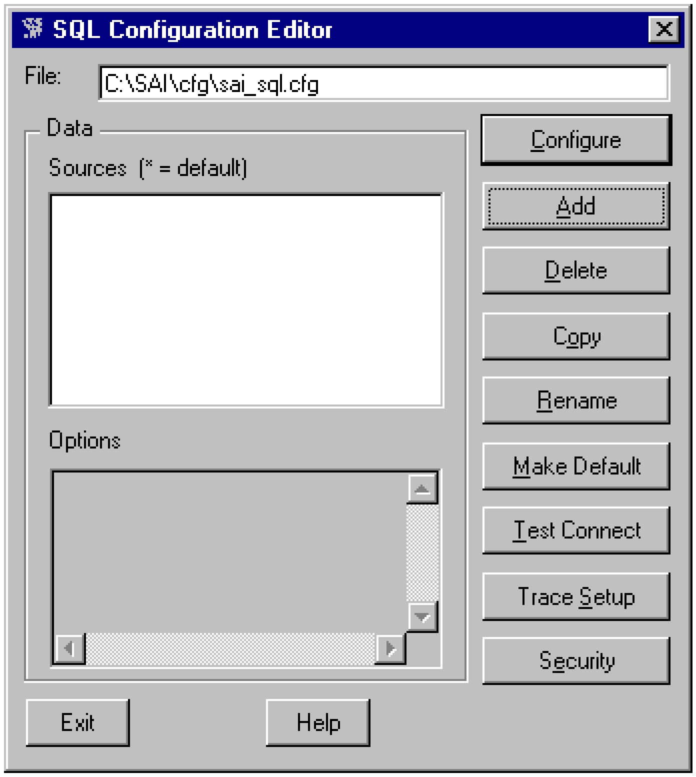


Figure 38. SQL Configuration Editor

If this is the first time you start the SQL Configuration Editor, you will get the question of whether you want to create the sai_sql.cfg file. Just click on **Yes** to create it. This file is used by all the other applications to contact the database.

To add a new source, click **Add**. You will then receive a question about which database driver you have installed; just click on the one that matches your driver. After clicking **OK**, the screen shown in Figure 39 on page 72 will appear.

.

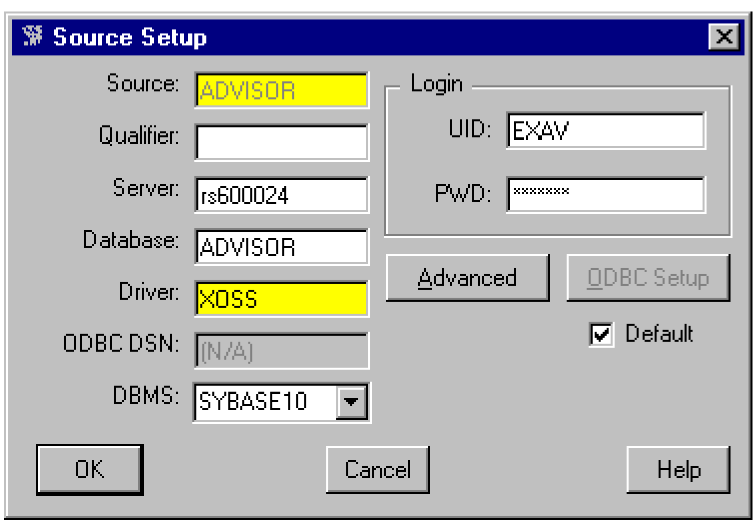


Figure 39. Source Setup

The Source value is the name of the source you are about to create; choose **ADVISOR**. This is important to think about if you have more than one Expert Advisor installation. The server is in this case the name of the sybase server instance found in the interfaces file on your sybase server machine. The Database is the name of the database you created in Part 4.6.4, “Creating the ADVISOR Database” on page 58.

In the DBMS field you just select the database type you have installed. In the Login box you specify the login ID and password to access the database.

Note

Remember to click the **Default** button to set this source as the default. If you do not do this, the application installations will fail with a parsing error.

To perform the advanced setup, click on **Advanced** and you will see Figure 40 on page 73.

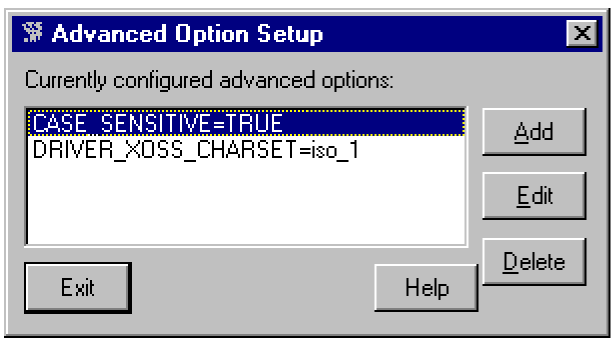


Figure 40. SQL Configuration Editor - Advanced Option Setup Window

Here we added CASE_SENSITIVE=TRUE since our database is case-sensitive. We also have to set the Character set to iso_1 to get the connection to work correctly.

Now go back to the SQL Configuration Editor window and click on **Test Connect**. If you have set up the connection correctly, you will see the following message:

Figure 41. Successful Connection to the Database

Note

If you install ASE as a networked client, it will use the same SQL configuration settings the file server has. In other words, the file server and the client will use the same sai_sql.cfg file.

Now the ASE is installed and configured.

4.9 Installation of Tivoli Advisor

Tivoli Advisor uses the ADVISOR database created by the RBMS software as previously discussed. After creating the database it is still not ready for use with Tivoli Service Desk. ESMBuild is a utility program that creates and maintains the database structure used by all Tivoli Service Desk applications.

4.9.1 Installing ESMBuild

ESMBuild can be installed on any machine that has an installation of ASE and a connection to the ADVISOR database. The ESMBuild provides scripts to maintain the ADVISOR database.

We installed ESMBuild on the machine WTR05368 machine. Running the setup program from the CD-ROM displays a dialog box for entering the authorization key. The product list dialog box appears and lists the available products based on the authorization key. We clear all the boxes except ESMBuild as shown in Figure 42 on page 74. Installation of ESMBuild provides only the application files and start icons. Database structure build still has to be defined and started.

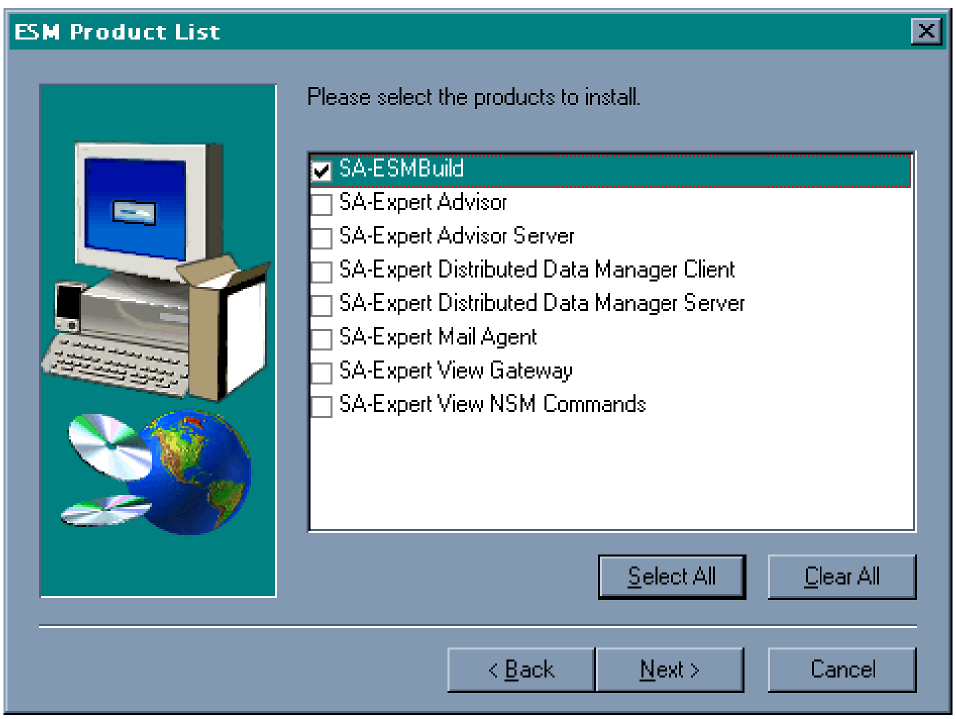


Figure 42. ESM Product List

ESMBuild can be used with any supported RDBMS. Because of differences in RDBMSs, ESMBuild has to be prepared to work with a specific RDBMS. We used sybase and defined the specific variables and ESMBuild-specific options as shown in Figure 43 on page 75.

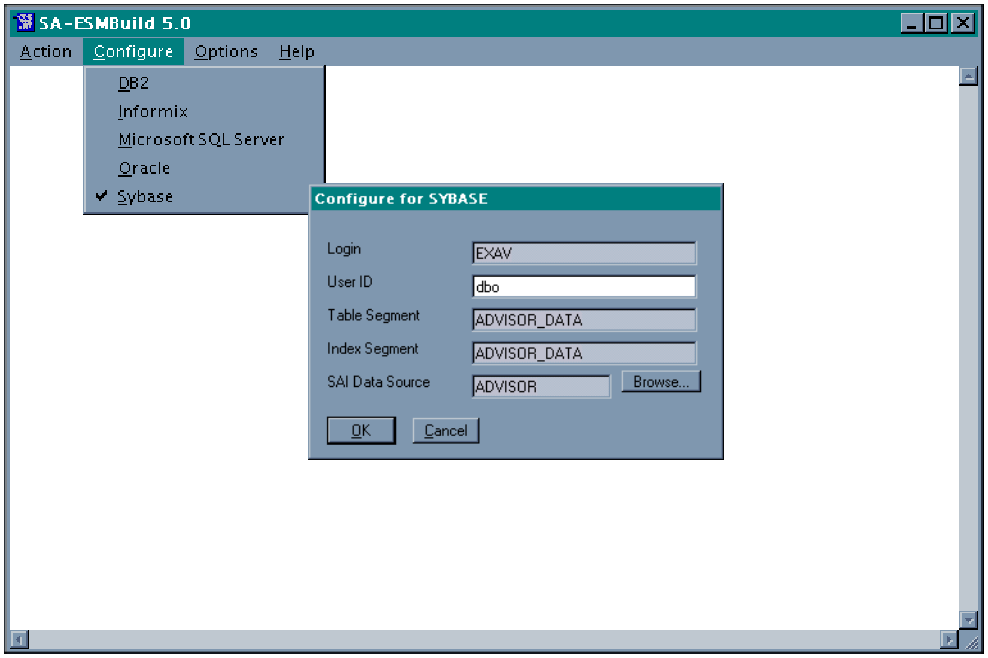


Figure 43. Preparing ESMBuild to Work with Sybase

Next we started to build the database structure. After entering the login and password for sybase, the Table Build Selections dialog window appears. We select actions for the Tivoli Advisor product as shown in Figure 44 on page 76.

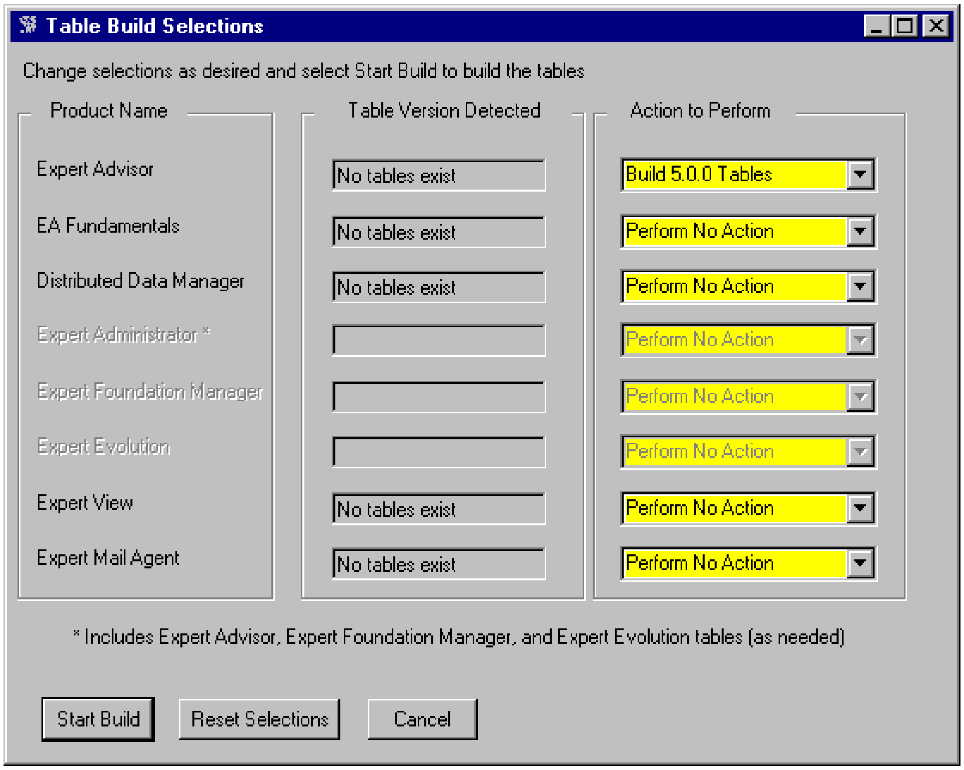


Figure 44. Table Build Selections

After building the ADVISOR database structure, privileges to the database have to be granted. Grant database scripts are provided for each Tivoli Service Desk application. For Tivoli Advisor, eagrant script has to be run to grant Tivoli Advisor rights in the ADVISOR database.

4.9.2 Installing Tivoli Advisor on the File Server

Next we can install the Tivoli Advisor application on machine WTR05368 by selecting the box for SA-Expert Advisor in the ESM Product List dialog box. Tivoli Advisor can be installed on a file server, the client workstation or a stand-alone workstation.

In the Tivoli Advisor dialog box we choose all boxes except the three-tier installation options as shown in Figure 45 on page 77.

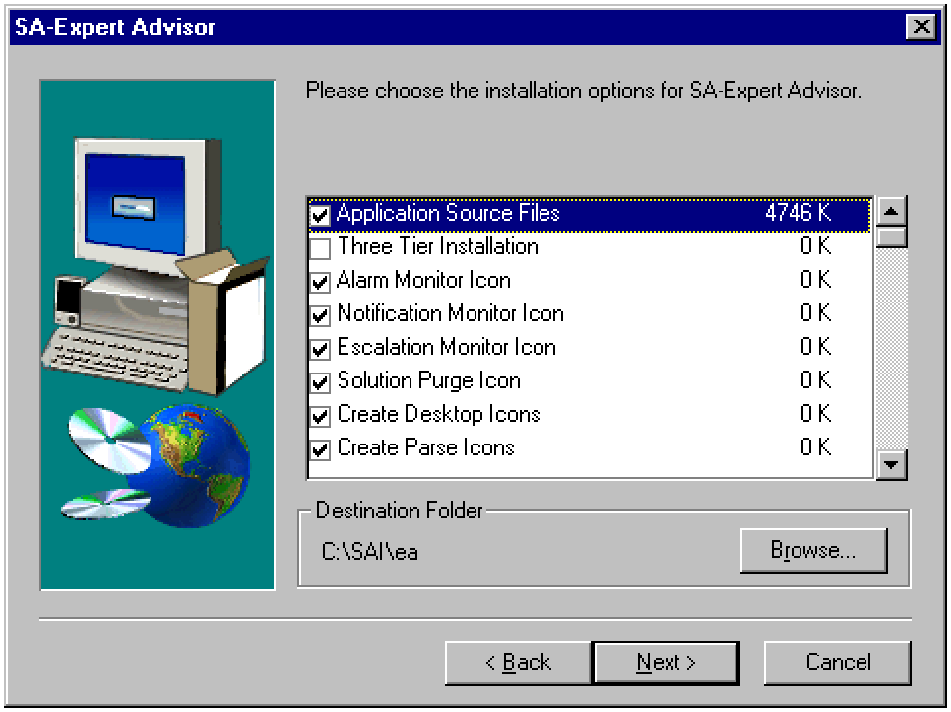


Figure 45. Tivoli Advisor Installation Options

In the second installation option dialog box we choose to have the same logon for the database and the application as shown in Figure 46 on page 78. By selecting this we choose that the user ID and password supplied in the Logon dialog box is used to log on to both the application and the RDBMS.

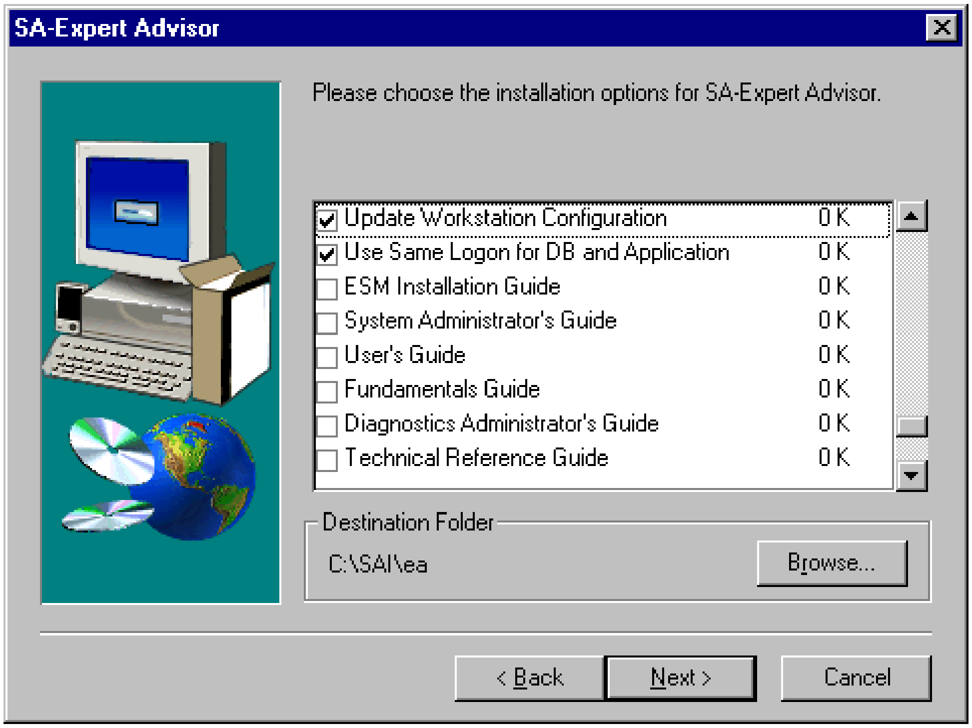


Figure 46. Choosing the Type of Logon

The installation of Tivoli Advisor on the file server is complete. After the installation of the application files the machine has to be rebooted for changes to take effect.

Before using the application all the Tivoli Service Desk applications have to be parsed. Parsing translates a file from ASCII text format into the binary format that SA-Script run-time system can interpret.

We parsed the Tivoli Advisor application by clicking on the **Parse SA-Expert Advisor** icon in the SA-Expert Advisor program group. In the Script Parser dialog box as shown in Figure 47 on page 79 we choose the **Build** option and start the SA-Script Parser on the file server.

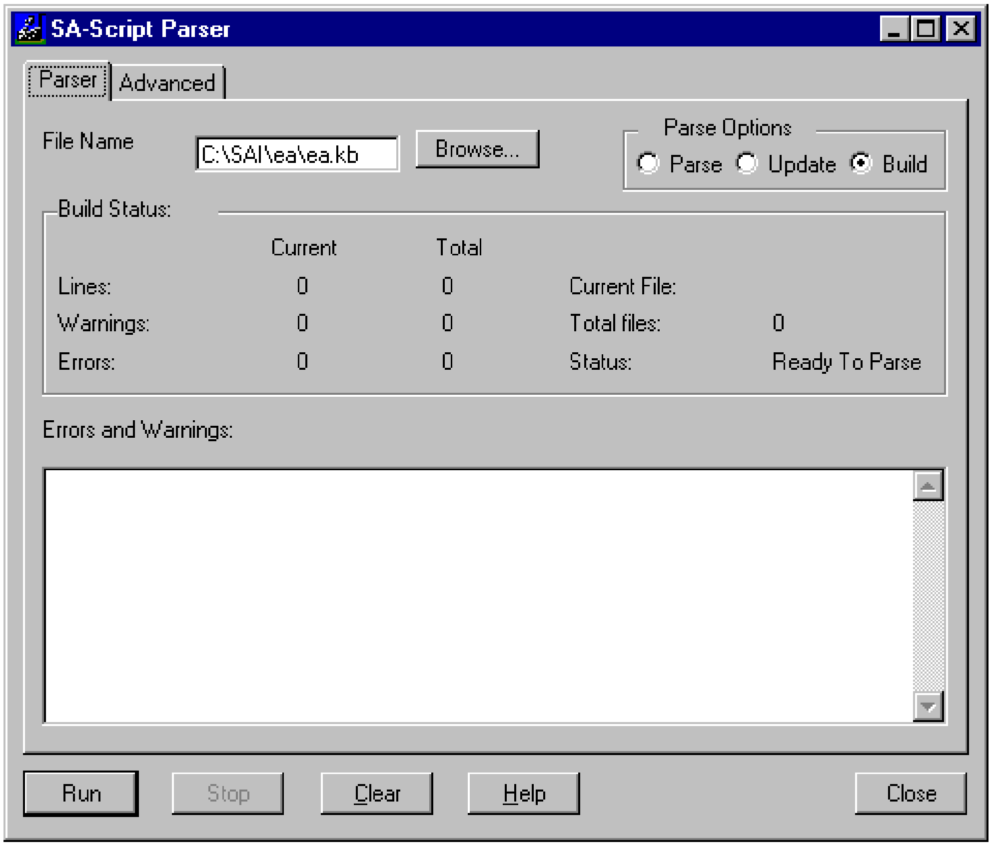


Figure 47. Parsing the Application

Once completed select **Close.**

4.9.3 Installing Tivoli Advisor on Client Workstation

To install the Tivoli Advisor on a client workstation, we followed the same steps as in the previous section. In the Installation Options dialog box we un check the box for the application source files as they are already installed on file server. In the Destination folder we set the path to the files on the mapped drive from the file server as shown in Figure 48 on page 80.

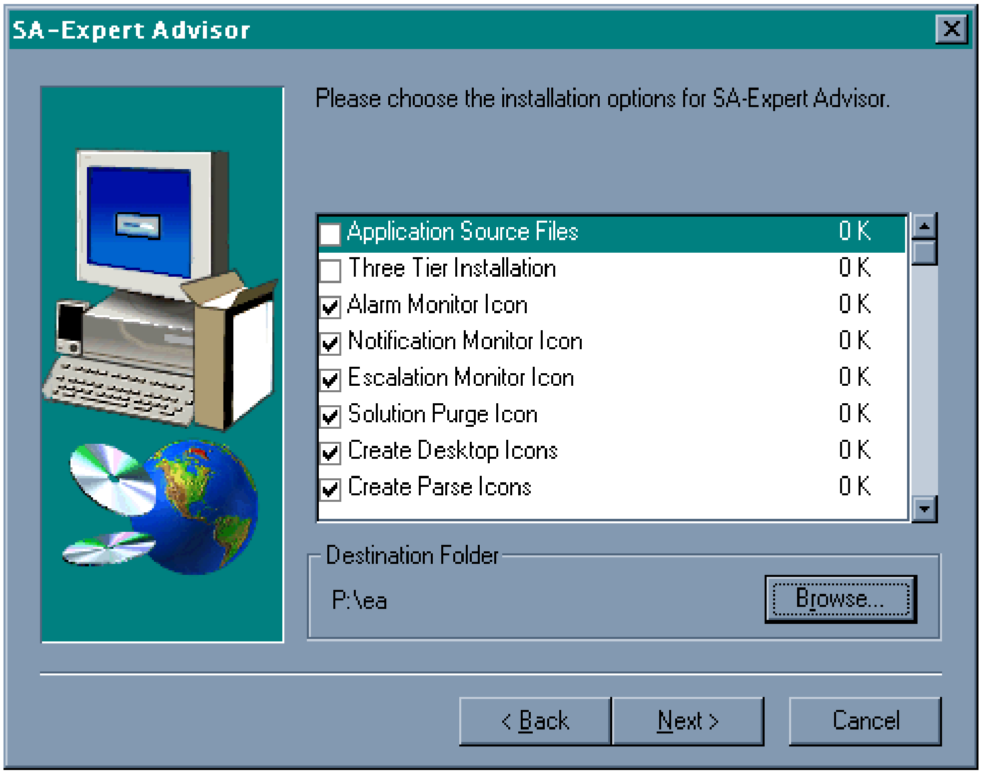


Figure 48. Tivoli Advisor Installation Options for Client Workstation

After the installation of the clients the workstation must be re-booted, as some entries are added to client startup procedures.

Note

After the client workstation installation there is no need to parse the Tivoli Advisor application again.

4.10 Installation of the Tivoli NSM Gateway

The Tivoli NSM Gateway application can be installed on a file server or stand-alone workstation and provides links to the NSM application.

Note

Before the Tivoli NSM Gateway installation, the TCP/IP services file on the gateway machine and all Tivoli Advisor workstations that will access NSM Gateway extensions have to be changed.

On the network/system management (NSM) machine we look in the services file where the port number to use for ASE extensions is assigned. In our case this port was set to 9400. On the NSM Gateway machine we edit the services file, which for NT 4.0 machines is:

<*drive*>:`\winnt\system32\drivers\etc\services`

We also add the following line at the end of the file:

```
asenet 9400/tcp
```

We installed the Tivoli NSM Gateway on machine WTR05368, which is also the file server machine where the Tivoli Advisor application is installed. We followed the steps as installing the Advisor. The main difference was in the Product List dialog check box we chose SA-Expert View Gateway installation. In the Expert View Gateway installation option we choose the options as shown in Figure 49 on page 81.

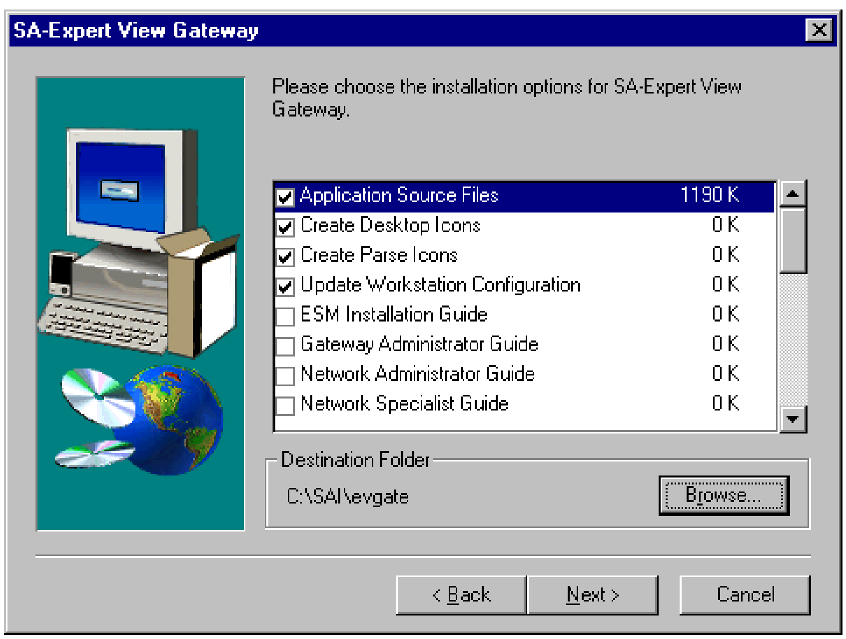


Figure 49. Tivoli NSM Gateway Installation Options for the Client Workstation

After the installation of the Tivoli NSM Gateway we parsed the Tivoli Advisor application again. This has to be done as installing the Tivoli NSM Gateway adds some functionality to Tivoli Advisor application. After the Tivoli NSM Gateway installation, the Tivoli Advisor GUI window list of failed nodes and status of the nodes managed by Tivoli NSM Gateway can be viewed.

4.10.1 Installation of Tivoli NSM Gateway on a Client Workstation

If the Tivoli Gateway GUI is to be used on the client workstations, the icons have to be installed and the workstation configuration has to be updated. This is done by installing Tivoli NSM Gateway on the client workstation. If this client workstation is used also for the Tivoli Advisor, then the /etc/services file has to be changed as described earlier.

If the client workstation is to be used only for Tivoli Advisor and Tivoli NSM Gateway extensions, installation of Tivoli NSM Gateway on client workstation is not necessary. In this case the only path to the NSM Gateway binaries on the mapped drive should be added to the PATH variable.

4.11 Installation of ExpertView NSM Commands

The ExpertView NSM Commands have to be installed on the same machine where the TEC is installed. In our scenario this is the machine rs600024. In fact this product provides the command EVProb used by the TEC to automatically open, update and close problem records in the Advisor database. At same time it provides also the EVQueryd daemon that processes the callback requests from the ExperView Gateway module.

To install the ExpertView NSM Commands the installation program install.esm has to be used. As we already stated the program doesn't support AIX Version 4.3 and a workaround has to be used. We propose here an alternative way to run the script for AIX Release 4.3.

The script install.esm calls the system command uname to get information about the system version and release. So, we created the following script called uname in the directory /tmp that will return the correct value.

```
PARM=$1

if [[ $PARM = "-s" ]] then

echo AIX

fi

if [[ $PARM = "-v" ]] then

echo 4

fi

if [[ $PARM = "-r" ]] then
echo 2

fi
```

We changed the permission of the uname file to:

```
chmod 755 uname
```

Then we placed /tmp as the first parameter for the PATH environment variable:

```
export PATH=/tmp:$PATH
```

Now, when install.esm calls the uname command will return the value for AIX Version V4.2.

To invoke the installation script mount the CD-ROM, for example:

```
mount -r -v cdrfs /dev/cd0 /infocd
```

where /infocd is the mount point. Then from the directory /infocd/unix issue the command:

```
./install.esm
```

The installation program will ask you for the product key and then shows a list of products like in the following figure.

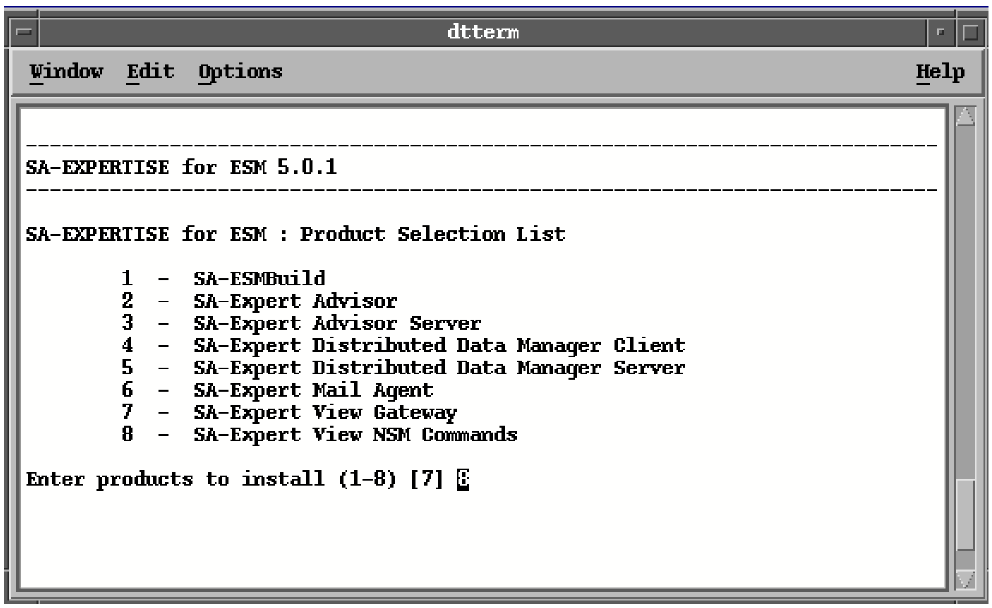


Figure 50. ESM Main Installation Panel

Select option **8** to install the NSM component. Next you will see a list of installation options for that component (see Figure 51 on page 84).

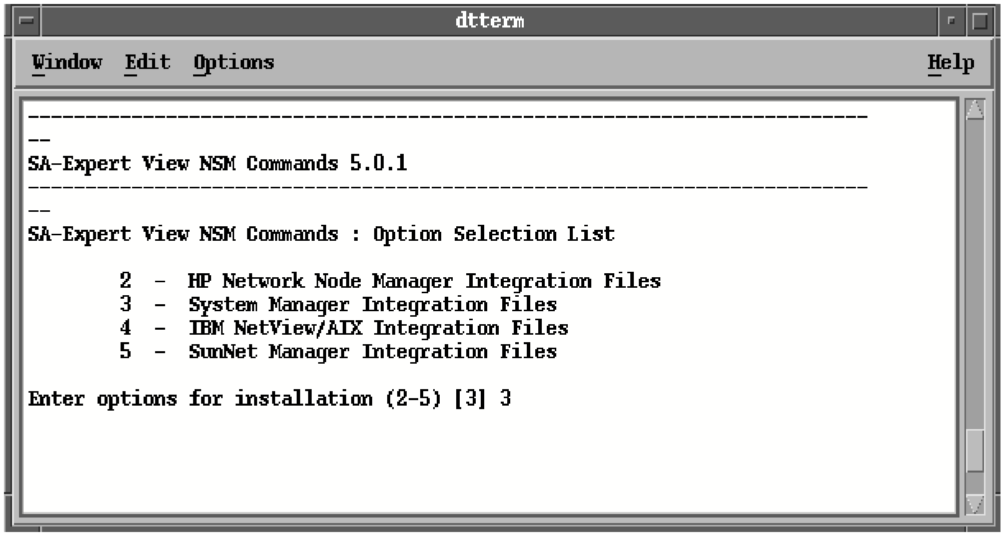


Figure 51. Expert View NSM installation Options

Select option **3** to install the System Management Integration option.

The installation program will ask for the path where you want to install the component. We selected the default value. When the installation is completed you should see the following messages:

```
Installing System Manager for AIX interface...

Extracting files from archive miscsysm.tar...

Extracting files from archive aixsysm.tar...

Adding product section to saiapp.ini ...

Product installation SUCCESSFUL.
```

We also added the directories for the NMS commands to our variable definitions.

```
PATH=$PATH:/usr/lpp/sai/aixASE/bin:/usr/lpp/sai/aixASE/Acrobat/bin:/usr/lpp/sai/\

                    evcmds/sysmgr/bin:/usr/lpp/sai/esmbin

LIBPATH=$LIBPATH:/usr/lpp/sai/aixASE/lib:/usr/lpp/sai/evcmds/sysmgr/bin

SAIPATH=$SAIPATH:/usr/lpp/sai/aixASE/bin:/usr/lpp/sai/aixASE/images:/usr/lpp/sai\

                    /aixASE:/usr/lpp/sai/evcmds:/usr/lpp/sai/esmicons

SAI_ROOT=/usr/lpp/sai

FULCREATE=/usr/lpp/sai/aixASE/indexes

FULSEARCH=/usr/lpp/sai/aixASE:/usr/lpp/sai/aixASE/bin:/usr/lpp/sai/aixASE/indexes

FULTEMP=/tmp

export PATH LIBPATH SAIPATH SAI_ROOT FULCREATE FULSEARCH FULTEMP
```

Now the NSM commands are available.

4.12 Installation of Tivoli Asset Management

The Tivoli Asset Management installation consists of the following components:

- Tivoli Asset Management Manager
- Tivoli Asset Management User

Tivoli Asset Management Manager can be installed on a file server, client workstation or stand-alone workstation. It is usually installed on the file server and the connected client workstations that require Tivoli Asset Management Manager utilities.

Tivoli Asset Management User can be installed on all three types of machines in the Tivoli Service Desk environment but it is usually installed on client workstations that do not require Tivoli Asset Management Manager utilities.

4.12.1 Installation of Tivoli Asset Manager on the File Server

We installed the Tivoli Asset Management Manager on machine WTR05368. The installation procedure is similar to installing the other applications on the file server described in previous sections.

The only difference is that we do not install the Tivoli Asset Management together with the Tivoli Advisor. We have to prepare the Tivoli Asset Management tables and views in the ADVISOR database with the ESMBuild procedure.

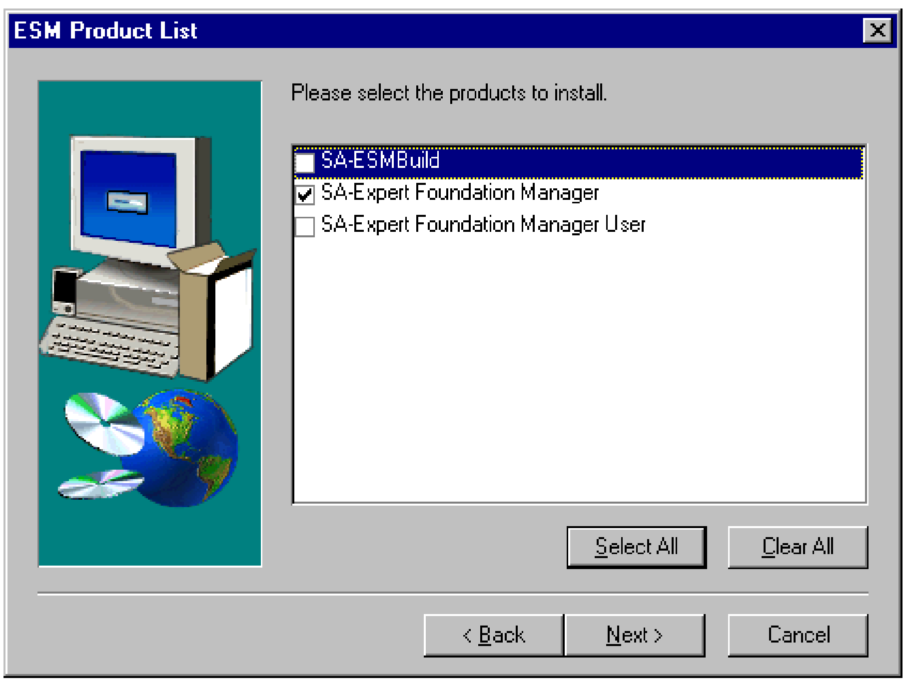


Figure 52. Selecting the Asset Manager Installation Options

Select the Foundation Manager application as shown in Figure 52 on page 85.

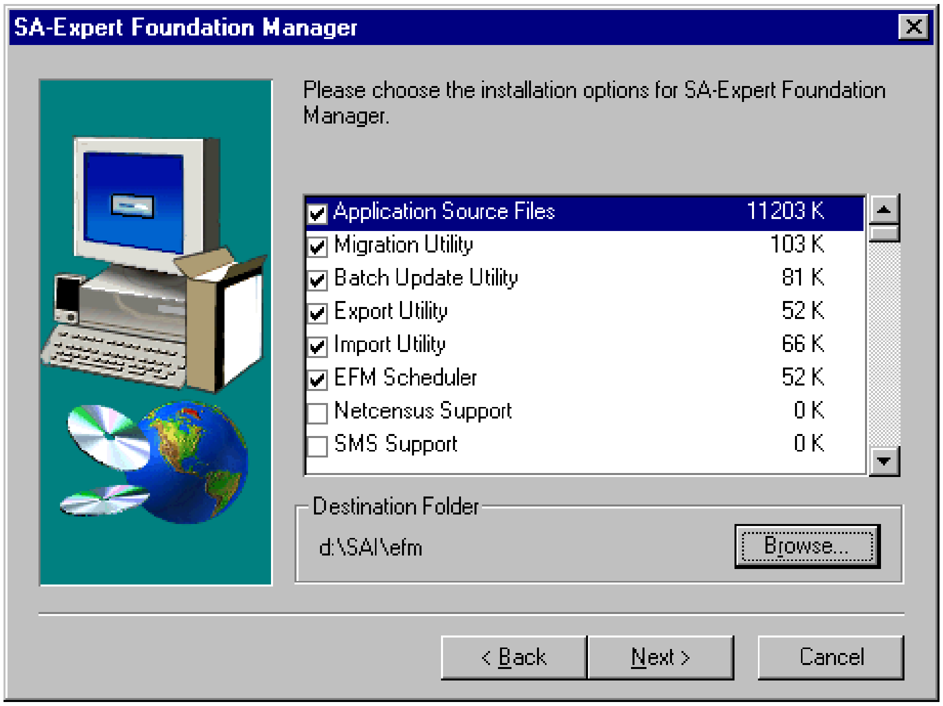


Figure 53. Selecting the Utilities

We selected the utilities as shown in Figure 53 on page 86.

We built the tables and views in the ADVISOR database using the ESMBuild script. With the correct authorization key the ESMBuild Selection dialog box shows the option to build tables for Tivoli Asset Management as shown in Figure 54 on page 87.

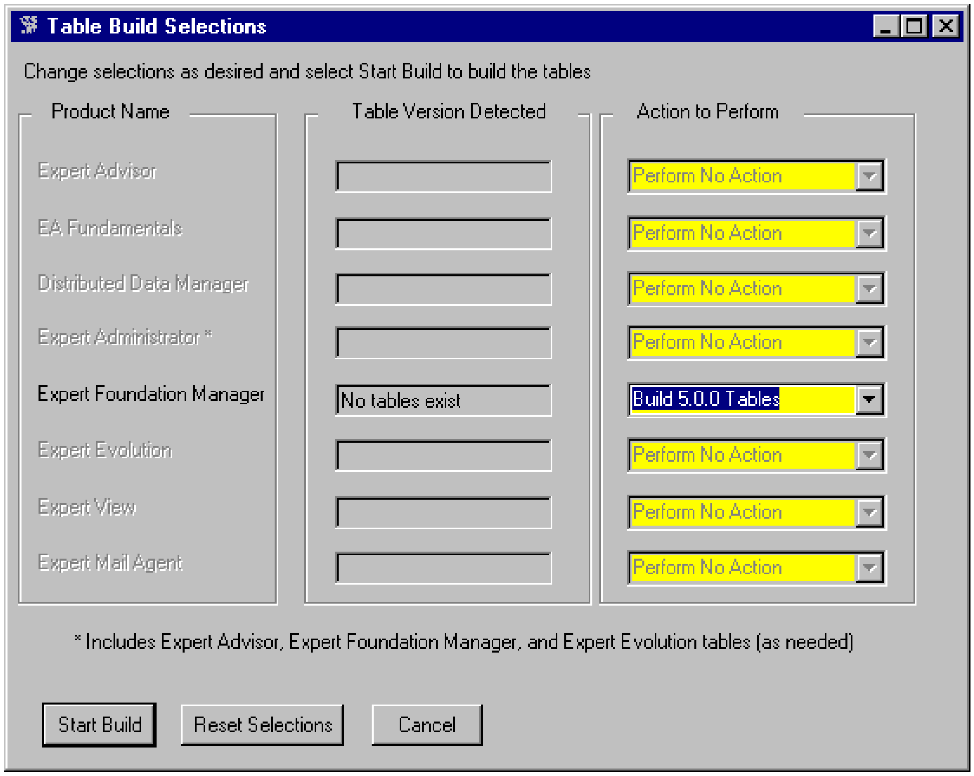


Figure 54. Tivoli Asset Management Table Build

After building the tables, we installed Tivoli Asset Management Manager code on the file server WTR05368.

After the installation of application files the machine has to be re-booted for changes to take effect and before parsing the application. We then parsed the Tivoli Advisor application.

4.12.2 Installation of Tivoli Asset Manager on a Client Workstation

There is no need to install any software on the client workstation if Tivoli Asset Management is going to be used through the Tivoli Advisor interface. The path to the Tivoli Asset Management binaries on the shared drive located on the server needs to be added.

If on other side the Tivoli Asset Management GUI is to be used directly with the icons on the client workstations, the icons have to be installed and the workstation configuration must be updated. This is done by installing Tivoli Asset Management on the client workstation. The installation follows similar steps as described in 4.9.3, “Installing Tivoli Advisor on Client Workstation” on page 79.

4.13 Installation of Tivoli Change Management

The installation of Tivoli Change Management follows almost the same steps as the installation of Tivoli Asset Management described in 4.12.1, “Installation of Tivoli Asset Manager on the File Server” on page 85.

Installing the stand-alone workstation version on NT requires additional parsing steps to be performed.

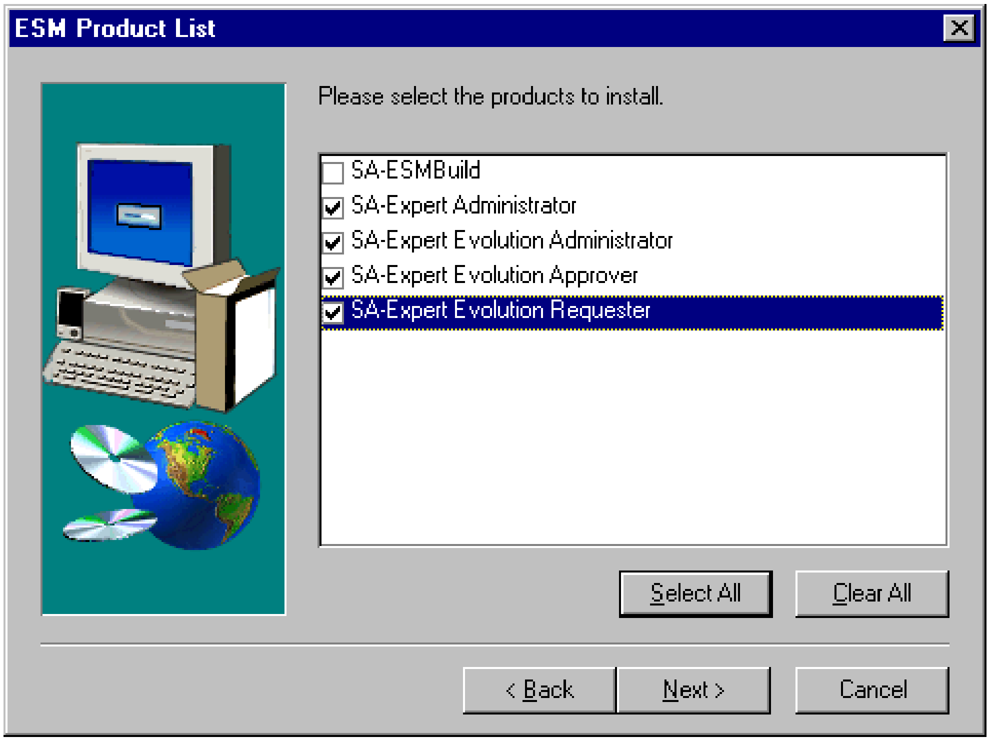


Figure 55. ESM Product for Change Management

We selected the options in Figure 55 on page 88.

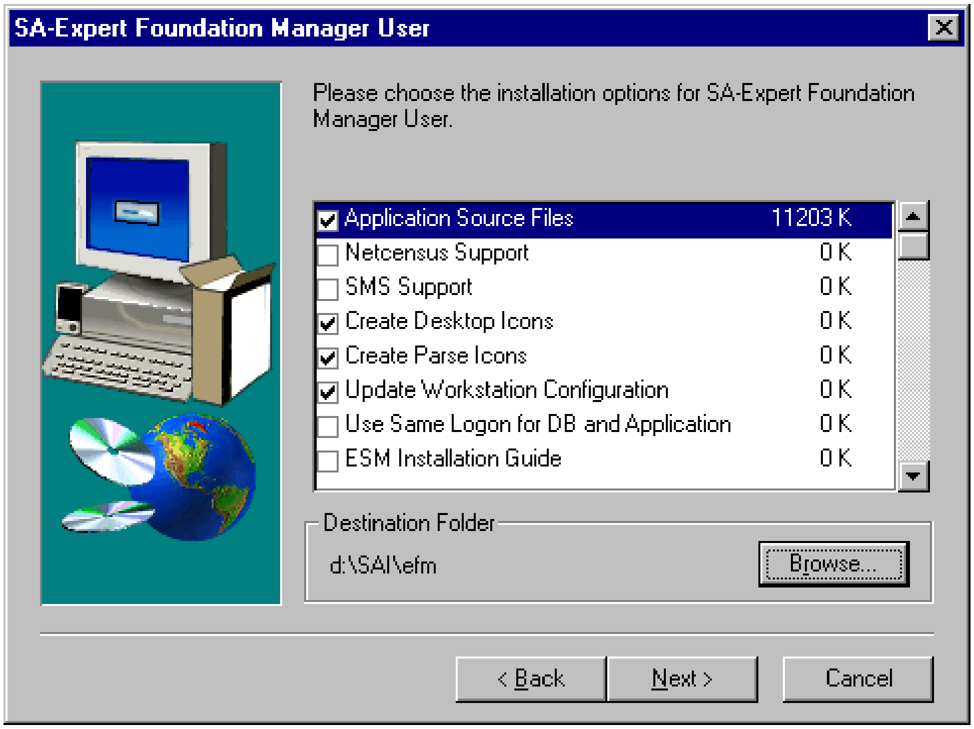


Figure 56. Foundation Manager User

We selected the default values for the installation as shown in Figure 57 on page 89.

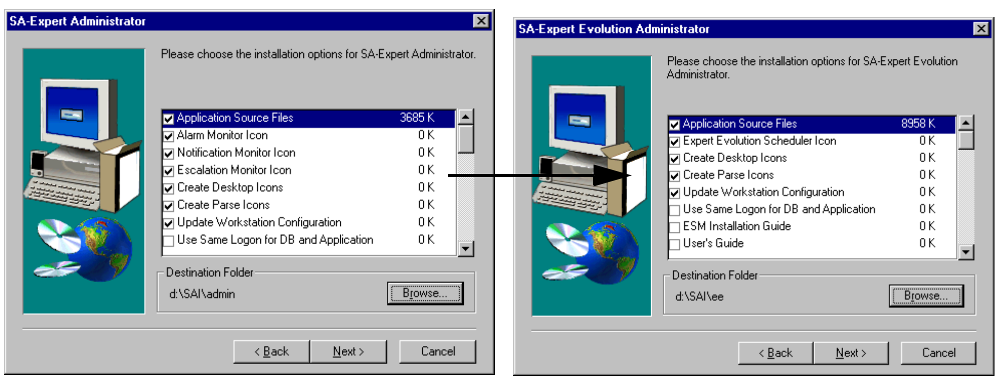


Figure 57. Change Management Configuration Screens

The approver and requester options are shown in Figure 58 on page 90.

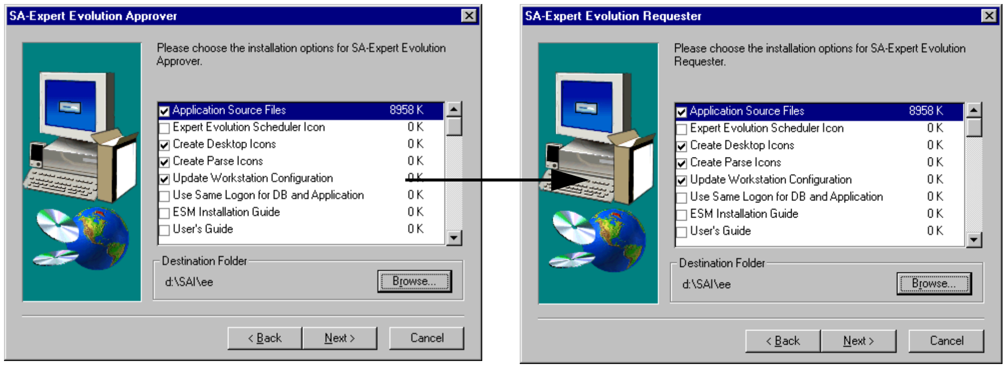


Figure 58. Change Management Installation Screens

After the installation and parsing of the Tivoli Advisor application, the Tivoli Advisor GUI looks like the screen shown in Figure 59 on page 90.

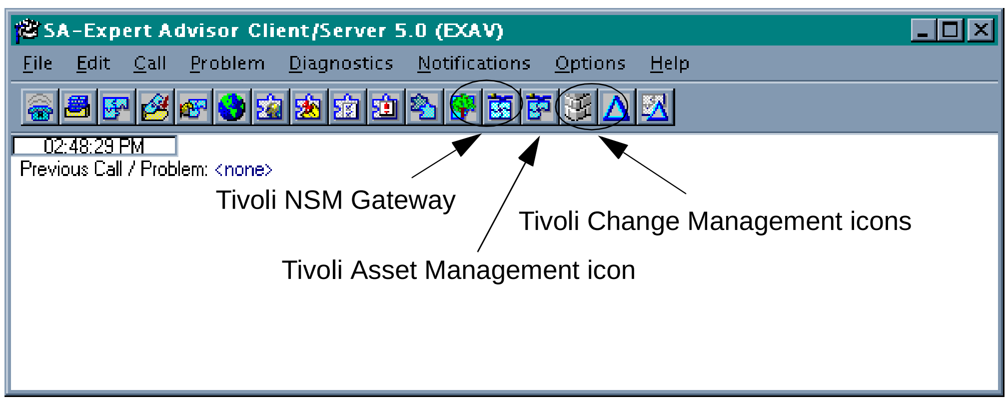


Figure 59. Tivoli Advisor GUI After Applications Installation

Note

Starting the Tivoli Change Management application produces an error because the Tivoli Advisor user has to map to a person in Tivoli Asset Management. This happens only when starting Tivoli Change Management and not when starting Tivoli Asset Management.

A person that belongs to an organization is not necessarily a user of the Tivoli Service Desk application. A user is a person who represents an end user of the Tivoli Service Desk applications. Users have user IDs and can belong to the groups.

We explain how to add Tivoli Service Desk users in section 5.6, “Adding New Users to the Tivoli Service Desk Applications” on page 195.

4.13.1 Verify the Sybase Tables Have Been Created

To verify that the sybase tables have been created we used the Sybase SQL server manager to view the configuration (see Figure 60 on page 91).

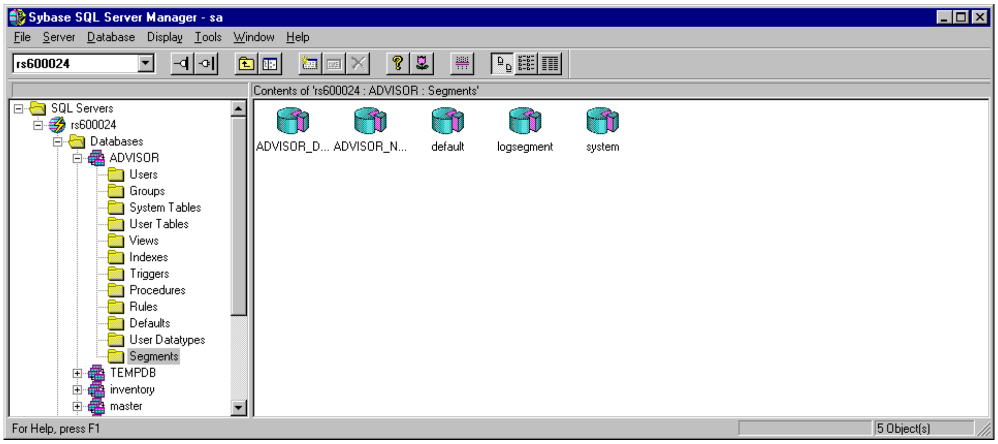


Figure 60. Sybase SQL Manager

The specific tables for the service desk application are shown in Figure 61 on page 91.

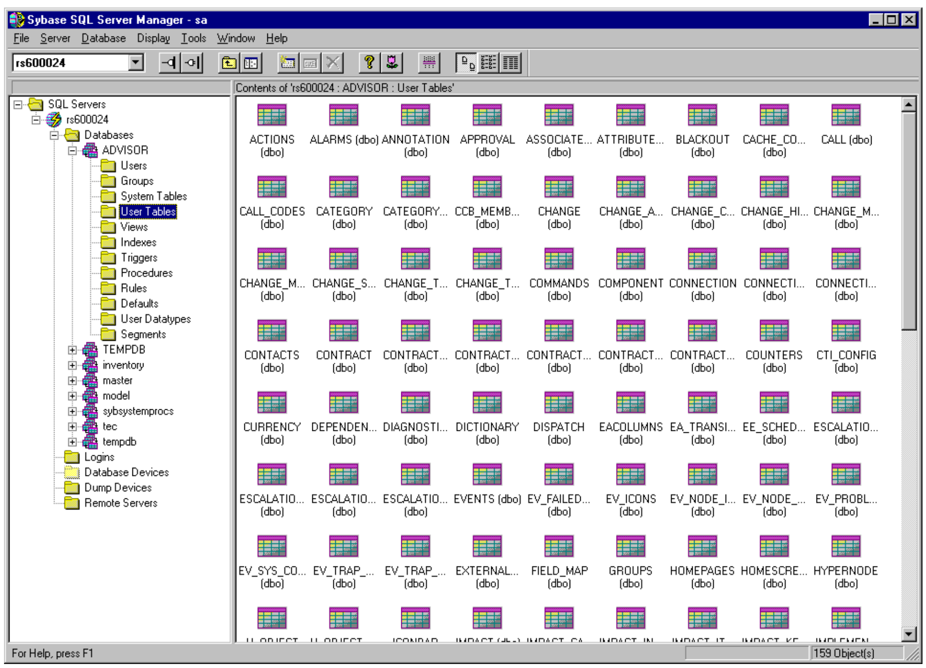


Figure 61. Tivoli Service Desk Tables

Next we can install the required TEC adapters.

4.14 Installing the TEC Adapters

The TEC adapters we installed for our management environment are listed below:

- Tivoli NetView Adapter for all SNMP-managed devices
- Distributed monitoring for NT and UNIX resources
- NT logfile adapters to monitor NT resources

- AIX logfile adapter to monitor AIX resources

Figure 62 on page 92 shows what adapters we installed and what resources they are managing,

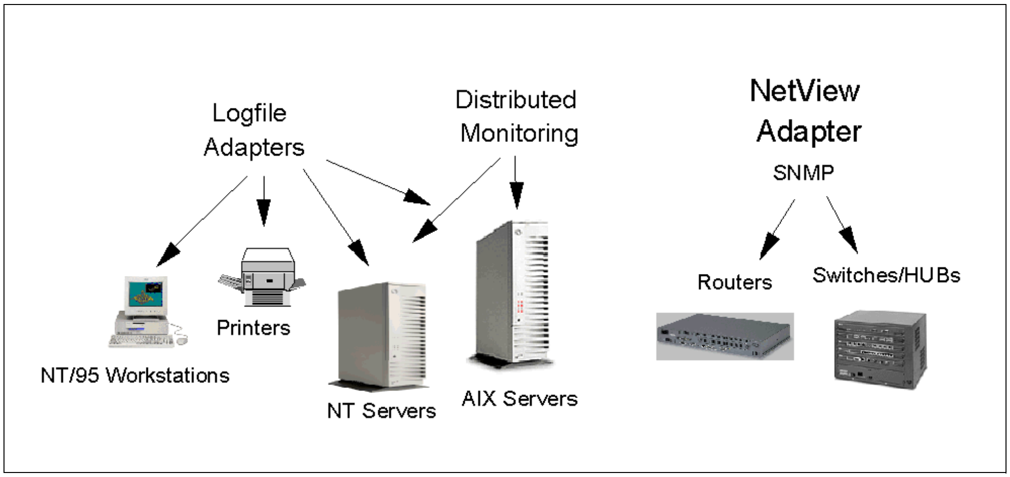


Figure 62. TEC Adapter Deployment

For our end-to-end example we use the NT adapter to show only the process for installing the NT adapter. For further information on installing TEC adapters see *TEC Implementation Examples,* SG24-5216 and the *Enterprise Console Adapters Guide.*

4.15 Installation of the TEC NT Adapter

In this section we discuss the installation of an NT adapter on an endpoint.

Here are the different steps we followed:

- Install the Adapter Configuration Facility
- Create an ACP profile
- Configure the profile and distribute it on an NT endpoint

4.15.1 Installation of the Adapter Configuration Facility

The Adapter Configuration Facility is a profile-based application that contains default profiles for all adapters and provides centralized configuration and control of these profiles.

The Adapter Configuration Facility should be installed on the TMR server and on all the Tivoli Management Gateways.

We started the install program for the Adapter Configuration Facility software as shown in Figure 63.

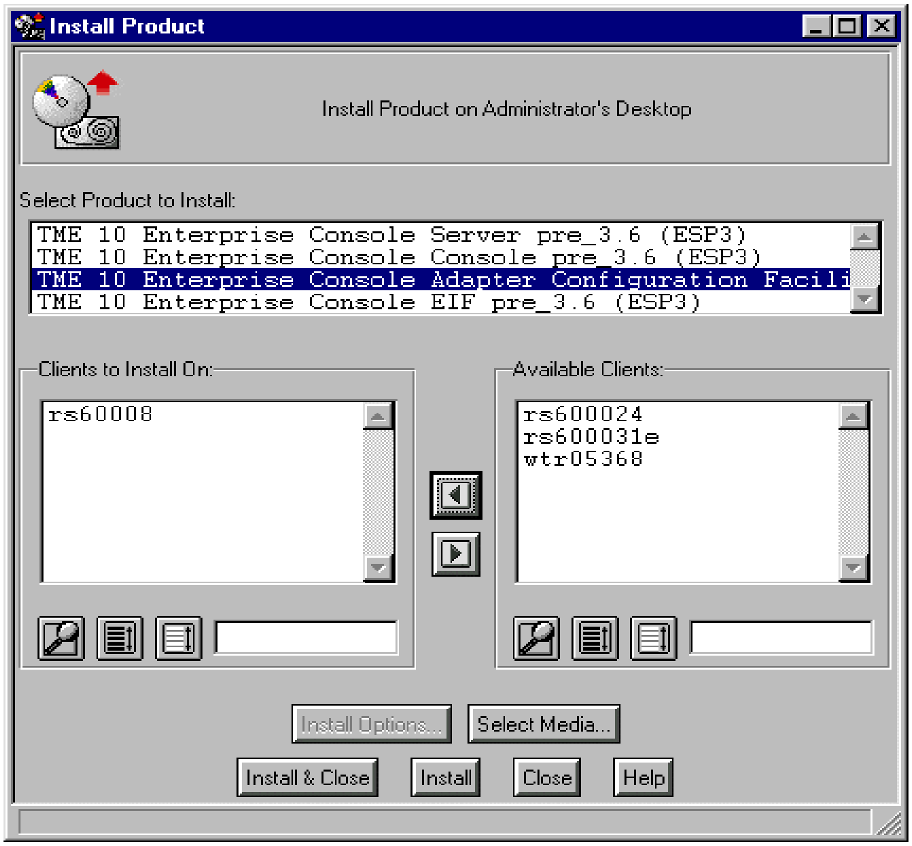


Figure 63. Install Product Window

We set the path to point to the source code, then proceeded to select **TME 10 Enterprise Console Adapter Configuration Facility**, and clicked on **Install**.

We installed it on rs60008, which is our TMR server and Tivoli Management Gateway.

4.15.2 Creating and Configuring the Adapter Configuration Profile

We created the ACP profile in a profile manager, as we would any kind of profile except that we selected the type of ACP.

Once the profile was created we configure it as follows.

The first item is what kind of adapter we want to use. In our case we created and configured an NT adapter as we wanted to distribute this profile on an NT endpoint. To configure the NT adapter we used the GUI.

We first opened the profile as shown in Figure 64 on page 94.

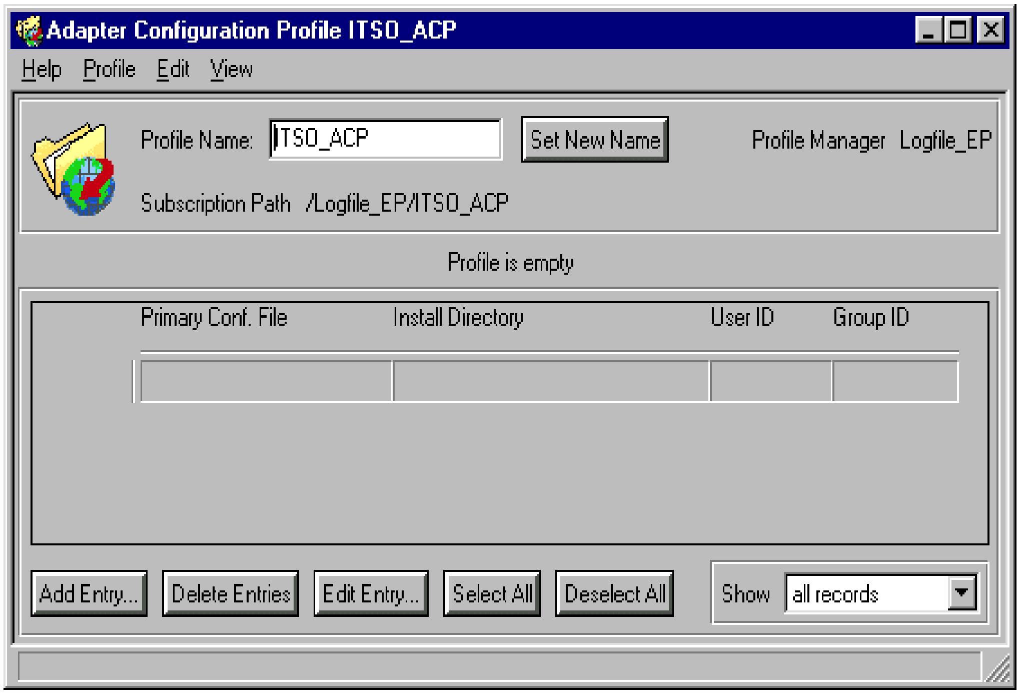


Figure 64. Adapter Configuration Profile

Next we created a new entry in this profile by clicking on **Add Entry.** The screen shown in Figure 65 on page 94 will appear. This window lists all the adapters we can configure. We selected the tecad_nt adapter.

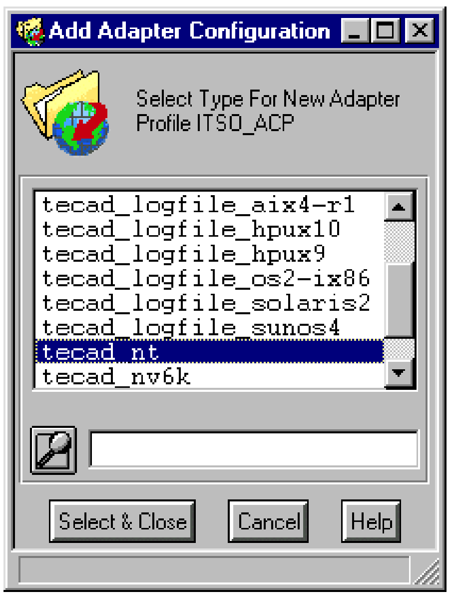


Figure 65. Add Adapter Configuration

You will see that you can select the tec_gateway adapter but we did not because the tec_gateway is automatically installed on the machine where the Adapter Configuration Facility is installed so we used the default configuration.

However, if you want to change the configuration of tec_gateway make sure the gateway to which you will distribute this configuration has the lcfd daemon running.

Now that we have selected the adapter there are a number of parameters we need to define. These are listed below:

- Filters
- Distribution requests
- Actions
- EIF environment variable setting
- General controls

4.15.2.1 Filters

Filters provide a way to prevent events from being sent from an adapter to an event server.

Figure 66 shows the filter's configuration dialog.

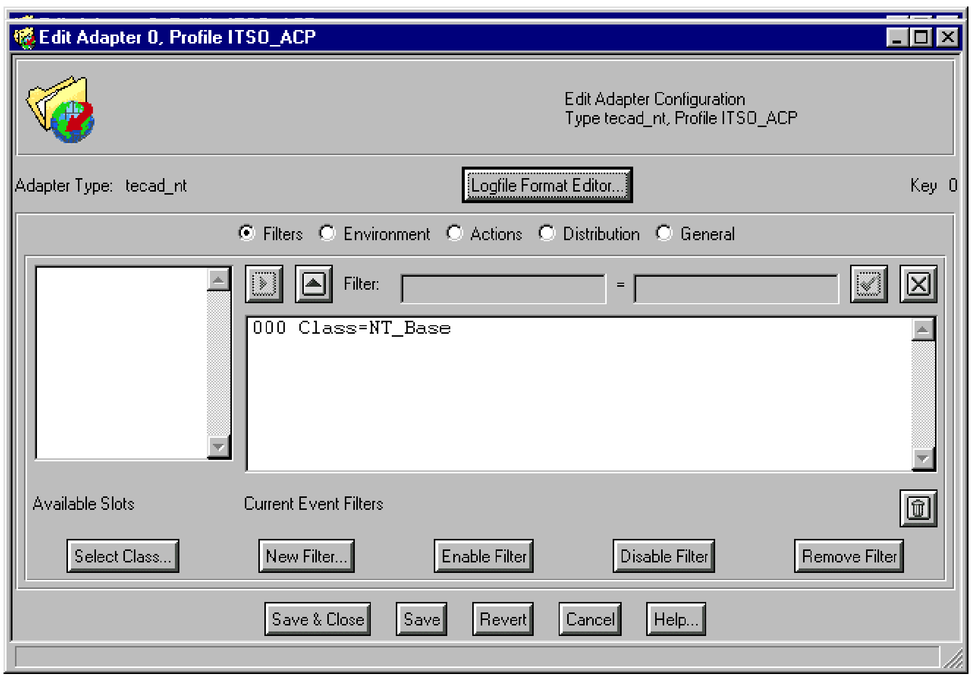


Figure 66. Filters Configuration Window

Each filter consists of a set of event slot values that describe the variety of event to be filtered out of the event stream. If an event comes from an adapter and its slot values match the ones defined in any of the filters, the event will be discarded.

To configure the filters you first have to select an event class by clicking on the bottom left button. On the left-hand side there will be a list of component slots.

You can select any slots and you can make comparisons in the two upper fields.

The left parameter is used for the name of the slot, and the right parameter for the slot value to be scanned for. Each class filter will be identified by a sequence number on the left of the filter.

If a # character appears after the sequence number on each slot comparison in the filter, this means that the filter is disabled.

You can enable it simply on clicking on the **Enable Filter** button.

4.15.2.2 EIF Environment Variable Settings

The EIF Environment configuration dialog allows you to configure environment variables that are set in the tecad_<adapter_name>.conf file (see Figure 67 on page 96).

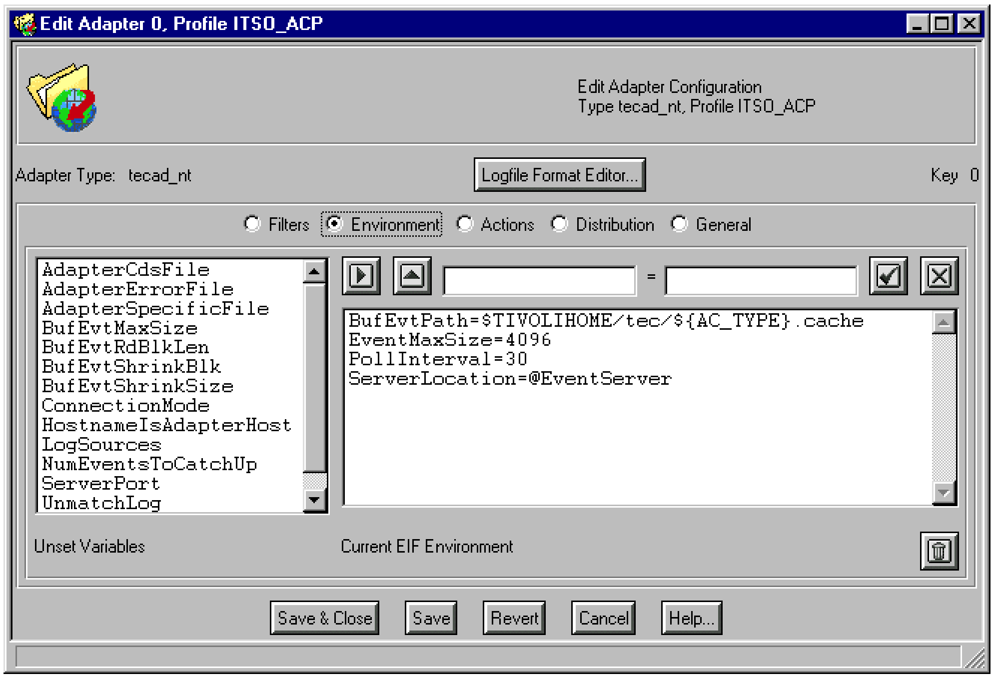


Figure 67. EIF Environment Configuration Dialog

This file is a repository of the environment variables used by the adapter to control its behavior.

Some of the variables are used directly by the EIF library itself, such as information about which event server to communicate with and how events should be buffered.

Other variables are available to the adapter, and each adapter is free to use as many variables as it needs.

On the left-hand side there is a list of all the variables, and in the central list all shown current environment variable settings are shown.

To define a variable, you must select it from the left list, put it in the left editing field and set its value in the right one.

4.15.2.3 Actions

The Actions dialog allows you to configure actions that are to be performed on subscribing endpoints upon distribution (see Figure 68 on page 97).

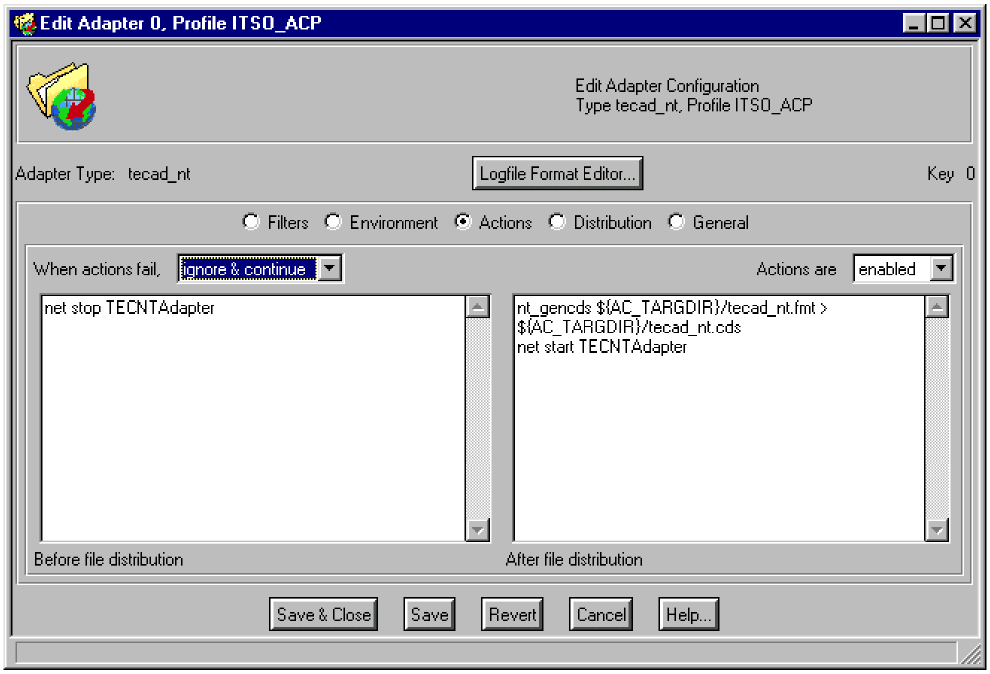


Figure 68. Actions Dialog

Actions can be performed both before and after configuration files are written.

The most common actions will stop an event adapter before distribution, clean out a configuration directory, and then after distribution restart the adapter.

Three different behaviors are available if an action fails. These behaviors can be selected in the When Actions Fail menu in the upper left. These behaviors are:

- Ignore & Continue
- Report & Continue
- Abort Distribution

As this is the first time we distribute this profile to an NT endpoint we use the **Ignore & Continue** behavior because we knew that stopping the NT adapter would fail as it has never been started before.

The control on the upper right corner allows you to enable or disable these actions.

4.15.2.4 Distribution

Figure 69 on page 98 shows the distribution options.

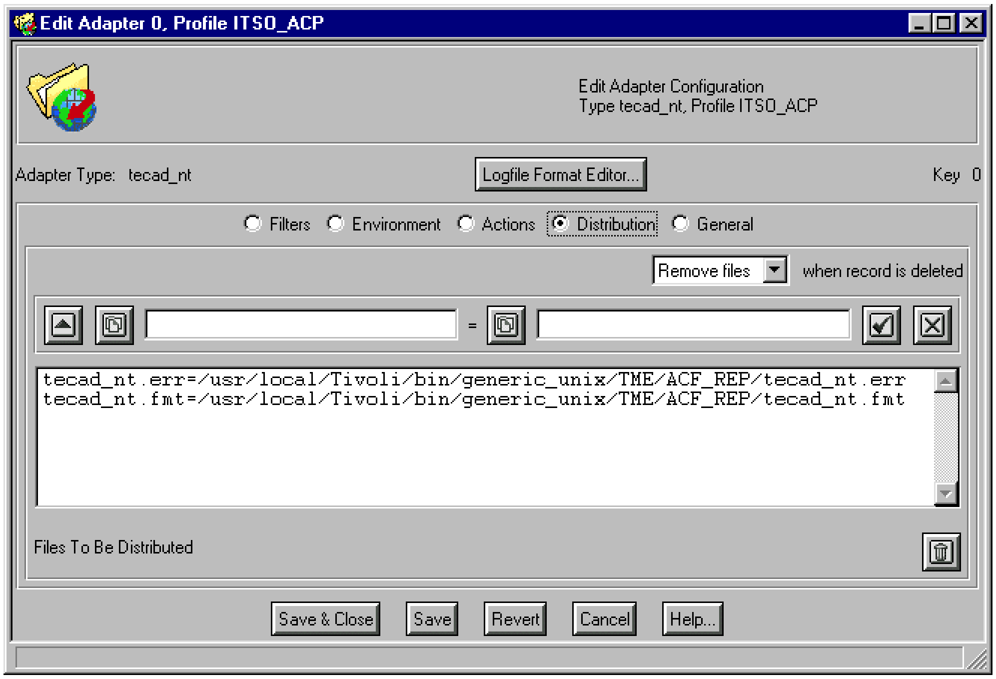


Figure 69. Distribution Dialog

The Distribution dialog lists all the files that are to be transferred along with the distribution of the adapter.

These files can reside on any managed node reachable by the endpoint.

The files are shown in the large center list and are in the format destination=source where destination is the pathname where the file will be deposited on each subscribing endpoint and source is the host and pathname from which the file is to be copied expressed as host:pathname.

If no host is explicitly named, then the file is taken from the TMR server.

When an entry in the profile is deleted, the ACF can remove all distributed files if that is appropriate. The control at the top allows that behavior to be set independently for each profile entry.

4.15.2.5 General

The General dialog box provides you with information that will control how an adapter configuration entry is installed at the endpoint.

Figure 70 on page 99 shows the edit adapter profile.

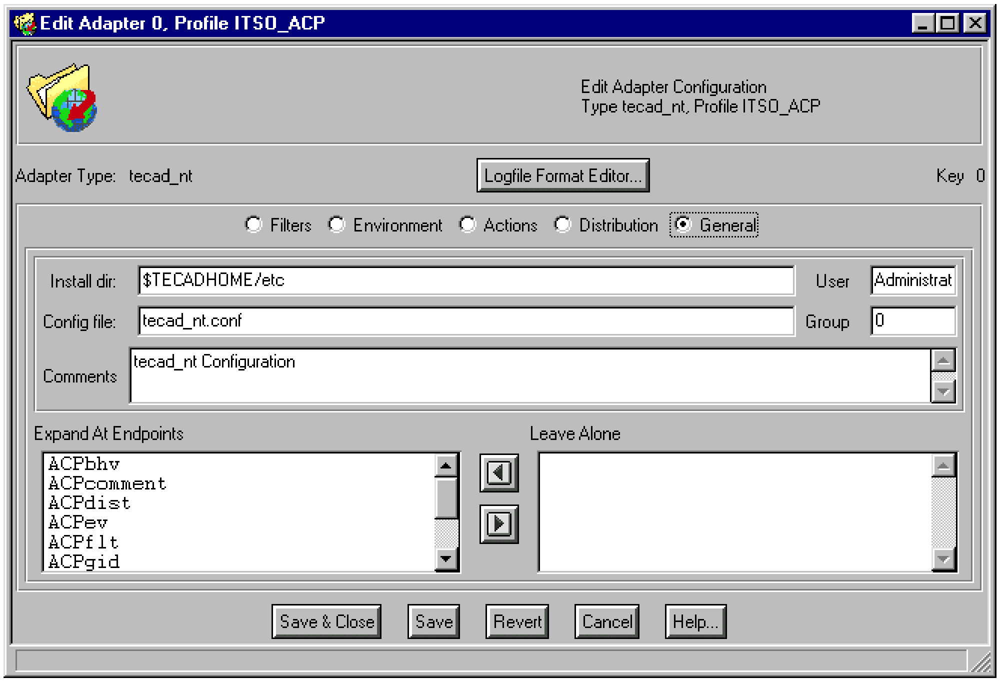


Figure 70. Edti Adapter Profile

An installation directory should be provided as a default location for files to be distributed and configuration files. The name of the primary configuration file can also be set.

Each entry is given a user ID and a group ID that will determine ownership of distributed files as well as the ID under which distribution actions will be performed.

The ID values can be numeric values or user/group name strings. At the end of the dialog there is a list of references that you can either expand at the endpoint or not.

This list is an environment list built by the ACF endpoint code and contains details of the adapter configuration.

This list refers to all the different parts of the profile entry as follows:

ACPbhv Behavior flags, this particular attribute will very rarely require expansion, and very rarely will there be environment variable references here.

ACPcomment The configuration file comment block.

ACPdist File distribution requests and before/after actions.

ACPev The EIF environment.

ACPflt Event filters.

ACPgid The group ID value.

ACPtarget The primary configuration file name.

ACPtype The type of the profile entry (unlikely to contain variable references anyway).

ACPuid The user ID value.

4.16 Installation of the SA Expertise Tivoli/Plus Integration Module

Here we describe the installation process of the SA Expertise Tivoli/Plus Integration Module 5.0.1.The steps to install the module are as follows:

1. Verify the installation of the SA Expertise components.

   Note that all the systems that provide any of the SA Expertise functions must be Tivoli managed nodes to make the Plus module work correctly.

2. Back up the Tivoli object database.
3. Make a note of all the server names for the systems that provide the different functions of the SA Expertise Suite.
4. Install the module on all required systems.

The installation of the module is a two-step process. The first step is the installation of the module; the second step is the configuration.

The Plus module consists of the following deliverables:

- Tasks to operate the SA environment
- Specific monitors for the SA environment
- TEC classes and rules
- Preconfigured file packages for software distribution

Depending on what core applications are installed, the appropriate deliverables of the Plus module will be installed in the TMR. In our environment, for example, we didn’t have software distribution deployed, so there were no software packages installed.

4.16.1 Installation Process

Before installing the module it is a good idea to back up the Tivoli object database as the Plus module creates lots of entries in the database, so that cleaning up the database becomes a very time-consuming process. There is no automated un-install process.

The module must be installed on the following:

- TMR server
- TEC server as the NSM machine
- ExpertView Gateway

To install the module on the TMR server use the Tivoli product installation dialog, set the media path and choose the appropriate systems from the installation target window (see Figure 71 on page 101).

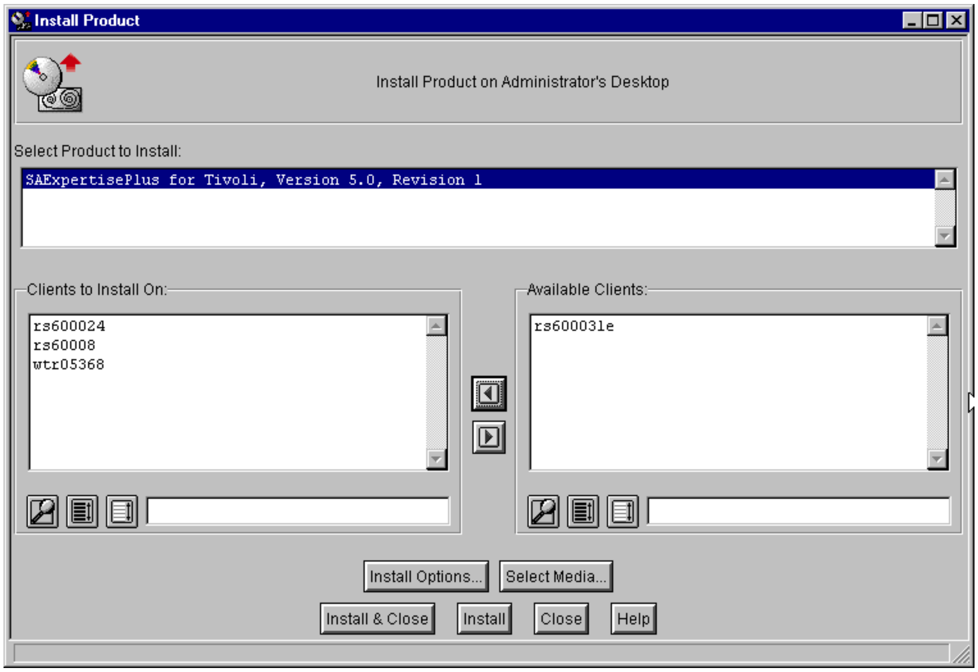


Figure 71. Install Product Dialog

After selecting the product **SAExpertise Plus for Tivoli,** Figure 72 on page 101 appears.

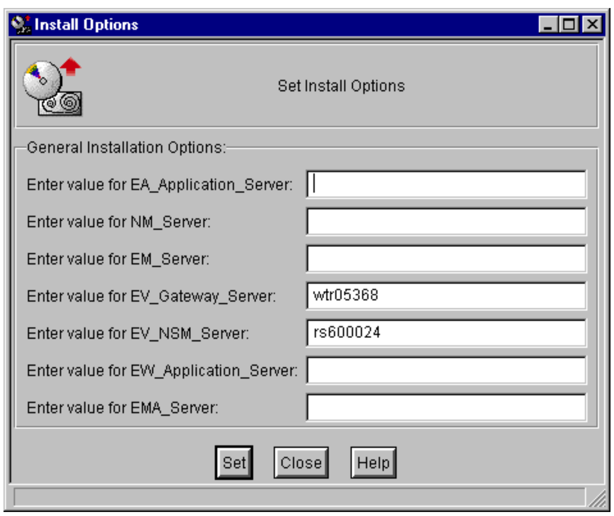


Figure 72. Install Options

We entered the appropriate values for our environment. The values of the entry fields map to the functionalities as follows:

Table 10. Label /Function Mapping

Label	**Function**
EA_Application_Server	Expert Advisor Application Server (only three-tier environment)
NM_Server	Expert Notification Server
EM_Server	Expert Escalation Server
EV_Gateway_Server	Expert View Gateway Server
EV_NSM_Server	Expert View NSM Commands Server
EW_Application_Server	Expert Web Application Server
EMA_Server	Expert Mail Agent Server

The names that you enter are the name of the nodes that are going to be subscribed to the profile managers that get created during the installation of the Plus module.

Note

During our installation we used Version 5.0.1 of the module. In Version 5.0.2 this installation screen does not exist anymore and the subscriptions have to be done manually.

The installation of the Plus module on the TMR server took a long time, approximately 15 minutes. This is due to the amount of database activity, such as creating collections, libraries, profile managers and tasks.

After the installation has completed successfully, double-click on the **SAExpertisePlus for Tivoli** icon, which has been created in the TivoliPlus collection (see Figure 73 on page 103).

.

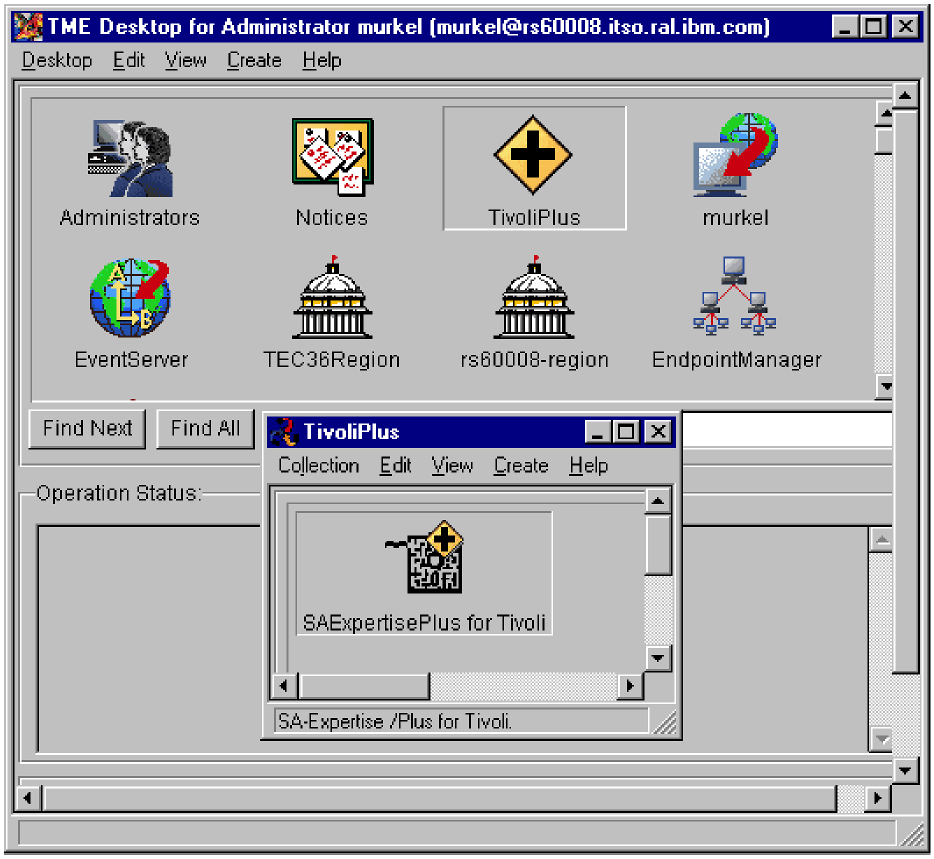


Figure 73. Tivoli Plus Collection

The screen shown in Figure 74 on page 103 contains the SA product-related collections, a task for informational purposes and a task to configure the TEC event server. The collections contain tasks to configure/operate the SA environment.

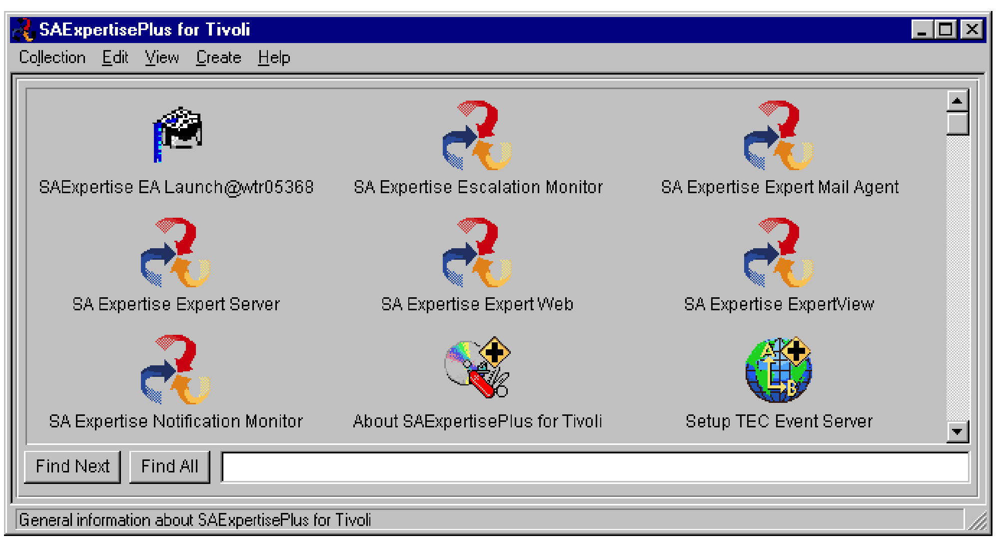


Figure 74. SAExpertisePlus for Tivoli Collection

The installation of the Plus module is now complete.

4.16.2 Configuration of the Plus Module

The configuration of the Plus module is done in multiple steps, again depending on what SA products are installed. In the lab environment, the components to be configured are:

1. The TEC event server
2. SA ExpertView including:
   - SA ExpertView Gateway
   - SA ExpertView NSM Commands

The configuration tasks are found in the appropriate collections, except for the TEC configuration task, which resides in the main SAExpertisePlus for Tivoli collection.

4.16.2.1 Configuring the TEC

The task setup shows two possible install options (see Figure 75 on page 104). The options are:

- Create new rule base
- Add to existing rule base

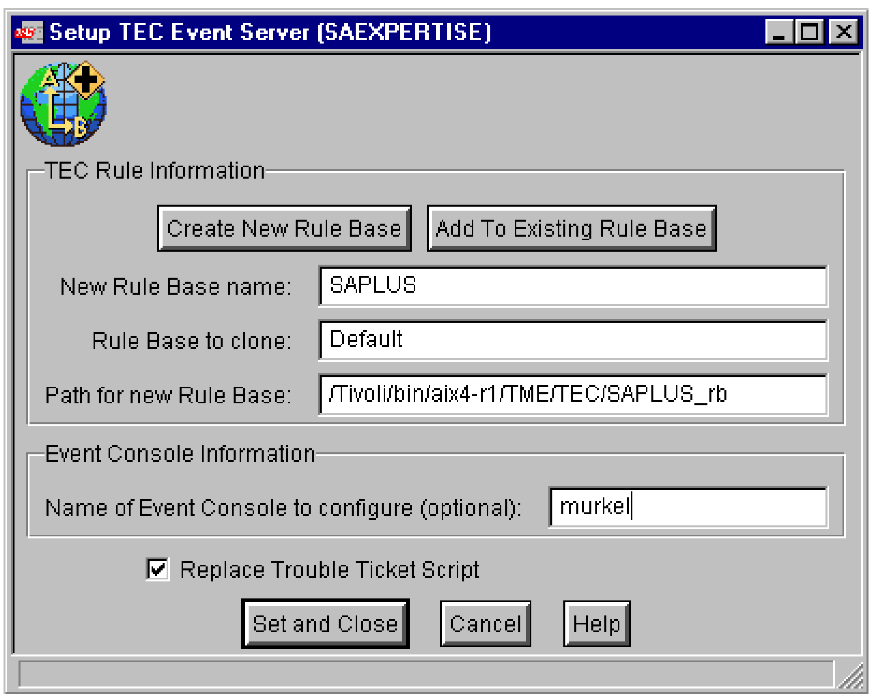


Figure 75. Setup TEC Event Server Task

Enter the appropriate values and click on **Set and Close**.

It is a good choice to create a new rule base instead of modifying an existing rule base. In case the newly created configuration does not work correctly, it is a simple task to revert to the original rule base.

After having clicked on **Create a new rule base**, it is possible to provide the necessary information for task completion.

Note

Checking the **Replace Trouble Ticket Script** option is not a destructive process.

The PLUS module installation script automatically does a backup of the three scripts sa_ticket.sh, notify_admin.sh and CloseTivoliTicket.sh. You will find these copies in the same directory as the original scripts with the extension *.sa*.

If the task completes successfully, the event server is restarted and the task should give you the output shown in Figure 76 on page 105.

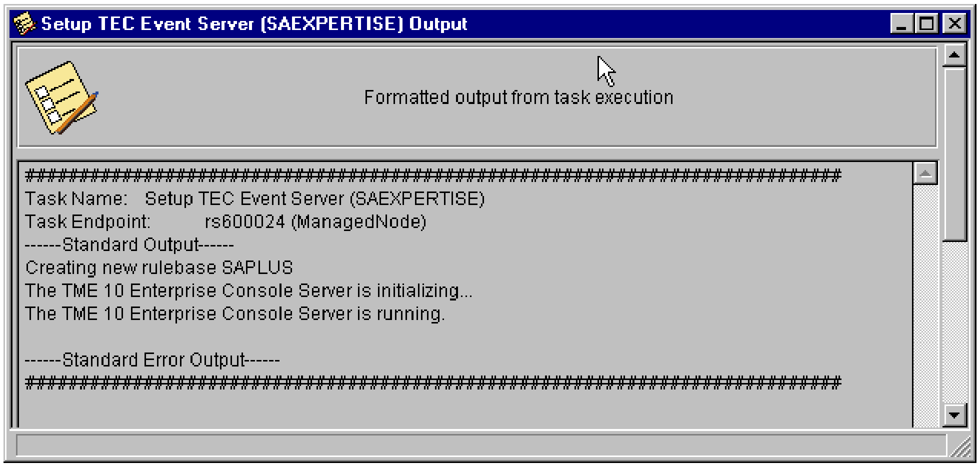


Figure 76. Successful Execution of Task Setup TEC Event Server

If you run the task the second time and every time after that, you will get a message in this screen informing you that the three above-mentioned scripts already exist and they have been copied to save them.

Taking a closer look at the newly created rule base, you can see that the module added new class definition files:

- sa.baroc
- SAExpertise.baroc

Two new rulesets are also provided for the distributed monitors. These are:

- sa.rls s
- aexpertise_monitors.rls

In the file sa.baroc the class SA_Base is defined. In this class definition the additional slot managed_object is defined, which later on provides the capability to forward the information in that slot to ExpertAdvisor as a unique identifier for the creation of a trouble ticket. We did not use the SA_Base class or one of its descendants for our environment, so we chose to add the managed_object slot to other classes. (For details see 6.2.6, “The Event Class Definitions” on page 273.)

4.16.2.2 Configuring SA ExpertView

To configure the ExpertView gateway, double-click on the collection **SAExpertise ExpertView** and Figure 77 on page 106 will appear.

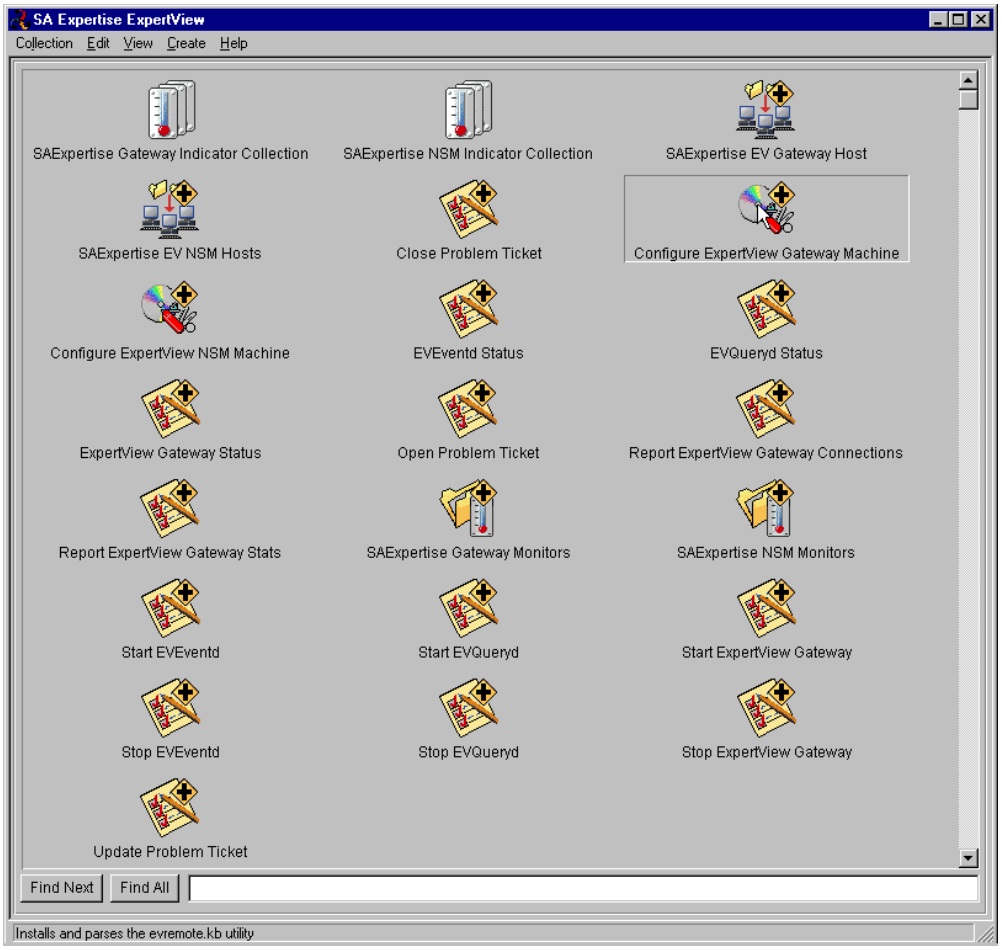


Figure 77. Contents of the Collection for SAExpertise ExpertView

Select the **Configure ExpertView Gateway Machine** task.

This task installs and parses the evremote.kb utility. We experienced two problems in running the task with our NT ExpertView gateway.

1. Failed subscription

   The Plus module creates various profile managers during the installation and then subscribes the managed nodes, whose names were entered in the installation dialog shown in Figure 72 on page 101, to the appropriate profile managers. In our installation, the subscription of the ExpertView gateway failed (with no visible error message in the installation output or anywhere else). As a result, the task did run without any output, but accomplished nothing. After subscription of WTR05368 (the ExpertView gateway) to the profile manager SA Expertise EV Gateway Hosts, the task completed successfully. However, in a second installation the subscription was done automatically and the problem didn't show up.

2. Task hanging

   The task started, but failed to complete without producing any output. There were no error messages or timeout warnings. We then started NT's task manager to see what happened during the task execution and realized that a process named kp.exe was using more than 90% of the system's processor time and was seriously affecting its performance. We killed kp.exe and the task produced the following error message.

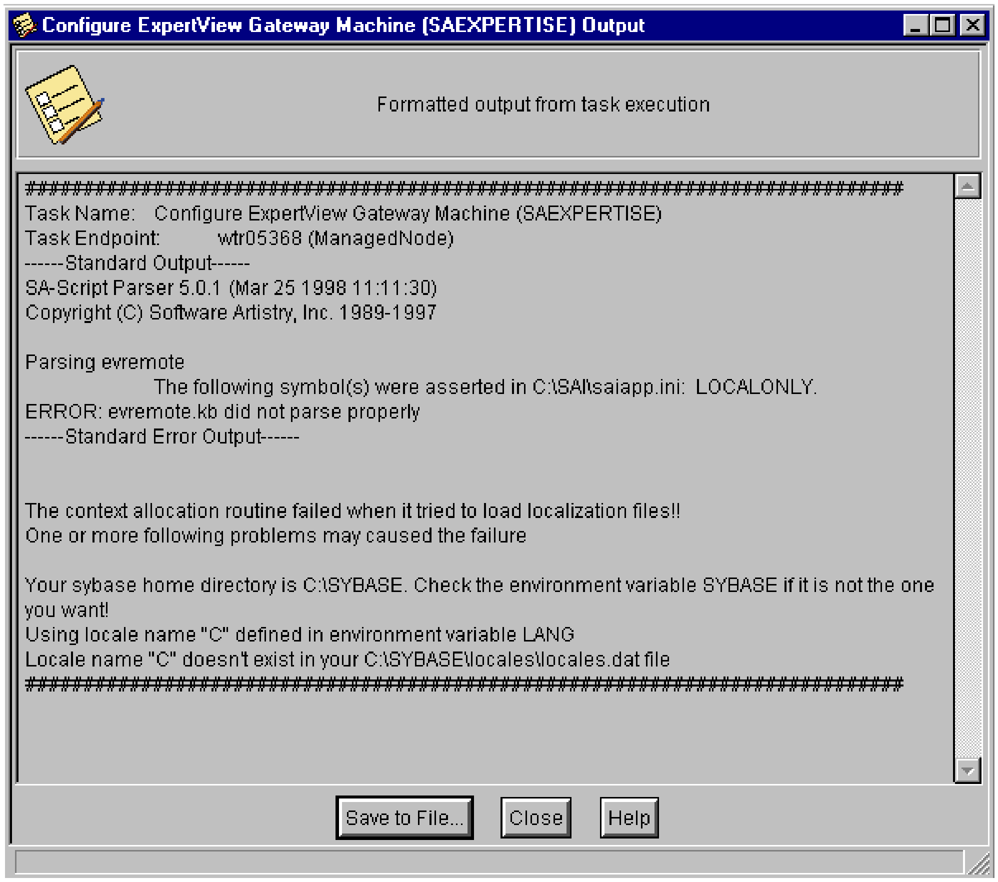


Figure 78. Configure ExpertView Gateway Machine

This is caused by the different language environments of the different Tivoli desktop systems. Depending on the environment the task is started from, the environment variable LANG is set. Therefore, depending on the LANG variable, a certain line has to be added to the file locales.dat in the sybase subdirectory locales (see Figure 79 on page 108).

Open the file and look for the NT section in that file.

```
; comment character is a semicolon
;
; @(#) locales.dat 1.2 12/8/93
;
; Sccsid: @(#) locales.dat 1.7 4/18/94
;
; LOCALES.DAT
;
; Flat-file database for converting vendor-specific international
; locale information to sybase proprietary format.
;
[file format]
version=10.0.1
                                   Task started from newman's desktop (NT)
...cut out parts of file...
[NT]
locale = C, us_english, iso_1
locale = en_US, us_english, iso_1
locale = default, us_english, cp850
locale = enu, us_english, cp850
locale = fra, french, cp850
locale = deu, german, cp850        Task started from rs60008's desktop (AIX)

...cut out parts of file...
```

Figure 79. File Contents for \sybase\locales\locales.dat

Add the appropriate highlighted line to the file and save it. The lines you would have to add may vary depending on your language environment. The output the task provides in case of failure is usually sufficient to know which line has to be added. You find lines that contain the correct locale configuration in other sections of this file. They just need to be copied to the NT section.

When we started the task again the output was now correct and is shown in Figure 80 on page 109.

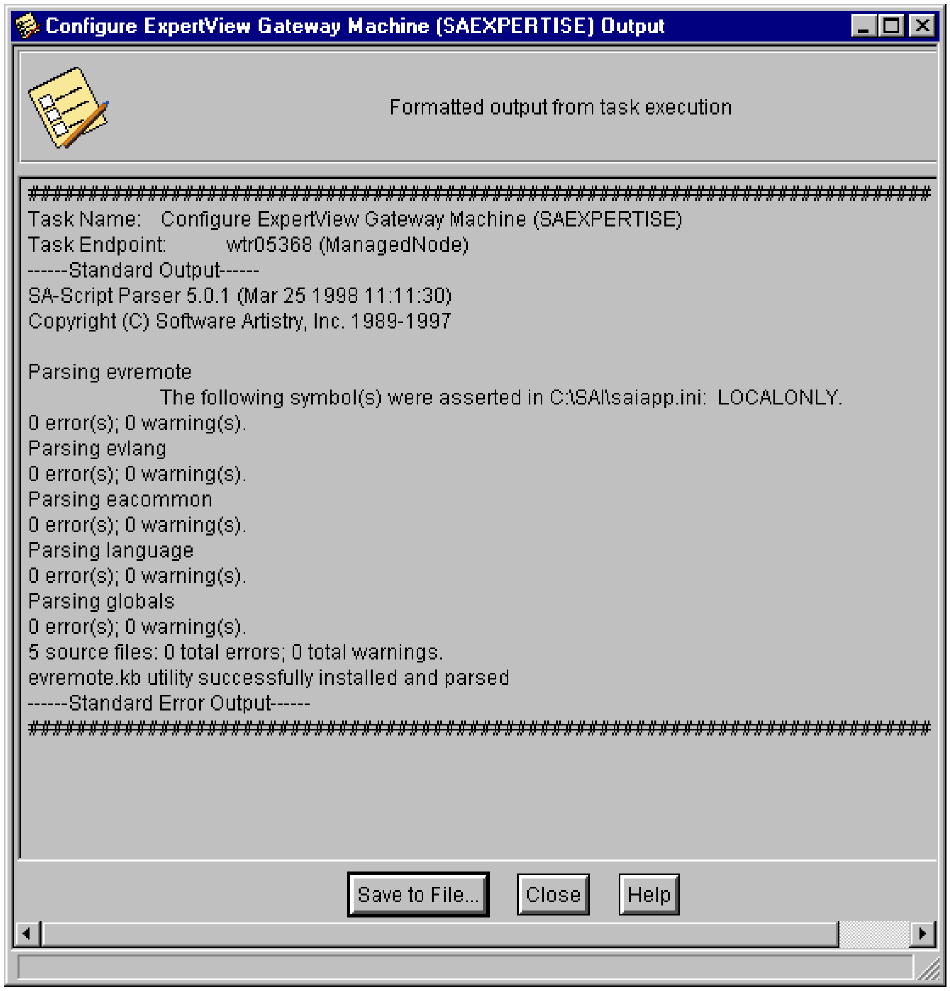


Figure 80. Correct Output of Task Configure ExpertView Gateway Machine

After completing this task, we ran the next configuration task in the collection, which is Configure Expertise NSM Machine.

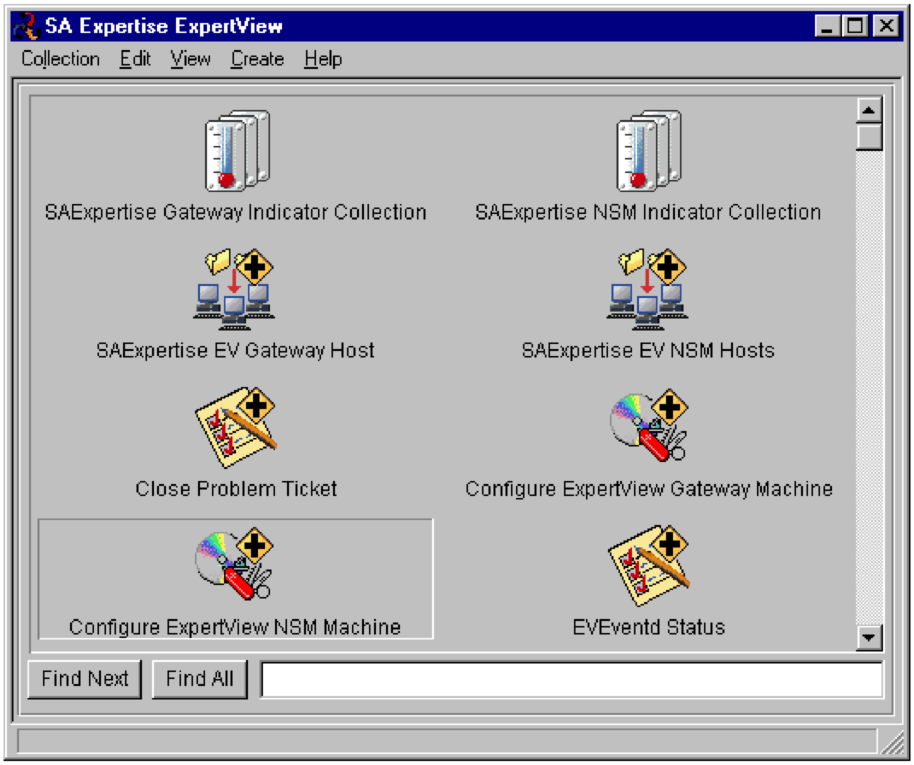


Figure 81. Configure Expertise ExpertView

Double-click on **Configure ExpertView NCM Machine** as shown in Figure 81 on page 110.

From Figure 82 on page 110 we entered the location of the EV_Base_Directory (the directory that contains the EVProb command) and the name of the ExpertView Gateway Machine.

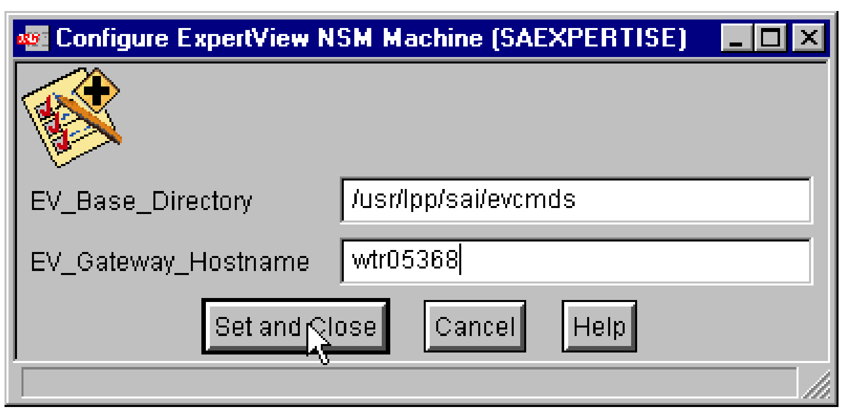


Figure 82. Task Dialog

Only enter the pathname where the NSM commands are located on the NSM machine. Do not enter the name of the machine in front of the path.

We entered the name of the ExpertView Gateway machine and clicked on **Set and Close**.

The next screen is shown in Figure 83 on page 111.

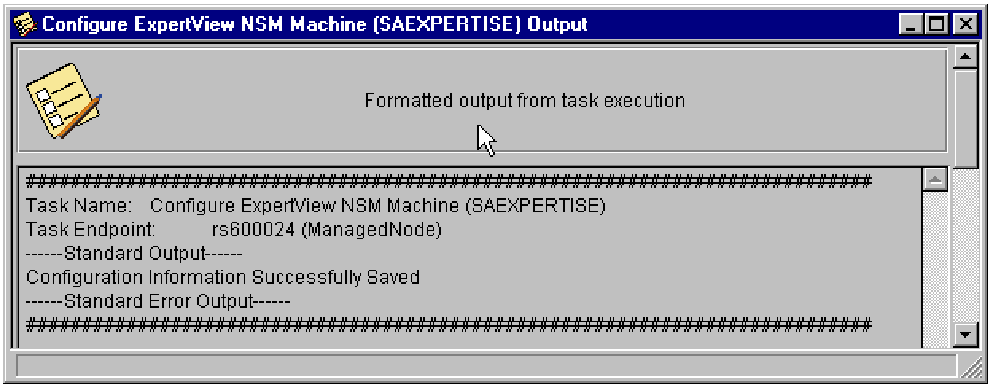


Figure 83. Task Output

These were the configuration tasks we had to run in order to have our environment up and running. In case you intend to deploy another environment containing one or more other SA function-specific servers such as a notification server or an escalation server, you would of course have to complete the other appropriate configuration tasks as well. For more information see the *SA-Expertise/Plus for Tivoli User's Guide*, which is delivered with the Plus module.

4.17 Tivoli Decision Support

In this section we explain the architecture of Tivoli Decision Support, also referred to as TDS, and the functionality of different modules. Next we go through the installation of all the modules for the Tivoli Decision Support application by performing the following installation steps:

- File server
- ADI client
- Administrator workstation

4.17.1 Installation Options

We can select to run Tivoli Decision Support in stand-alone mode or network mode. In stand-alone mode everything runs on one system, and none of the modules are shared with anyone else in the network. The network mode configuration implies a client/server connection, with shared files on the server and the client interface, the Tivoli Discovery Interface (TDI), installed on the end-user workstation, plus the PowerPlay module as a prerequisite on any of the machines.

When multiple users want to run Tivoli Decision Support, the best way is to install the program's component in network mode. Alternatively, if TDS will be used by a single user, we can choose to install it in stand-alone mode, where all the components are in a single machine.

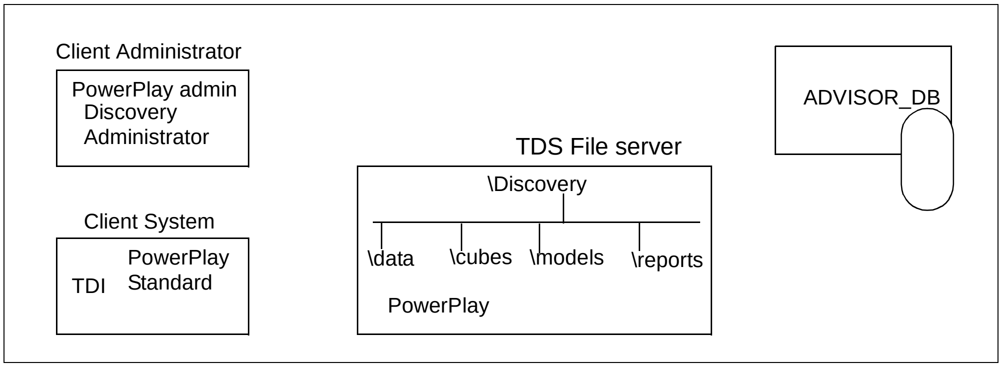


Figure 84. Expert Discovery Modules in a Typical Network Installation

Figure 84 on page 112 shows the modules for a typical network installation. The file server, in our case WTR05368, contained the binaries. PowerPlay is installed on the client system and the client administrator.

4.17.2 Component Installation

When we first launch the setup program we can see different options to install:

- Discovery Interface (TDI)

  This is the client interface, the end user's entry point into Tivoli Decision Support. Using the TDS Discovery Interface, the user can navigate the support center's database and retrieve information from it. The requisite is a 32_ODBC connection to the database server and shared access to the file server.

- Server components

  In a client/server environment, such as the one we decided to install, all views, reports and models are located on a shared network drive for use by the TDS and administrator.

  - PowerPlay reports (PPR files)
  - PowerPlay models (PYG files) or the model definition files, the link between real data and cubes

    This is where all the cube generation is done, and we need a fast machine with room on the hard disk to create cubes. Cubes will vary depending on how much data is captured in them, mostly within a range of 2 MB to 7 MB each.

  - Crystal reports (RPT files)
  - Content database (MDB files)
  - Administrator database (MDB files)

- Administrator module

  The administrator module performs configurations tasks, maintains all the cubes used by TDS, provides specific parameters for building or customizing cubes, and makes calculations when generating views.

- Cognos PowerPlay

This third-party application must be installed on any of the systems that will run Decision Support, so in a network mode installation we need to install it both on the server and on the client machine.

- Crystal Reports

  Although the end user does not necessarily need to use Crystal Reports tools or commands, Decision Support uses these tools to create views of data. The Crystal Reports are installed from a separate CD-ROM. It only needs to be installed if modifications need to be made to the default reports.

- ODBC

  The connection to the Problem Management database is done via the ODBC. The required 32-bit ODBC drivers are installed for the RDBMS you are using.

4.17.3 Installation of Tivoli Decision Support

In our environment, we chose to install the Discovery server component on the ASE file server. To start the installation run setup.exe from the CD-ROM.

Figure 85 on page 113 will appear. Select **Install Tivoli Decision Support**.

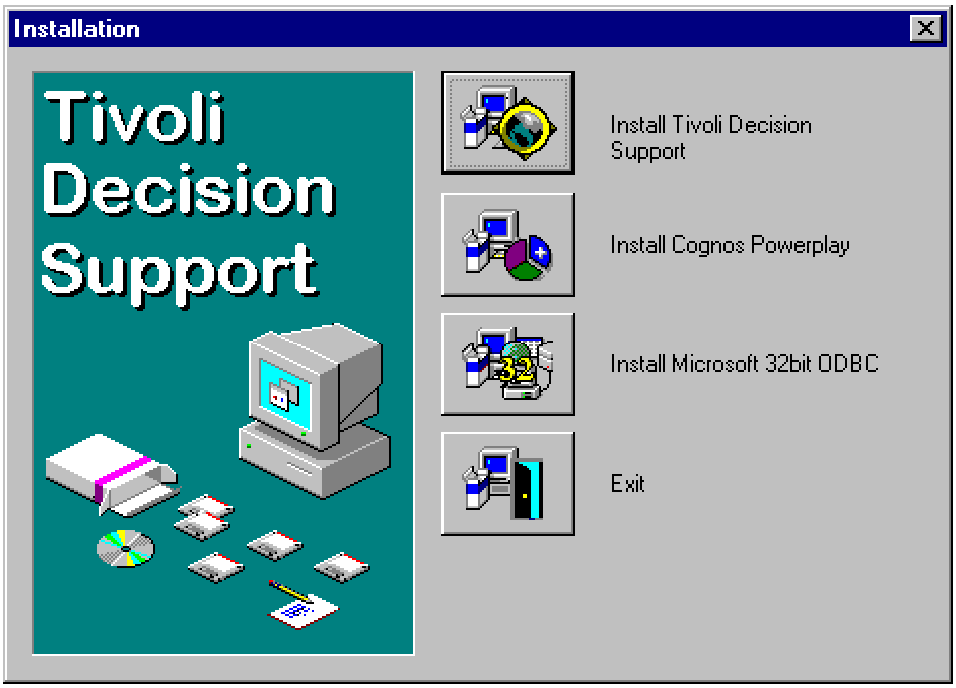


Figure 85. Tivoli Decision Support Installation Dialog Box

Figure 86 on page 114 prompts for the component to install, and subsequently asks for a destination directory. We entered `C:\Program Files\TDS` which is the default.

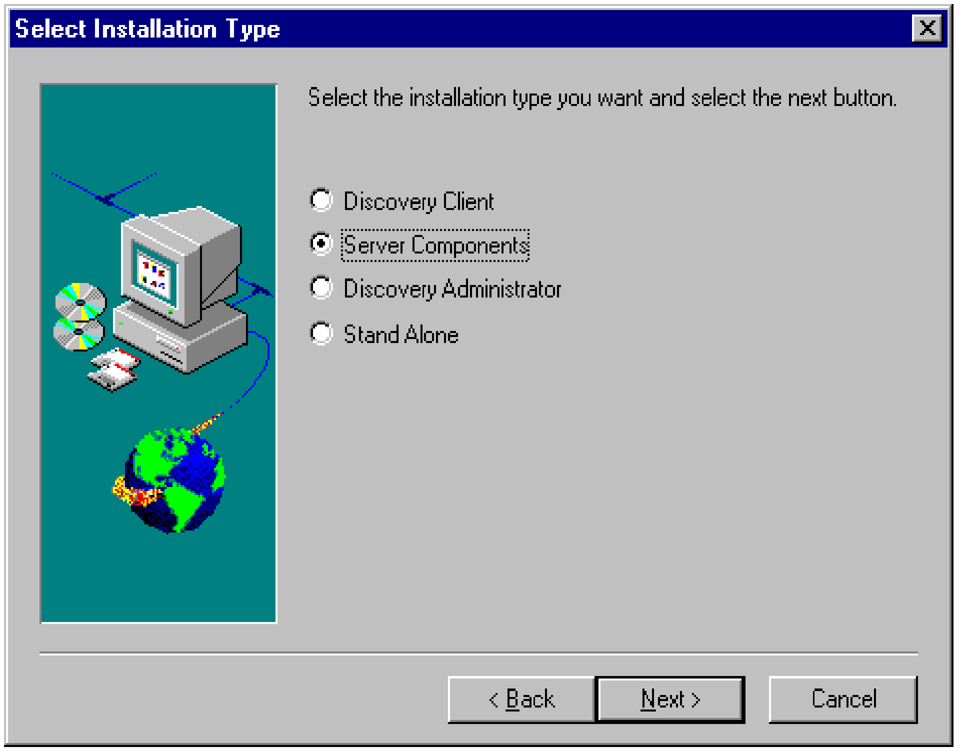


Figure 86. Select Installation Type

Now the server component is installed. As specified in the *Tivoli Discovery Installation Guide*, we need to install the PowerPlay component on every machine in our environment.

From the Installation dialog box, we chose **Install Cognos PowerPlay**, selecting the standard module, as you can see in the following screens.

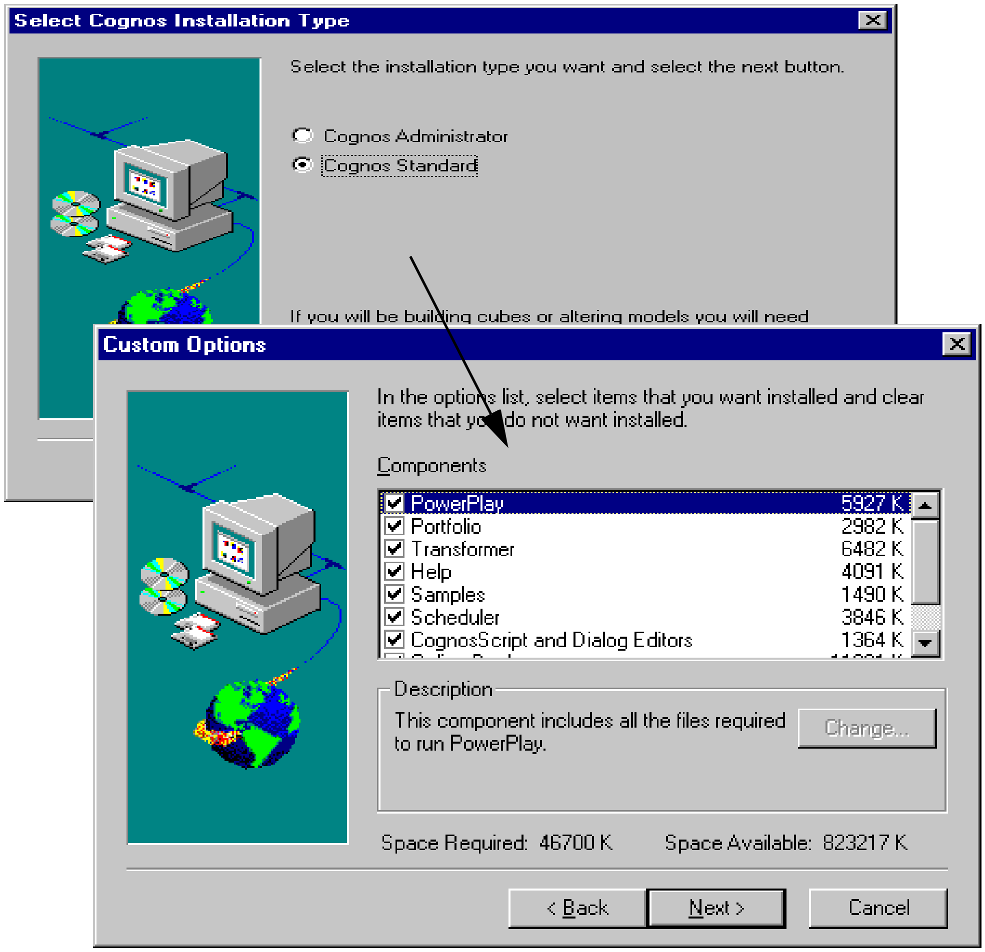


Figure 87. Tivoli Decision Support: installing Cognos PowerPlay Standard

Choose the custom installation to see more information on which components are to be installed. One of the components is Acrobat Reader used to view the .pdf manuals online.

4.17.4 Installation of TDS Client

To install the client workstation, we started the setup program just like in the server’s installation, but this time we select the client component (see Figure 86 on page 114).

Next we are asked where to put all the files, icons and TDS components. Since we have a file server in our networked configuration we select the shared disk of our file server as shown below:

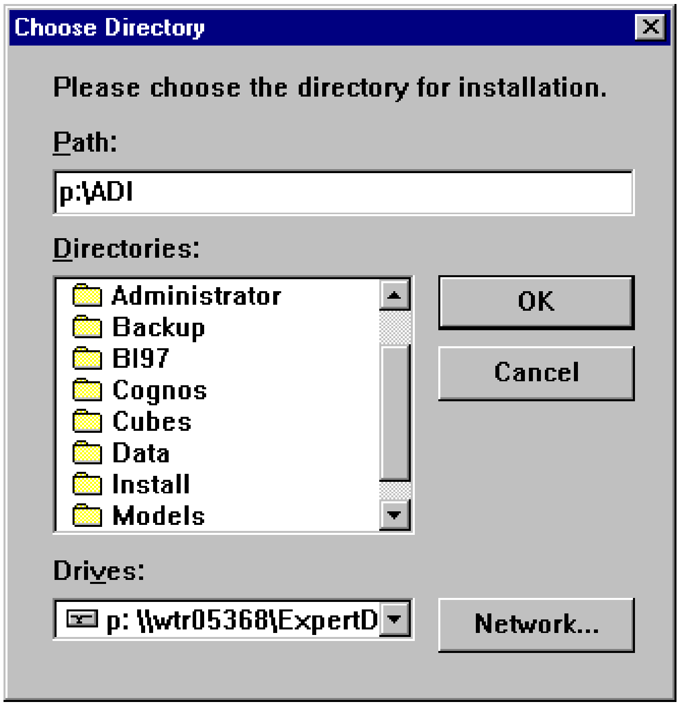


Figure 88. Decision Support: Selecting File Server Directory for ADI Installation

TDS, the user's entry point to Decision Support, is now installed and a new program group is added to NT Program Manager.

We still need to install the Cognos PowerPlay component on this client as it is a prerequisite for running any view or Crystal Report. We do this in the same way as for the file server. For the client we selected drive p:.

4.17.5 Installing Decision Support Administrator's Workstation

We now prepare the Decision Support administrator's workstation. This machine should be able to connect to both the Decision Support file server and the problem management databases, which in our environment is ADVISOR_DB.

The Administrator Component is installed and operated separately from TDS. The Discovery Administrator must be installed on at least one system at the support center.

From the CD-ROM run setup.exe. From the setup dialog select **Discovery Administrator** from Figure 86 on page 114.

As you can see from Figure 89 on page 117 we selected to use the same file server configuration as we had for the TDS installation, putting all the files in a server's shared disk, but creating a specific directory for the Administrator's module (see Figure 89 on page 117).

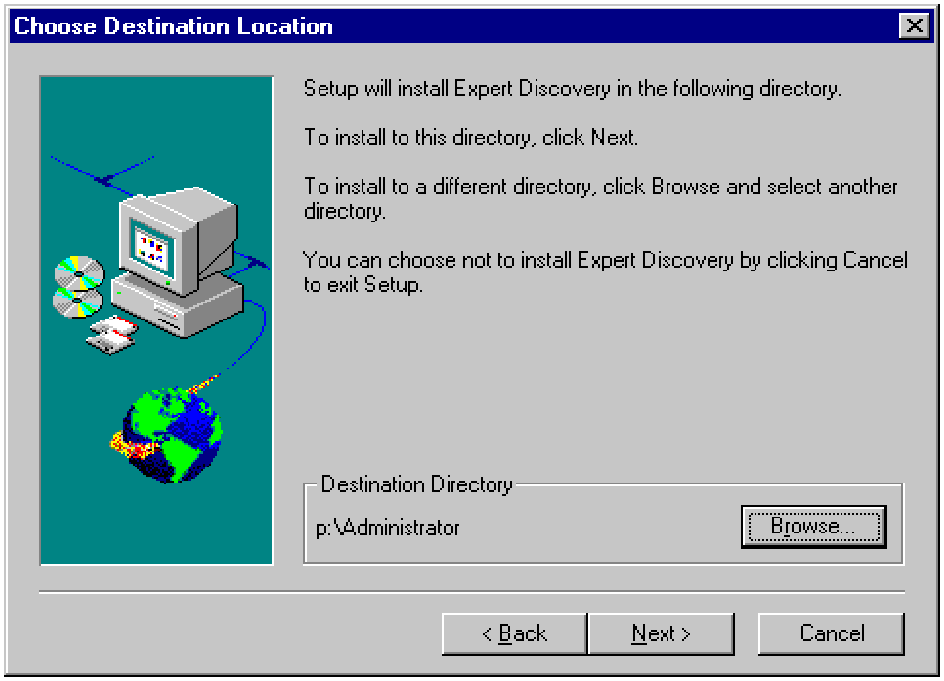


Figure 89. Tivoli Decision Support: Installation of Administrator Module

Click on **Next** to install the application.

This time we were asked to restart our NT Workstation.

4.17.6 ODBC Driver Installation

The connection to the ADVISOR database is achieved via the ODBC. In our environment we are using sybase for the ADVISOR database.

From the install screen shown in Figure 85 on page 113 we selected **Install Microsoft 32bit ODBC.** Then we continued to install the ODBC drivers for sybase.

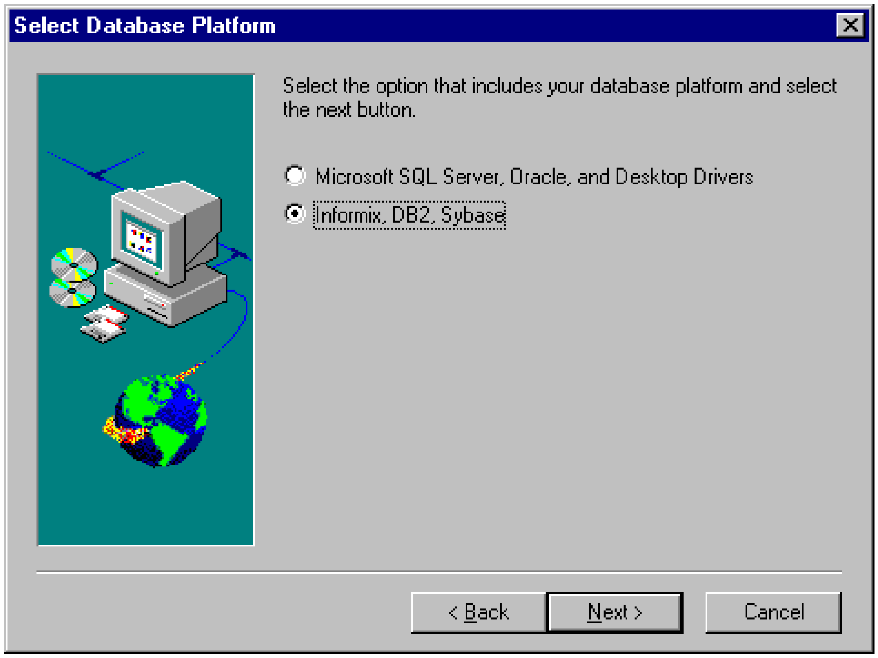


Figure 90. Selection of Sybase Drivers for Tivoli Decision Support

From Figure 90 on page 118 select the Sybase option followed by **Next**.

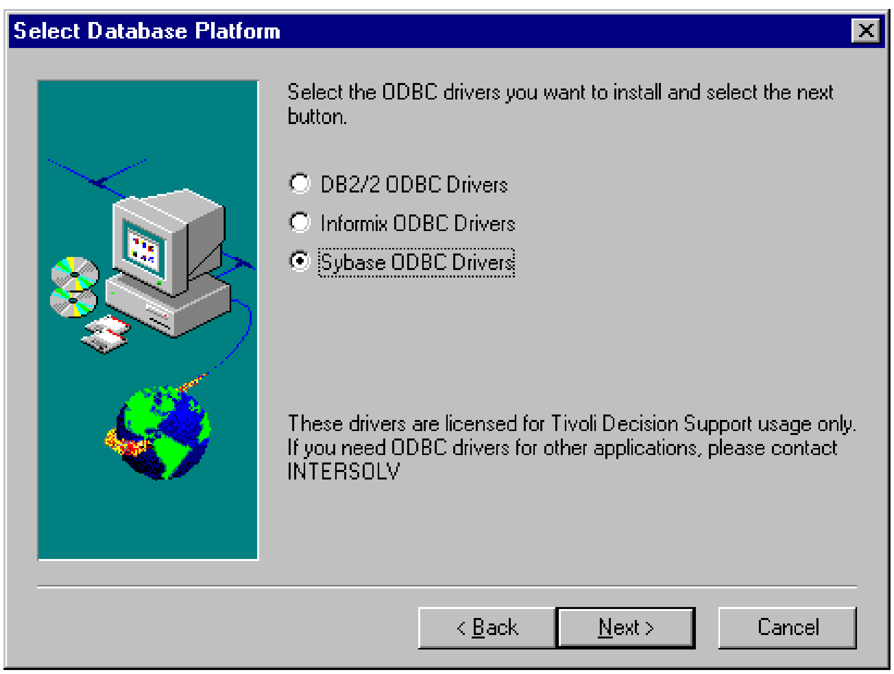


Figure 91. Sybase ODBC Driver

Click on **Next** to install the drivers (see Figure 91 on page 118).

4.18 Knowledge Paks

This section discuss the installation and the functionality of the Expert Advisor version of the Knowledge Pak Desktop Suite by ServiceWare Inc.

4.18.1 Installation

To perform the Knowledge Pak installation you need the following:

- The Knowledge Pak software and license key
- The Interface Generator and the Script Parser tools to be installed
- The Expert Advisor installed

The installation process described consists of the following main phases:

1. Install temporary Knowledge Pak files locally on the Windows (NT in our case) system. This is a temporary installation on the local disk. It will install the Installation and User Guide in HTML format, some script files and the Knowledge Pak data files.
2. Parse the installation utility source files. This will create an installation program to load the Knowledge Pak data on the ADVISOR database.
3. Load the Knowledge Pak data on the ADVISOR database. This will load new SCIM entries and new data for the AIDs tools on the ADVISOR database.

4.18.1.1 Temporary Knowledge Pak Installation

To start the installation process run the setup program located on the CD-ROM. You will be prompted for the license key. Once entered the screen shown in Figure 92 on page 119 will appear.

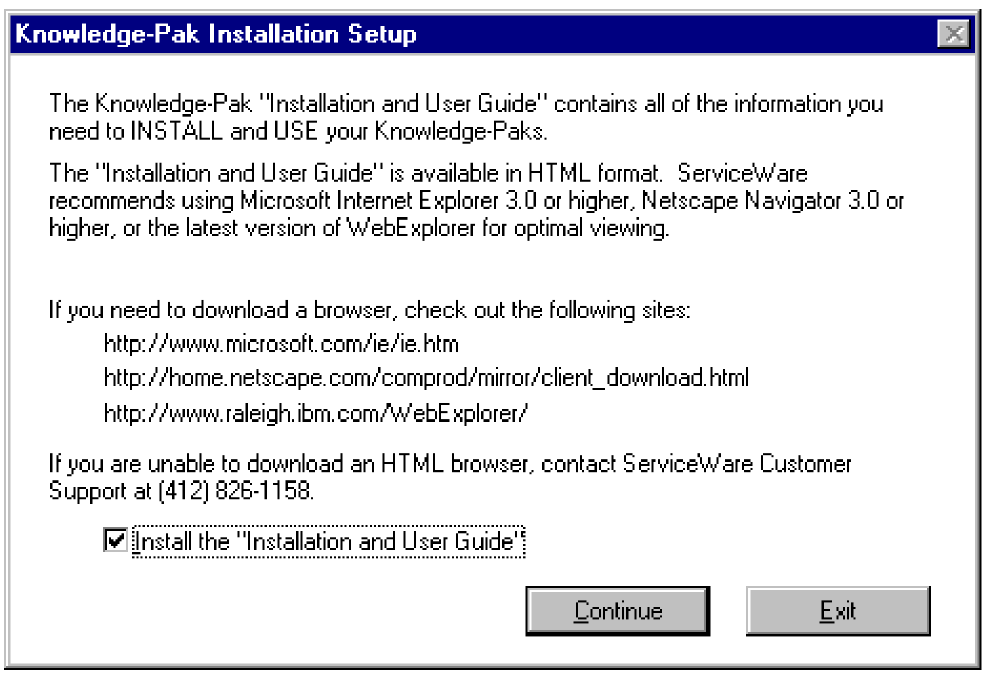


Figure 92. Knowledge Pak Installation

Ensure that the **Install the Installation and User Guide** box is selected, then select **Continue**. The installation program will prompt you for the path where you want the Knowledge Pak files and the Installation and User Guide installed.

The screen shown in Figure 93 on page 120 will appear.

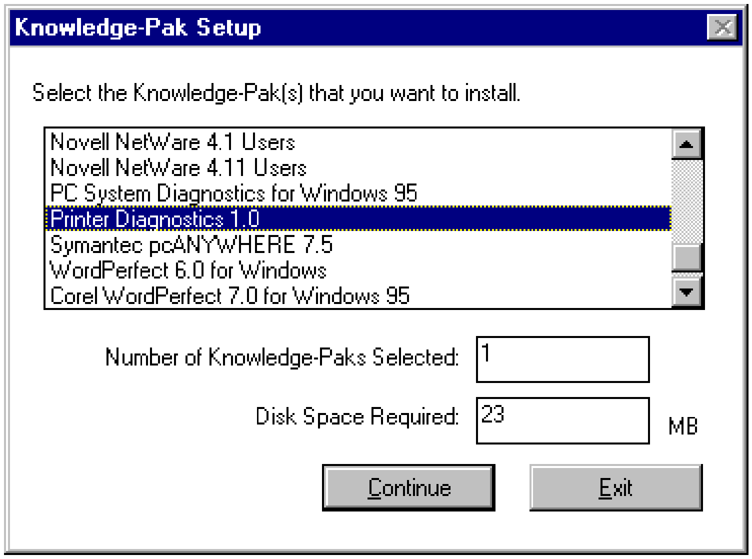


Figure 93. Knowledge Pak Selection List

From this panel we selected the Knowledge Paks we wanted to install, followed by **Continue**. This will install the data files on the local disk.

Once installed a message box will confirm the successful installation of the local install.

4.18.2 Parse the Installation Utility

Next we parsed the Knowledge Pak Installation Utility screen and the Knowledge Pak Installation Utility program.

Before performing the parsing you may need to edit the file .SCM located under the subdirectory script of the installation path you chose. This is only needed if you want to modify the SCIM entries that are going to be created by the Knowledge Pak installation.

For the scenarios we show we do not change the file.

For a more detailed description of the parsing please refer to the *Installation and User Guide.*

As part of the installation you must rename the file KPFINST5.DF, located under the directory c:\SAI\WINASE\32 to KPFINST.DF.

Start the Interface Generator and parse that file.

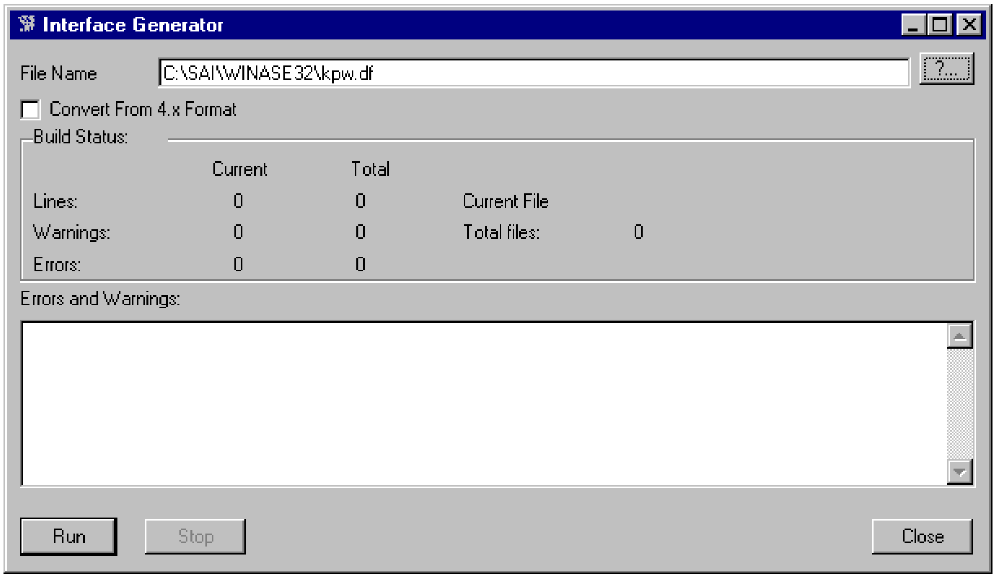


Figure 94. Parsing the Knowledge Pak

Start the script parser select the file KPFINST.KB.

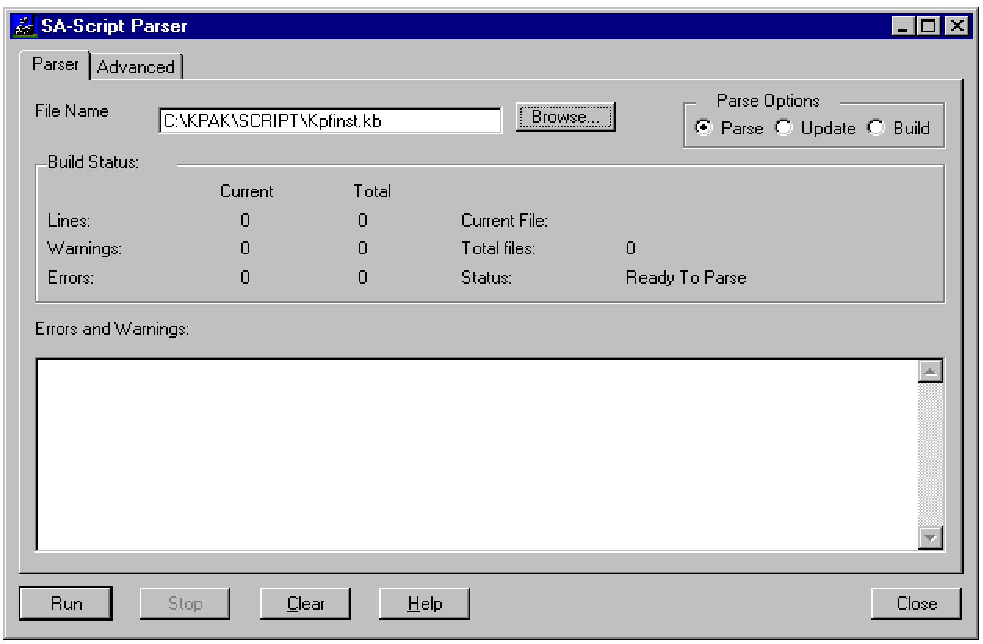


Figure 95. Parse the Knowledge Pak Installation Program

4.18.3 Loading the Knowledge Pak Data

So far we have created the Import utility that loads the Knowledge Pak data into the ADVISOR database, but nothing has been yet loaded into the database.

To start the Installation utility, double-click the icon of the file KPFINST.KBC, which you will find under the script subdirectory. This file should be associated with the KML.EXE program. The Installation utility will start and prompt for the user and password. Use the administrator user of Expert Advisor so that the

Installation utility will have full access to the database. The Installation utility main window is shown in Figure 96 on page 122.

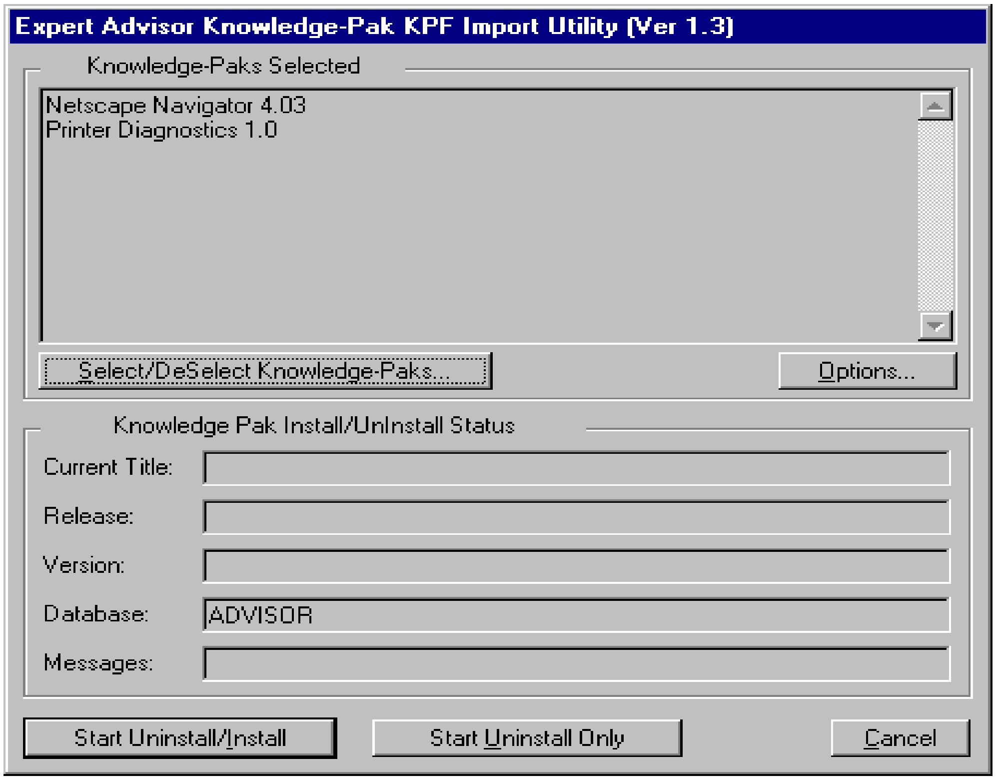


Figure 96. Import Utility

By the Select/DeSelect Knowledge Paks button select the Knowledge Paks you want to install. We selected Netscape and printer diagnostics (see Figure 97 on page 122).

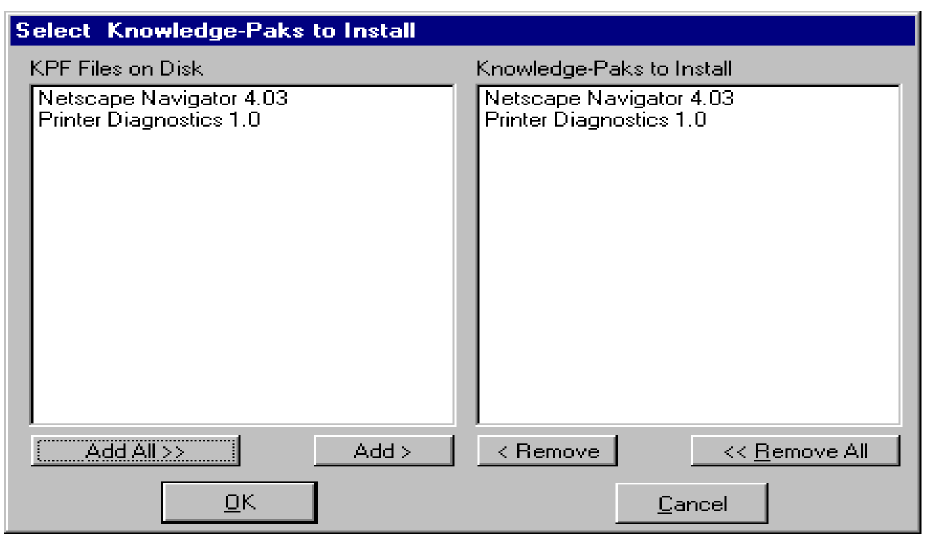


Figure 97. Select Knowledge Paks

After selecting the Knowledge Paks click on **OK**. Next select the **Options** button and set the import options shown in Figure 98 on page 123.

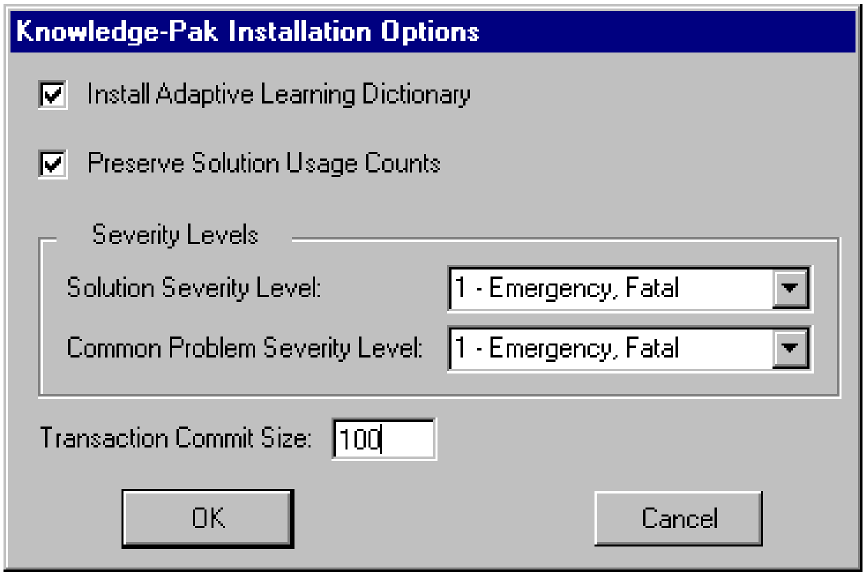


Figure 98. Select Installation Options

Click on **OK**. After that select the **Start Uninstall/Install** button on Figure 96 on page 122 to initiate the data import. This process will load the new SCIM and solution data.

4.18.3.1 Clean Temporary Files

At this point you can remove the temporary installation directory on the local Windows machine.

4.18.4 Knowledge Pak Data

After the installation of the Knowledge Pak, you will see more SCIM entries in the ADVISOR database and new solutions created. The adaptive learning information will have been added. Additional information will be added to the error messages and Hyper-Trees.

For example, during a call registration if you enter the word print in the Description field and then click the **ADL** button, you will see a list of possible solutions related to printer problems (see Figure 99 on page 124).

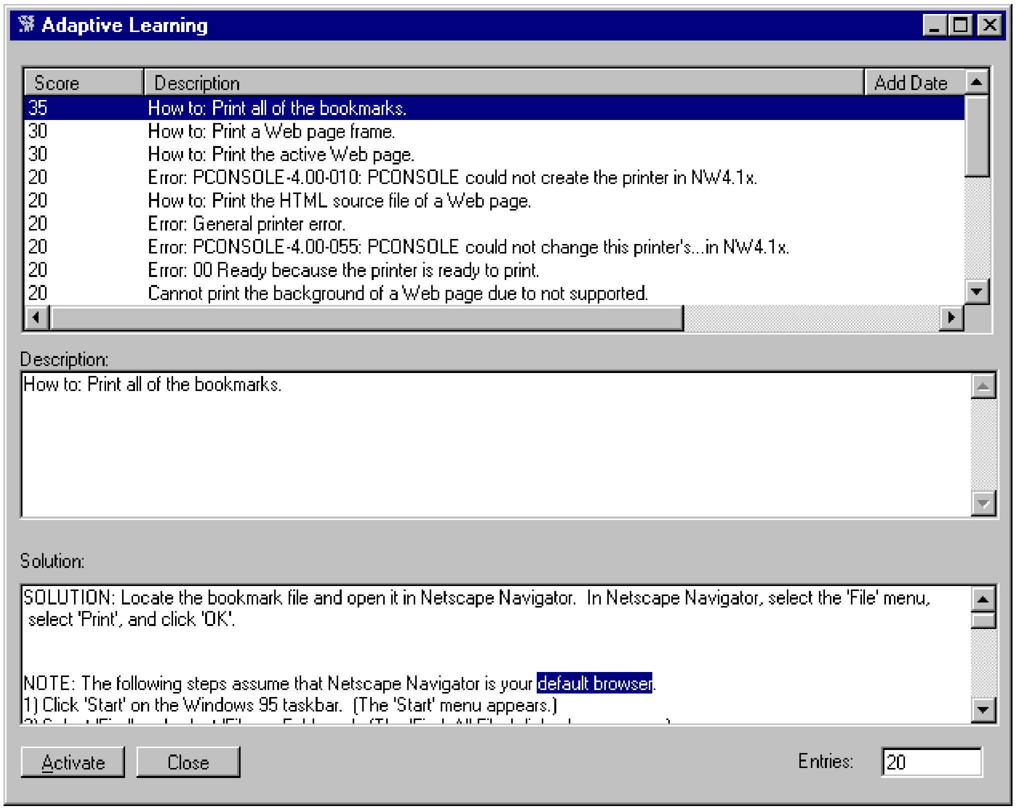


Figure 99. ADL Solutions

The Knowledge Pak provides a large set of solutions covering many common products such as Lotus Notes, Microsoft Office, and Netscape. The Knowledge Paks provide immediate benefits to a support desk.

Without the Knowledge Pak the initial solution database is virtually empty. It is the advisor user's and administrator's responsibility to add solutions while they resolve problems. This means that when the IT support organization starts to support a new product, there is no significant help from the AIDs tools. The Knowledge Pak provides a set of solutions that can be pre-loaded so that at the initial phase of new product support you can have benefit from the AIDs tools. New solutions can still be added and the solutions database expanded, during the day-by-day support activities.

Chapter 5. Setting Up the Problem Management Environment

This chapter shows the step-by-step guide on how to configure the management applications.

At this stage all the applications have been installed and are functioning correctly. The next stage is to configure the environment. This involves entering specific data into the Tivoli Service Desk database for our environment, with the main focus in the Tivoli Service Desk applications. The TMR environment is explained in more detail in Chapter 6, “Problem Management Scenario Using the TEC” on page 241.

Table 11 on page 125 outlines the steps we performed to set up our environment.

Table 11. Setting Up - Task List

Task	Title	Description
Phase 1		
1	TMR Environment	Setting up the TMR environment and configuration of the Tivoli applications.
Phase 2		
2	Location Definitions	Adding the locations and contact information.
3	Workflow Definitions	Define the call registration options.
4	Defining the SCIM	Creating the SCIM definition for our environment.
5	Diagnostic Aids	Setting up the diagnostic functions for our environment.
Phase 3		
7	Network/Systems Management Tools	Discussion on creating the level 2 support tool structure.

First we discuss the TMR configuration.

5.1 Base Configuration for the TMR Environment

The Tivoli Management Environment was configured based on information we developed from the event methodology requirements. This is the main input to the design of the TMR environment. Other parameters are what the software distribution requirements are.

The configuration of the TMR is not relevant to the overall configuration of the problem management solution. We could have chosen any number of configurations such as multiple TMRs.

The important configuration as far as the TME environment is that of the TEC. This is where the trouble tickets will be generated and forwarded to the Tivoli Problem Management application.

The base TMR configuration was established and the following TMR environment was created. Although not the main focus of this chapter, the TMR environment setup is summarized below:

- Create the managed nodes (AIX and NT servers)
- Create Tivoli management gateways
- Distribute the logfile adapters to the AIX and NT servers based on the event management requirements
- Set up distributed monitoring based on the event management requirements
- Configure the polling and threshold parameters for Tivoli Distributed Monitoring
- Configure Tivoli Inventory
- Define administrators
- Define access rights
- Create a new rulebase
- Activate the new rulebase

Figure 92 on page 128 shows the policy region rs60008-region we defined for our environment.

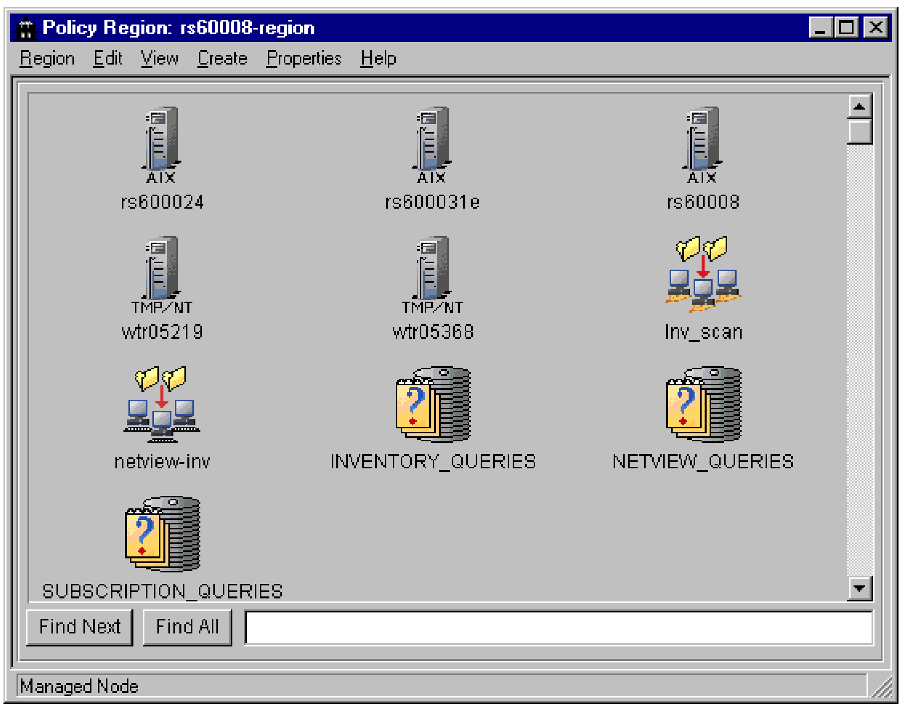


Figure 100. rs60008-Region

For the purpose of testing the new features in TEC 3.6 we also defined a number of gateways as shown in Figure on page 127.

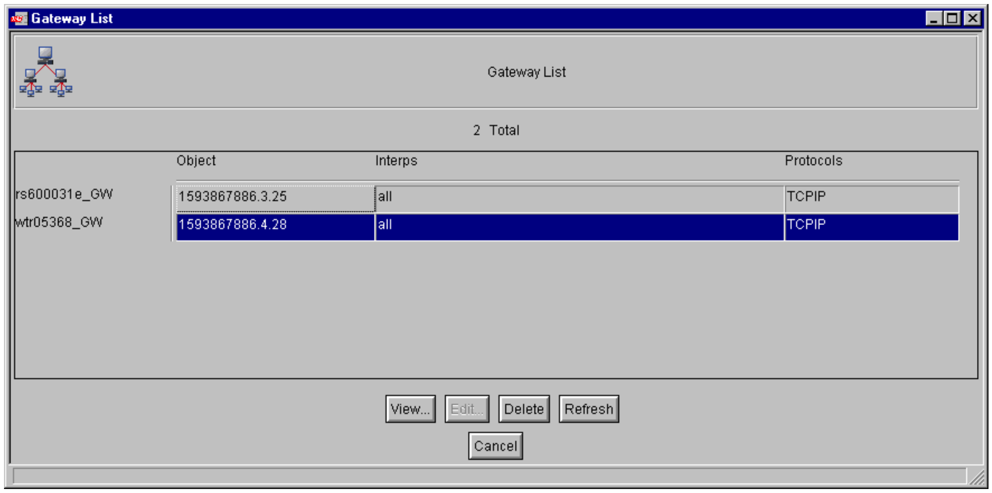


Figure 101. Gateway List

Once the TMR environment has been set up we can look at the problem management applications. First we defined the users and location information.

5.2 Setting Up Locations and Caller Information

When a call is registered using the Problem Management application the initial screen shown in Figure 102 on page 127 appears. The more information we can enter here the better chance that a problem can be resolved quicker.

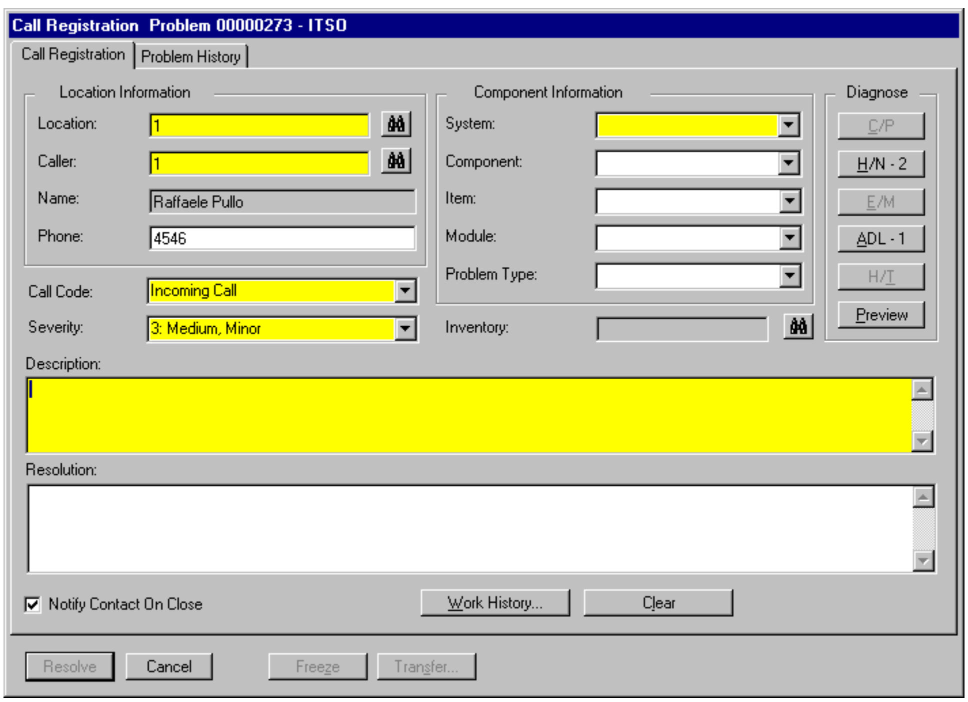


Figure 102. Main Call Registration

The user and location information is entered using the following procedure:

First select **Edit** followed by **Locations** from the main window (see Figure 103 on page 128).

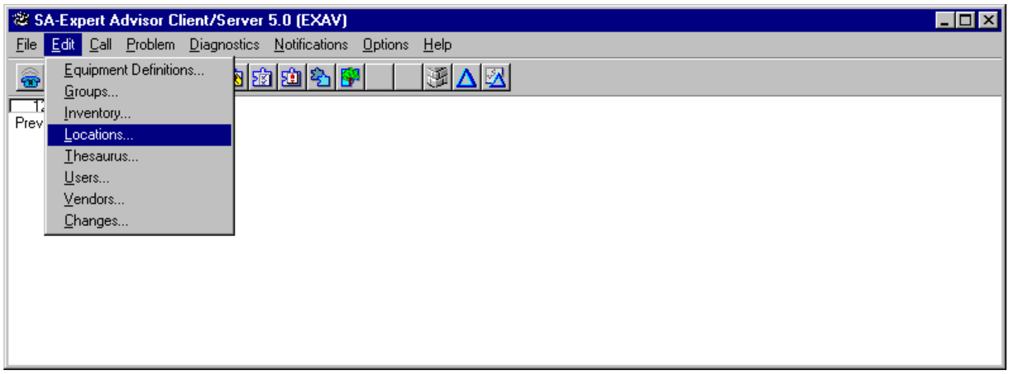


Figure 103. Main Screen

You can enter an asterisk in the Id field to reveal all the defined location codes (see Figure 104 on page 128).

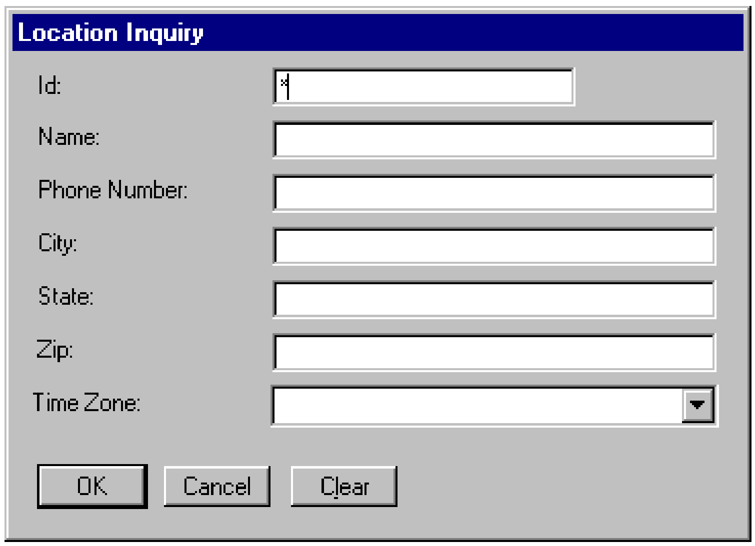


Figure 104. Location Enquiry

Click on **OK** to show the defined locations (Figure 105 on page 129).

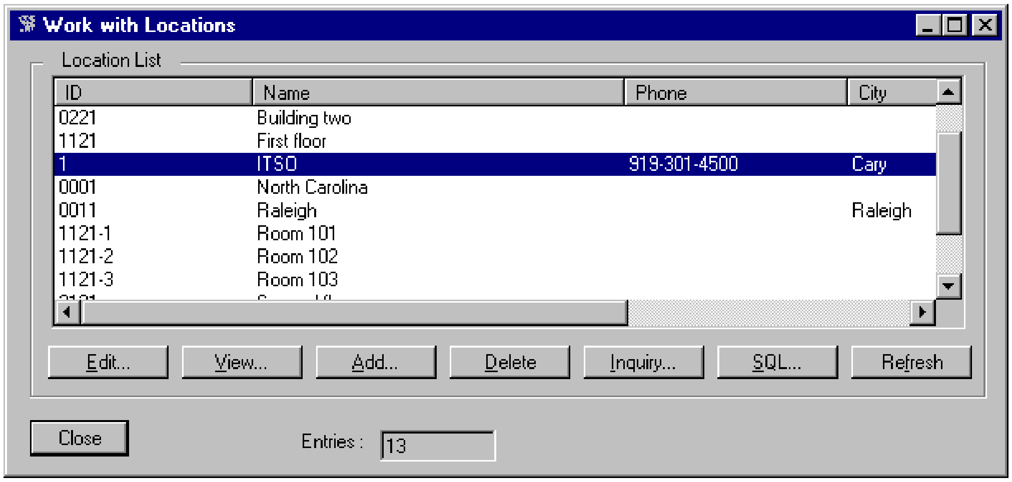


Figure 105. Work with Locations

From here you can add, delete, and edit the locations. We select **Edit** to see our existing location information (see Figure 106 on page 129).

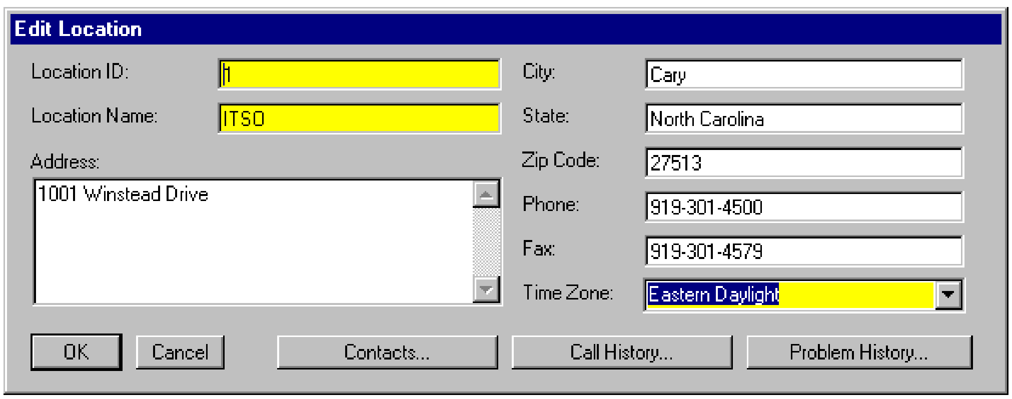


Figure 106. Edit Location

The ITSO is defined as Location ID number 1. Next we set up the contacts for this location by clicking on **Contacts** (see Figure 108 on page 130).

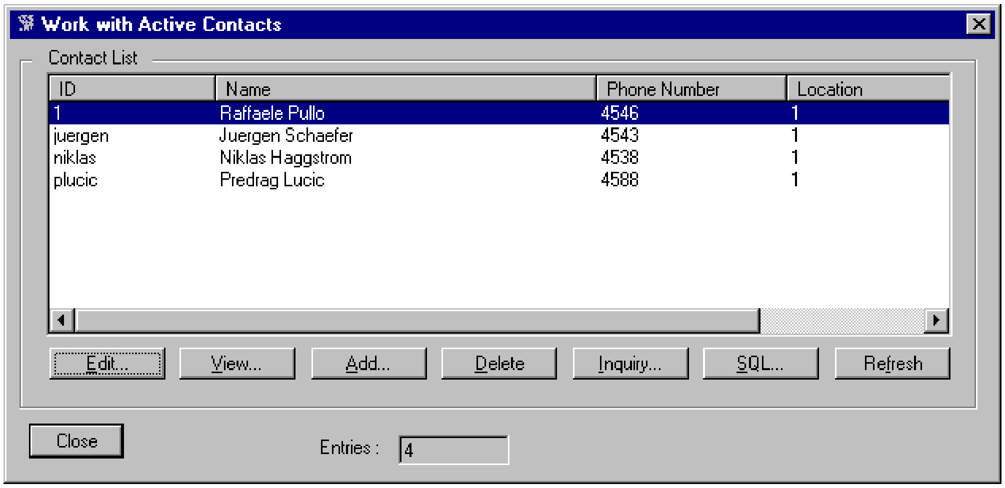


Figure 107. Work with Active Contacts

Figure 107 on page 130 reveals the list of contacts. We defined the user ID of 1 as shown in Figure on page 130.

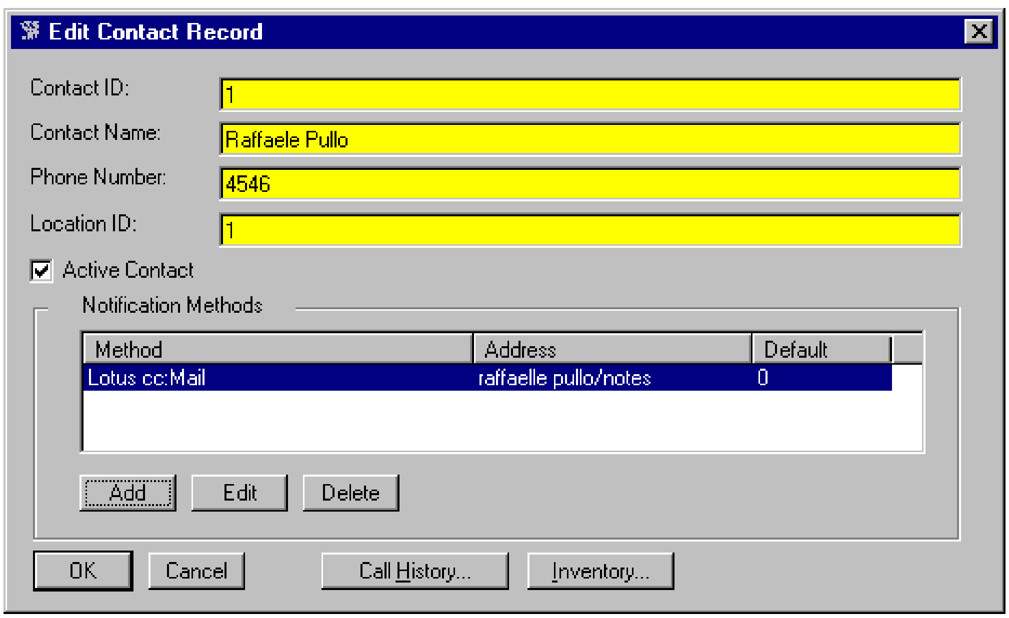


Figure 108. Contact Record

Now, when we receive a call we can enter the value 1 for both the location and caller fields from the entry screen shown in Figure 102 on page 127.

5.3 Workflow Definitions

In the Tivoli Advisor there are tools to configure the workflow when a call is registered. The workflow in Tivoli Advisor is totally dependent on the structure and rules of a business. For example, the Service Level Agreements (SLAs) will decide the escalation and notification policies.

This chapter describes the customization of the workflow process. By using examples we show how to create a typical workflow environment.

To modify the workflow options, from the Advisor startup window select **Options** as shown in Figure 109 on page 131.

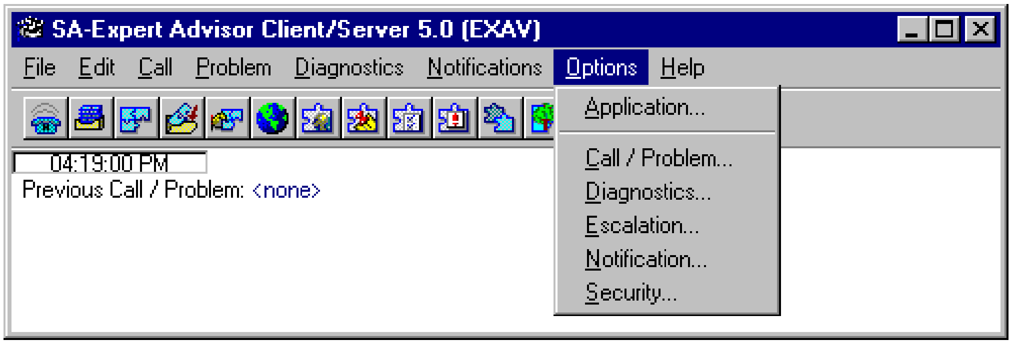


Figure 109. Tivoli Advisor Client/Server Window

All of the customization we do in this chapter is accessible through the above menu. First we look at the notification definition.

5.3.1 Notification

A notification contains status information about a call or a problem that is sent to an Advisor user or a caller. A notification is always created within Advisor, but can be delivered outside of the Advisor application. If a notification is configured to be delivered within Advisor, it is called an alarm. An example of a notification could be to inform an Advisor user via e-mail that an escalation threshold has been met. It could also be used to inform a caller that his or her problem has been resolved.

There are three configuration options for notifications:

- Methods
- Users
- Groups

The Notification Options window can be selected from the Options pull-down menu by selecting **Notifications** (see Figure 110 on page 132).

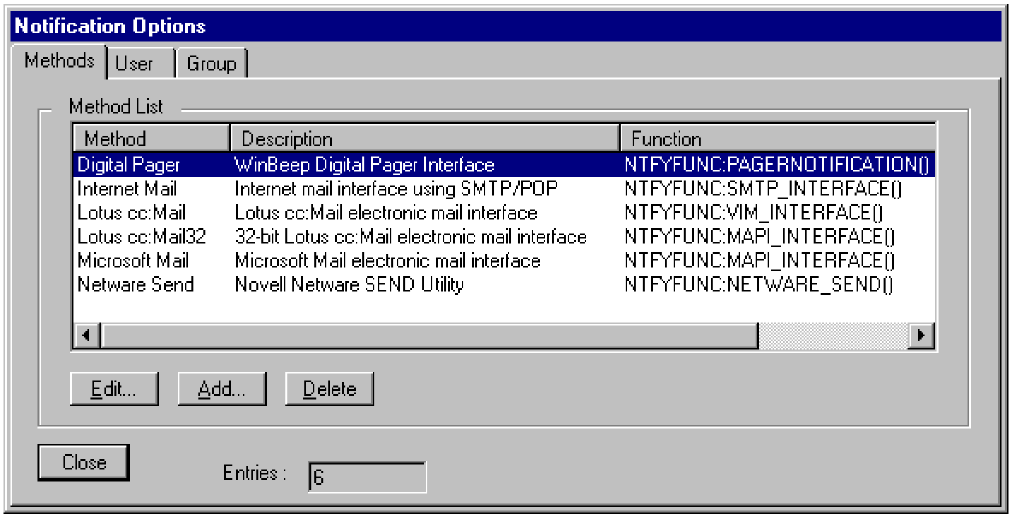


Figure 110. Notification Options Window, Methods Tab

To see the notification options for a user, click on the **User** tab (see Figure 111 on page 132).

Note

In the Methods tab above, you can see the notification methods that are shipped with Tivoli Advisor. The different methods are in fact functions within the $SAI_ROOT/ea/ntfyfunc.kb file. If you would like to add a new method to this list, you would have to create the function first in the ntfyfunc.kb file and then add the method from this window.

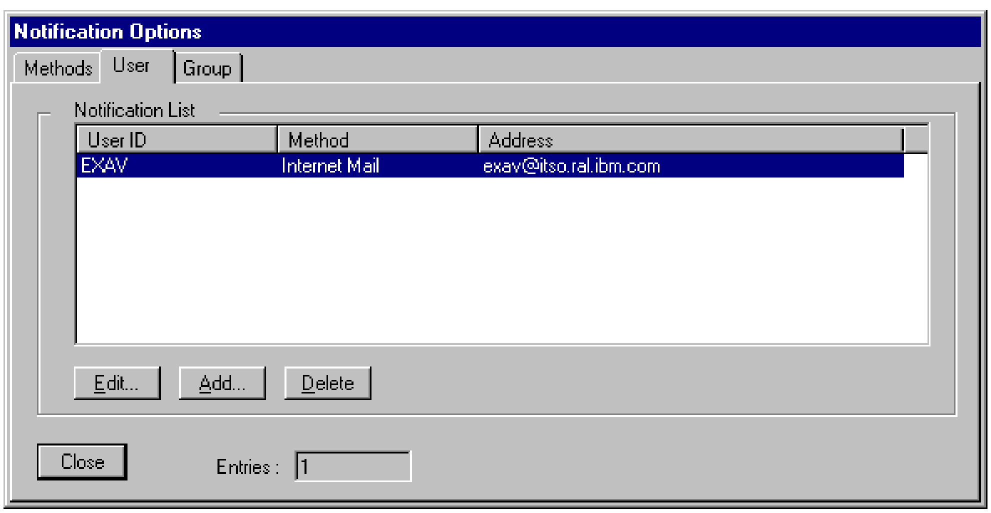


Figure 111. Notification Options

Here you can define the notification method and the e-mail address for the Advisor user. To add a user click on **Add**.

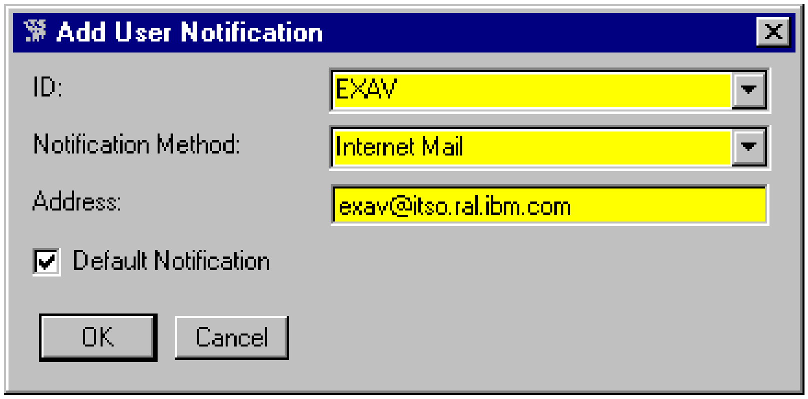


Figure 112. Add User Notification Window

We entered EXAV for the User ID, the Notification Method was set to the Internet address of the Advisor user and the e-mail address of exav@itso.ral.ibm.com.

To see the options for a group, click the **Group tab**.

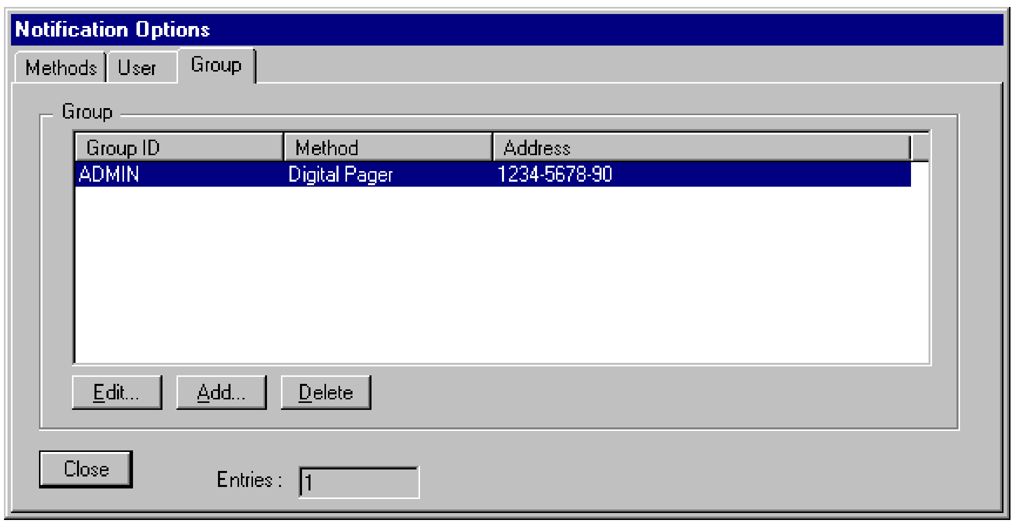


Figure 113. Notification Options Window for Group Tab

Figure 113 on page 133 shows the group notifications. This function is very powerful, but bear in mind the potential cost involved, for example, paging large groups of users at a time.

5.3.1.1 Notification Monitor

To keep track of the notification requests that users or the system commits there is a monitor called the Notification Monitor. This is an application that is installed at the same time as the Tivoli Advisor. The Notification Monitor's task is to scan the database for notification requests and send them to the specified notification method.

In order for any notifications to be delivered, at least one Notification Monitor per Advisor database has to be running. This means that it should be running on a machine that is never shut down and also should not be running any

CPU-intensive applications. The ASE file server could be a suitable location for the Notification Monitor.

5.3.2 Alarms

Alarms are notifications that are generated and delivered within Tivoli Advisor. They can be generated manually or by the system.

5.3.2.1 Sending Alarms

To send an alarm to a user select **Notifications** followed by **Send Notifications** from the Tivoli Advisor startup window (see Figure 114 on page 134).

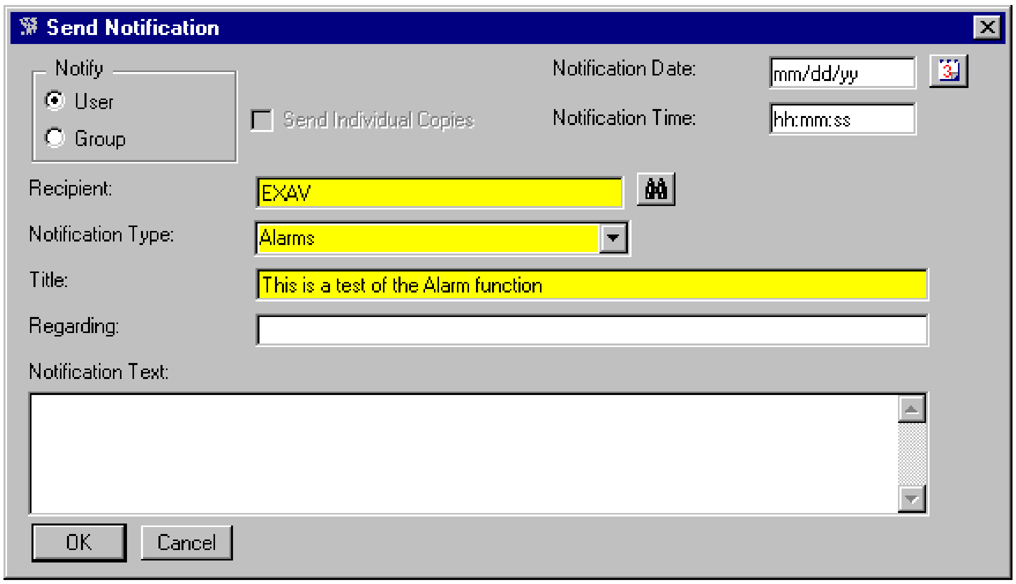


Figure 114. Send Notifications

When you have selected a recipient, the different notification types that are specified for that user or group will appear in the list. There will also be an option to send an alarm. If the user or group doesn't have any specified notification methods, only the alarm option will appear.

When the required fields are entered, click on **OK** to send the alarm.

There are two ways of looking at alarms. These are the following:

- Through the Tivoli Advisor
- Through the alarm monitor

5.3.2.2 Receiving Alarms through Tivoli Advisor

If you have the Tivoli Advisor started, it will automatically scan the database for alarms. You can specify how often it will scan the database from the Edit User window (see Figure 115 on page 135).

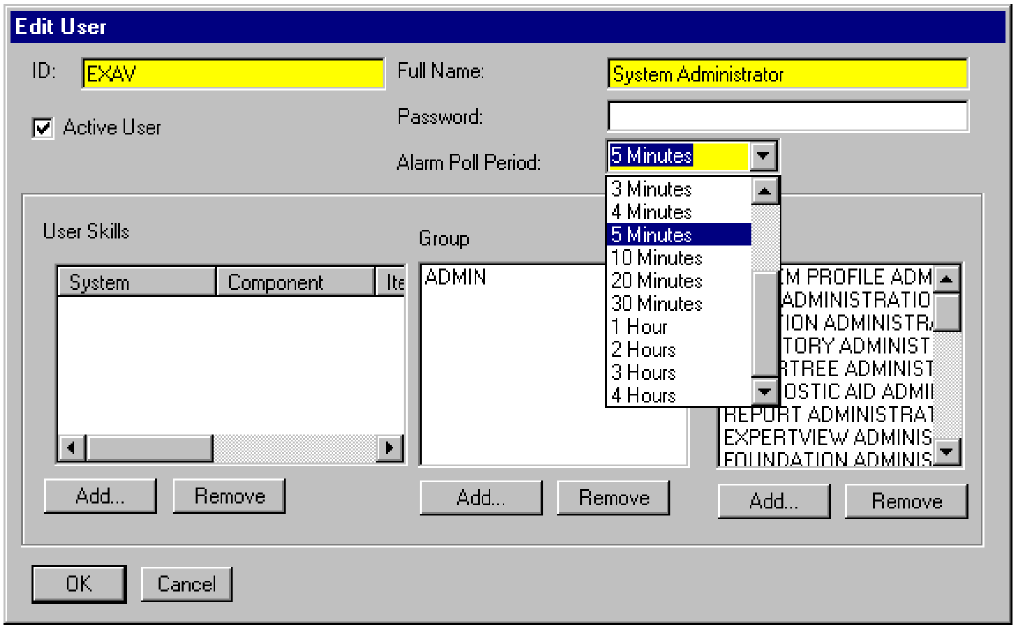


Figure 115. Edit User Window, Alarm Poll Period Menu

If you select Off, the Advisor will not check for alarms.

When Advisor has discovered that you have alarms, it will notify you by putting the message "You Have Alarms" in the upper right corner of your Advisor startup window.

Figure 116. Read Alarms Window

To read the alarms that you have, choose **Notifications** followed by **Read Alarms** (see Figure 116 on page 135).

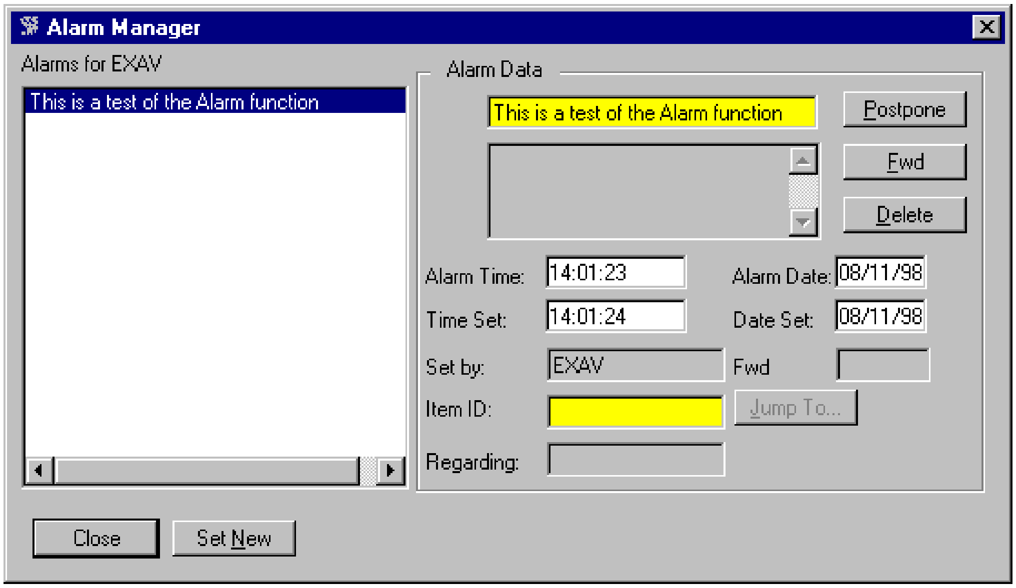


Figure 117. Alarm Manager Window

In the Alarm Manager, you can choose to postpone, forward or delete an alarm.

5.3.2.3 Receiving Alarms through Alarm Monitor

If you do not have Tivoli Advisor started, you can start the Alarm Monitor application directly.

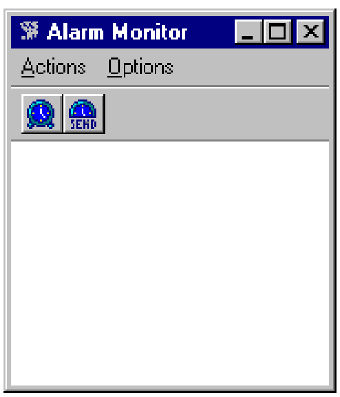


Figure 118. Alarm Monitor Window

The Alarm Monitor runs totally independent of the Advisor. Here you can choose to view or send alarms. If you choose the View Alarms icon, you will see the Alarm Manager window as shown in Figure 118 on page 136. Select the **Send Notifications** icon and you will see the Send Notification window (see Figure 114 on page 134).

When Alarm Monitor detects a new alarm, a window will pop up on the screen.

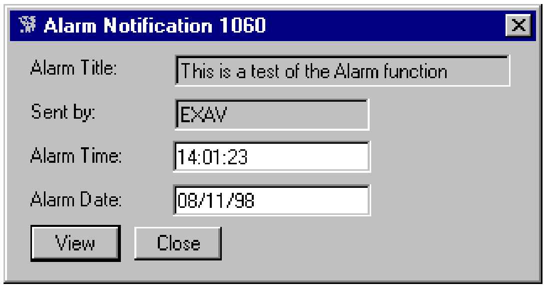


Figure 119. Alarm Notification Window

Here you can either choose to close or view the alarm.

5.3.3 Escalation

The escalation sets up the method used to raise the awareness of a call or a problem to a user or group. The escalation rules are defined by your business policies, for instance, SLAs.

There are different types of escalations, these are:

- Escalation by default
- Escalation by level
- Escalation by severity
- Escalation by condition

5.3.3.1 Escalation by Default

The default escalations are base settings for the other escalation types. For example, if an escalation rule of type level does not have a specified user or group to notify, it will use the default settings. The different options that the default settings have can be seen by choosing **Options -> Escalations** from the Advisor startup window.

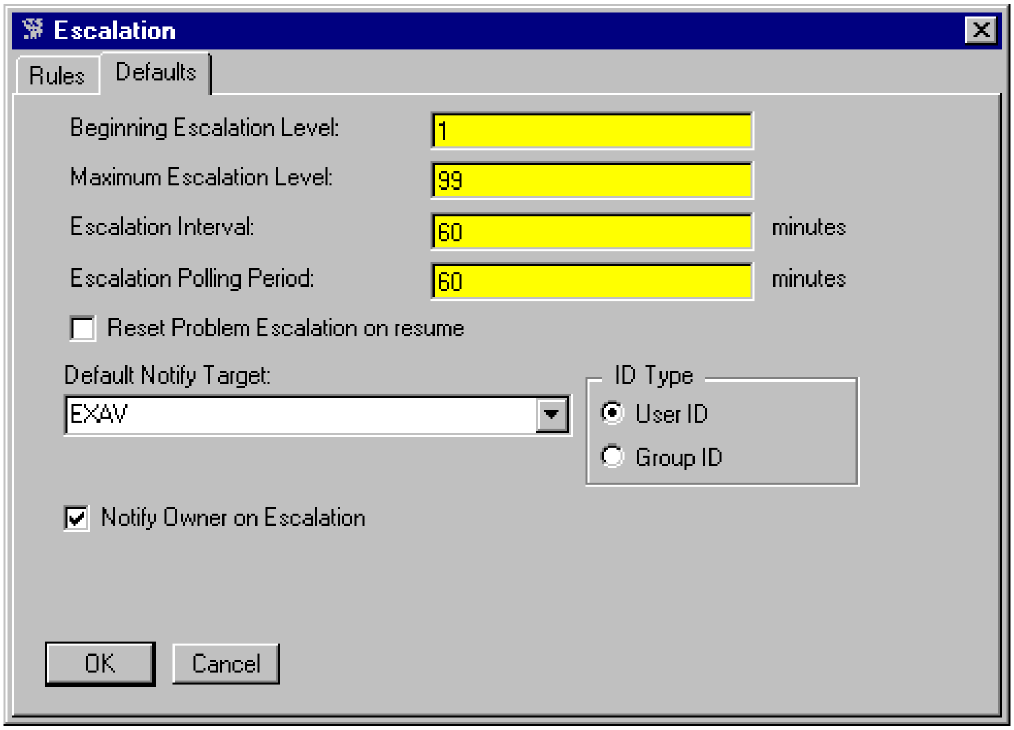


Figure 120. Escalation Window

Figure 120 on page 138 shows the default options.

5.3.3.2 Escalation by Level

Escalation by level is a simple way to escalate. You specify different levels and a pre-defined time limit for each level. The levels must be in order for the escalation rules to execute.

If they are not in order, the defined rules will not continue. For example, the levels have to be in the order 1, 2, 3, 4. If they are defined as 1, 2, 4, the rule will stop after the 2nd escalation level.

Table 12 on page 138 shows three example escalation rules that execute on escalation levels.

Table 12. Escalation Rules Based on Escalation Level

Escalation-level	Interval (Minutes)	Notify	Notify Owner?
1	60	EXAV	YES
2	90	SD-MAN	YES
3	180	CO-MAN	YES

If a problem is outstanding for say 60 minutes, the escalation rule 1 will activate and notify the user EXAV. At the same time it will notify the user that EXAV is now the new owner of this problem. The Notification Monitor will also increment the escalation level by one.

After another 90 minutes the escalation monitor will trigger the next escalation rule and notify the Service Desk Management SD-MAN. The Notification Monitor will also increase the escalation level by one.

If another 180 minutes passes, the company management CO-MAN will be informed about this problem. The notification monitor will in this case not increase the escalation level since there is no level 4 specified.

5.3.3.3 Escalation by Severity

The escalation by severity adds another dimension to the escalation rules. Now we can have two rules for each escalation level: one that triggers by severity and another that triggers by time only. Table 13 on page 139 shows an example set of escalation definitions for severity and escalation.

Table 13. Escalation Rules Based on Escalation Level and Severity Level

Escalation level	Severity level	Interval (Minutes)	Notify	Notify Owner?
1	1	30	EXAV	YES
1		60	EXAV	YES
2	1	90	CO-MAN	YES
2		180	CO-MAN	YES

Here we can see that an escalation rule will execute if a problem with severity 1 is opened for more than 30 minutes. A problem with any other severity will not be escalated until it has been opened for 60 minutes.

It is very important to set the most generic rule at the end of all the rules within that escalation level. This is because when a rule is executed on a problem, it will ignore the rest of the rules for that escalation. In this case the rule with no severity level specified is the most generic.

5.3.3.4 Escalation by Condition

A problem can be escalated by the value in the different fields of the problem record. This can be useful is you want to escalate a problem that has been created from a business critical system.

The condition is formed by the name of a database field in the ADVISOR database and the conditioning data. The different conditions can be connected using AND or OR statements so that you can define rules that act on more than one field. An example of this is shown in Table 14 on page 139.

Table 14. Escalation Rules Based on Escalation Level, Severity Level and Condition

Escalation-level	Severity-level	Condition	Interval (Minutes)	Notify	Notify Owner?
1		LOCATION='V.I.P' AND PROBLEM_TYPE= 'HARDWARE'	60	SD-MAN	YES
1		LOCATION='V.I.P'	30	SD-MAN	YES
1	2		90	EXAV	YES

Escalation-level	Severity-level	Condition	Interval (Minutes)	Notify	Notify Owner?
1			120	EXAV	YES

Here we see that a HARDWARE problem that was generated from a caller in location V.I.P is not escalated as fast as any other problem coming in from that location. But the calls coming in from the V.I.P location are still escalated faster than severity 2 calls. A call coming in from location V.I.P with severity 2 would be escalated in 30 minutes, since the rule with the LOCATION='V.I.P' condition is before the severity level rule in the list of rules.

5.3.3.5 Creating Escalation Rules

The escalation rules can be edited by selecting **Options -> Escalations** from the Advisor startup window (see Figure 121 on page 140).

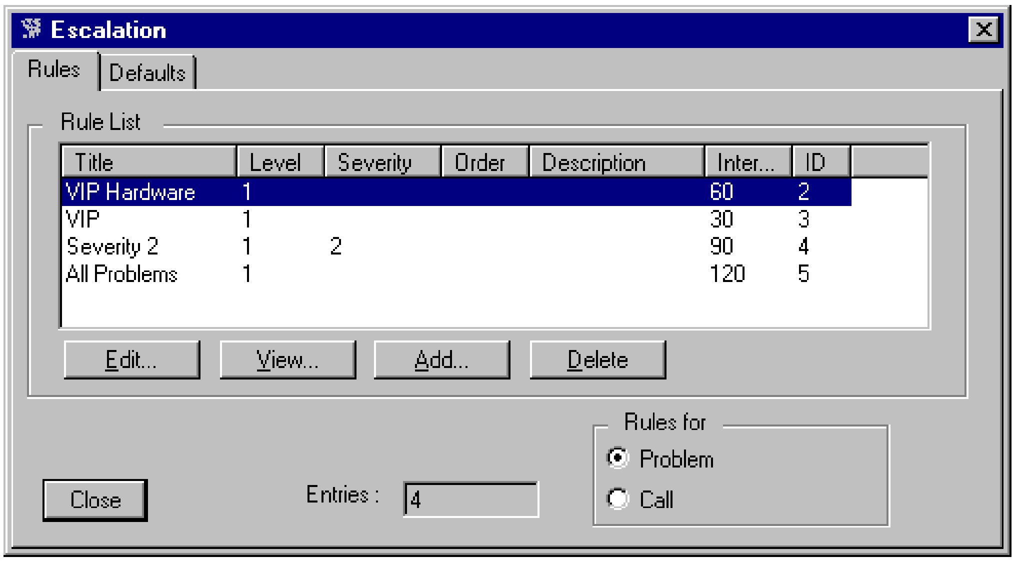


Figure 121. Escalation Rule

To add a new rule, click on **Add** (see Figure 122 on page 141).

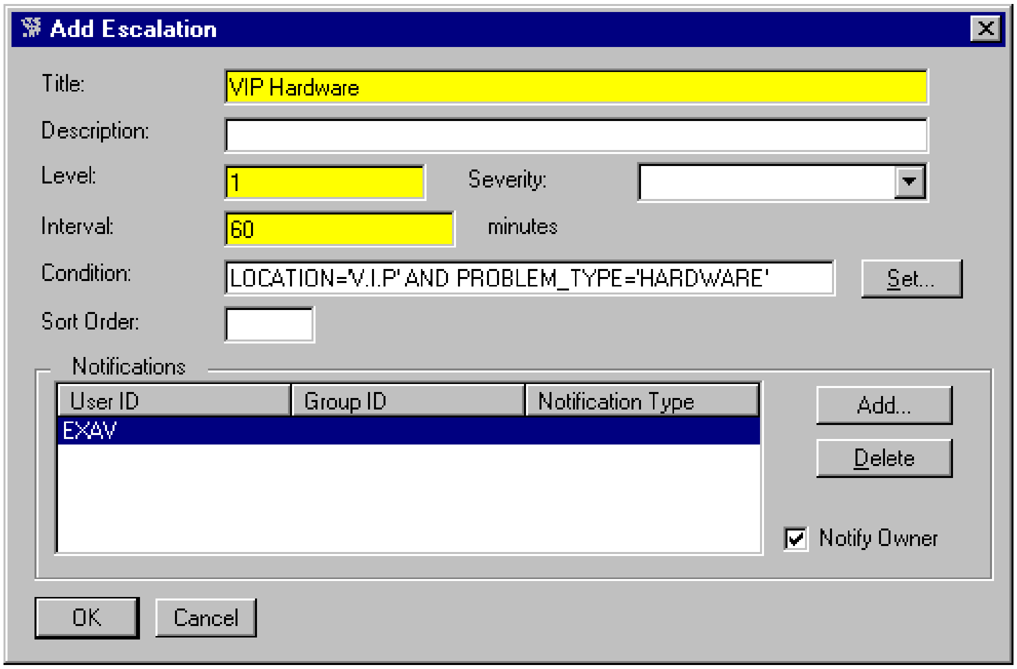


Figure 122. Add Escalation

We entered the escalation rule with the condition being set to:
`LOCLOCATION='V.I.P' AND PROBLEM_TYPE='HARDWARE'`

Click on **OK** to add the new escalation rule.

5.3.4 Escalation Monitor

The Escalation Monitor is an application that scans the database for escalations according to the escalation rules that exist. When the Escalation Monitor finds a call or a problem that has been open too long, it will do the following:

- Use the escalation rules to send a notification to the user or group defined in the rules. If no escalation rules exists, it will use the default settings.
- Raise the escalation level of the call or problem.
- Calculate when the next escalation will occur according to the escalation rules.

You can run several copies of Escalation Monitor, for example if you have many escalations or if you want to be sure that the escalations really occur.

Like the Notification Monitor, the Escalation Monitor should be run on a machine that is only running monitoring processes, for example, an ASE file server. If no Escalation Monitor is running, there will be no escalations.

5.3.5 Problem and Call Options

You can alter the way the Advisor should behave to better fit your organizational needs. For example, if your service desk has agreed with users to accept call registration via an answering machine, you can create a call code, called answering machine, to get a better registration for those calls.

There are many parameters that can be set that will drive the management of a call. The ones we are going to cover in this section are:

- Call Management
- Inquiry
- Call Code
- Problem Code
- Problem Type
- Severity
- Call Defaults
- Auto Hot News

To better understand what parameters can affect the analyst when they register a call, see the Call Registration window shown in Figure 123 on page 142.

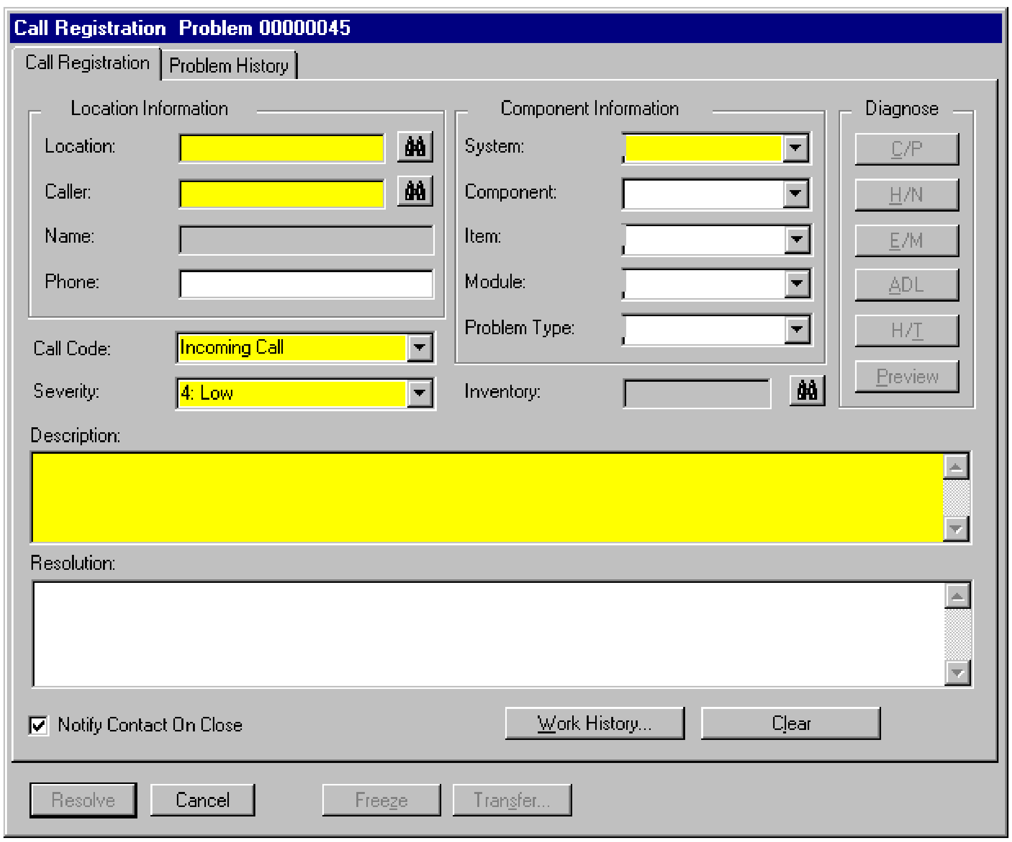


Figure 123. Call Registration

The configuration screen to start to configure the call options can be started by selecting **Options** followed by **Call/Problem** from the Advisor startup window.

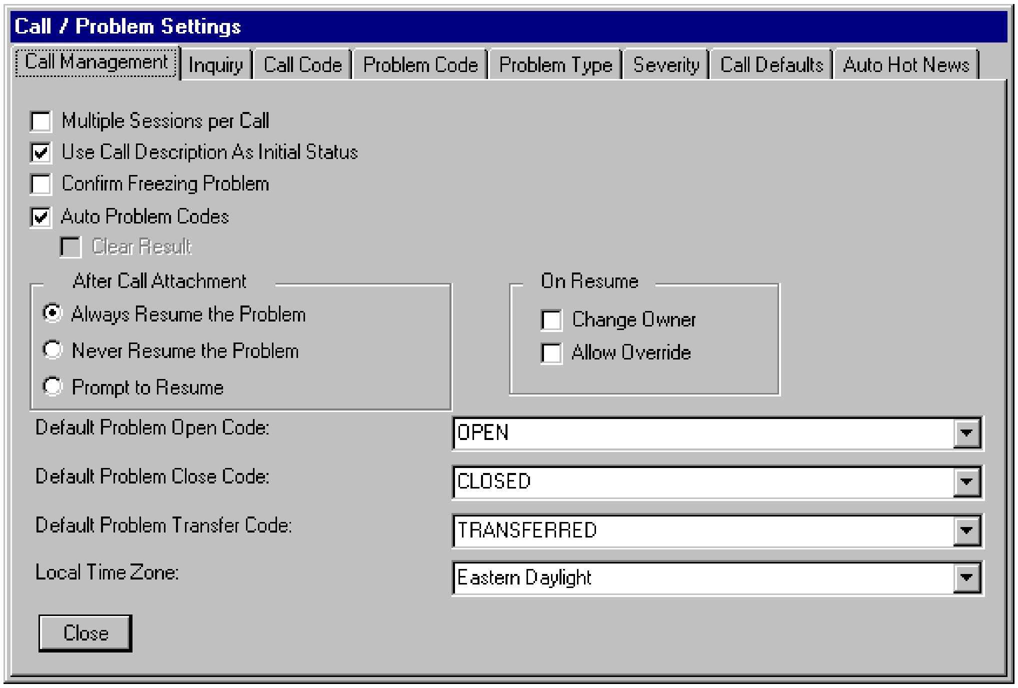


Figure 124. Call/Problem Settings Window, Call Management Tab

5.3.5.1 Call Management

By selecting the **Call Management** tab we can set the way a call should behave when it is altered. For example, we can set the default problem open code to inform the system that when a new call is opened, it will be defined with the status of OPEN.

Another parameter we can set is if you want problems to which a call is subsequently attached to a prompt to appear with the option to resume the call regardless of whether the original problem has been closed or frozen.

5.3.5.2 Inquiry

By selecting the **Inquiry** tab you can specify if there should be an inquiry window before you do certain actions. The inquiry window allows you to narrow down the number of results in the window that you are about to open. This is mostly used in when working with windows, for example, the Work with Common Problems window.

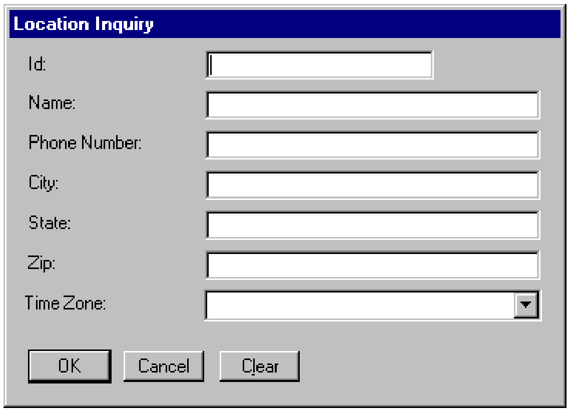


Figure 125. Inquiry Screen

The modifications made will change the behavior of the application.

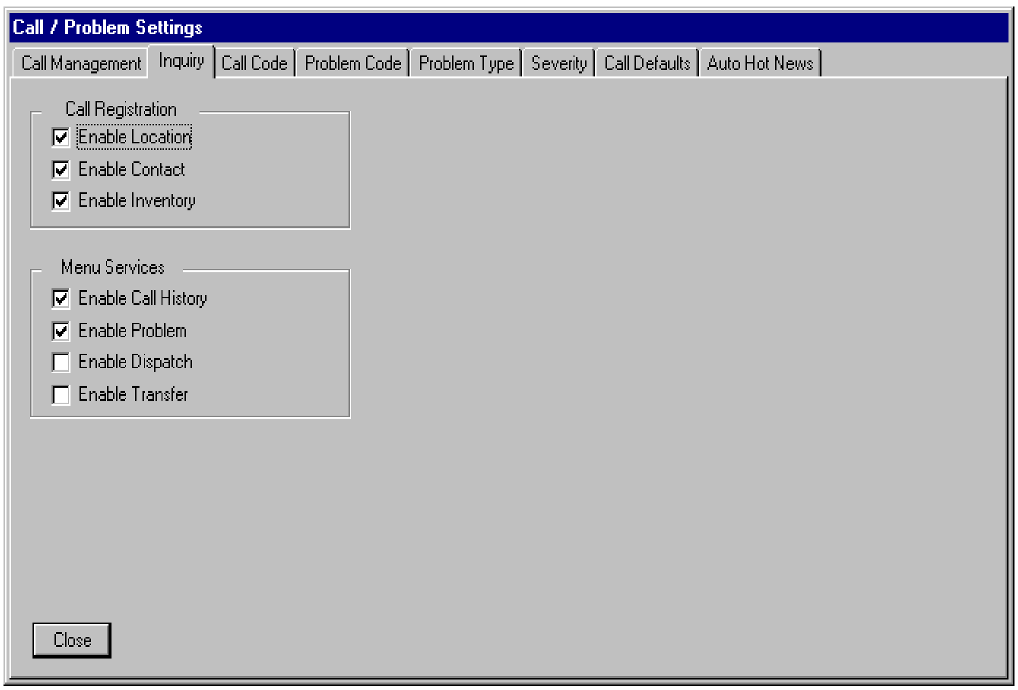


Figure 126. Call/Problem Settings

The Inquiry window could be useful, for example, if you have 2000 contacts specified in your environment and you do not want to get a list of all of them every time you are going to search for a contact. If you mark the Enable Contact box, you will get an Inquiry window everytime you access the contacts in your environment. We selected the options for location, contact and inventory.

5.3.5.3 Call Codes

Call codes are used to specify where a call comes from.

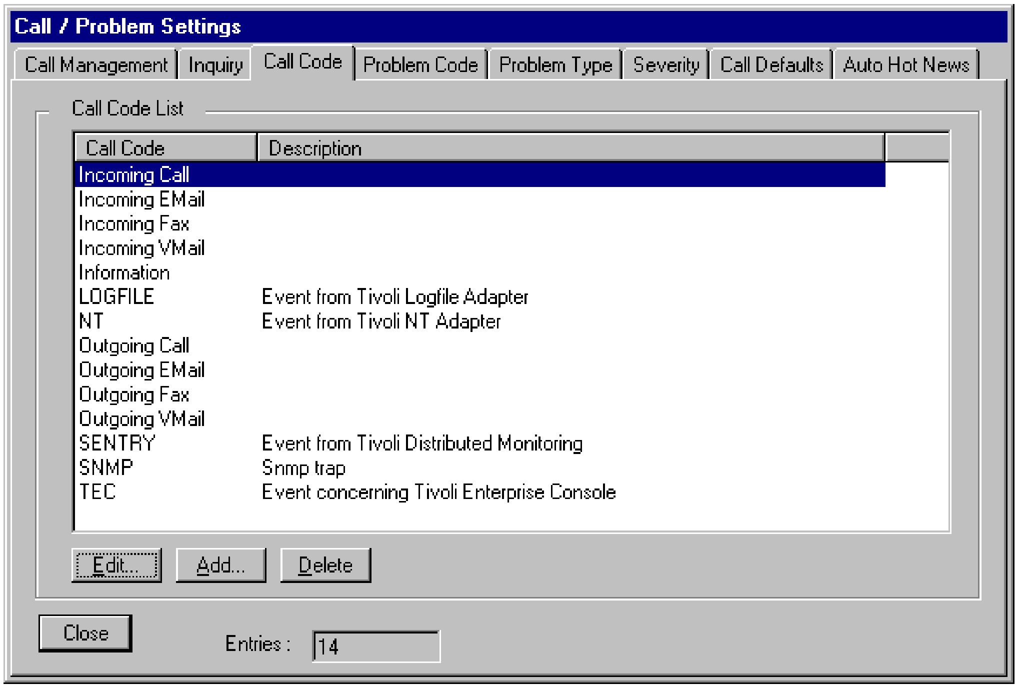


Figure 127. Call/Problem Settings

A call code could be, for example, an incoming telephone call or fax.

We added additional call codes for our integration with the Tivoli Management Environment. The codes we added where NT, SENTRY, SNMP and TEC.

5.3.5.4 Problem Codes

Problem codes are used to shows the current status of a problem.

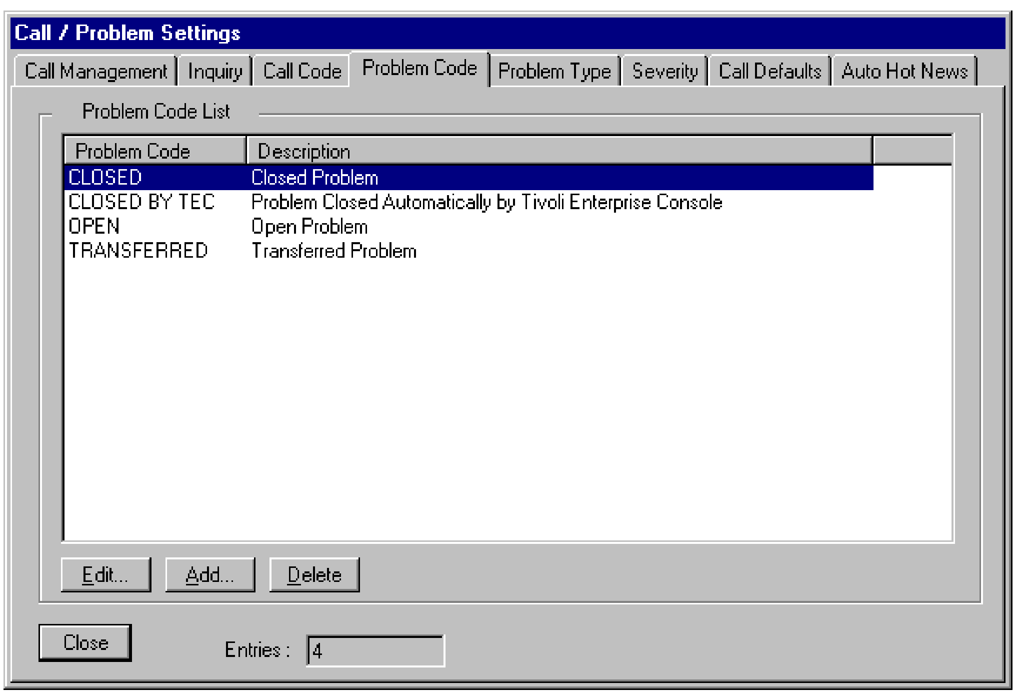


Figure 128. Call/Problem Settings - Problem Code Tab

Figure 128 on page 145 shows the definitions for our environment. We used the default setting with the addition of the problem code CLOSED BY TEC that informs us when a problem is automatically closed by the Tivoli Enterprise Console.

5.3.5.5 Problem Types

The Problem Type is used to describe the type of problem.

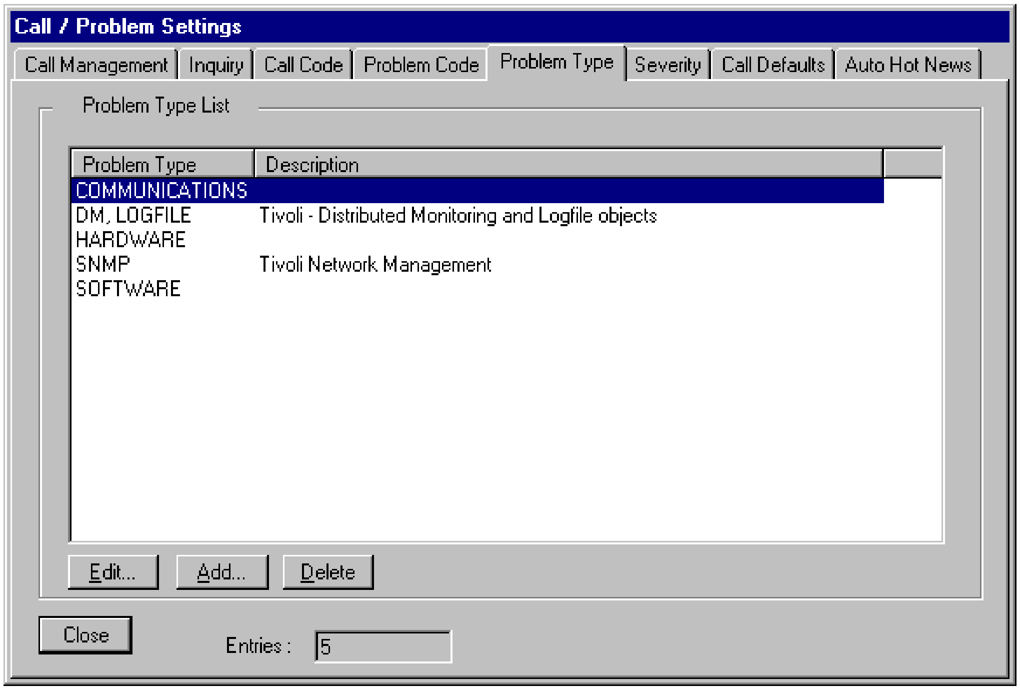


Figure 129. Call/Problem Settings - Problem Type

The default problem types are COMMUNICATIONS, HARDWARE and SOFTWARE. Here we can define additional problem types. We added the problem type of SNMP and DM LOGFILE.

5.3.5.6 Severity

Severity is used to assign the importance a call or problem has.

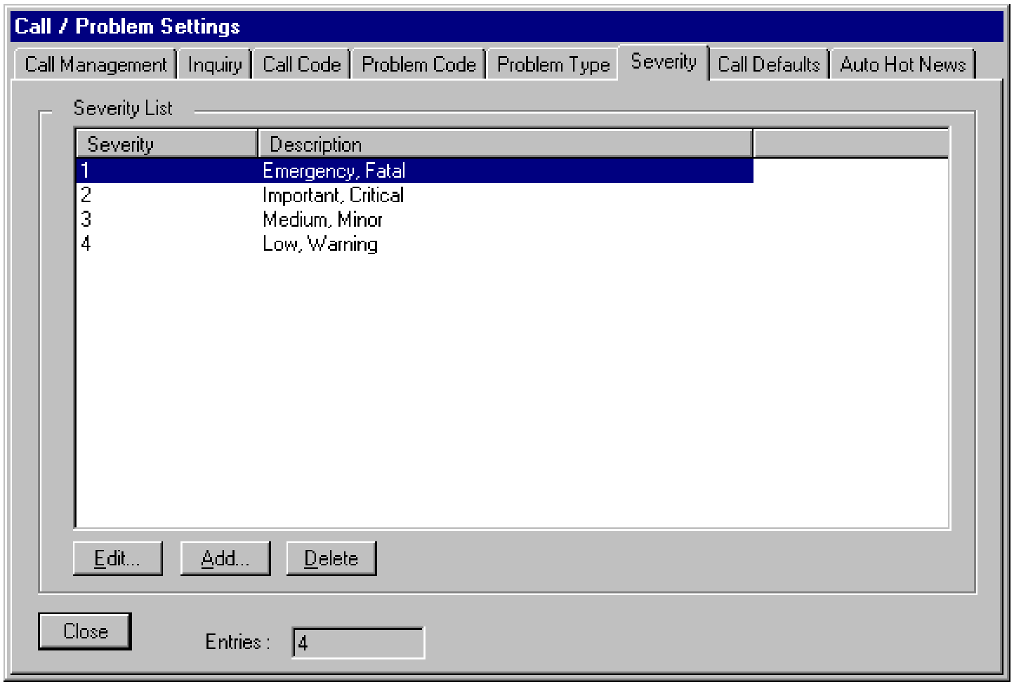


Figure 130. Call/Problem Settings Window, Severity Tab

Figure 130 on page 147 shows the default severities. We did not need to add any new severity values for our environment.

5.3.5.7 Call Defaults

The call defaults are used when an Advisor user registers a new call.

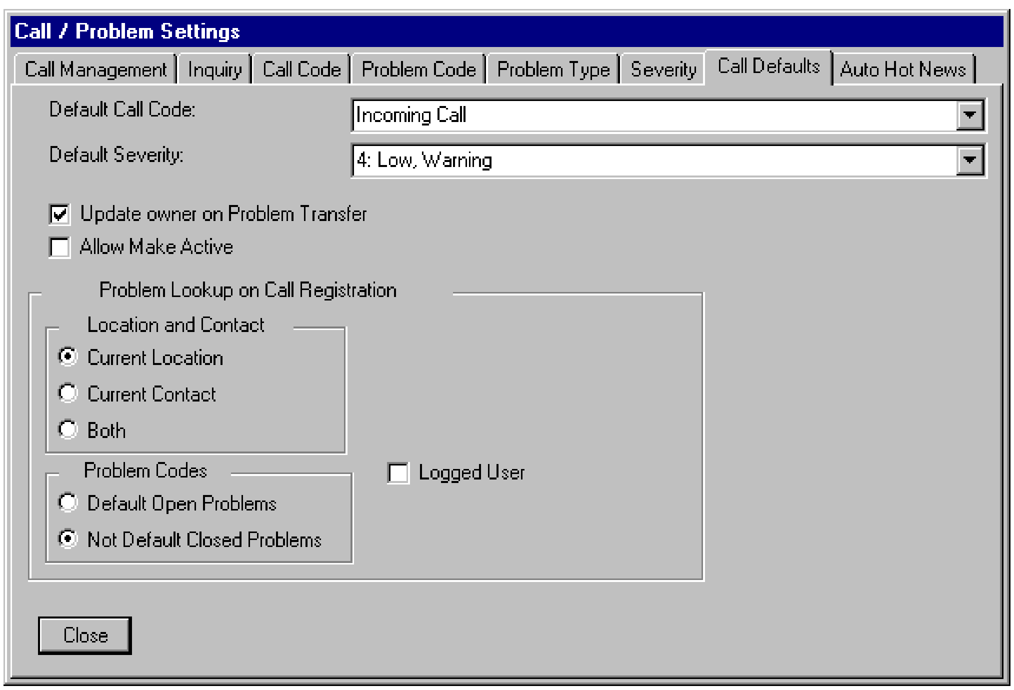


Figure 131. Call/Problem Settings - Call Defaults

Here you can set the default call code and the severity.

5.3.5.8 Auto Hot News

This option tab enables you to automatically display any new hot news records.

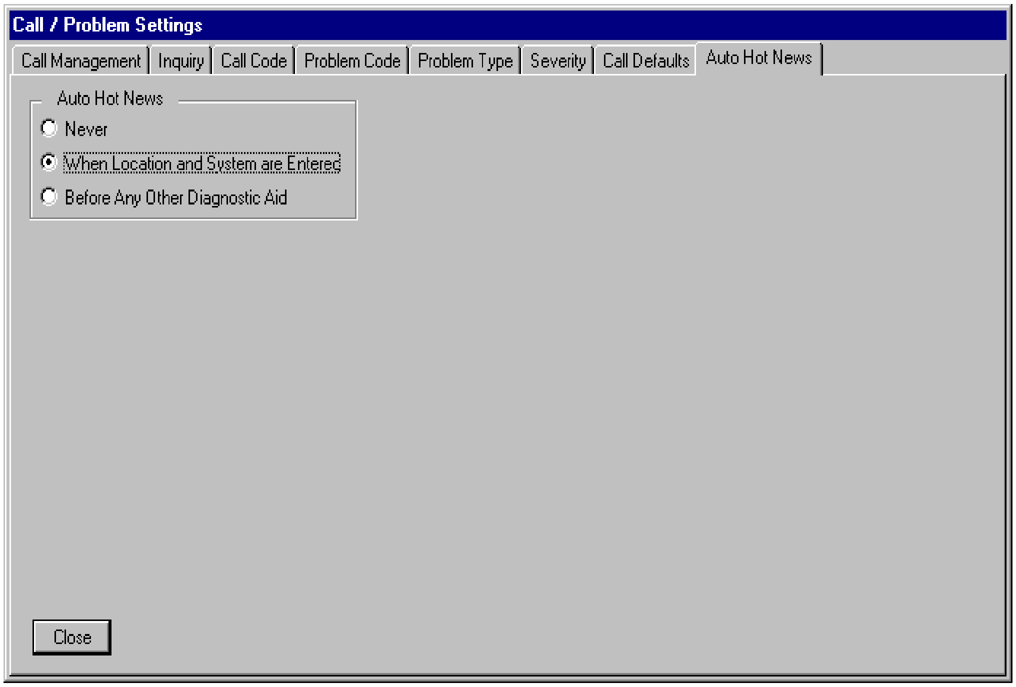


Figure 132. Call / Problem Settings Window, Auto Hot News Tab

This enables you to force the Advisor user to see a hot news item when a call is entered. Figure 132 on page 148 shows that we selected the option to display hot news records when the Location and System fields are entered when a new call is being registered.

5.4 System Component Item Module (SCIM)

The SCIM structure is the way to define your environment to assist the Tivoli Advisor users. The SCIM can be an inventory-based view of the environment or it can also be a problem-based view of your environment.

The SCIM is used by the analyst when working with a problem. The diagnostic tools also use SCIM to better evaluate and identify a problem. The SCIM definition is depicted in the Figure 133 on page 149.

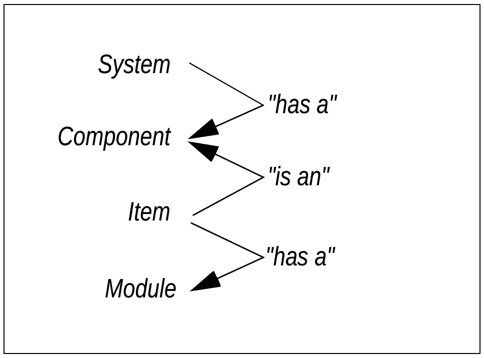


Figure 133. SCIM Definition

The definition of a system is that it has a component. Likewise, the definition of an item is that it is a component. The item also has a module.

An example of this could be:

- Hardware (system) has a monitor (component).
- IBM P200 (item) is a monitor (component).
- IBM P200 (item) has a blank screen (module).

As you can see, this works out perfectly when you approach SCIM with the inventory-based view. But, if you try to do this with a problem-based view of the SCIM you will end up in trouble. It is very hard to get a good structure for the item and module levels of the SCIM with a problem-based view. Therefore, we recommend you just define the system and component levels when you are defining the problem-based view.

An example of a problem-based view could be, a mail application (System) has a communications error (Component).

To SCIM definitions are listed below:

- System

  The system is a collection of different components. Each system in the Advisor environment has to have a unique identifier.

- Component

  Component is a general collection of items. The component is not something that has a serial number or an inventory number. It is a general description of a group of items that have serial numbers or inventory numbers. A server is a typical component. Each component in your environment must have a unique identifier.

- Item

  An item is something that you can assign a number to, for example a serial number or a license key. An item is easy to define since it is something that you can see or touch. That is why the item level of SCIM is a good entry point when you are starting to build your SCIM structure.

- Module

  A module can be a part of an item that can experience a problem; it can also be the problem itself.

5.4.1 Defining a SCIM Structure

When you are defining the SCIM structure of your environment it is easier to start by putting down your structure on paper before entering it into the SCIM database. When you start thinking of your structure it is best to start with the items, then do the components, followed by the systems and finally defining the modules.

5.4.1.1 Defining the Items

An item is anything that has a vendor, a serial number or model number, for example, AIX 4.3, Framework 3.6, Proliant 5000 and Pentium Pro.

5.4.1.2 Defining the Components

Next we can define the components. Components are a category of items, something you cannot buy. The components are divided into problem types, such as software, hardware or communication. For example:

Table 15. Items and Their Associated Problem Types

Component	Problem type
Operating system	Software
Systems management	Software
Server	Hardware
Processor	Hardware
Problem management	Software
Network	Communication

5.4.1.3 Defining the Systems

The next step is to define the system levels. Systems are groups of components that are related to each other, for example, hardware or applications.

5.4.1.4 Defining the Modules

When you have the system, component and item specified, it is time to do the module level. The module could be a part of the item or a state of the item, for example, NT Service Pack 3 or a distorted screen.

Figure 133 on page 149 shows an example of how part of a SCIM structure could look:

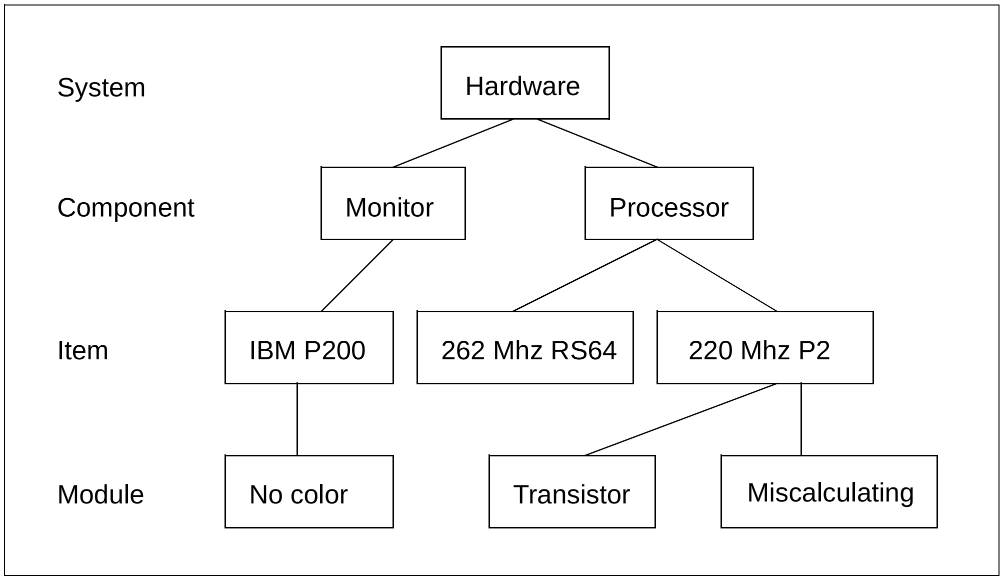


Figure 134. SCIM Example

This example structure will be implemented in the next section.

5.4.2 Implementing a SCIM Structure

When you are about to enter your SCIM structure into the Advisor application, you must start with the system level. To do this click on **Edit** followed by **Equipment Definitions**.

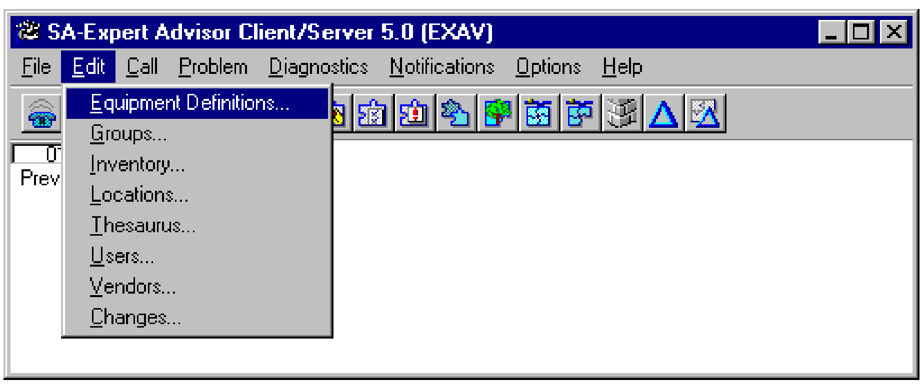


Figure 135. Edit - Equipment Definitions...

Figure 136 on page 152 will appear. From here you can edit, view, add or delete a system. To add a system, you just have to fill in the unique identifier called the System ID and a description.

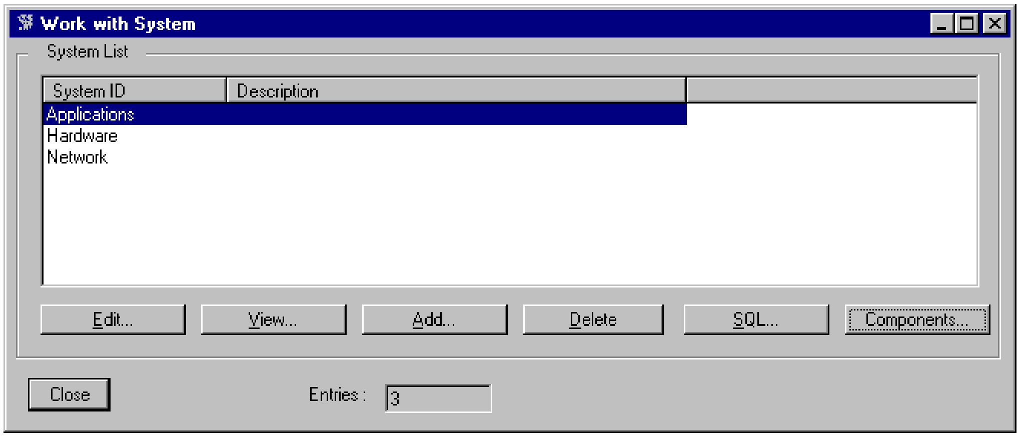


Figure 136. Work with System

To see the components associated to a system, select the system you prefer and click on **Components**.

In Figure 137 on page 152 we selected the **HARDWARE** system and clicked the **Components** button.

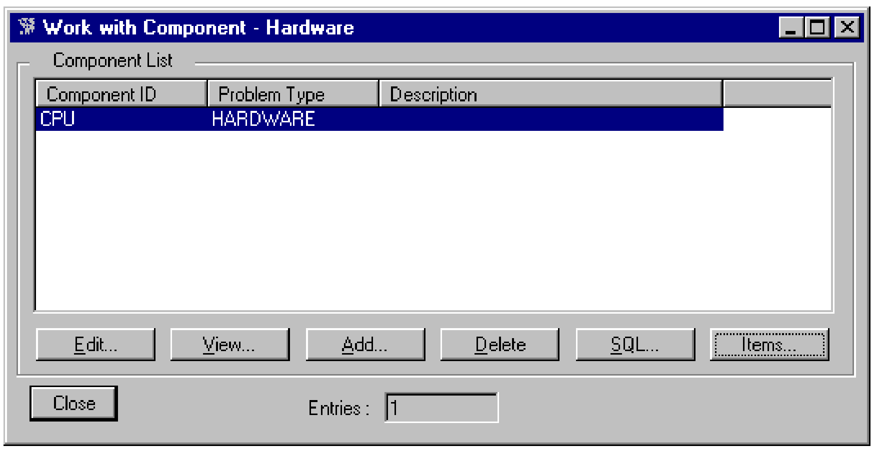


Figure 137. Work with Component - Hardware Window

To add our monitor component, we click on **Add**.

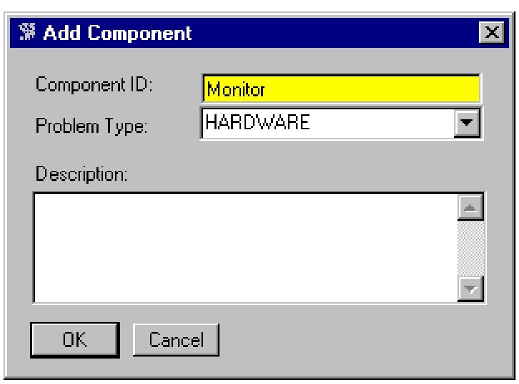


Figure 138. Add Component

Our components are now defined and we move on to the items. To see the items associated to a component, select the component and click on **Items**.

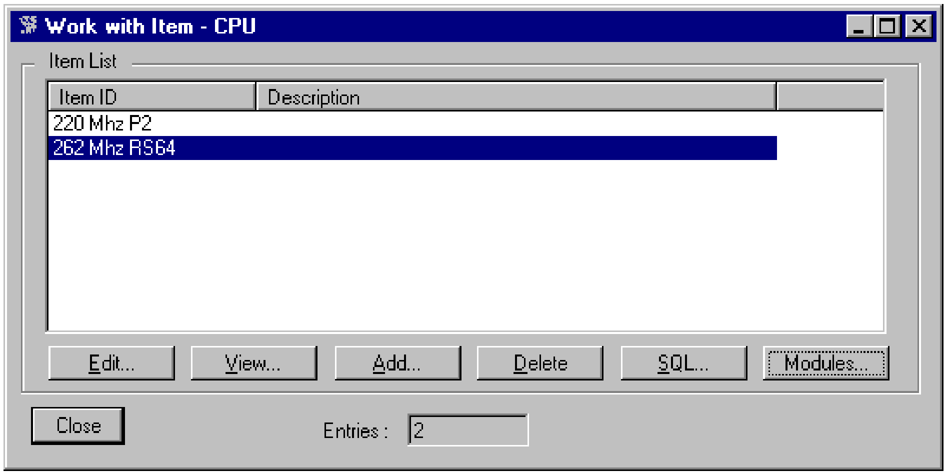


Figure 139. Work with Item - CPU Window

Here we selected the component CPU, then clicked on **Item**. To add an item click on **Add**.

.

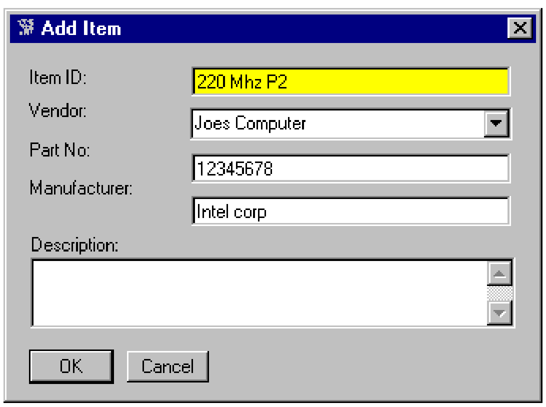


Figure 140. Add Item

When the item is defined you can drill down to the module level by clicking on **Modules** from the Work with Item window.

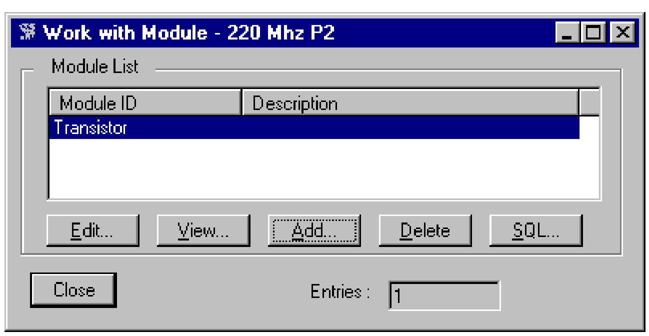


Figure 141. Work with Module - 220 Mhz P2 Window

The procedure for adding, viewing, deleting and editing a module is the same as for the system, component and item.

The SCIM structure is now ready to be used with the Advisor application.

5.5 Diagnostic Aids

The diagnostic aids described in this section are very powerful and important tools to exploit the capabilities of the Tivoli Advisor. If they are set up, they can provide the knowledge of a whole service organization to the service desk analyst. These tools are quite complex to understand and set up.

The different aids that are included in Tivoli Advisor are:

- Common Problems (C/P)
- Error Messages (E/M)
- Hot News (H/N)
- Hyper Trees (H/T)
- Adaptive Learning (ADL)
- Solutions

All of the aids, except Solutions, can be found in the Call Registration window in Advisor. At the beginning, when no data is entered into the call, the buttons that represent the aids will be grayed out. When the analyst starts to fill in data about the call, he or she can make a request to the diagnostic tools to see if they can provide a solution. The request is made by clicking the **Preview** button. If a possible solution is found, the button of the corresponding aid will show how many solutions it found.

5.5.1 Common Problems (C/P)

The Common Problem diagnostic aid is a tool for specifying and searching through the most frequent problems in your environment. It enables the analyst that is receiving a call or is working on a problem to find a common problem by entering the following information:

- Location
- Component
- Problem Type

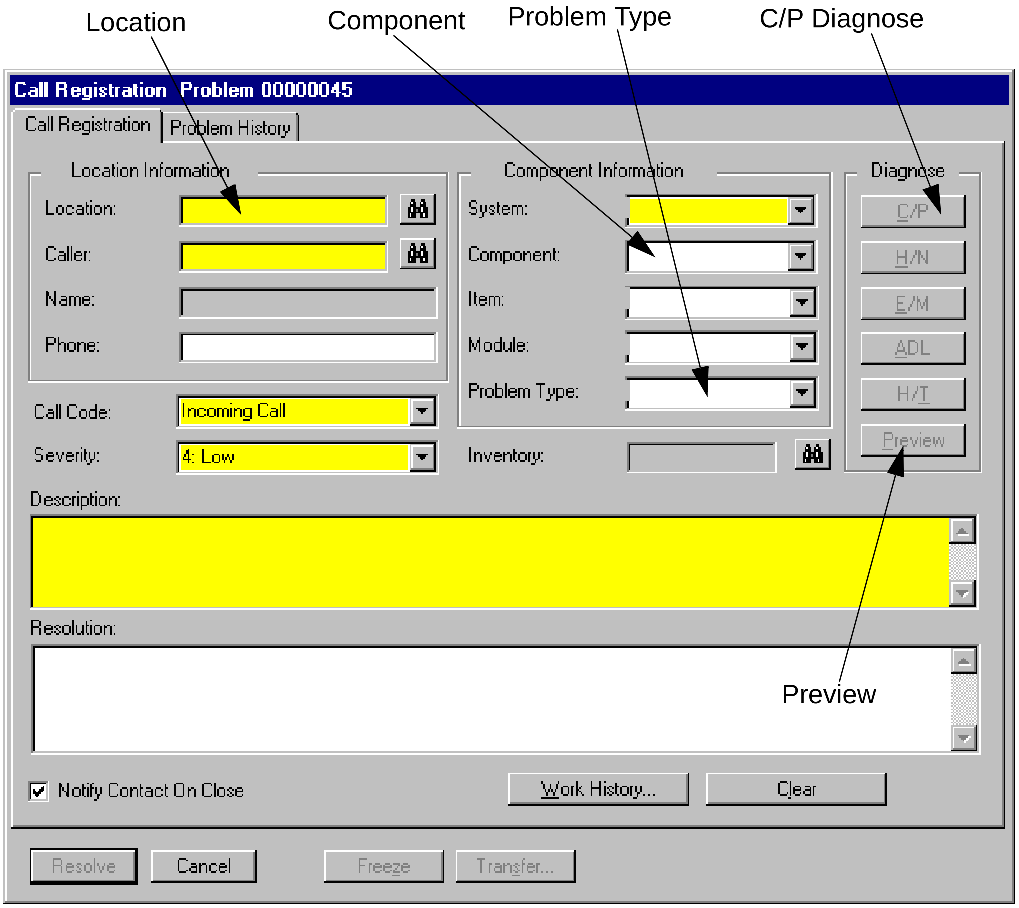


Figure 142. Call Registration - Common Problems

When these fields are entered and the Preview button is selected, the common problems that are specified for this criteria will be accessible through the C/P Diagnose button. You can also tell the system to go directly into Common Problems by clicking the C/P Diagnose button.

When the analyst finds a matching common problem, he or she can associate the call with that common problem therefore providing an immediate solution to the call.

The number of common problems in a normal size environment is usually around 100.

5.5.1.1 Common Problems - Record Hierarchy

The common problems are organized by group records and individual records.

The group records are like folders; they can contain individual records or group records. In this way you can form a tree structure of common problem records which makes the records more easy to find. A group record cannot be activated as a solution to a problem by an analyst.

The individual records are Common Problem records that can be activated as solutions to problems by a analyst.

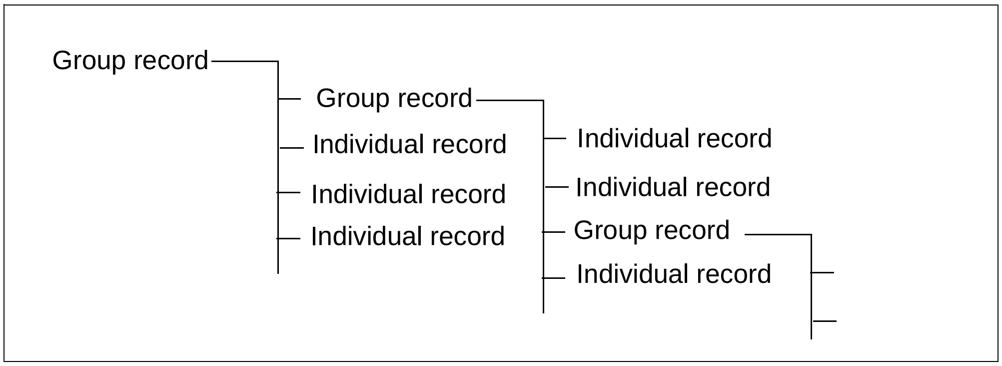


Figure 143. Common Problems, Record Hierarchy

5.5.1.2 Creating Common Problems Records

To display the Work with Common Problem window, click on **Diagnostics -> Work with Common Problem** from the Advisor window (see Figure 144 on page 156).

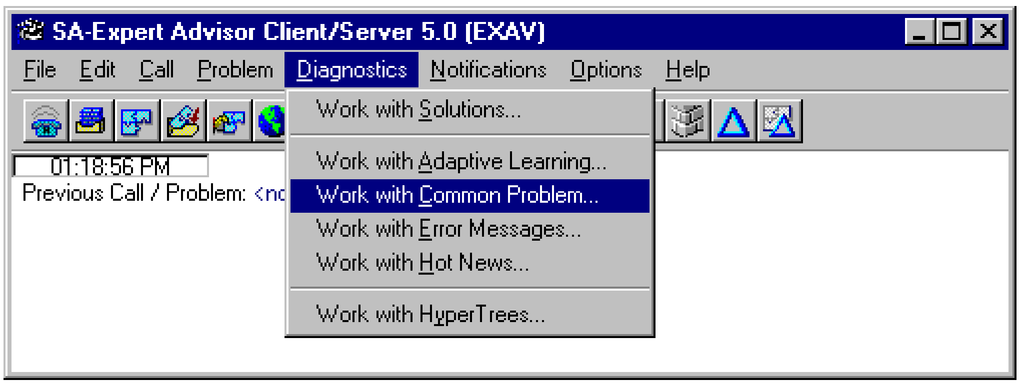


Figure 144. Work with Common Problems

Once we have selected the menu as shown in the above figure, the window in Figure 145 on page 156 will appear.

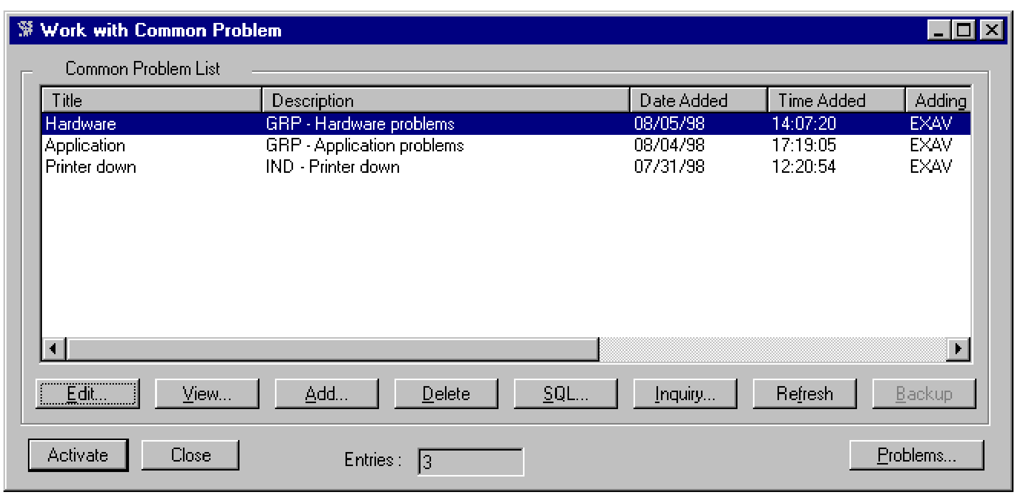


Figure 145. Work with Common Problem

Note

We added an index, GRP or IND, in the Description field to differentiate groups and individual records. There is no way you can see if the records are group or individual records in this window if you do not specify it in the Title or Description field.

To add a new common problem record, click on **Add** (see Figure 146 on page 157).

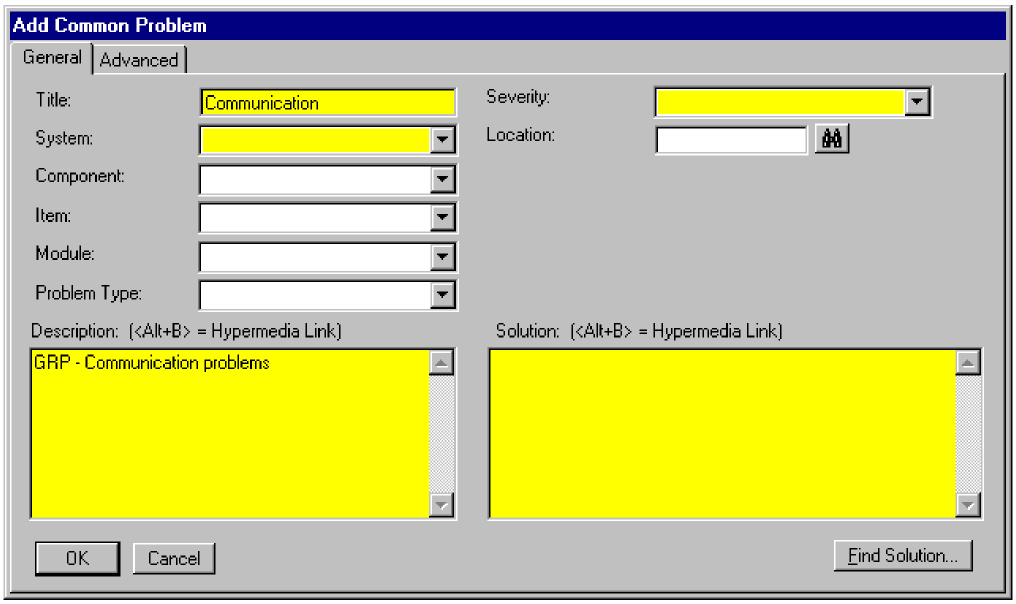


Figure 146. Adding a Common Problem

Click on **OK** when complete.

5.5.1.3 Creating a Group Record

In this case we wanted to add a group record for all the communication problems. We set the index in the Description field to GRP, but to specify to the system that this is a group record, we have to select the **Advanced** tab (see Figure 147 on page 158).

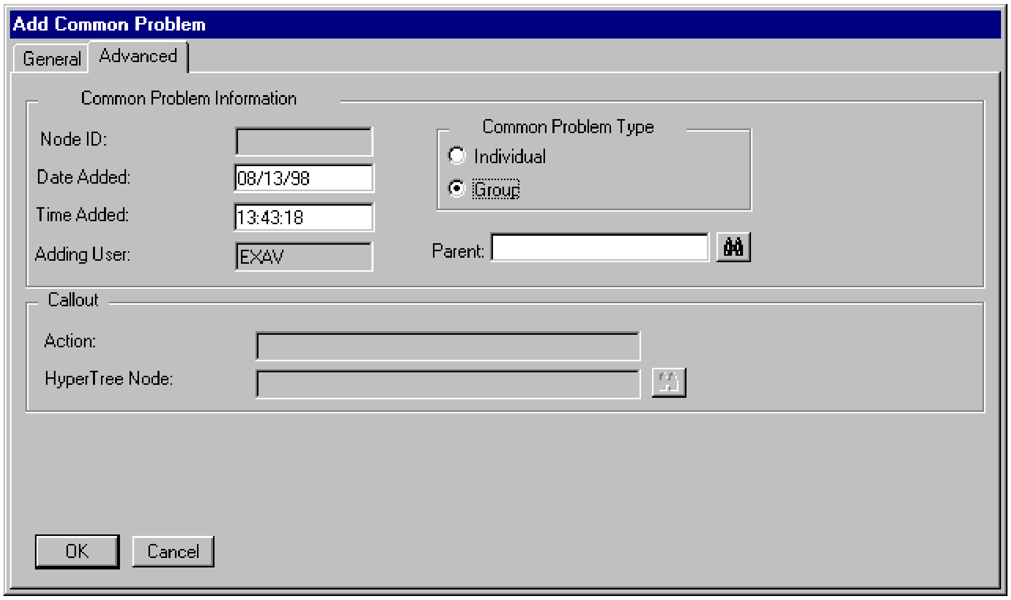


Figure 147. Common Problem, Advanced Tab

In the Common Problem Type field, we chose **Group**. This resulted in some of the required fields in the General tab to be set as optional.

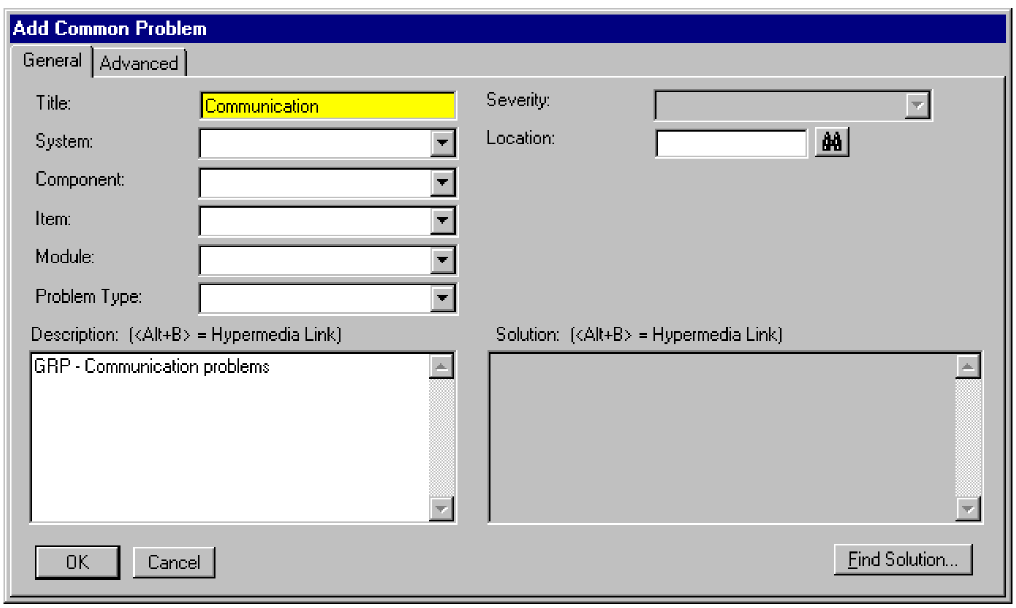


Figure 148. Common Problem Window with Common Problem Type Set to Group

We clicked on **OK** to create the new Common Problem record.

5.5.1.4 Navigating in the Hierarchy

To see which individual records are associated with a group record, you can either double-click on the group record or select it and click on **Activate**.

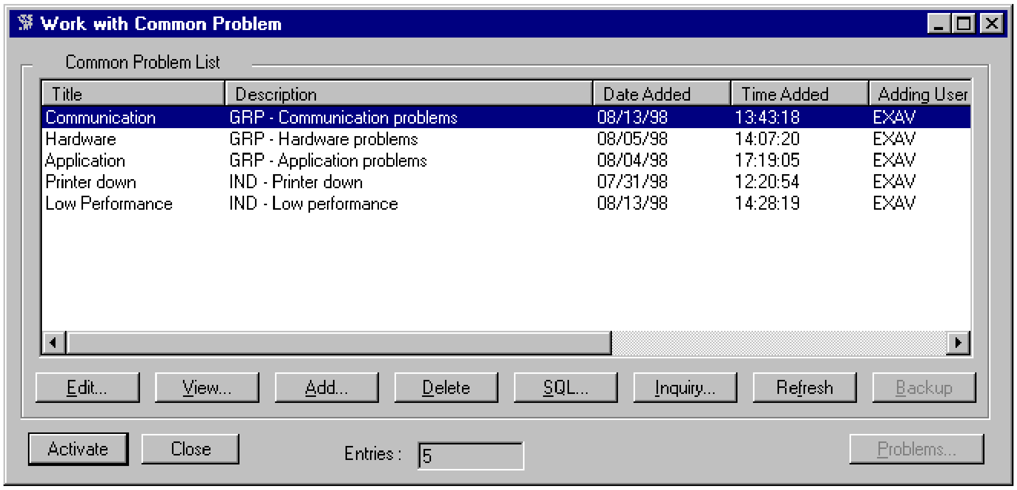


Figure 149. Work with Common Problems

The window now refreshes and shows only the records associated to that group record.

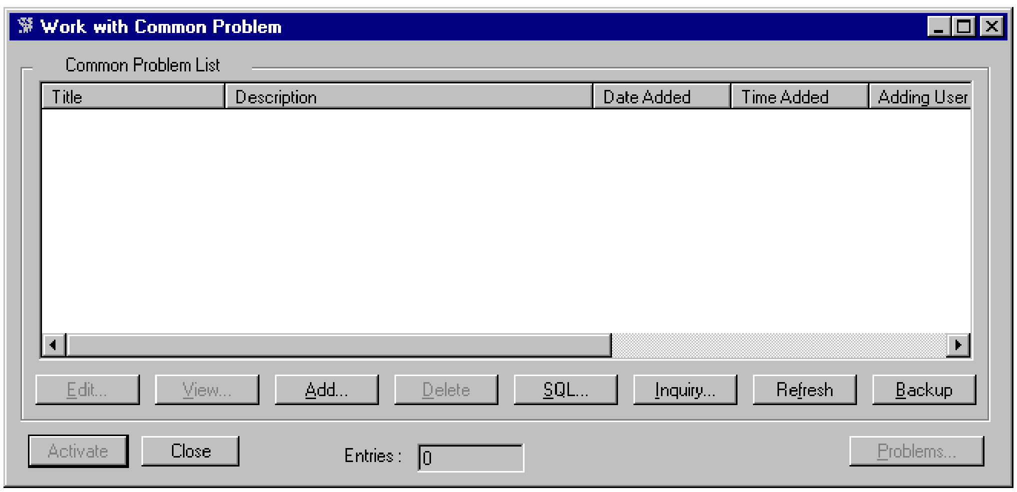


Figure 150. Work with Common Problems

To return to the previous window, click on **Backup**. As you can see in Figure 150 on page 159 the Communication group record does not have any subsequent records. To create an individual record perform the steps discussed in the following sections.

5.5.1.5 Creating an Individual Record

When you create an individual common problem record there are two ways of adding data to the fields:

- Manually enter all the data for the record
- Search for a solution that can provide the data for the record

To add the common problem click on **Add.**

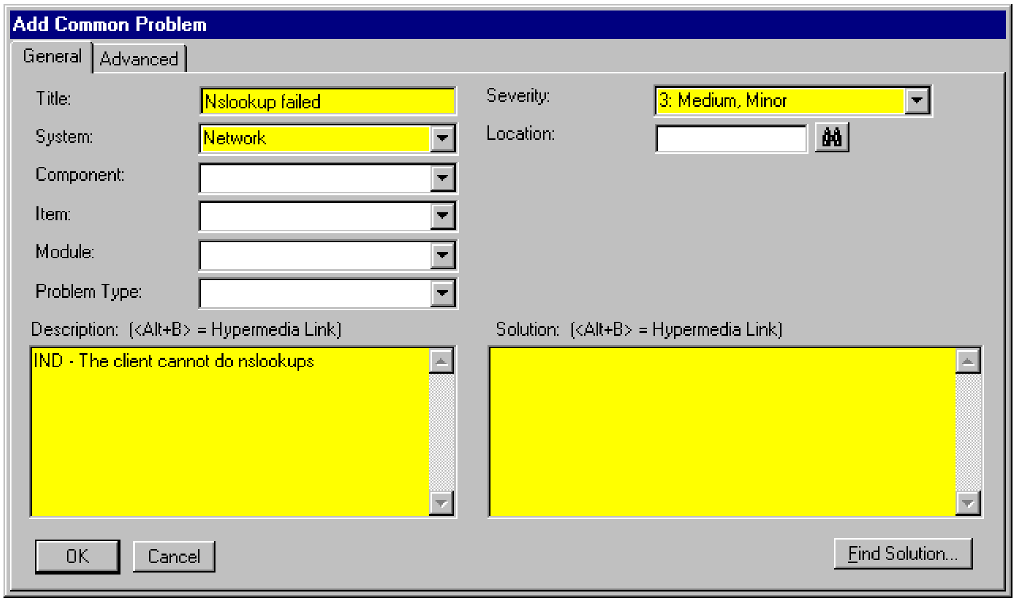


Figure 151. Add a Common Problem

To find a solution to this common problem, click on **Find solution** from Figure 151 on page 160.

The inquiry window is displayed. It will be prefilled with the values of the different fields that the Add Common Problem window had defined.

Do the inquiry that you want so that the Work with Solutions window is displayed.

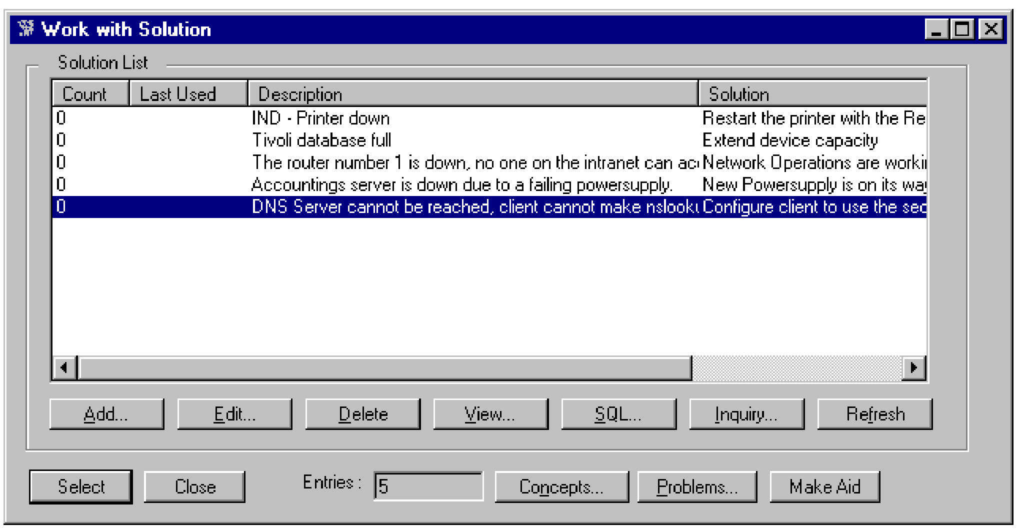


Figure 152. Work with Solutions Window

Select the solution that you want to create a Common Problem record for and the data from that solution will be transferred to the Add Common Problem window.

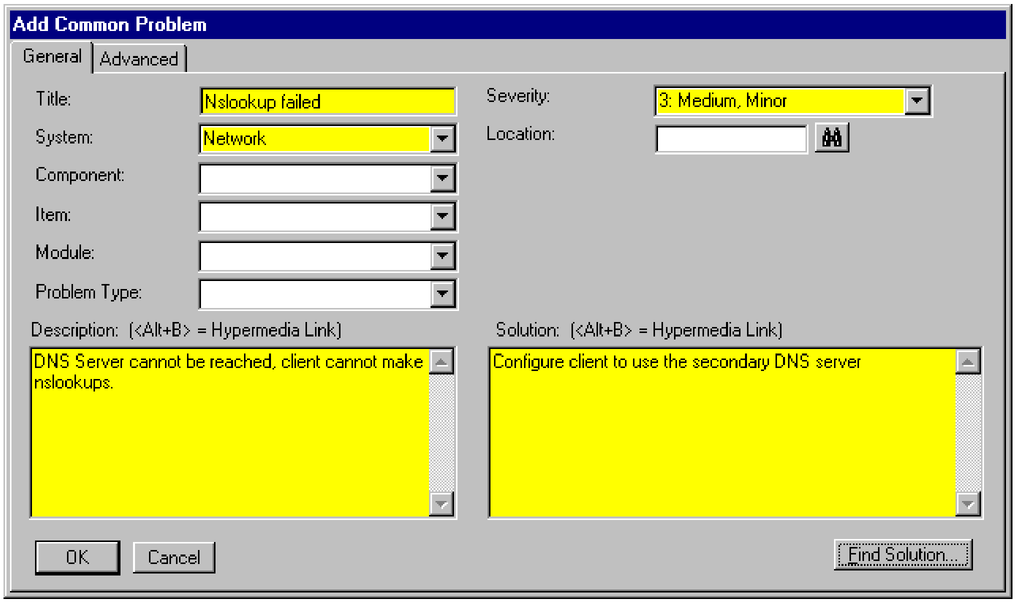


Figure 153. Add Common Problem Window after Solution Is Found

If you now click on **Advanced** you will see Figure 154 on page 161.

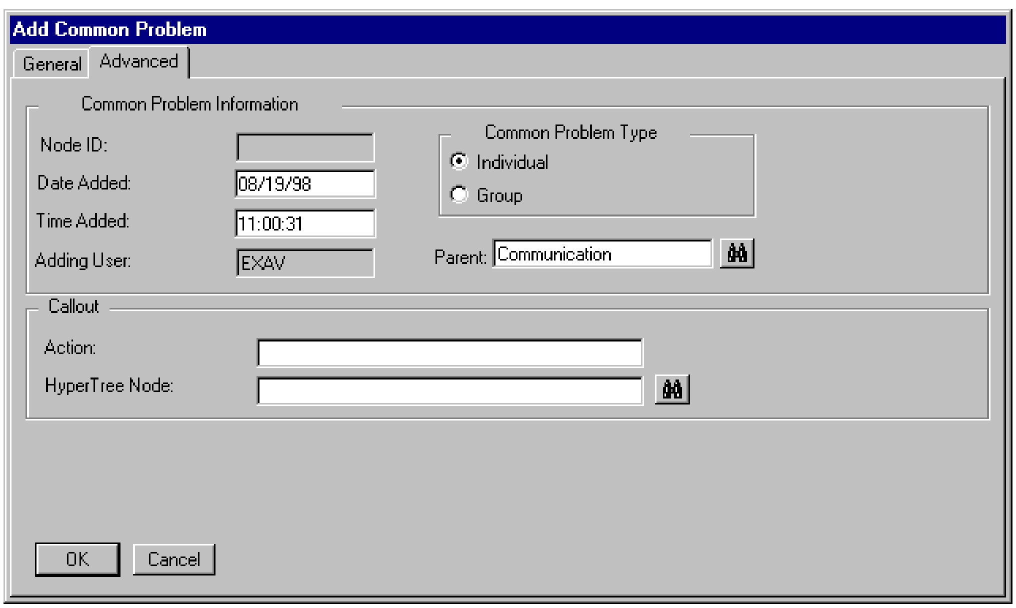


Figure 154. Add Common Problem - Advanced Tab

Here we can see that the record type is Individual and the parent record is Communication.

As you can see there is also a Callout section. Here we can set actions that the Common Problems tool will perform when we activate the Common Problem record.

This feature is useful when, for example, you want to notify someone that the Common Problem record has been used, or if you want to execute a help file to the analyst that contains more detailed info about the common problem.

You can also execute callout to a script program or to any external program. The syntax for a Tivoli Script program is either Module:procedure () or module:function ().

The syntax for a external program is D:\directory\filename.ext.

You can also display a hyper tree node when the record is being activated. To do this enter the Node ID in the Hyper Tree Node dialog box or browse to the node you want through the Browse button beside the dialog box (see also Chapter 5.5.4, “Hyper Trees (H/T)” on page 166).

If you now click **OK**, you will see the Common Problem record that was created (see Figure 155 on page 162).

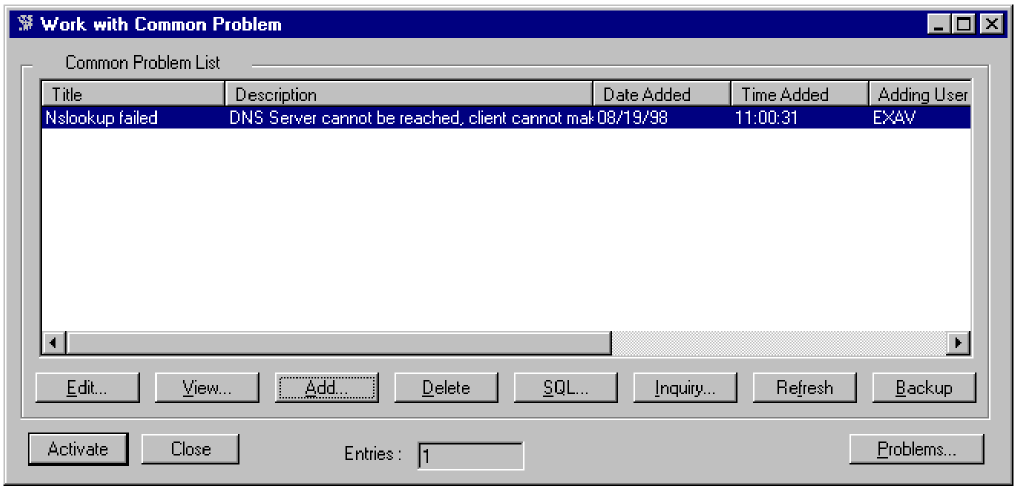


Figure 155. Work with Common Problems Showing the New Record

5.5.2 Error Messages

The Error Messages diagnostic helps an analyst to search for error messages that a caller reports. Error messages use the following fields to find matching solutions:

- System
- Component
- Item
- Module
- Problem Type

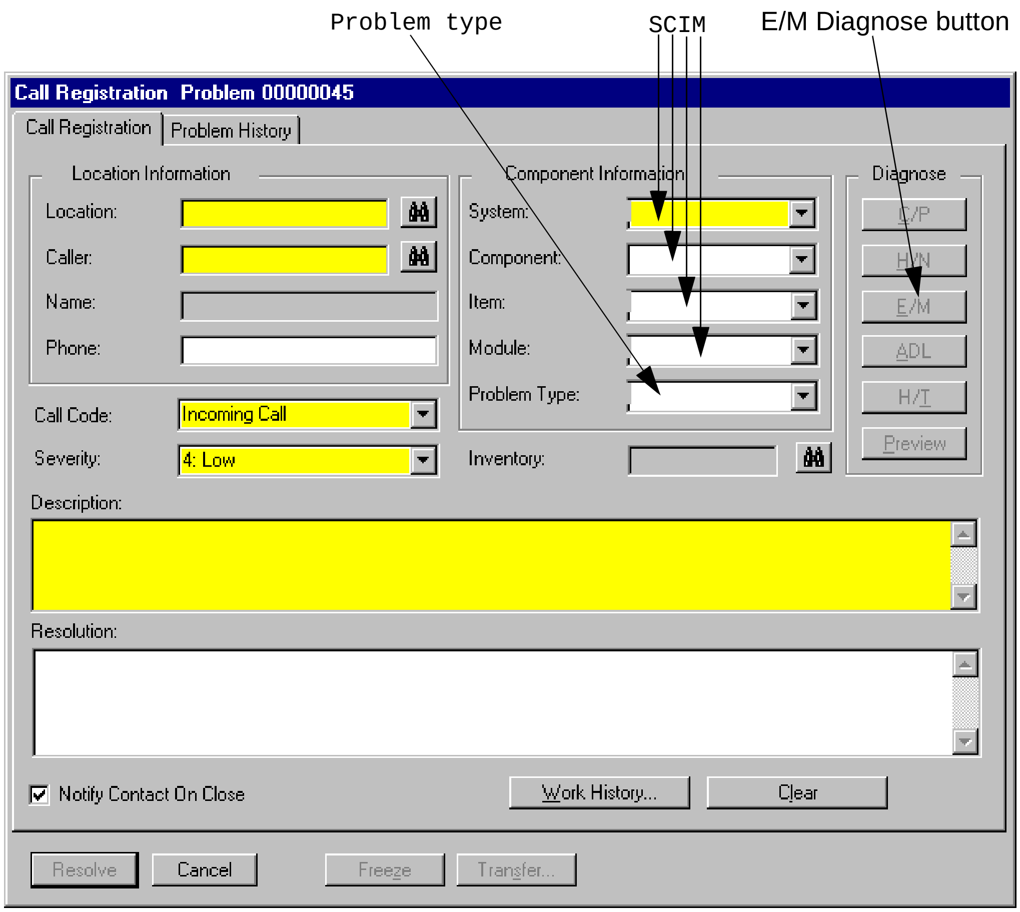


Figure 156. Call Registration - Error Messages

As with the other diagnostic tools, you can diagnose problems based on the data you have entered in the different fields and then clicking on **Preview**. In addition to this you can also access the error messages directly by selecting the **E/M** diagnose button.

The structure for the error messages is basically the same as the common problems, apart from the following:

- The windows look the same, except that the Location field is not displayed.
- The records are hierarchically organized into the following types of records:
  - Group records
  - Individual records
- You can navigate down in the hierarchy by double-clicking a group record or by selecting it and clicking on **Activate**. To view up the hierarchy, click on **Backup**.
- You can create individual error message records by filling in the data manually or by taking the data directly from a solution.
- You can do a callout to a Tivoli Script application, external program or to a hyper tree node.

5.5.3 Hot News

The Hot News diagnostic aid works with time-based solutions. This means that all the Hot News records have a starting date and an ending date. This is a very good way of informing an analyst that for the call they are working with there is a solution, but it will not be available until a certain date or time, for example, if a patch for an application will not be available for a few days.

The data that Hot News uses to find a matching problem when an analyst enters the call screen is:

- Current date and time
- System
- Component
- Item
- Module
- Location
- Problem type

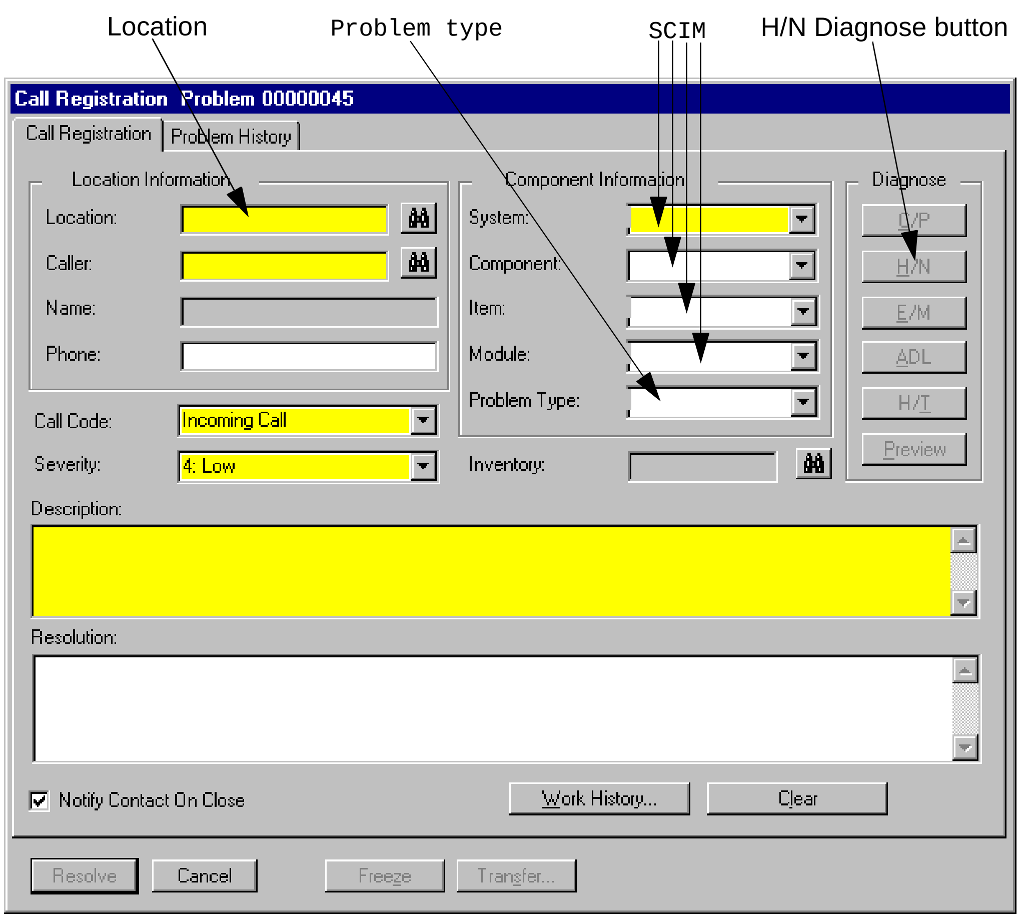


Figure 157. Call Registration, Hot News Working Fields

When any of these fields are entered, the H/N diagnose button will become available to search for Hot News records. You can get a preview of how many records are matching by clicking on **Preview**.

5.5.3.1 Adding a Hot News Record

The Hot News records are not divided into a hierarchical structure as common problems or error messages. A Hot News record can only be an individual record.

The Work with Hot News window is accessed by clicking on **Diagnostics** followed by **Work with Hot News** from the Advisor startup window.

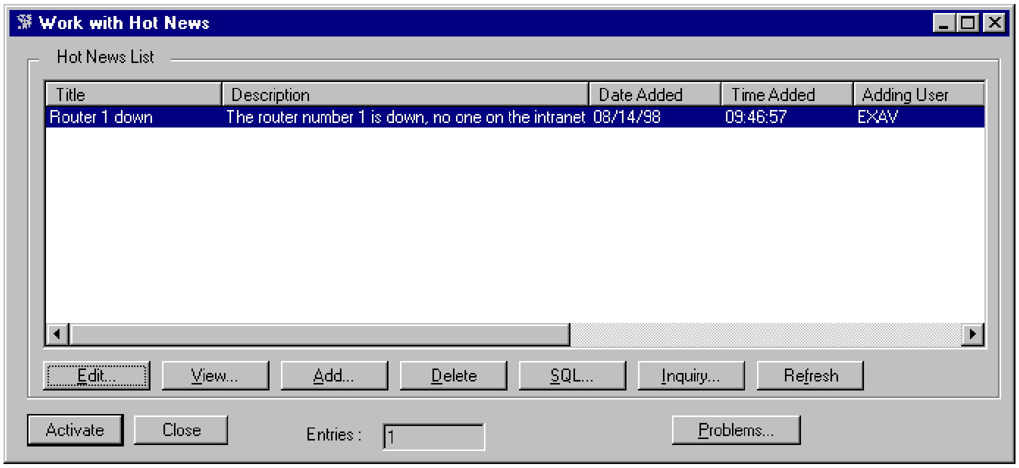


Figure 158. Work with Hot News

From the screen shown in Figure 158 on page 165 you can edit, view, add or delete a Hot News record. We can also select a record and click on **Activate** to copy a record's data into a Call Registration window. To see all the problems that are associated with this Hot News record, click on **Problems**.

The Add Hot News window is shown in Figure 159 on page 165.

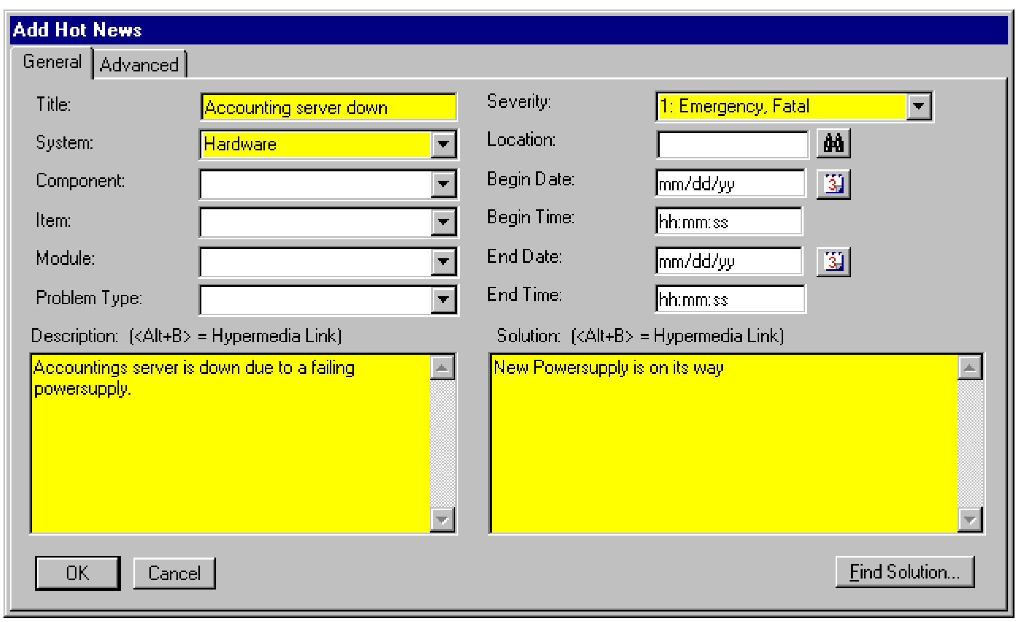


Figure 159. Add Hot News, General

Here you can see all the fields that can be filled in manually. You can also click on **Find Solution** to pull data from an existing solution into a Hot News record.

By selecting the **Advanced** tab, we see Figure 160 on page 166.

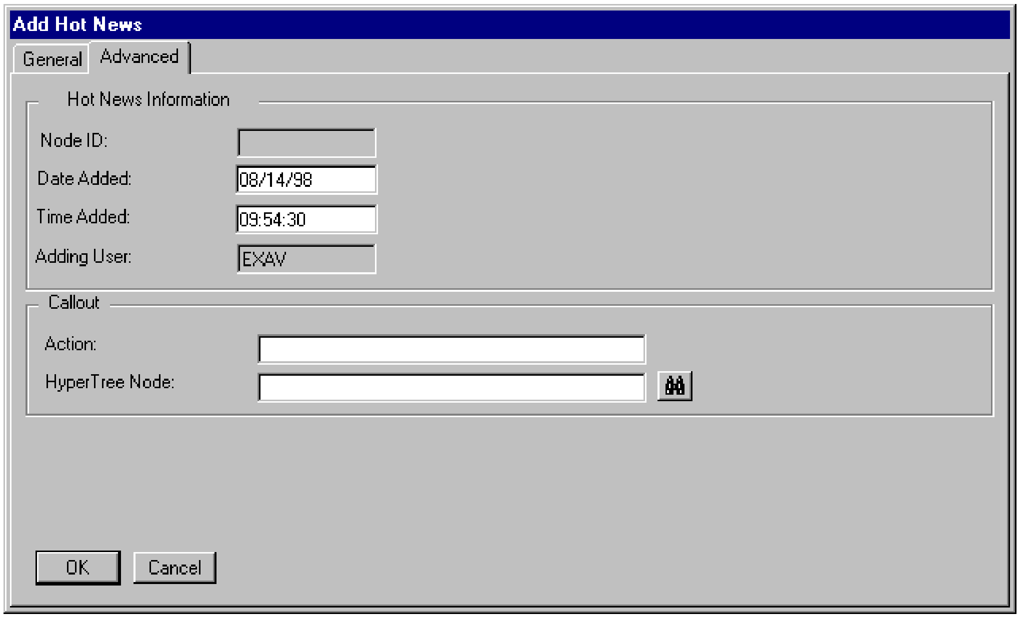


Figure 160. Add Hot News, Advanced

The box where you can choose if the record should be an individual record or a group record as in Common Problems and Error Messages is removed. The callout box that enables us to execute a Tivoli Script program, any external program or activate a Hyper Tree Node still exists.

5.5.4 Hyper Trees (H/T)

The Hyper Tree diagnostic aid helps the analyst to solve a problem by formulating questions to the user that is calling. Depending on what answer the analyst gets from the user, the Hyper Tree will go down different branches in the Hyper Tree structure. The hierarchy is based on a tree structure. The tree is built up by nodes. These nodes can be of two types:

- Question/response node
- Solution node

A question/response node can contain one question, one or more responses and one or more actions.

The response part in a question/response node can be linked to another node. This node can be either a question/response node or a solution node.

The solution node contains the instructions on how to solve or transfer the problem. The solution node is the leaf node and cannot be linked to anything else other than a response node.

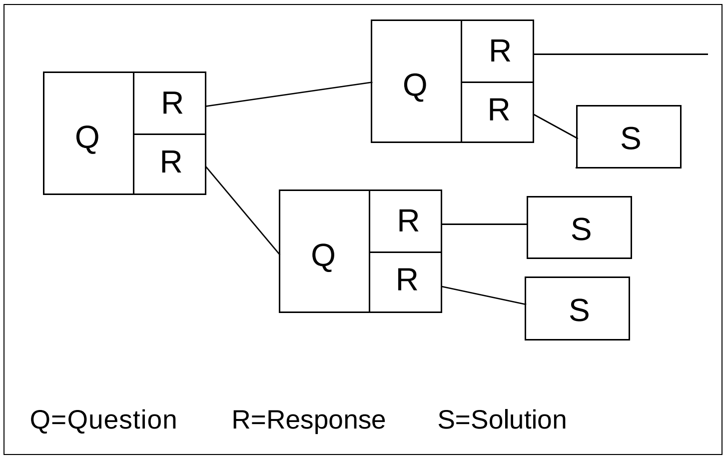


Figure 161. Hyper Trees, Node Structure

Figure 161 on page 167 shows the relationships and Figure 162 on page 167 shows an example tree:

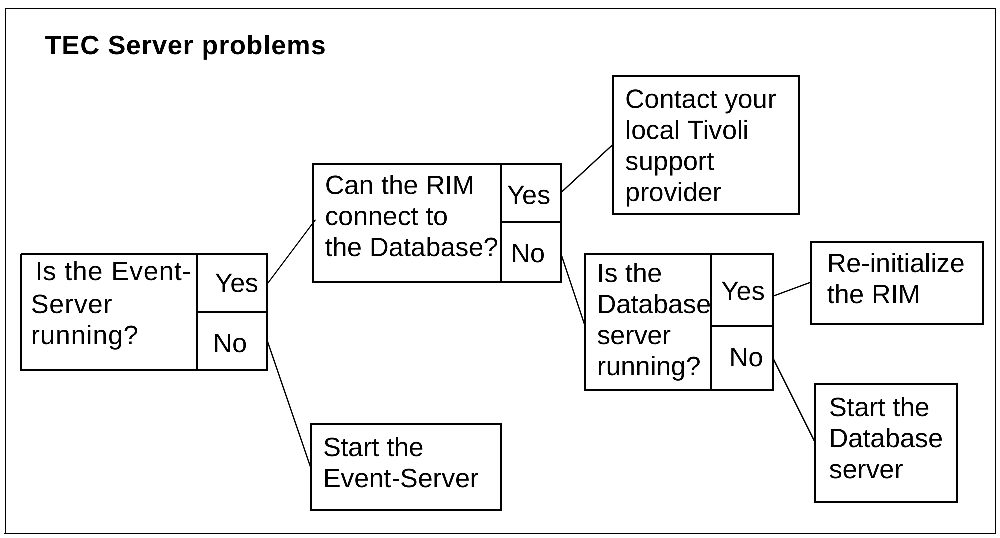


Figure 162. Example Hyper Tree, TEC Server Problems

The different trees can also be linked together so you do not have to create the same information over again. Here is an example of a tree for the TEC Console problems:

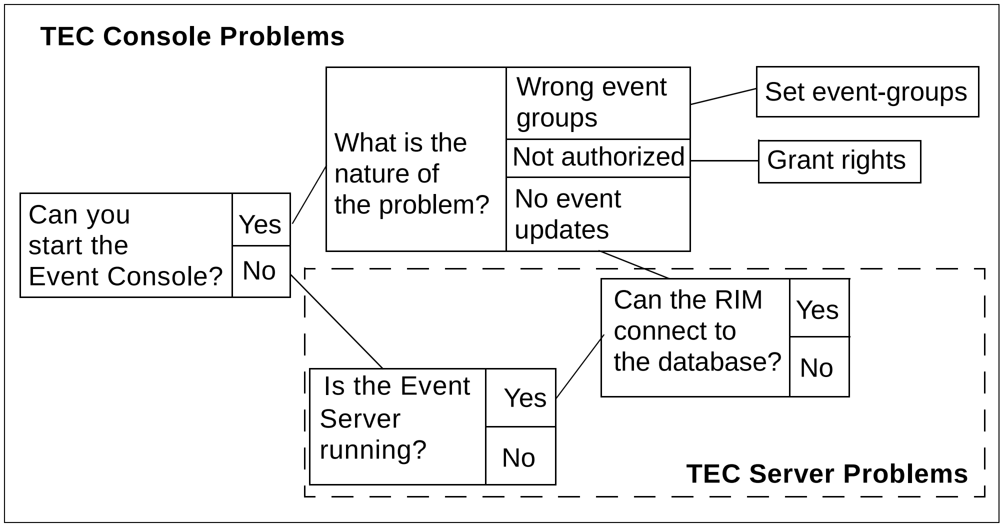


Figure 163. Example Hyper Tree, TEC Console Problems

The two nodes that are inside the dotted line belong to the TEC Server Problem tree.

If we combine these two trees, we get the following result:

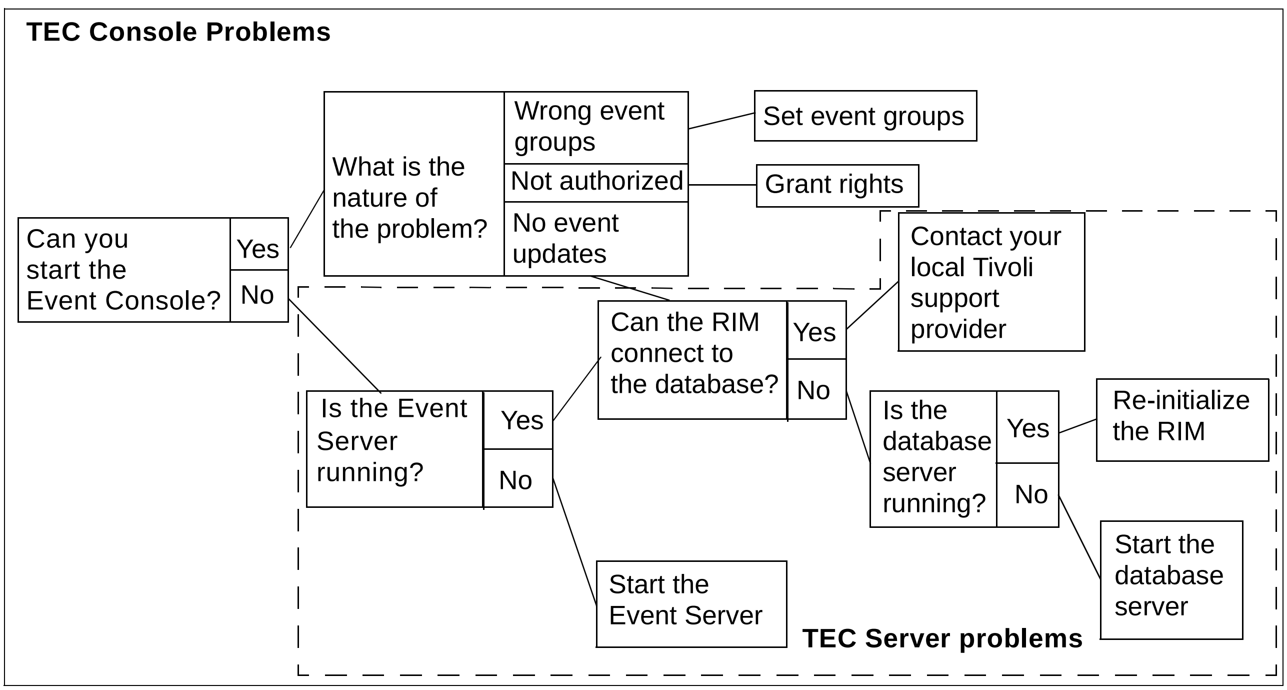


Figure 164. Hyper Tree Example, TEC Server Problems and TEC Console Problems

You can read the flow of the process from left to right. Each decision is based on the response from the caller.

5.5.4.1 Working with Root Nodes

A root node is the first node in a Hyper Tree. This node should have the general description of the Hyper Tree, for example, printer problems or application problems. To work with HyperTrees select **Diagnostics** followed by **Work with HyperTrees** (see Figure 165 on page 169).

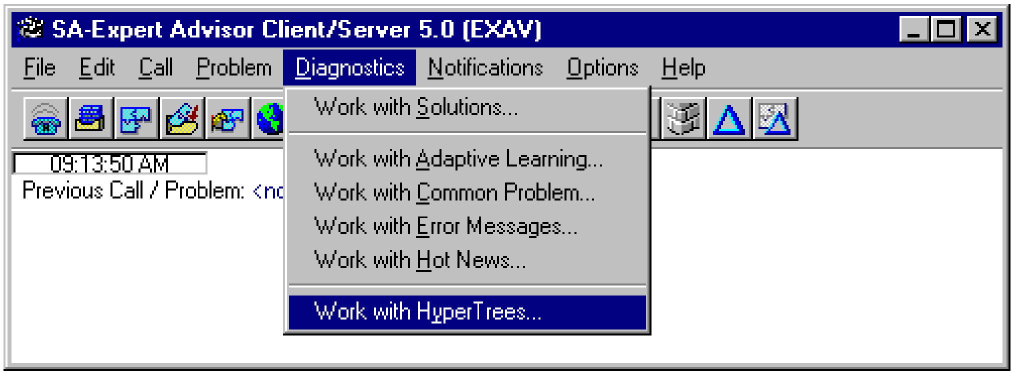


Figure 165. Work with HyperTrees

Select the **HyperTree Editor** icon shown in Figure 166 on page 169.

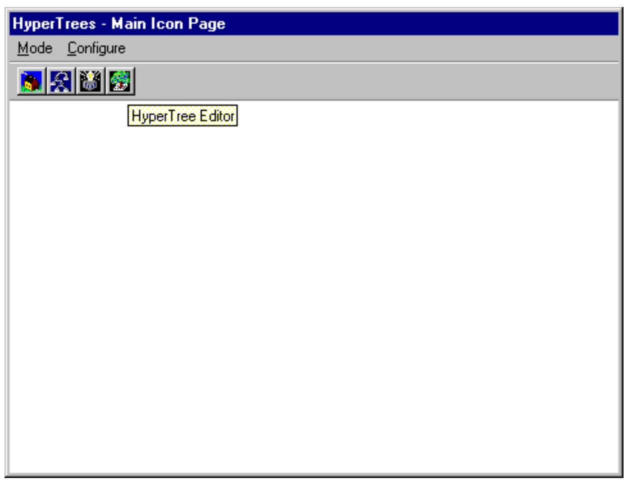


Figure 166. Main HyperTree Editor

Figure 167 on page 170 shows the list of HyperTree definitions for your environment. We have a number of pre-defined HyperTrees that were installed with the Knowledge Pak.

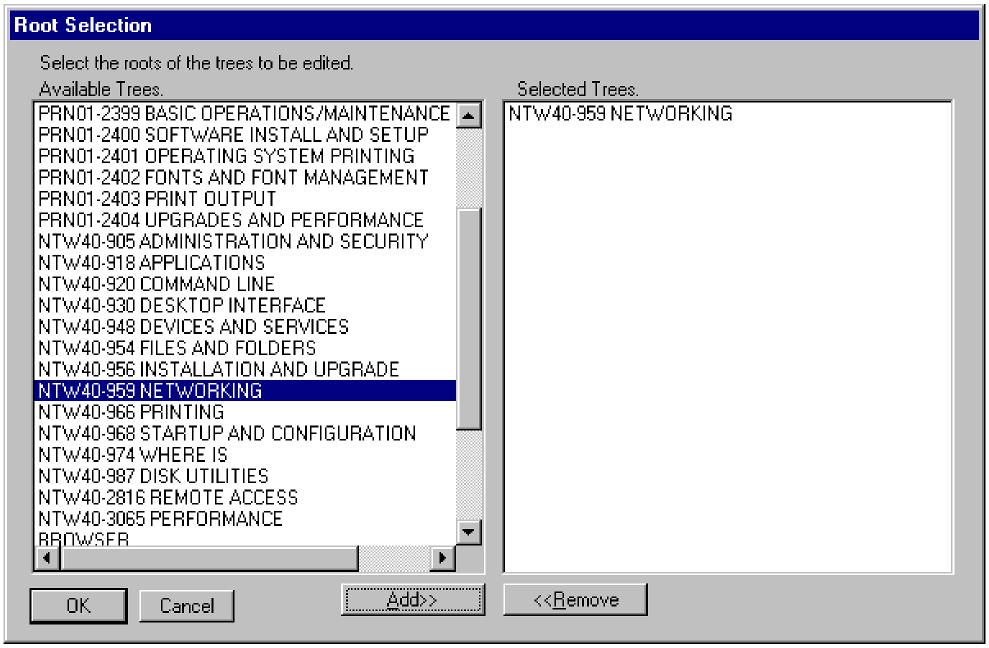


Figure 167. Work with Hypertrees

We selected **NTW40-959-NETWORKING** for NT networking problems. To work with this HyperTree click on **Add>>** followed by **OK**.

From the screen shown in Figure on page 170 double-click the HyperTree definition.

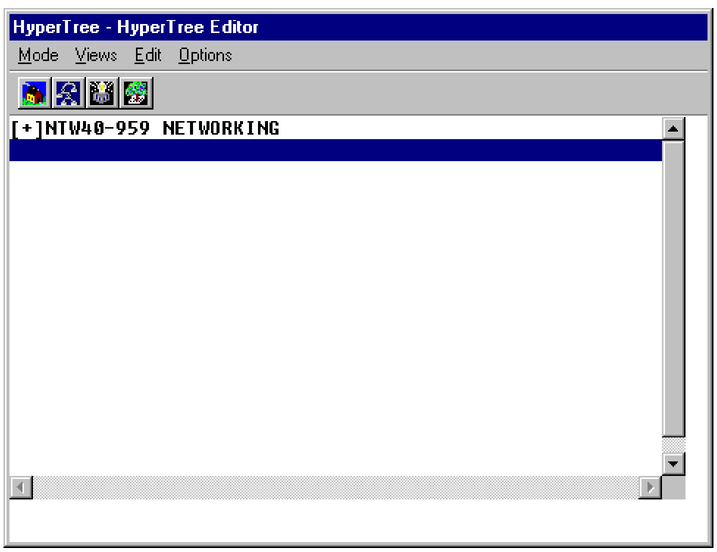


Figure 168. Viewing the Hypertree Definition

Figure 169 on page 171 will appear.

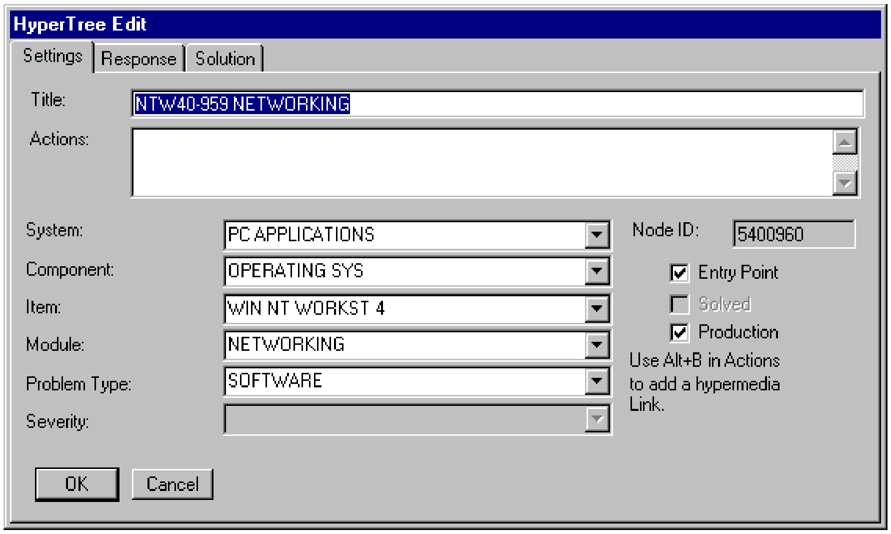


Figure 169. HyperTree Definition

Figure 169 on page 171 shows the definition. This includes the SCIM defined for this HyperTree.

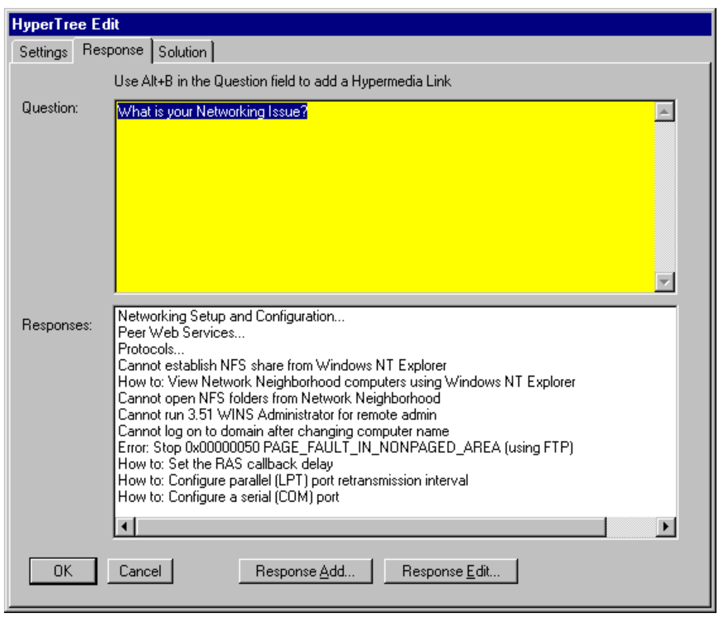


Figure 170. Response Window

To view the response click on the **Response** tab shown in Figure 170 on page 171.

Next we show how to define a new HyperTree.

5.5.4.2 Adding a Root Node

To add a new root node, from the Tivoli Advisor Diagnostics menu select **Work with HyperTrees** (see Figure 165 on page 169).

To add a new root node, select the first empty line below the existing root node and by using a single click on the left-hand mouse button, select **Edit -> Insert.**

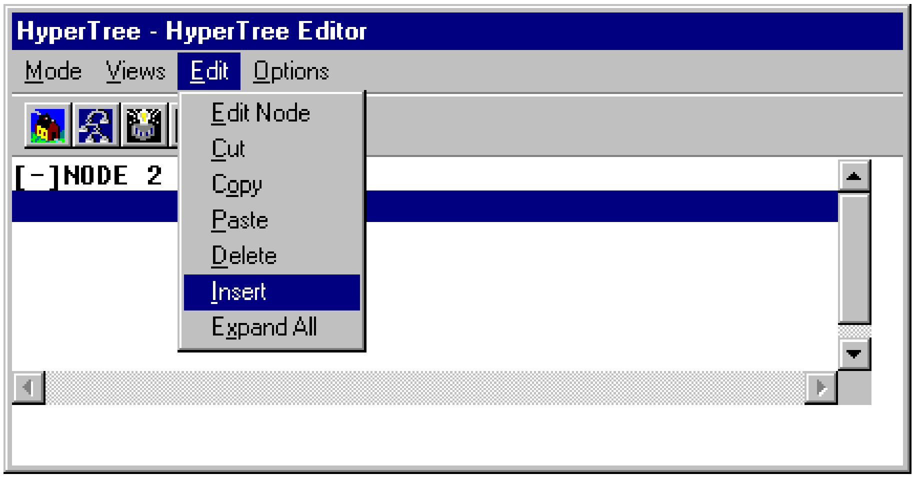


Figure 171. HyperTree - HyperTree Editor, Edit - Insert Menu

Note

If you are inserting the first root node, you will not see any root nodes and will not be able to select an empty line. Therefore, you can choose **Edit - Insert** directly.

When you have done this, the node just created appears in the window as shown in Figure 172 on page 172.

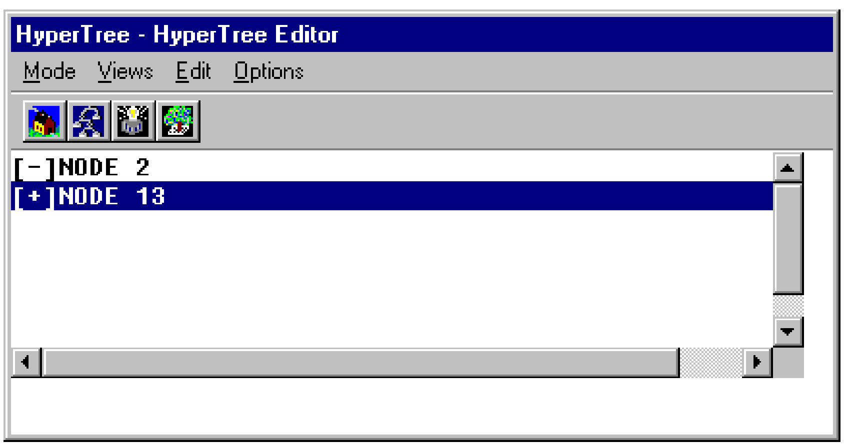


Figure 172. HyperTree Editor, New Root Node

To edit this node, select it by double-clicking on it, or choose **Edit - Edit Node** from the menu. If you do this, the HyperTree Edit window appears (see Figure 173 on page 173).

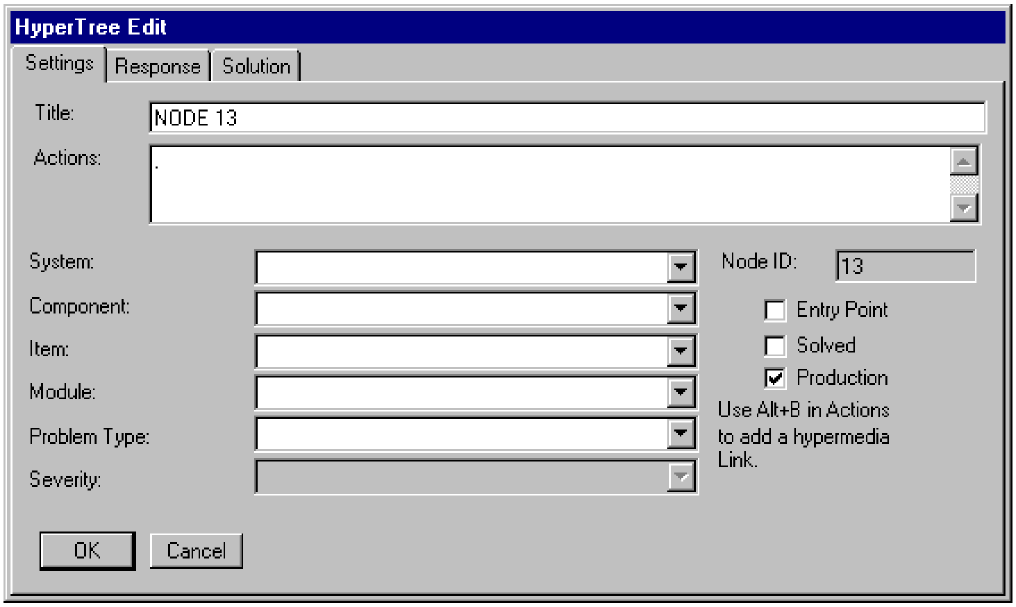


Figure 173. HyperTree Edit Window

The fields are explained below:

- Title

  This is the field where the title of the node goes. This title appears when an analyst works in sequential response mode and it is also written to the audit trail in Advisor. Therefore, it is important to choose a good title for the node.

- Actions

  In this field you should put instructions of actions that either the analyst or the user should perform before the analyst can ask the question of this node to the user.

- SCIM

  Here enter the SCIM information about the node. The more information, the better results you will get when the analysts are working with HyperTrees.

- Problem Type

  Enter the type of problem. If nothing is entered in this field, the node matches for all problem types.

- Entry Point check box

  The entry point indicates that this is where an analyst can begin the problem resolution. All root nodes should be entry points. Also, the nodes where you plan to have access from another tree should also be entry points.

- Solved check box

  This box is only used if the node is a solution node. When this box is selected, the description and solution fields in the Solution tab will become available.

- Production check box

  This check box indicates whether the node is available to the analysts or not. If it is cleared, it is not available. This could be used in a test phase, for example.

This is how we filled in the fields for our example tree:

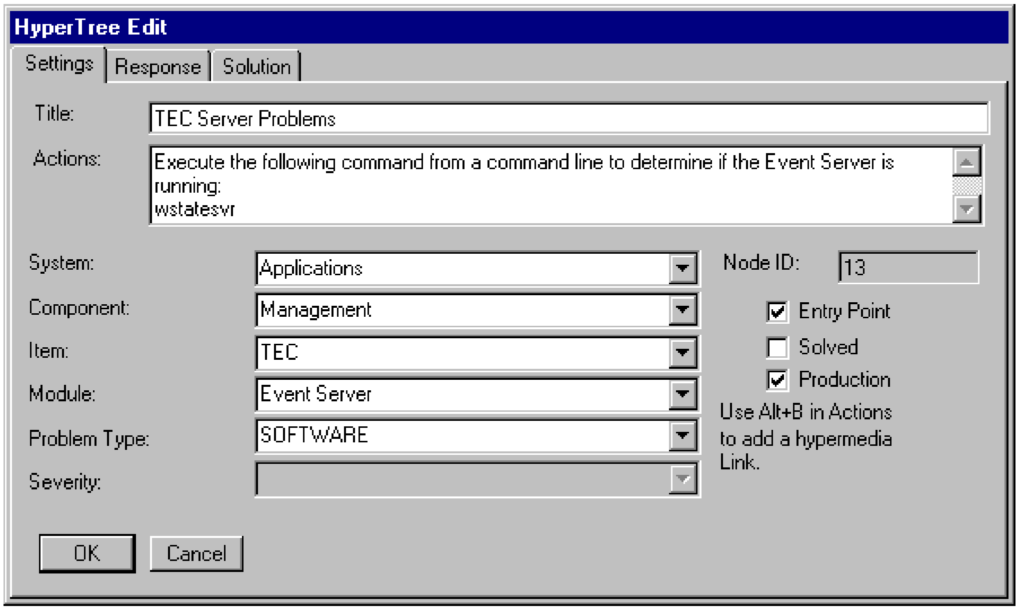


Figure 174. HyperTree Edit Window, Example Node

The root node for the TEC server problem tree is now created. The next step is to create the question/response nodes.

5.5.4.3 Working with Question/Response Nodes

From the HyperTrees main icon page select **Mode - Sequential Response**.

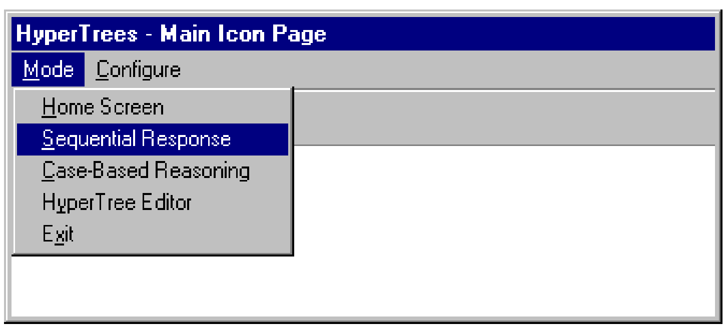


Figure 175. HyperTrees - Main Icon Page, Mode - Sequential Response

Now the Root Selection window appears as shown in Figure 176 on page 175.

.

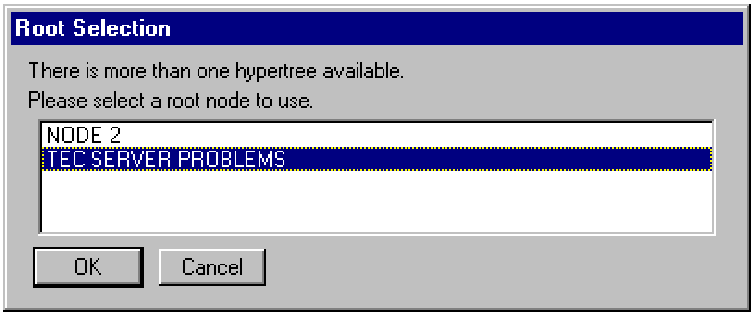


Figure 176. Root Selection Window

We chose the **TEC SERVER PROBLEMS** root node and clicked on **OK**.

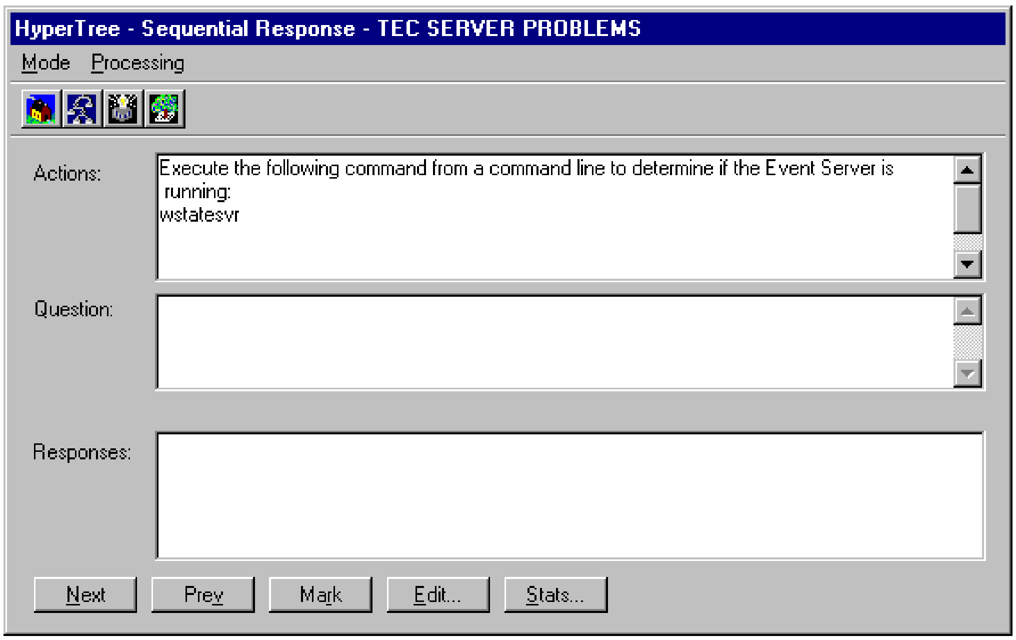


Figure 177. HyperTree, Sequential Response for TEC SERVER PROBLEMS

Figure 177 on page 175 shows the action that we filled in when we created the root node in this window.

Now we have to add a question and responses to that question to the root node. Otherwise we cannot create any subsequent nodes to this root node. Select **Edit**.

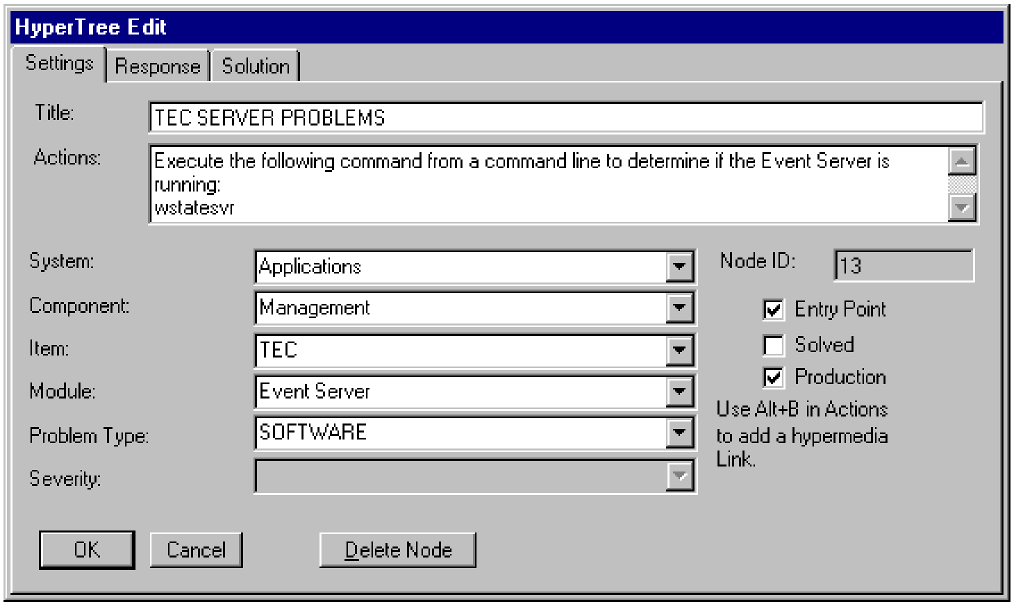


Figure 178. HyperTree Edit

You can see that this is the same window as we saw earlier when we created the root node.

To add a question to this node, choose the **Response** tab (see Figure 179 on page 176).

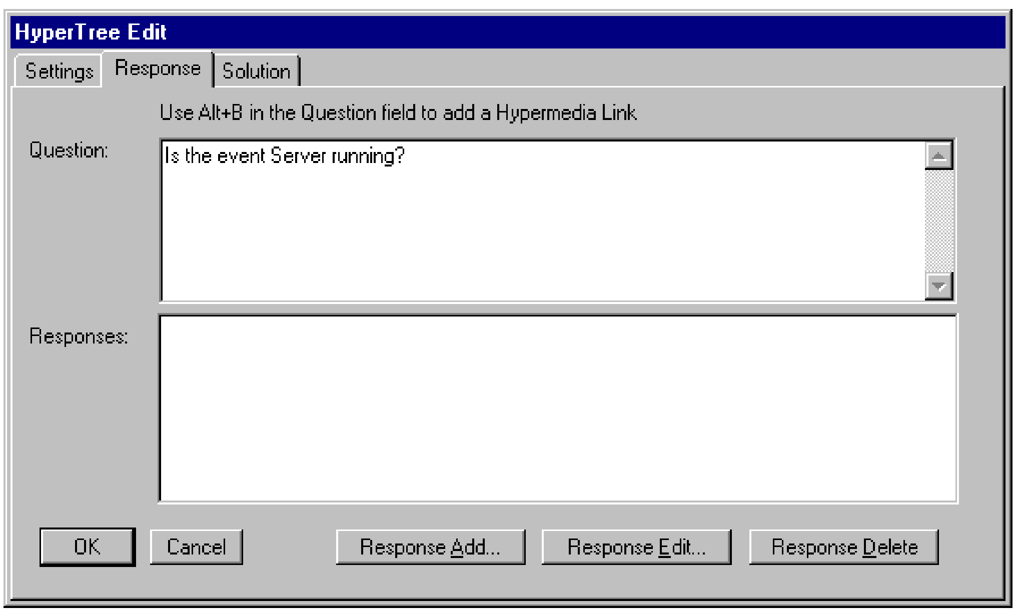


Figure 179. HyperTree Edit Window, Response

Next we added the action and the question was created. We need some responses too. To create a response, click on **Response Add** (see Figure 180 on page 177).

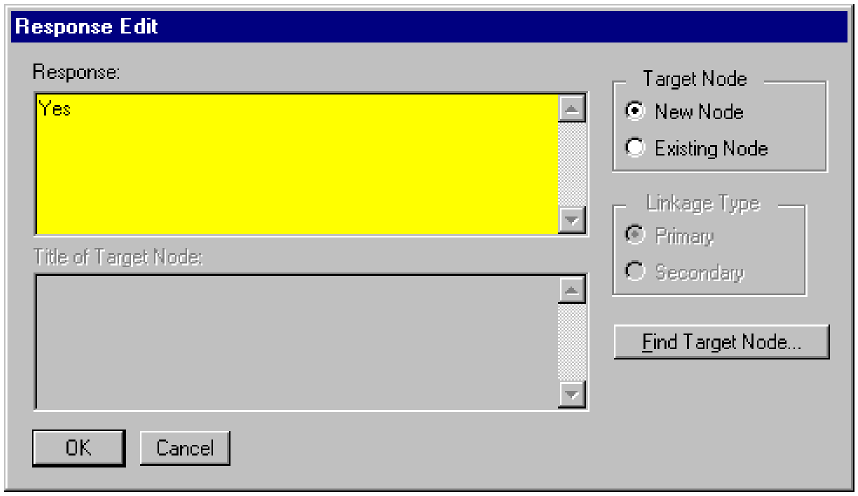


Figure 180. Response Edit

Here you can specify if the node that this response is linked to is a new one or a node that is already created. Since we did not create the node that this response will point to, we selected **New Node**.

Click **OK** to create the new response.

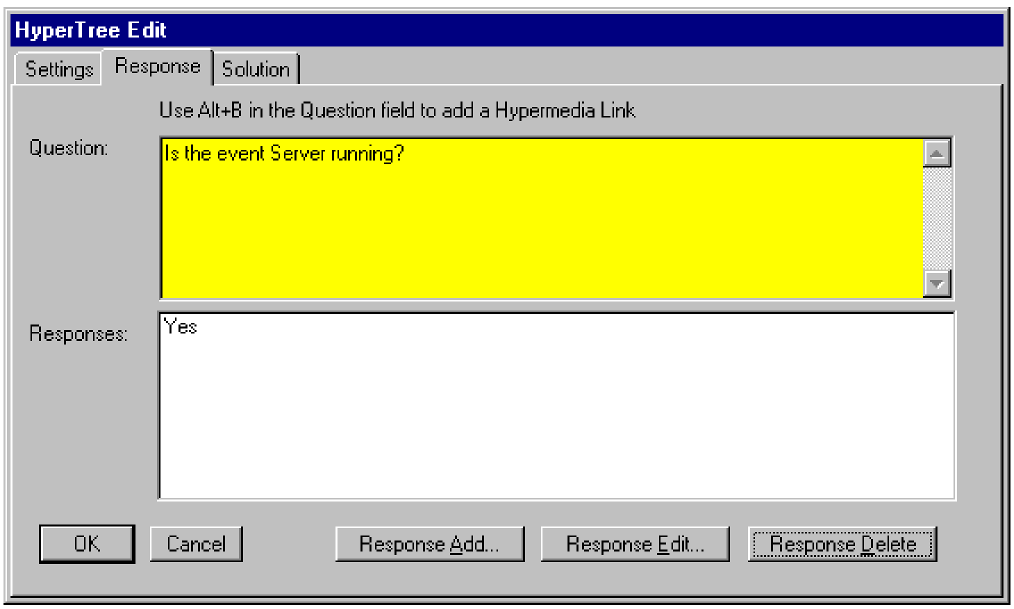


Figure 181. HyperTree Edit Window, Response

When the information has been added click on **OK**.

Now that we have a response for this node, we can create a substantial node that contains a question and responses. To do this you select the response that you want to create the next node for and click on **Next**.

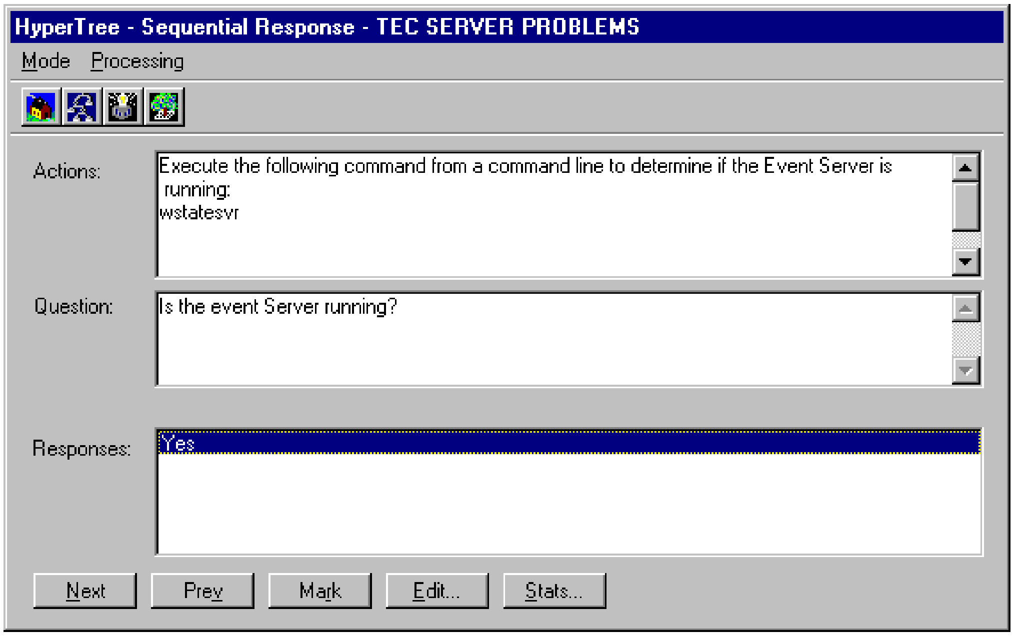


Figure 182. HyperTree - Sequential Response - TEC SERVER PROBLEMS

Figure 182 on page 178 will clear and the header of the window will change. This indicates that you have created a new question/response node, just as we specified it to do when we created the response (see Figure 180 on page 177).

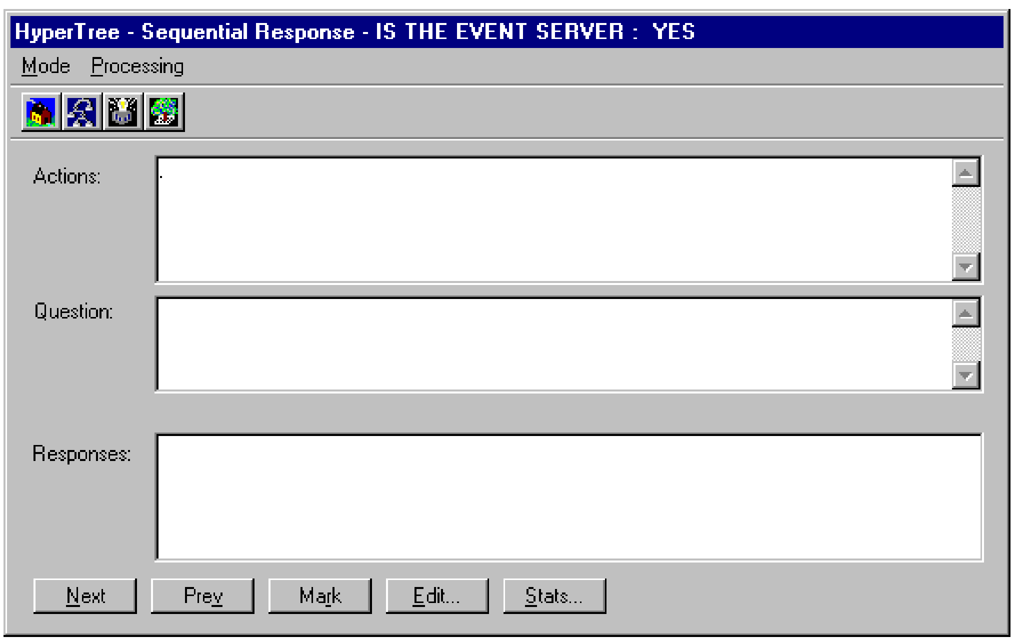


Figure 183. HyperTree - Sequential Response - IS THE EVENT SERVER: YES

To edit this node click on **Edit**. Repeat this procedure until you have defined the whole HyperTree structure.

5.5.4.4 Working with Solution Nodes

To create a solution node, in the HyperTree Edit window of the selected node, check the **Solved** box (see Figure 184 on page 179).

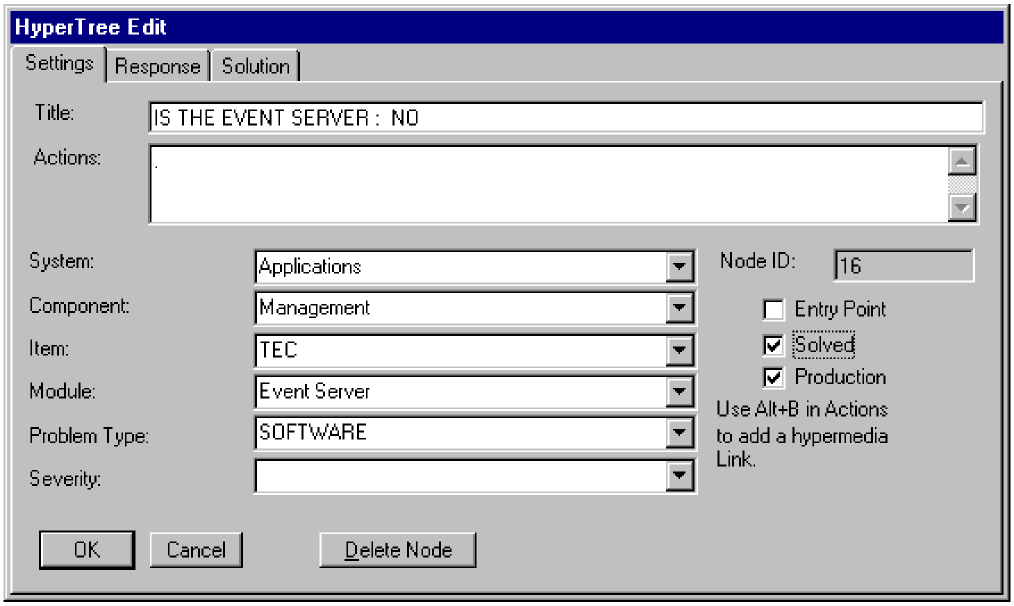


Figure 184. HyperTree Edit, Settings, Solved Check Box Selected

Then click the **Solutions** tab and enter the solution.

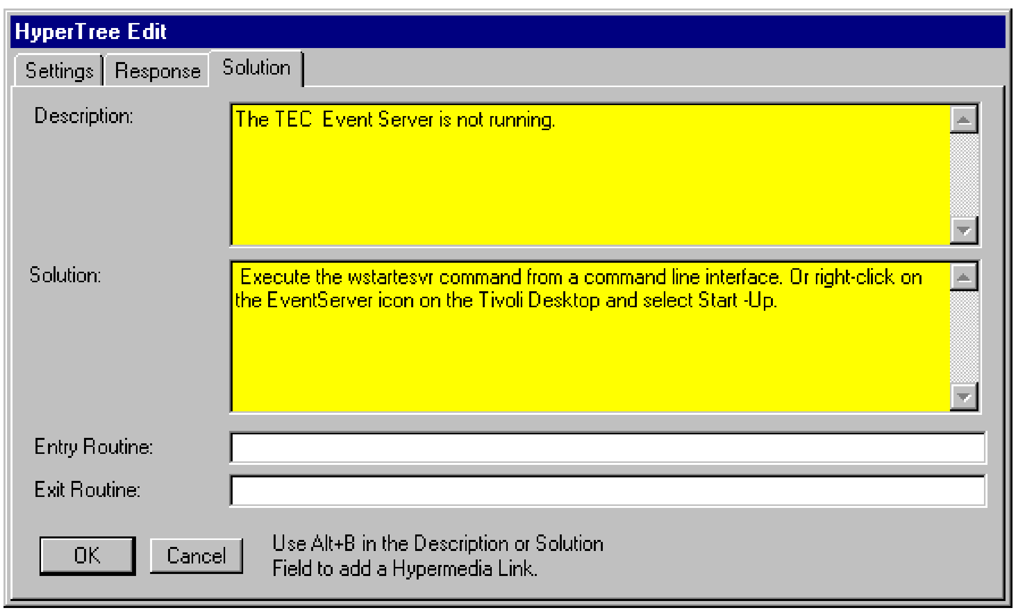


Figure 185. HyperTree Edit, Solution

You can specify Tivoli Script routines or external programs in the Entry Routine and Exit Routine boxes. If you set the program in the Entry Routine box, the routine will execute when the solution node is reached by an analyst. If you put the routine in the Exit Routine box, it will execute when the analyst decides to use that solution for the call and activates the solution node.

5.5.5 Adaptive Learning (ADL)

This section describes the Adaptive Learning (ADL) functionality of Tivoli Advisor, which is a very powerful and useful tool for problems resolution.

Adaptive Learning is a *description-based* diagnostic aid. It uses the free-text description of a problem to find possible solutions.

For example, when registering a new call you can describe the problem in free text and then click the **ADL** button to search for possible solutions.

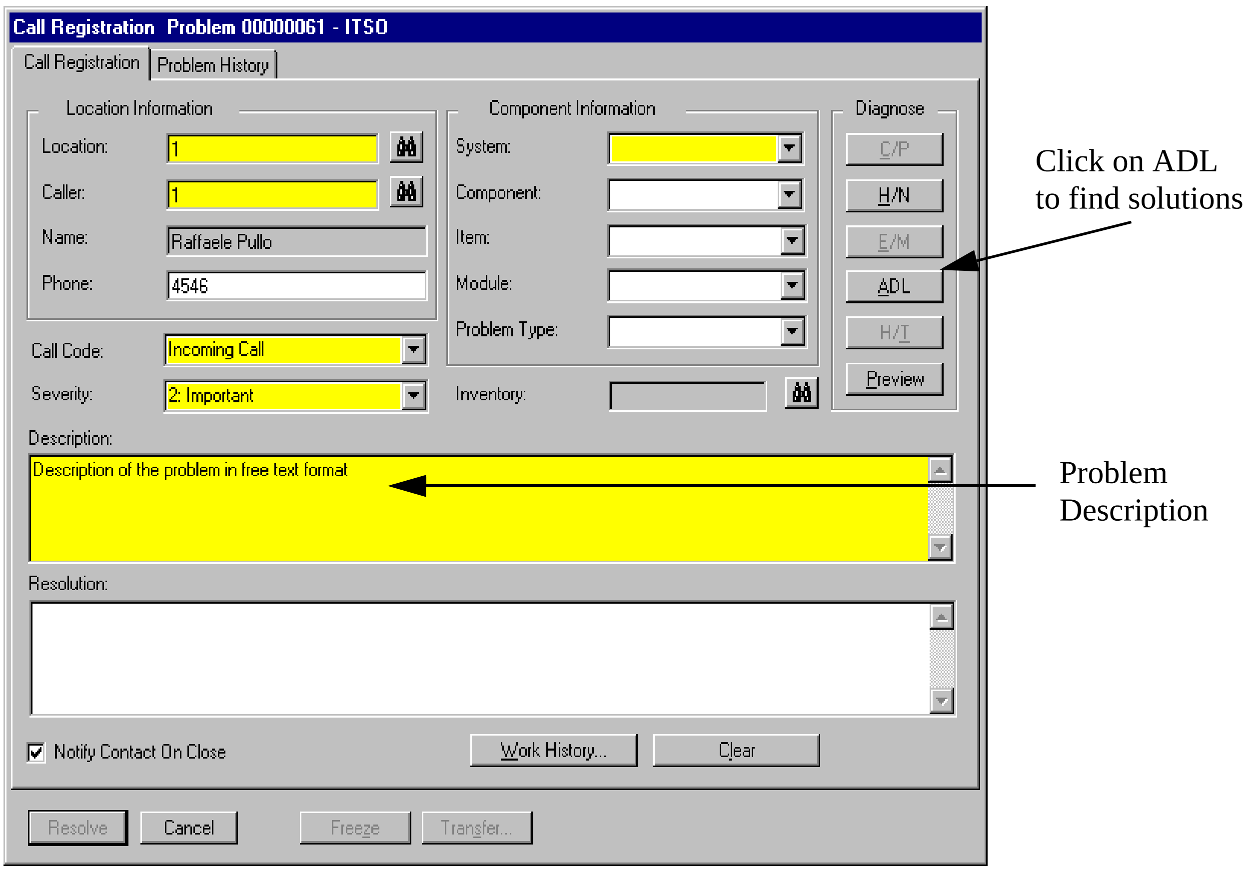


Figure 186. Using ADL When Creating a New Call

The ADL reports the possible solutions to the problem you describe, assigning a *score* to each one based on their probability to be the right solution.

The solutions contained in the database are defined by the administrator or by the users, or could even be imported from an external source. While solutions are used to solve new problems, Adaptive Learning can dynamically learn which solution is more suitable to solve a particular problem. So the more you work with the Advisor and with the diagnostic tools, the more Adaptive Learning becomes an intelligent way to find the correct solutions to any incoming problems.

The solutions used by ADL can be any of the Advisor solution sources such as:

- Common Problems
- Error Messages

- Host News
- Solution from HyperTrees
- Solutions added through the Work with Solutions dialog box

5.5.6 How ADL Works

The functionality of ADL is based on an association between a concept and a solution. Concepts are synonyms for a list of words. ADL uses a thesaurus to match keywords with concepts, so a concept is associated to one or more words or keywords. For example, the words TEC rules and event server can be associated to a common concept, which is TEC.

To better understand ADL let's discuss the relations between word-concept and concept-solution as depicted in Figure 187 on page 181.

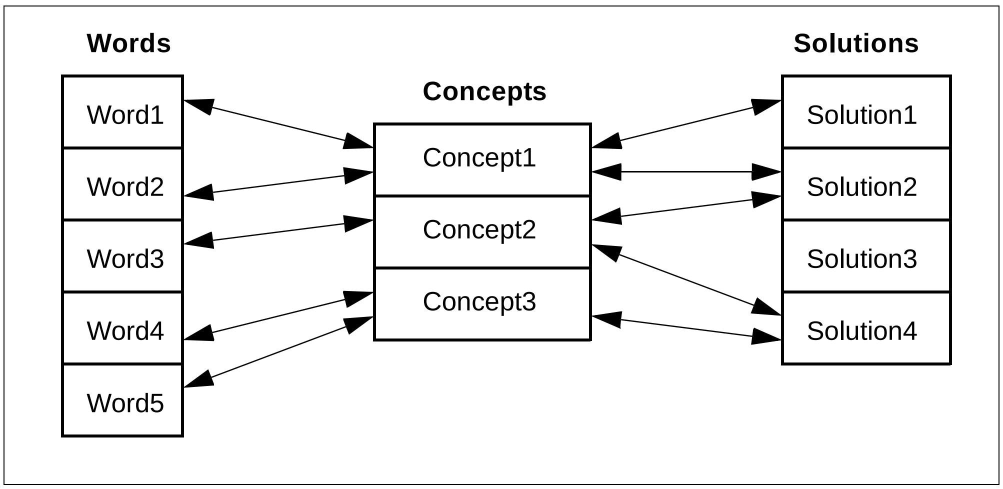


Figure 187. Relation Word Concept Solution

The key relationships in order to understand the ADL functionality are:

- A concept can be associated to many words that can contain a single word or a sentence.
- A solution can be associated to one ore more concepts.

The associations between words and concepts are provided by the diagnostic administrator using the thesaurus tool in Expert Advisor. When completed you will have a list of words and concepts. We see later how entries on this list are created.

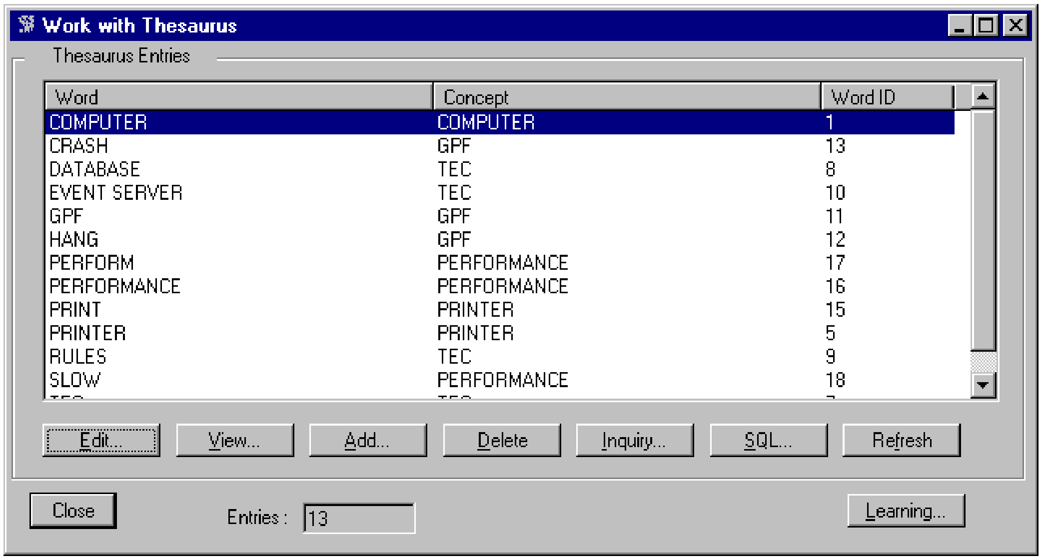


Figure 188. Working with Thesaurus, List of Words and Corresponding Concepts

The association between concepts and solutions can be done in two ways:

- Manually
- Automatically using the learning utility

We describe later how the association can be performed using the learning capability.

The ADL working can be summarized as follows:

- The user describes a problem in free-text format. This is entered into the description field. Then the ADL button can be selected.
- Adaptive Learning tries to find words in the thesaurus that match the words present in the problem description.
- If any words are found, ADL looks for corresponding concepts (see Figure on page 181).
- Next the ADL searches in the solution database to find solutions that are associated to one or more of those concepts.
- If any concepts are found, the solutions are returned with a score. The higher the score, the higher the probability that the solution is the correct one. The score is based on the number of concepts found that are associated to a solution and on the number of times a solution has been used.

An example follows to help better understand the ADL functionality.

5.5.6.1 Example Showing How ADL Learns and Finds Solutions

This section provides an example to show how Adaptive Learning learns from a previous problem's history and finds a solution for a new problem. This example does not cover all the possible ways to create new solutions, how to make the ADL learning or to find solutions, but instead is intended to help in understanding how ADL works. We perform the following steps:

1. Create a solution in the database.
2. Resolve an incoming call using a common problem solution.

3. Define the words and concepts.
4. Use the learning utility.
5. Show what the system has learned.
6. Resolve a new call using what the system learned before.

We describe all steps in detail. The first step is to create a solution.

- Create a solution

  You define a solution to a particular problem so that this solution can be used in the future by any user. There are many ways to create a solution. Figure 189 on page 183 shows the dialog box to create a solution to a common problem.

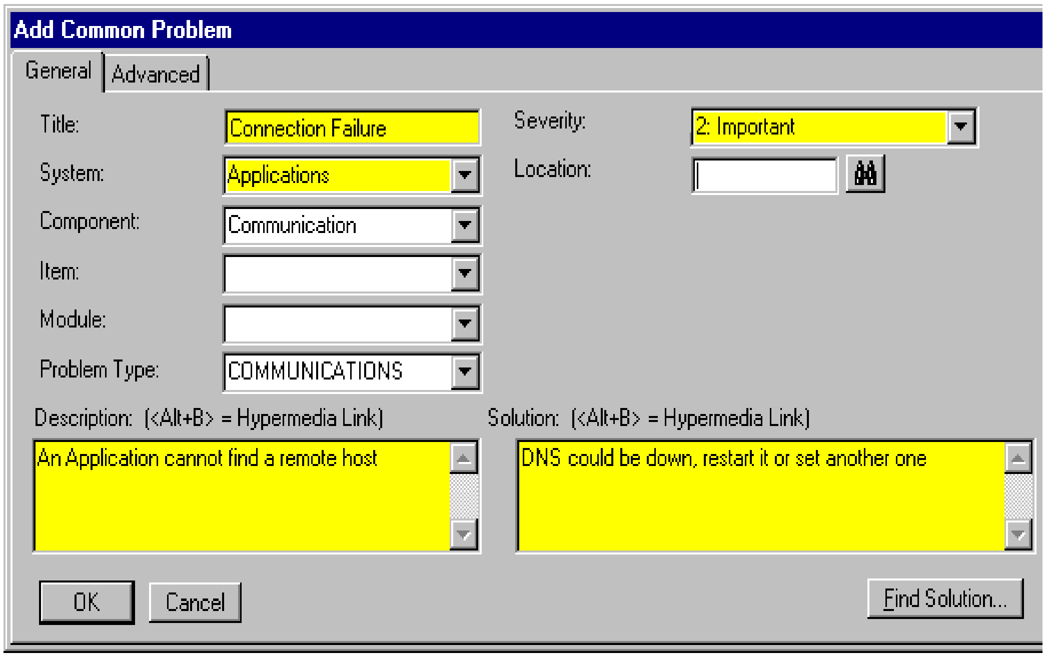


Figure 189. Adding a Common Problem

  This mechanism is the way to create a possible solution and is useful when an application fails to connect to a remote host. The solution indicates the DNS as a possible cause. After the solution as been created and activated, in the solution database you now have a new record.

- Resolve a call using a solution

  Suppose that a call is received for a Web browser problem and that from the caller's description you suspect the problem is related to a communication issue. You would enter the SCIM and the description similar to the data entered in Figure 190 on page 184.

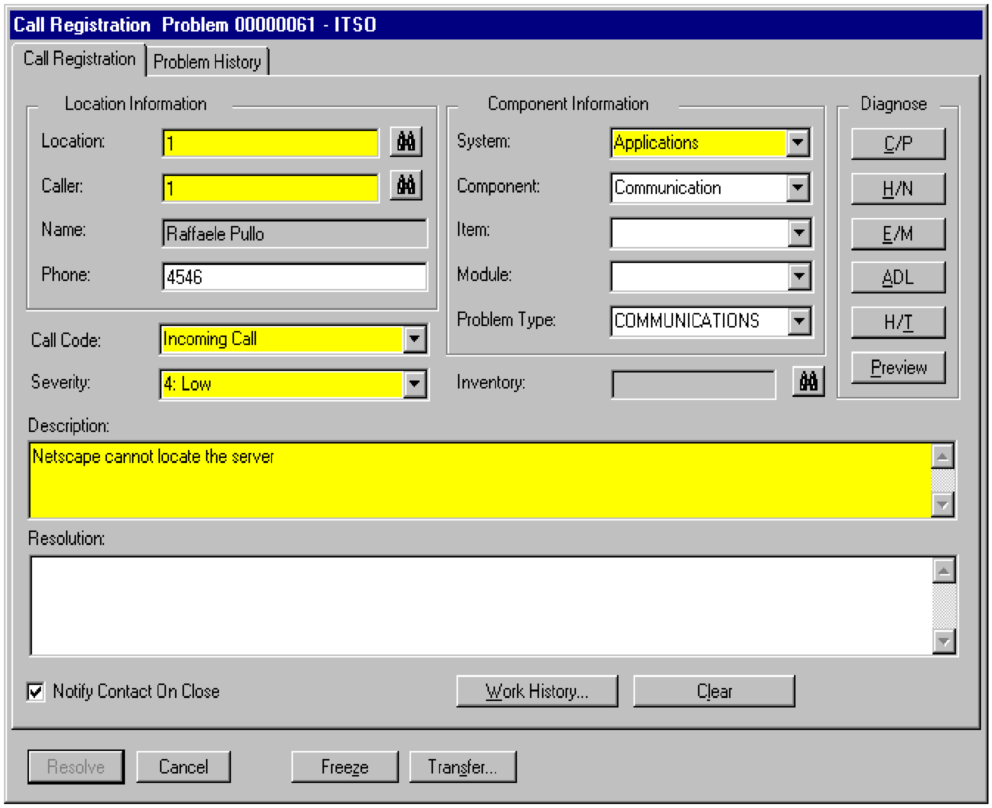


Figure 190. Call Registration

For the call registration we entered the description as:

`Netscape cannot locate the server`

This is a real message that is returned by Netscape.

Then you click the **C/P** button to search for a solution and the system comes up with the previously created solution. Use this solution to resolve the call.

Note that now in the problem database there is a new problem closed using the solution that you have created before.

- Define words and concepts

  Now we can define a concept called CONNECTION that is related to the communication problem using the following word association:

  CONNECTION

  LOCATE

  PATH

  To perform this operation we used the thesaurus utility and the dialog box similar to the one shown in Figure 191 on page 185.

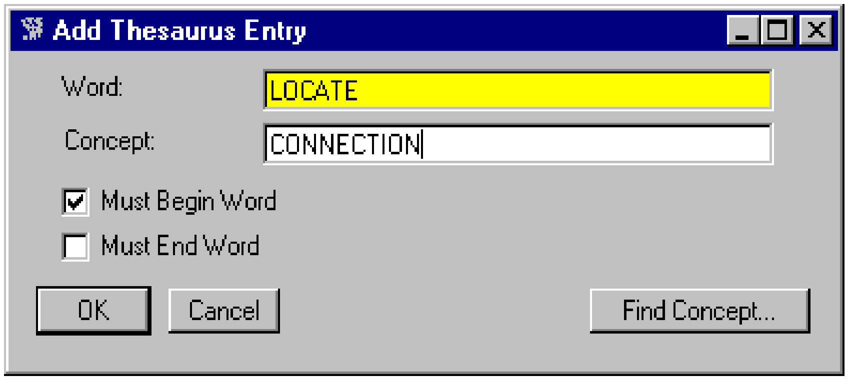


Figure 191. Adding Words and Concepts

- Use the learning utility

  Up to this point the solution we created before (see “Create a solution” on page 183) has had no concept associated with it. Using Expert Advisor we can list all the solutions. To do this, from the action bar select **Diagnostics** and then **Work with Solutions** (see Figure 192 on page 185).

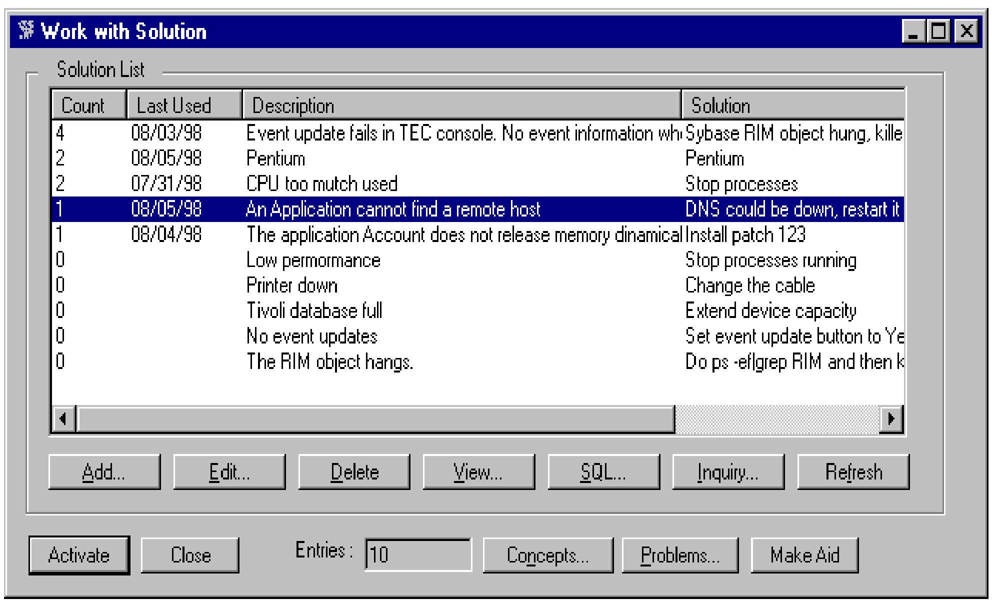


Figure 192. List of All Solutions

By selecting the solution we created and selecting **Concepts**, we got the screen shown in Figure 193 on page 186.

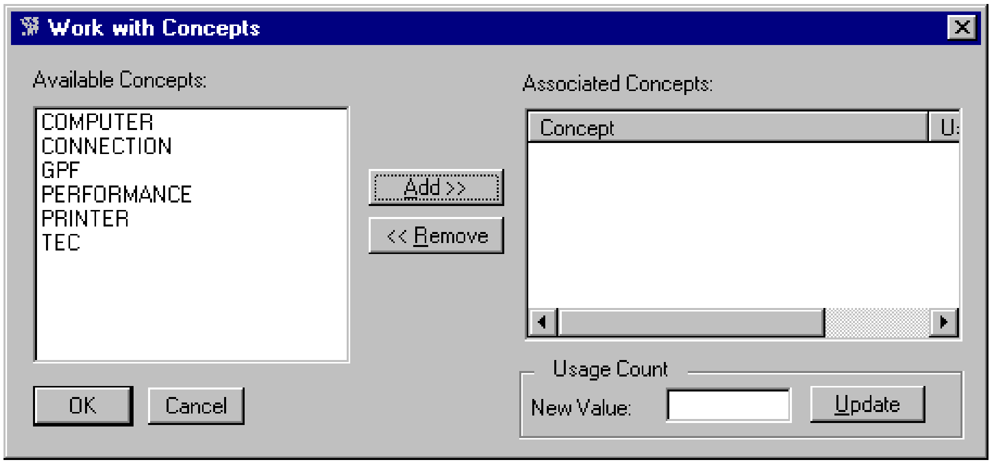


Figure 193. Concept Associated to the Solution

As you can see no concept is associated with the solution. Now we can ask the learning utility to learn from the previous history. The description of learning is described in the next example. Expert Advisor lists all the words using the thesaurus (see Figure 194 on page 186).

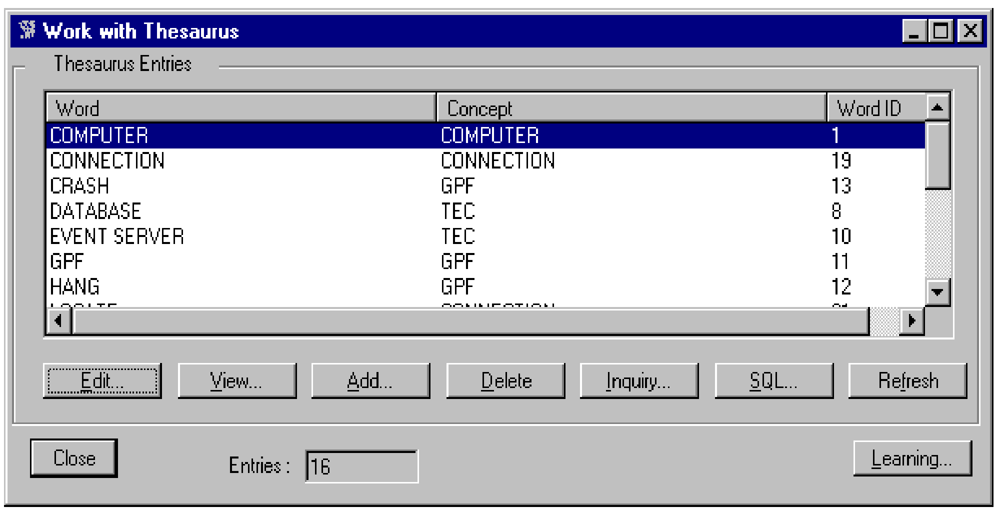


Figure 194. Thesaurus Words and Concepts

Click on **Learning** to start the learning utility (see Figure 195 on page 187).

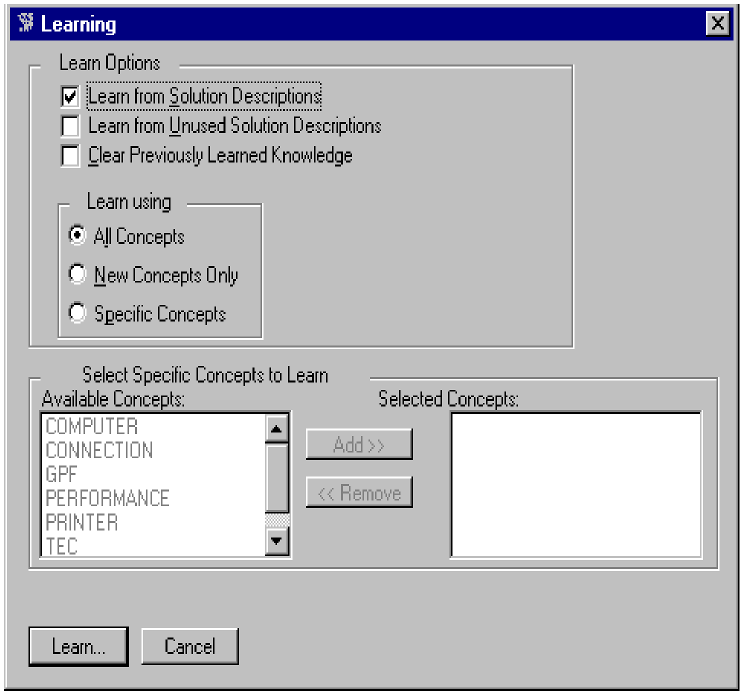


Figure 195. Learning

By selecting the **Learn...** button the system starts to learn. It will request some query criteria on the problems the system is going to analyze. Here we choose **OK** in the Query box displayed. At the end you receive the window shown in Figure 196 on page 187.

Figure 196. Learn Complete

This message shows that the learning utility has scanned the problems in the database. It has found 27 of them CLOSED and has used those to learn. What it learned is described in the section below.

- What the system learned

  You can see now that the solution you created before has a concept associated with it. In Figure 193 on page 186, we can see the following:

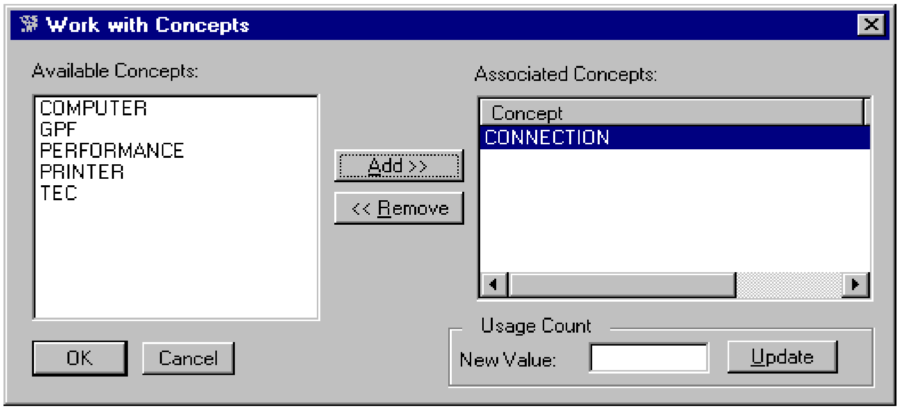


Figure 197. Concept Associated to the Solution after Learning

There is the CONNECTION concept associated to the solution we created before. This is what the system has learned. The system learned something from the activities done in the past. This knowledge will be useful in the future for a problem that can appear completely different from that just resolved. We see in the next section how this will be used as an aid.

Let's see what the system did to create such an association:

1. The system scanned the database looking into the CLOSED problems.
2. It found the problem that was resolved before using an existing solution.
3. In the problem description there is the word locate and this word is associated to the concept CONNECTION.
4. Because that problem has been closed with that solution the system has associated the concept CONNECTION with that solution.

Note that in order to reuse the new knowledge, the ADL must be initialized. To do this, from the Expert Advisor graphical interface choose **Diagnostic** in the action bar and then **Work with Adaptive Learning** and click on **Initialize** (see Figure 198 on page 188).

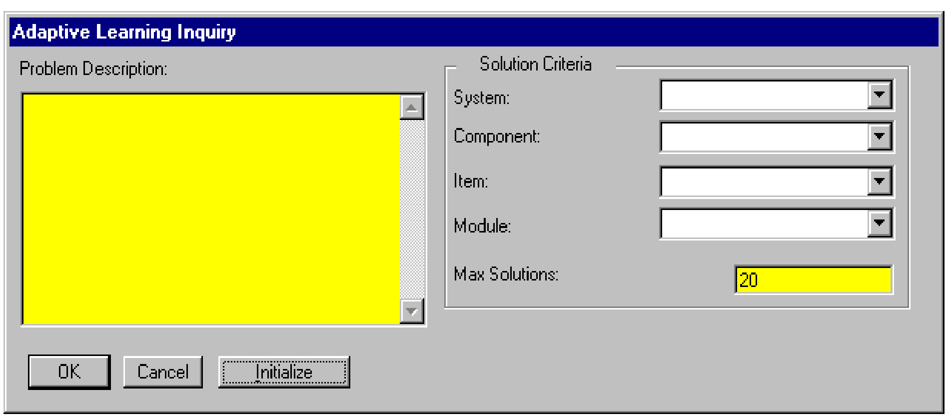


Figure 198. Initialize the ADL

The ADL should be initialized regularly to take advantage of the most current solution data. This operation updates the KEYWORD_MATRIX table, which stores all the thesaurus data. In two-tier environments the initialization occurs automatically when Expert Advisor is started. In three-tier environments the ADL does not need to be initialized because KEYWORD_MATRIX resides on the server and is updated on an hourly basis.

- Resolve a new call using Adaptive Learning

  Now we can resolve a new incoming call using the knowledge the system learned.

  For instance, if a call is being registered for a problem with Lotus Notes. The Lotus Notes application cannot find the server and returns the message "Unable to find path to server".

  In this example we chose a problem that could sound similar to the previous one; however, the general problems could look different from each other but have the same cause.

  This could be that a problem really looks different from another, or the way a caller describes the problem to the help desk person. It may not be clear what SCIM information the problem is related to.

  All we have is a vague description of the problem and we need to find a quick solution. It could be a solution previously created as a common problem or a Hot News or other.

  In our case, the call description is entered as shown in Figure 199 on page 189.

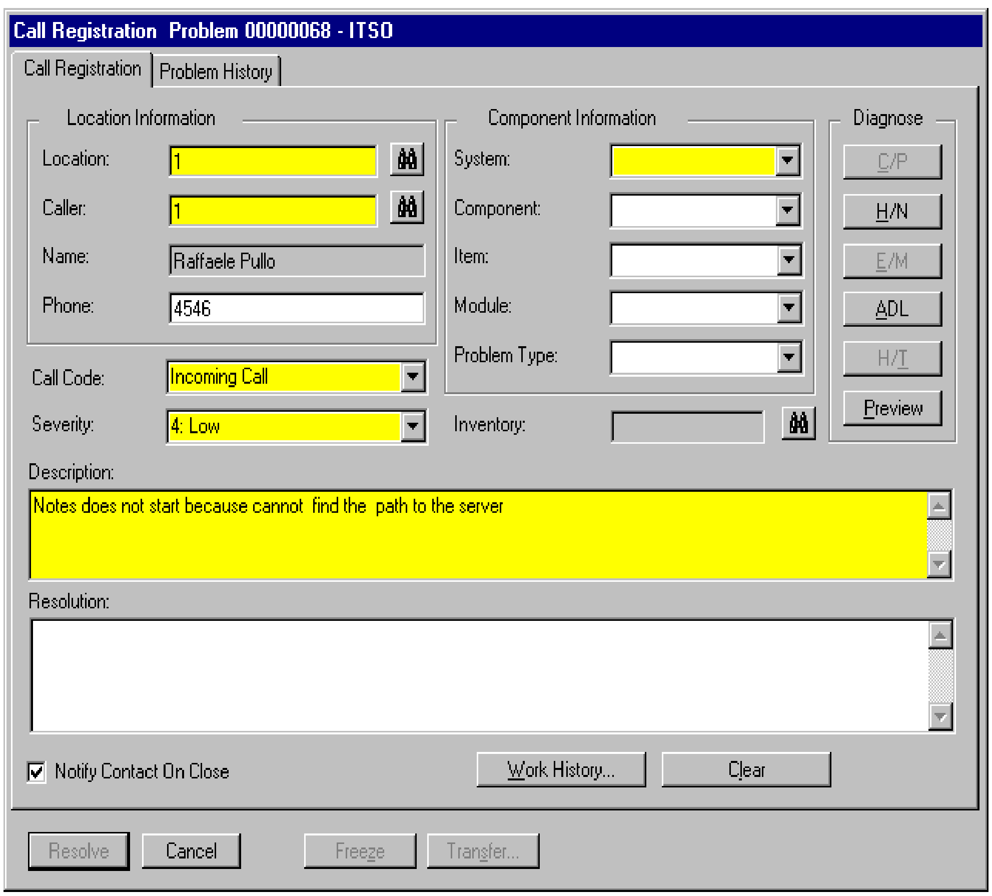


Figure 199. New Call Registration

The Description field entered is:

```
Notes does not start because cannot find the path to the server
```

Click on the **ADL** button to ask Adaptive Learning to search for a solution based on the problem description you entered. We can see that ADL finds the solution that was already used for the different problem with Netscape.

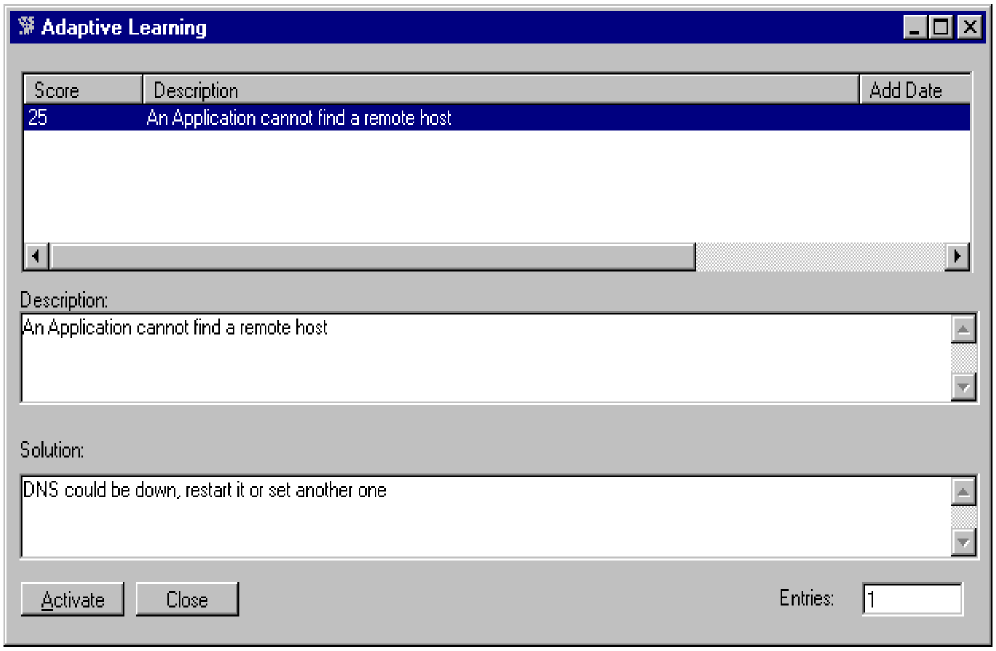


Figure 200. Solution List

You can use this solution immediately to resolve the problem. The system found the solution by:

- Finding the word path in the free text description.
- That word path exists in the thesaurus and is associated to the concept COMMUNICATION.
- The system looked for solutions associated with the concept CONNECTION.

After you used this solution the system learned something more. The fact that this solution has been used one more time increases its score for future problems.

5.5.6.2 Considerations

From the examples we have discovered what Adaptive Learning uses to learn, but there are some other considerations. ADL does not only learn from common problem solutions but from all the diagnostic sources, and it does not only learn when you ask it to learn like in our example. ADL learns in many ways from many sources and this is a tremendous capability to acquire knowledge.

What the users have to do is find new solutions and work with ADL as much as possible. In such a way ADL can learn more and more while the users do not perceive what new knowledge Adaptive Learning is acquiring and where, but they get the benefits from what the system learns.

The Adaptive Learning administrator has to create the correct thesaurus words and concepts. For directions about how to do this in the more efficient way refer to the manual *Tivoli Diagnostic Administrator Guide.*

Adaptive Learning, together with the other aids, makes the Expert Advisor diagnostic functionality very powerful and extremely useful in a help desk environment.

5.5.7 Solutions

In the Tivoli Advisor database there is a table called SOLUTIONS. This table contains all the information about how calls or problems were solved or how they should be solved.

The entry for one solution in the SOLUTIONS table is called a solutions record. A solutions record can be defined as:

- A common problem record
- An error message record
- A Hot News record
- A solution node in a HyperTree
- A record added through the Work with Solutions window
- A record added by an analyst in the Call Registration window

All the solution records have a usage count. This is a counter that tells us how many times a specific solution has been activated. By activated, we mean used as the solution to a problem.

A solution record has a status associated with it. This is either active or inactive. When a solution record is active, it can be used as a solution to a problem. When it is inactive, it can no longer be used.

5.5.7.1 Working with Solution Records

To view the Work with Solutions window, select **Diagnostics** followed by **Work with Solutions** from the Tivoli Advisor startup window.

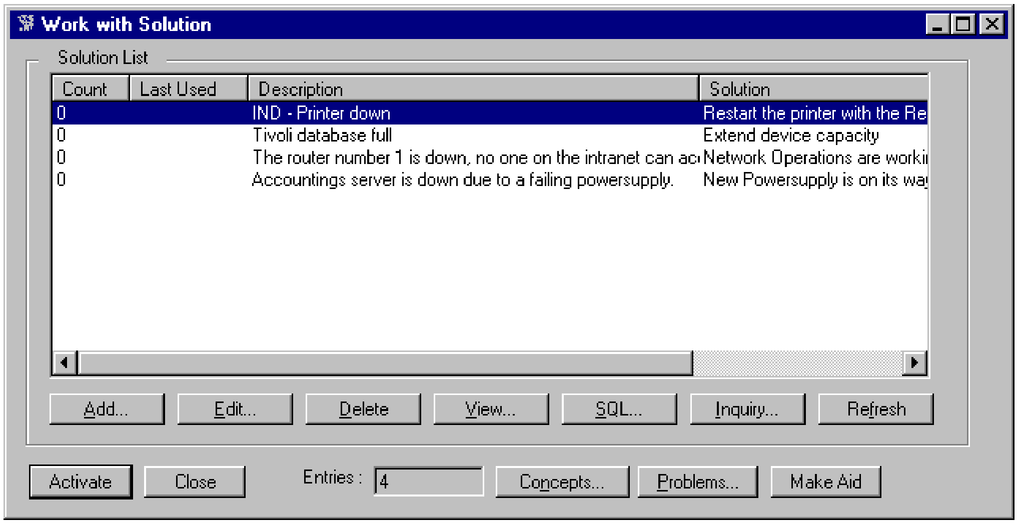


Figure 201. Work with Solutions

From the screen shown in Figure 201 on page 192 we can add, edit, delete or view a solution. There are also a couple of special features that are available here:

- Activate

  If an analyst is working on a call and finds a solution to the problem via the Work with Solutions window, he or she can transfer the data of that solution to the Call Registration window by clicking on **Activate**.

- Concepts

  If you select an event then click on **Concepts**, you can associate this solution with concepts. The Adaptive Learning diagnostic tool can then use these concepts to find solutions to a problem.

- Problems

  If you select an event and click on **Problems**, a list of all the problems where this solution has been used will appear.

- Make Aid

  This button is used to create a diagnostic aid record from a solution record. The different type of records that can be created are:

  - Common Problem record
  - Hot News record
  - Error Message record

  If you select an event and click on the **Make Aid...** button, you will get a window asking you what type of aid you want to create.

Figure 202. Choose Aid Window

When you have chosen what type of record you want to create, you will see the Add Common Problem, Add Hot News or the Add Error Message window depending on your previous choice.

5.5.7.2 Removing Solution Records

To remove a solution record you can do one of the following:

- Inactivate the record

  If you inactivate the record, it will become unavailable for the analyst. It will still be in the Work with Solutions window, but no one can use it to solve a problem.

- Delete the record

  If you delete a record, you remove it from the Work with Solutions window. The record will still be in the SOLUTIONS table in the database, but with a flag that says that this record has been deleted.

  If you want to delete a record, the usage count of that record has to be zero (0). If the record has been used to solve a problem, the usage count raises beyond zero and you cannot delete the record.

- Purge the record

  If you want to completely remove Solution records from the database, you have to run the Solution Purge utility. This is an application that is installed at the same time as Advisor.

  The Solution Purge utility purges records that are older than 30, 60, 90, 180 or 365 days depending on what is specified. You can also decide how many times a record must be used as a solution to a problem before it can be purged.

  The Ignore ADL option enables you to not remove those records that have been used by Adaptive Learning. You should run the Solution Purge utility periodically to keep down the number of solutions in the database.

5.5.8 Hypermedia Links

Hypermedia links are used to add information to some of the description and solution fields, for example Description: [<Alt=B> = Hypermedia Link].

This indicates that you can insert a link to some other source of information such as a sound file or picture. This is a great way to give the analyst more detailed information about how to solve a problem.

The different types of hypermedia links you can use are:

Search Search enables you to scroll to the line where the next instance of a word or string occurs.

View file The View file opens any flat file into Tivoli Advisor’s Hypermedia Viewer. The file is going to be in read-only mode. This Hypermedia Viewer is automatically installed with Advisor.

Display image This does the same thing as the View file except for the fact that the file now is a graphics file.

Play sound This function opens a sound file and plays it.

Run program This calls an executable file to start another application.

The windows that you can add hypermedia links to are the following:

- Add Common Problem window
  - Description box
  - Solution box
- Edit Common Problem Window
  - Description box
  - Solution box
- Add Hot News window
  - Description box
  - Solution box
- Edit Hot News window
  - Description box
  - Solution box
- Add Error Message window
  - Description box
  - Solution box
- Edit Error Message window
  - Description box
  - Solution box
- HyperTree Edit window
  - Description box
  - Solution box
  - Settings box
  - Question box

When you press Alt+B the screen shown in Figure 203 on page 195 appears.

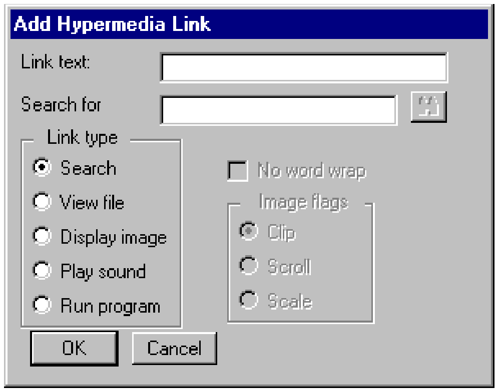


Figure 203. Add Hypermedia Link Window

Here specify the text to be linked. Depending on what link type you choose the box name of the box just below the Link text box changes. This box is the argument to the link text. If it is a Search link, as in this case, the Search for box will appear. If it is a View file type, the box will be named File as in Figure 204 on page 195.

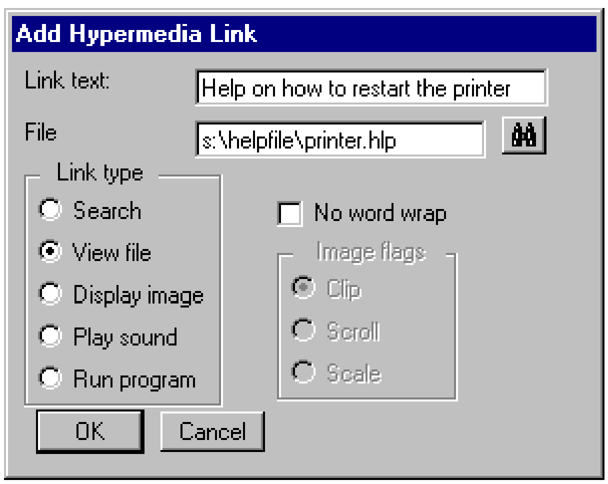


Figure 204. Add a Hypermedia - Link Window

This link would appear in the Edit window as shown below:

```
[[ Help on how to restart the printer|FILE s:\helpfile\printer.hlp ]]
```

If the link is not viewed in the Edit window, it will only display the Link text, but this will be highlighted to indicate that it is a hypermedia link.

5.6 Adding New Users to the Tivoli Service Desk Applications

Before a new user is added to the service desk they must have the correct rights database. Therefore, the first task is to add the sybase users and grant them access to the ADVISOR database as described in Chapter 4.3.6, “Changing the Owner of the ADVISOR Database and Creating Users” on page 62.

We installed the applications with the option **Use Same Logon for DB and Application**. We added a new Tivoli Advisor user with the same user ID and password as used for the sybase user.

Add the user through the Tivoli Advisor dialog box as shown in Figure 205 on page 196.

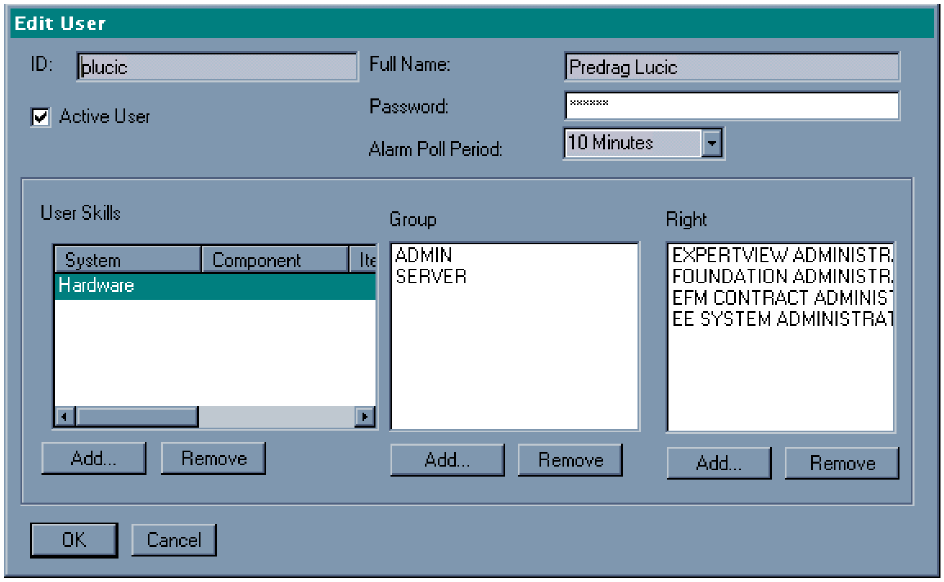


Figure 205. Adding Tivoli Advisor User

The system administrator called EXAV then runs the eagrant script through the ESMBuild window to allow the Tivoli Advisor user to access the ADVISOR database.

This script is located in the directory \SAI\esmbuild\ea. The eagrant script must be run for every new Advisor user added to the service desk.

Database grant scripts should be also run for granting other Tivoli Service Desk application-specific rights to users. These scripts are provided together with Tivoli Service Desk applications as shown in Table 16 on page 196.

Table 16. Scripts for Granting Rights to Users

Script name	Tivoli Service Desk Application
eagrant.*	Tivoli Advisor
evgrant.*	Tivoli NSM Gateway
i_grant.*	Tivoli Asset Management
eegrant.*	Tivoli Change Management

For example, to grant users the rights to work with Tivoli Asset Management tables in the ADVISOR Sybase database we would execute the script i_grant.syb from the ESMBuild window.

This grant script is provided together with the Tivoli Asset Management application and is located in \SAI\esmbuild\efm.

We added the users as shown in Figure 206 on page 197 to set them as members of the ADMIN group.

The user plucic will also be given rights to Tivoli Asset Management, Tivoli Change Management and the Tivoli NSM Gateway.

The users’ rights are associated with the ADMIN group definition. We added additional rights to this user ID as shown in Figure 205 on page 196.

User plucic and juergen are also second-level users so we associate user plucic to the SERVER group and hardware-related problems and user juergen to the NETWORK group and communication-related problems.

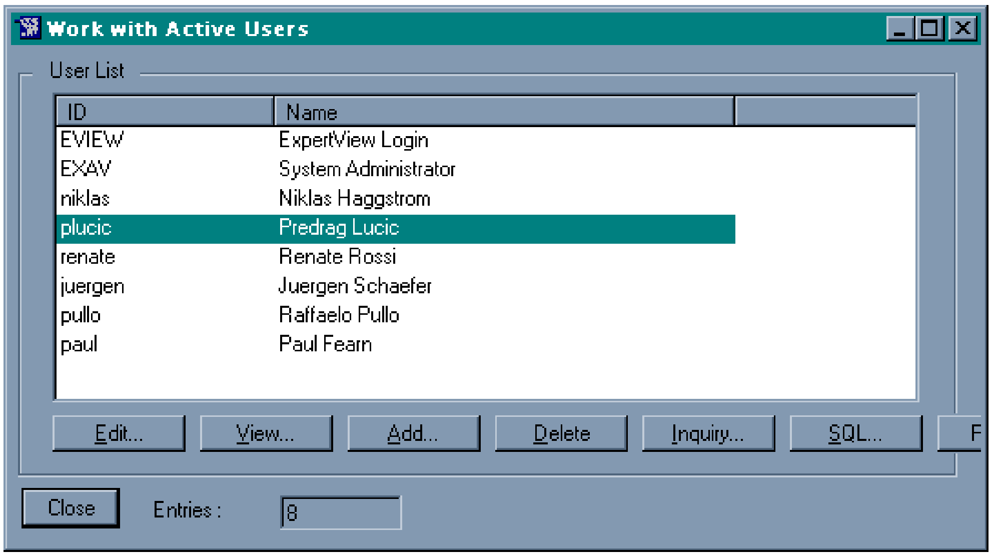


Figure 206. Tivoli Advisor Users

At this point we are ready to associate Tivoli Adviser user plucic to a person in Tivoli Asset Management. Open the Tivoli Asset Management window and associate a person with the user as shown in Figure 207 on page 198.

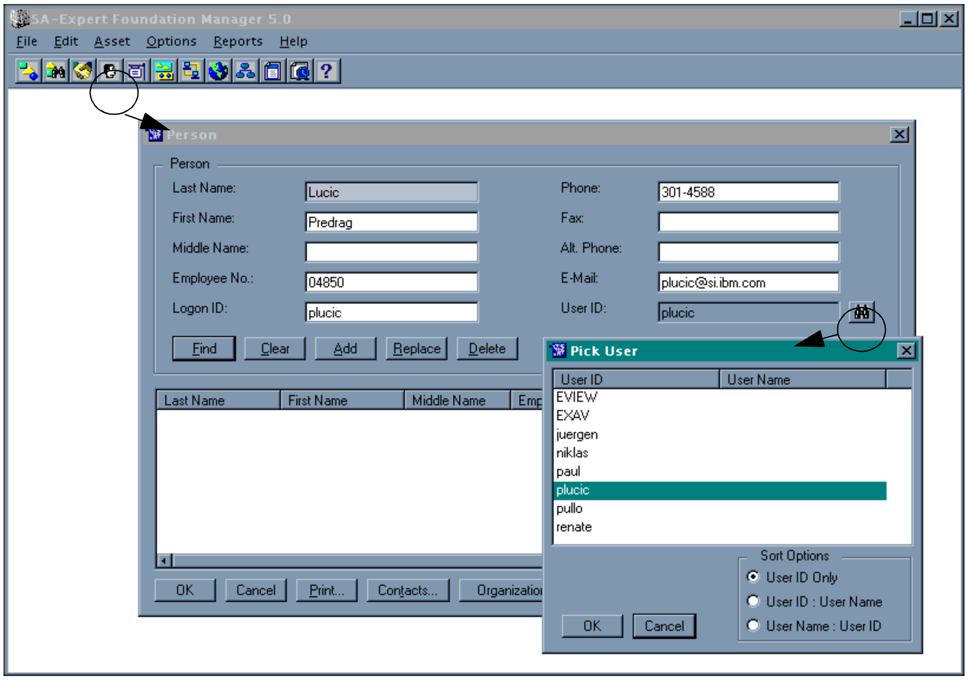


Figure 207. Mapping the User or Person

Select the **Asset** icon, select the user ID plucic and finally click on **OK** (see Figure 207 on page 198).

5.7 EV Gateway Configuration

This section explains the parameters that can be configured for the gateway. All the parameters are set from the gateway screen shown in Figure 208 on page 199.

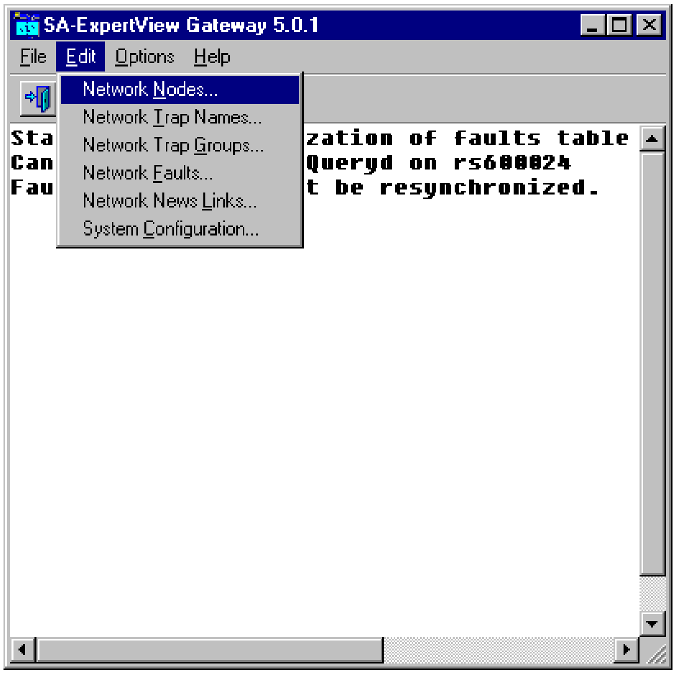


Figure 208. ExpertView Gateway Commands

From here we can configure the following:

- General information
- How ExpertView opens and closes problems
- Callbacks

To configure the system options select **Edit** followed by **System Configuration**. Figure 209 on page 199 will appear.

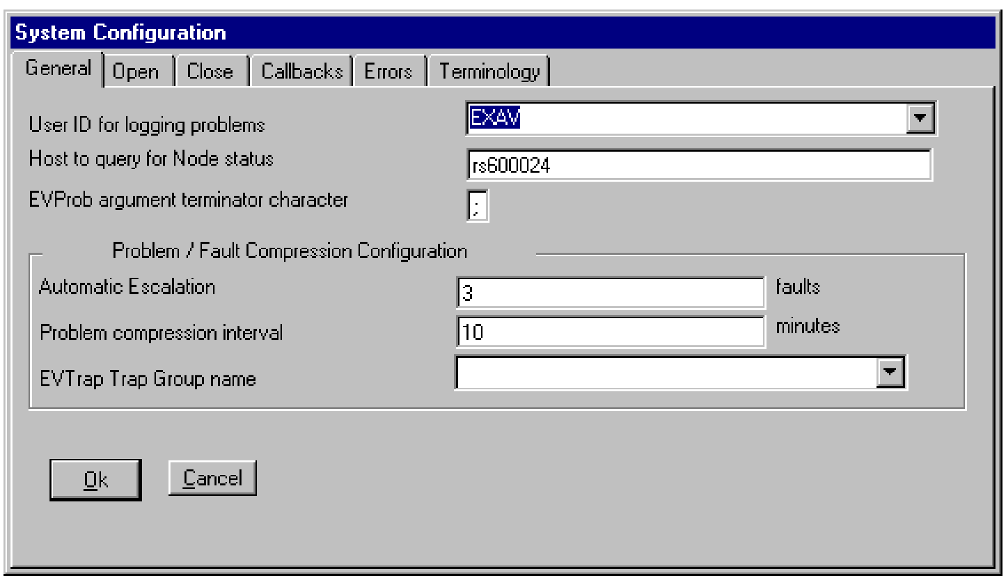


Figure 209. System Configuration Options

The user ID to send problems is EXAV. The host is the hostname where the NSM will run. In our case this was rs600024. The EVProb argument separator is set to a colon. This is used when executing the EVProb command.

The compression configuration is used for escalation purposes. ExpertView will only escalate a problem once. If a value is entered in the Automatic Escalation box, then when a device fails this number of times it will activate escalation.

The problem compression interval causes the gateway to re-open a ticket when a similar problem occurs, that is, re-open a problem that was CLOSED. The compressed node failures are added to a single problem in the Advisor.

To set the options for the OPENING of a ticket, click on the **Open** tab (see Figure 210 on page 200).

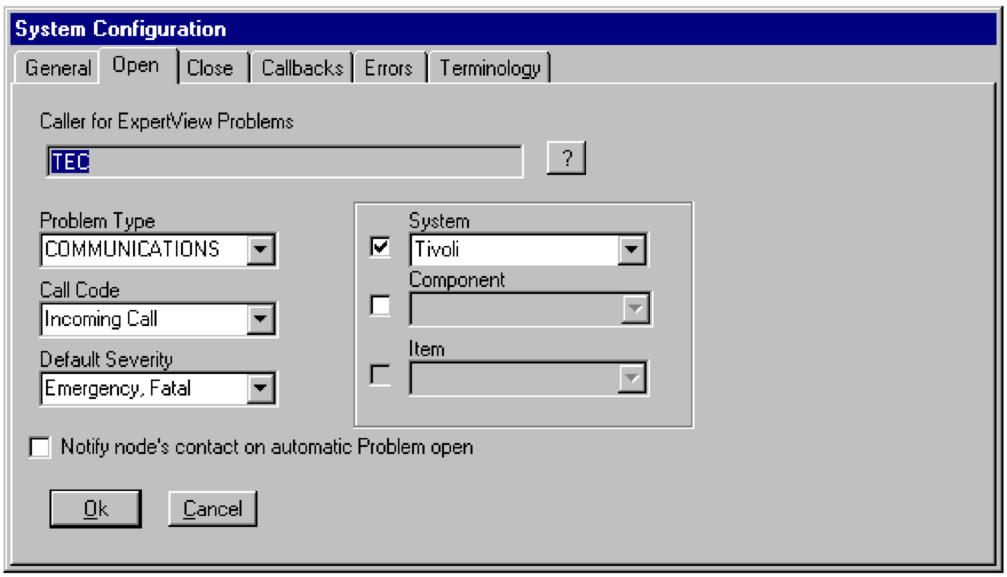


Figure 210. System Configuration - Open Options

The default SCIM parameters can be set here. However we create the SCIM options directly using the EVProb command. The default call code and severity values are set here. To configure the error options click on the **Error** tab (see Figure 211 on page 201).

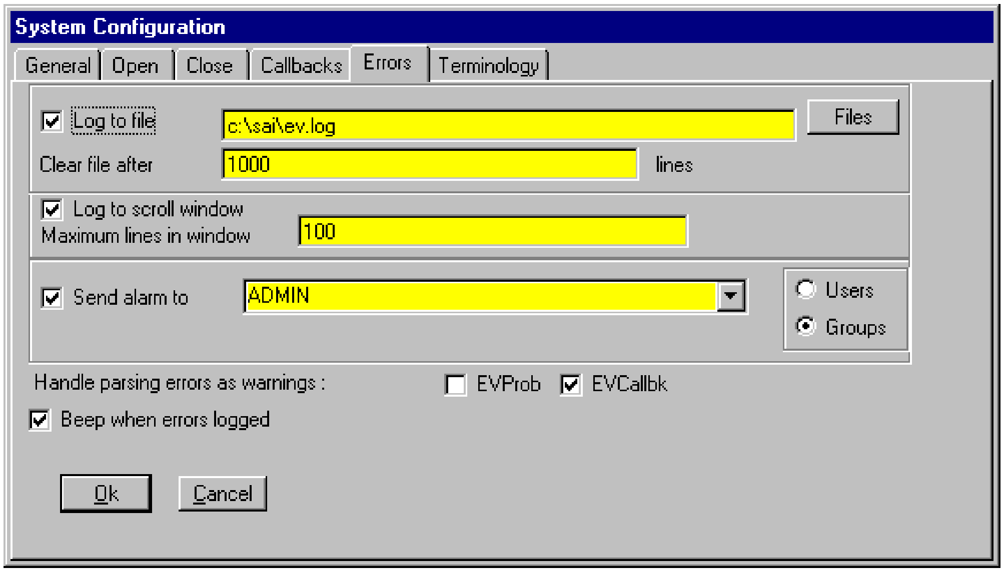


Figure 211. System Configuration - Error Options

Here we set the various error handling options for the gateway.

Next we look at notifications.

5.8 Notifications

Notifications can be defined from the ExpertView gateway. This can be useful when the TEC generates trouble tickets and the Advisor user can be notified. This is activated as follows:

From Figure 208 on page 199 select **Edit** followed by **Network Nodes** (see Figure 212 on page 202).

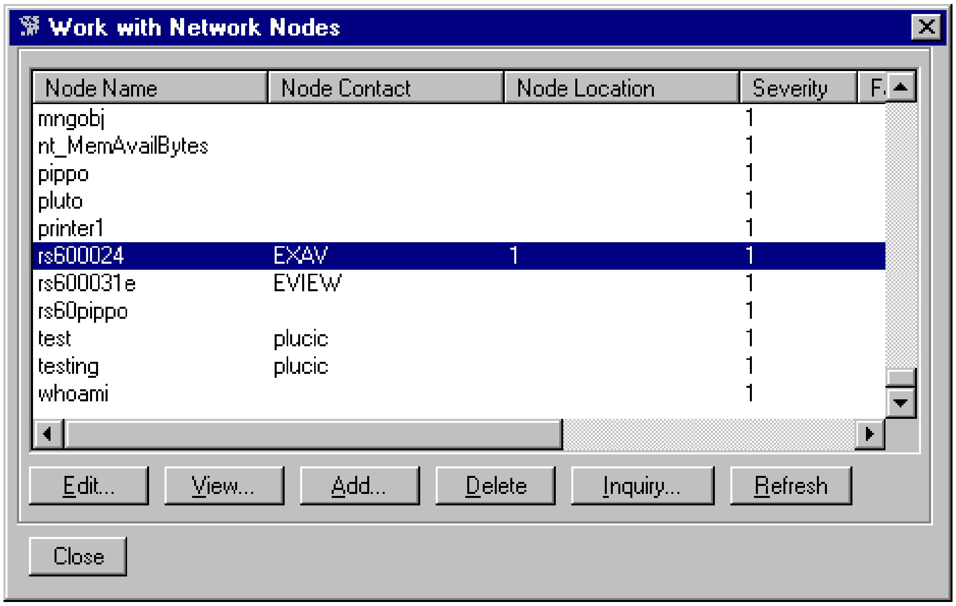


Figure 212. Network Node Options

Select the node, in our case rs600024 and click on **Edit**.

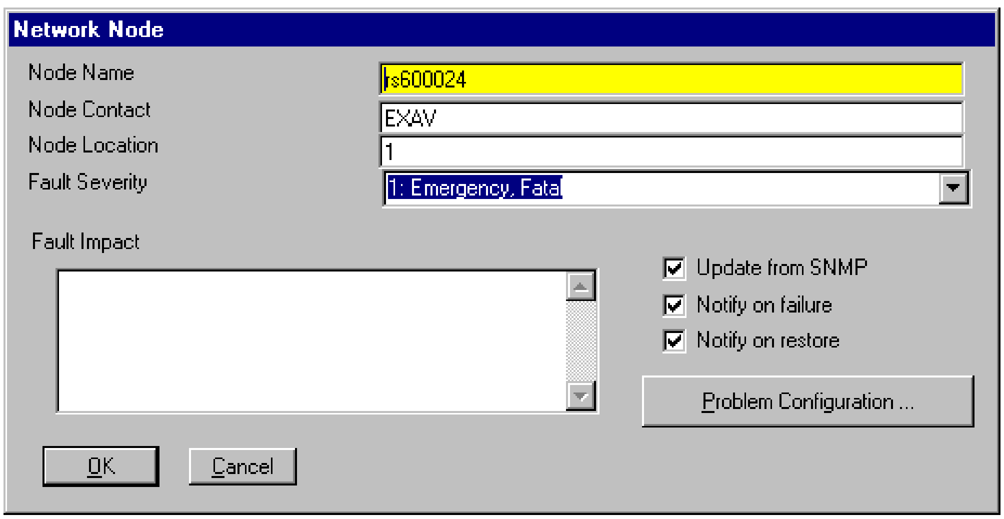


Figure 213. Network Node Configuration

Set the options to Notify on Failure and Notify on restore.

5.9 Tivoli Asset Management Configuration

At this point we do not have any entry in the Tivoli Asset Management database. There are a few methods to populate this database as described in Chapter 3.1.10, “Tivoli Asset Management” on page 30. We installed the Tivoli Inventory integration module to link the Tivoli Inventory database with entries in the Tivoli Asset Management database.

5.9.1 Installation of the Tivoli Inventory Integration Module

The Tivoli Inventory integration installation process consists of steps that have to be performed before the Tivoli Inventory data can be displayed through the Tivoli Service Desk application. These steps are described below for our environment shown on Figure 218 on page 231.

We used the Tivoli Inventory Integration Module 5.0.1 which does not have support for sybase as the Tivoli Inventory repository, so we installed Oracle on machine rs600031e. We also installed the Oracle client on machine wtr05368, because the Tivoli Advisor file server must have an SQL connection to the Tivoli Inventory database.

Every machine that would need access to the Tivoli Inventory data through the Tivoli Service Desk application must have an SQL connection to the Tivoli Inventory database. In our case this means that we install the Oracle client on the file server and stand-alone workstations, as well as on client workstations.

5.9.1.1 Applying the File Pack

The file pack has to be applied on all machines that will use the module. This is the case for file servers and stand-alone workstations, but not for client workstations. We applied the file pack from the Tivoli/Inventory Module 5.0.1 CD to the directories /sai/efm and /sai/esmbuild/efm on the Tivoli Advisor file server machine wtr05368.

5.9.1.2 Define a TIVOLI Data Source

We created a TIVOLI data source on the Tivoli Advisor file server to establish database connectivity with a Tivoli Inventory database. We define this through the ASE SQL Configuration dialog as shown in Figure 214 on page 203.

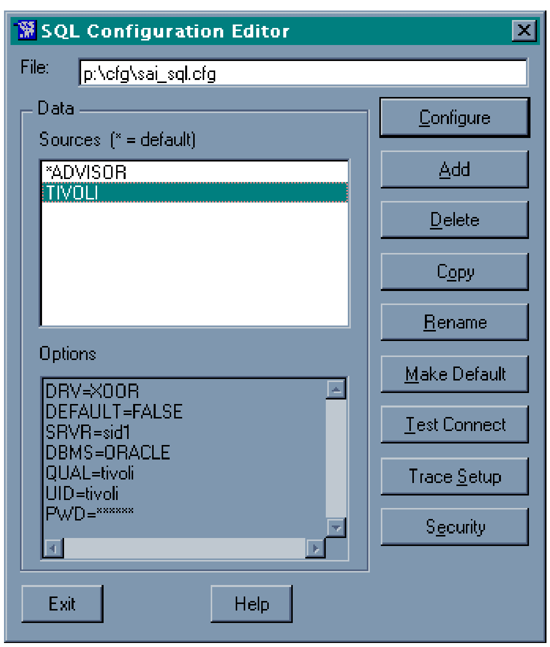


Figure 214. Tivoli Inventory Data Source

5.9.1.3 Modify the Tivoli Advisor and Tivoli Inventory Database

We modified the Tivoli Advisor database with the script that is installed as part of the Tivoli Inventory integration module. We ran this script through the ESMBuild dialogs. The name of the script depends on the database used for Tivoli Advisor. In our case we use the i_tivsysb.tab script as shown in Figure 215 on page 204.

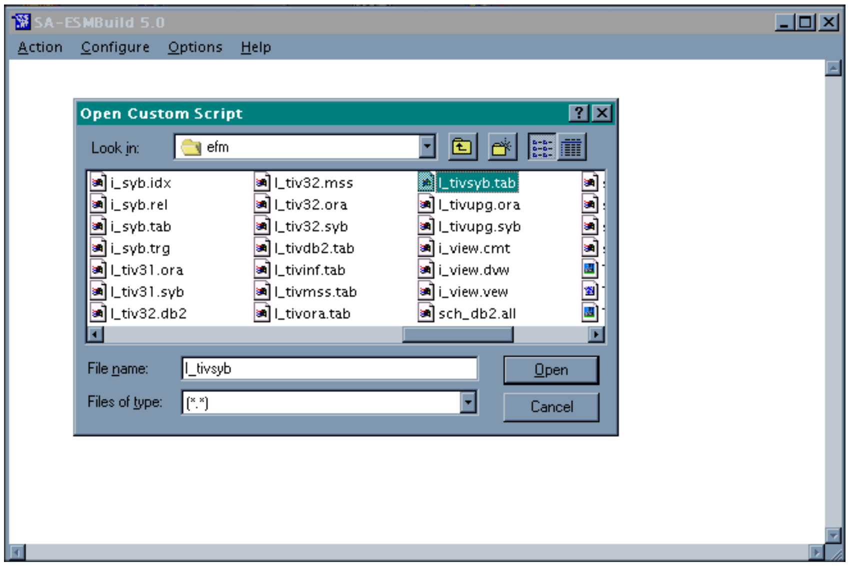


Figure 215. Modifying the Tivoli Advisor Sybase Database

After completion of this script we did the same with the Tivoli Inventory database. We executed the script, which added new views required for the integration. These views do not modify any of the existing Tivoli Inventory database tables.

Before running this script we configure ESMBuild to use the Tivoli Inventory database as shown in Figure 216 on page 204.

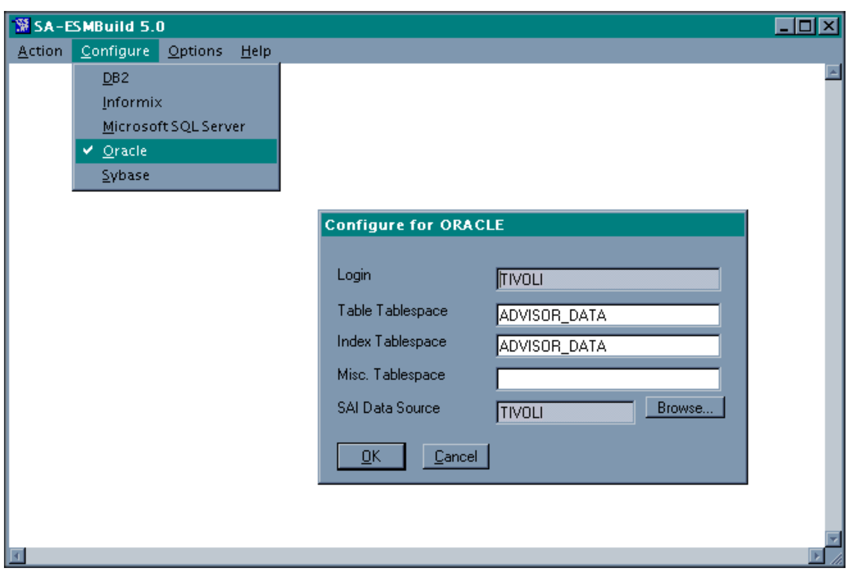


Figure 216. ESMBuild Configuration for Oracle

We then executed the script i_tiv32.ora which is used to modify the Oracle database with Tivoli Inventory Version 3.2. We have Tivoli Inventory Version 3.6, but this script can also be used for this version. With these two procedures the databases are ready for migration.

5.9.1.4 Parse the Dialog Files and the Tivoli Advisor Application

When we installed the Tivoli Inventory Integration file pack there is a lot of new dialog files that change the Tivoli Asset Management dialogs and add buttons for the Tivoli Inventory attributes. We parse these dialog files using the ASE Interface Generator as shown in Figure 217 on page 205.

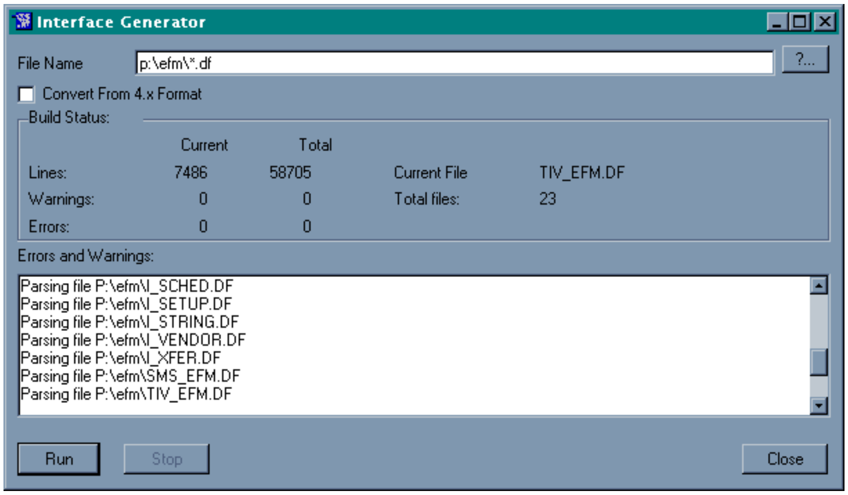


Figure 217. Parsing the Dialog Files

After this we parsed the Tivoli Advisor application.

5.9.1.5 Migrating the Asset Records

The migration process establishes a link between the asset records in the Tivoli Advisor database and the records in Tivoli Inventory database. We started the migration by opening the Tivoli Asset Management Migration dialog box from Tivoli Asset Management folder.

From the **Actions** menu we choose **TME 10 Inventory** and fill in the selection fields in the TME 10 Inventory Migration dialog box.

By selecting **Find** we now can retrieve all the Tivoli Inventory records that match the selection criteria shown in Figure 218 on page 206.

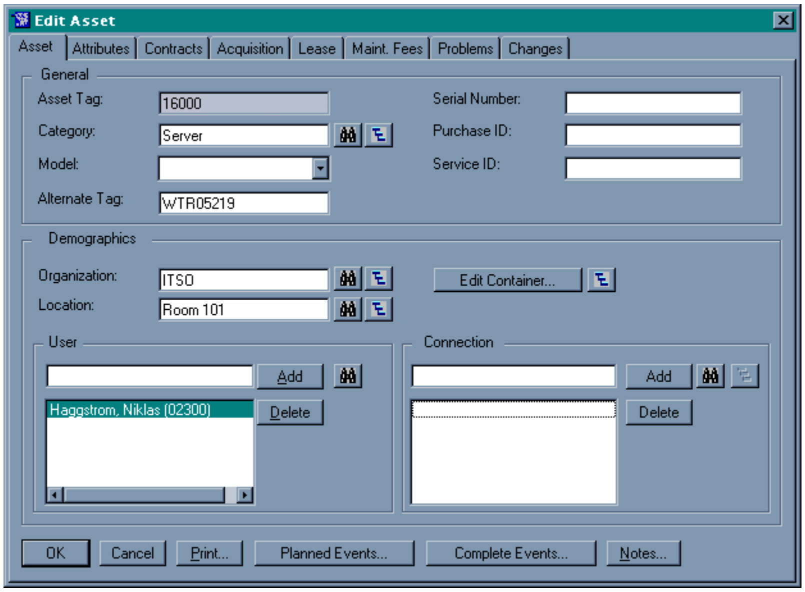


Figure 218. Tivoli Inventory Migration Selection Criteria

For mapping we defined the primary asset tag to be equal to the Tivoli object ID and for alternate asset tag we define the TME object label.

Note

Using the Tivoli object ID for the asset tag can be very confusing. You can use your own tag labels here, but then you have to customize scripts for integration between TEC and Service Desk products to map the Tivoli object ID of the affected machine to its asset tag.

We then started to migrate all the selected items. To check migration statistics we set up the log file name and location in the setup menu of the Tivoli Asset Management dialog box.

We choose not to migrate people and location records, because we do not have any data associated with these tables. With some Tivoli Inventory customization this data can also be migrated. After migration, the Tivoli Asset Management dialog box for assets looks as shown on Figure 219 on page 207.

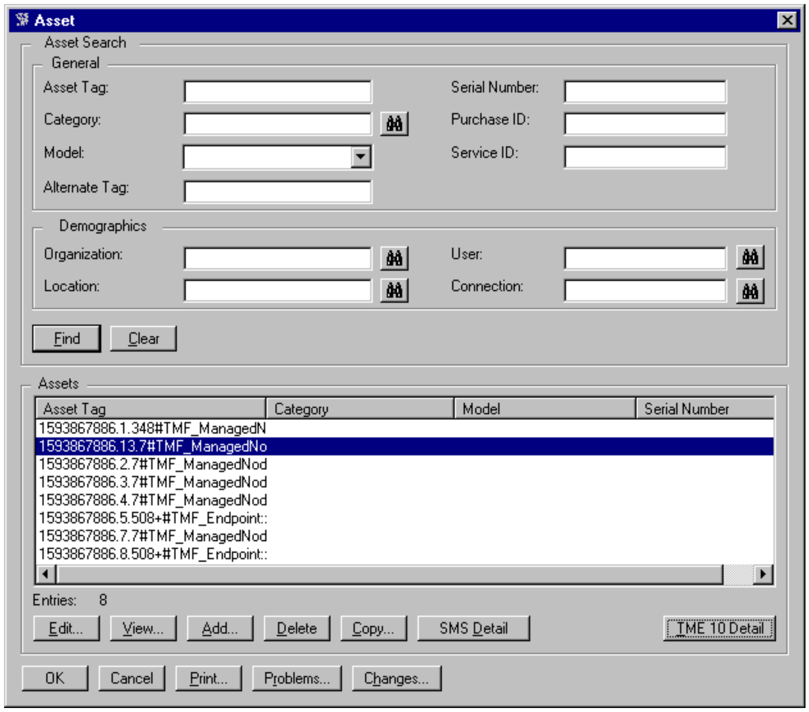


Figure 219. Display of Migrated Assets

By clicking the **TME 10 Detail** button we can see an extended data view of a selected asset as shown in Figure 220 on page 207.

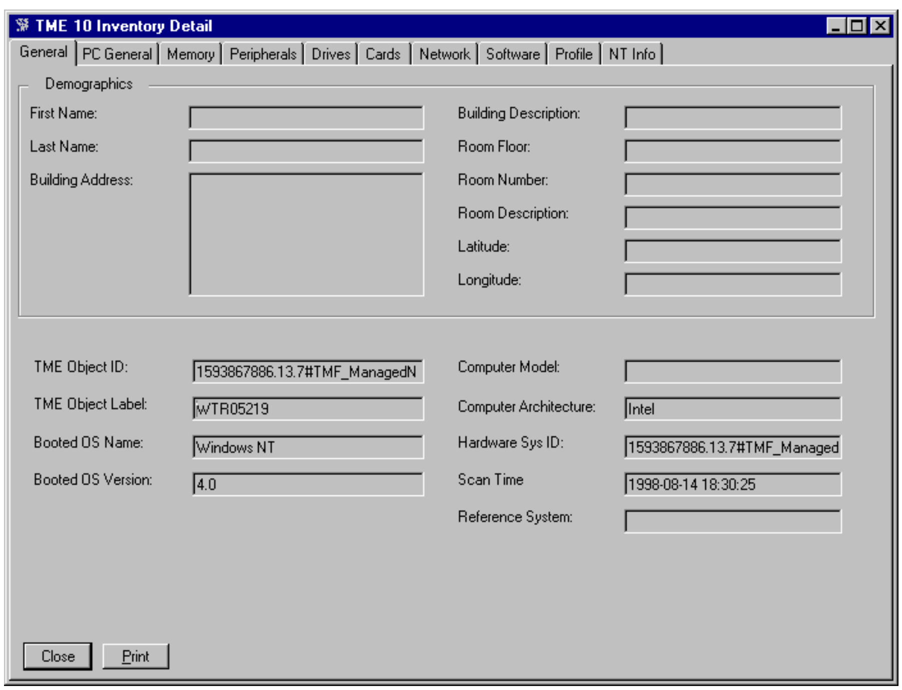


Figure 220. Tivoli Inventory View of Selected Asset

This view shows only the attributes that are predefined by scripts that come with the Tivoli Inventory Integration Module. With some customization, additional data can be populated by the migration utility and shown also in the Attributes window of Tivoli Asset Management.

5.9.2 Adding Tivoli Asset Management Attributes

The migrated assets do not have any additional attributes associated to them. For efficient use of Tivoli Asset Management at least the category and owner of this asset should be populated in the appropriate asset field. This can be customized to be automatically populated with the migration process, but we choose to do this manually as shown in Figure 221 on page 208.

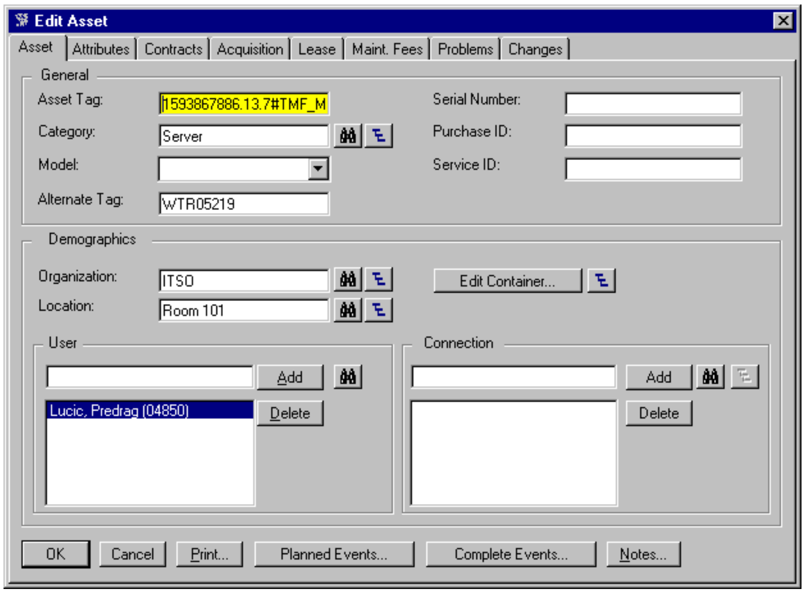


Figure 221. Adding General Attributes to Asset

We added all the managed nodes to the category of Server and all endpoints to the category of Desktop. For both categories we entered values to the attribute fields defined for each category separately as shown in Figure 227 on page 238. We enter these values manually because there is no default mapping of Tivoli Inventory data to these fields.

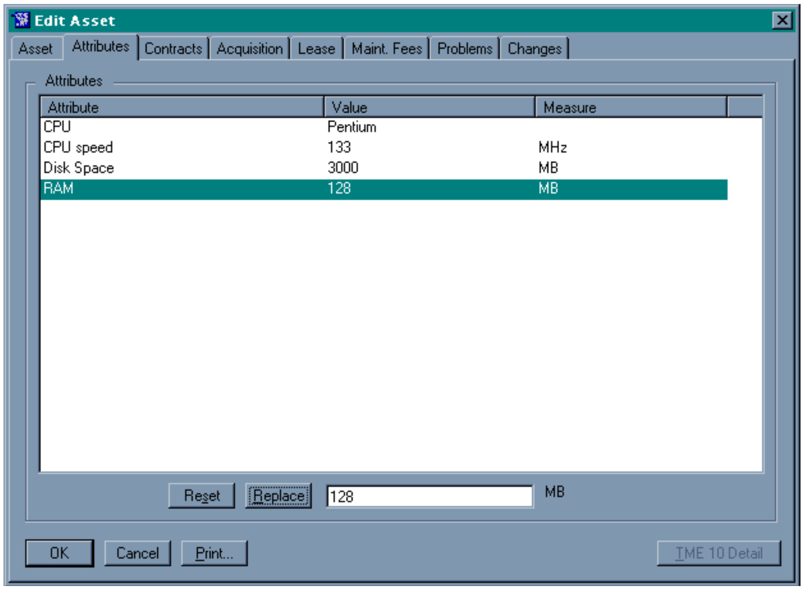


Figure 222. Edit Assets

We also associate assets with the user and defined location and organization attributes. Location and organization attributes are linked with categories, connections and container attributes as part of the data manager hierarchies.

These provide methods for identifying and tracking the asset. For more information about data managers and their definitions you can refer to *SA-Expert Foundation Manager 5.0 System Administrator's Guide* or *SA-Expert Foundation Manager 5.0 User's Guide.*

5.10 Tivoli Change Management Configuration

The change management application requires information for the following areas:

- People
- Resources
- Processes

This section describes the steps necessary to complete a change using Tivoli Change Management. We also show what we need to define for the change management process for our lab environment.

5.10.1 Change Request

The requester initiates a new change by assigning it a name, category and status. If there is a model for the selected category, additional information is added to the change automatically. The model is a template for changes that occur often or for changes with common data elements.

Here we define a model for change in the category Upgrade HW as shown in Figure 223 on page 210. Change categories provide classification of changes and are added in the Change Category Manager. A detailed description of the Change Category Manager can be found in *SA-Expert Evolution Manager 5.0 System Administrator's Guide* or *SA-Expert Evolution Manager 5.0 User's Guide.*

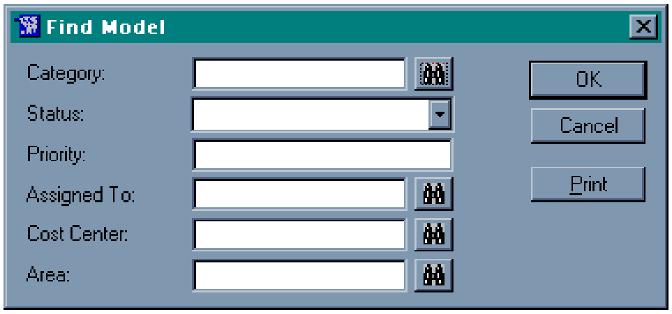


Figure 223. Model for Changes in Category Upgrade HW

Select Category by clicking on the icon. The Change Category Manager window will appear as shown in Figure 224 on page 210.

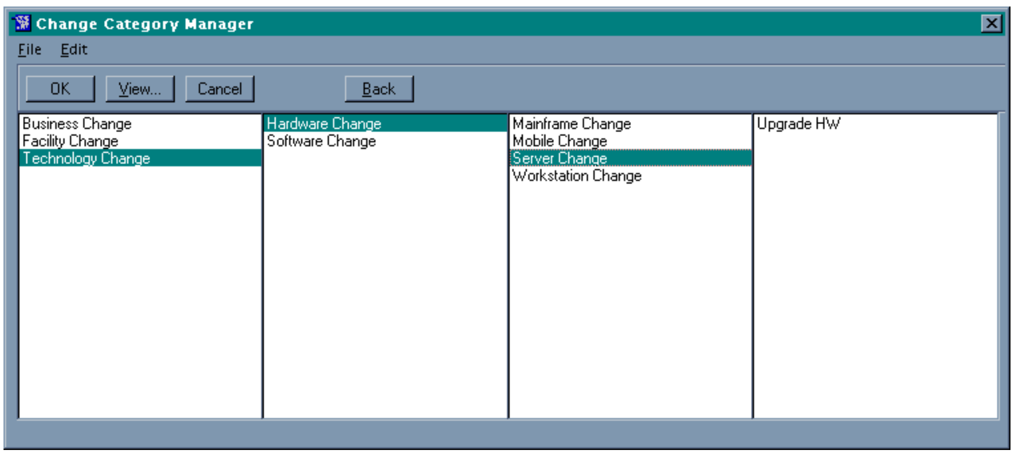


Figure 224. Change Category Manager

Select **Technology Change/Hardware Change/Server Change** followed by **OK**.

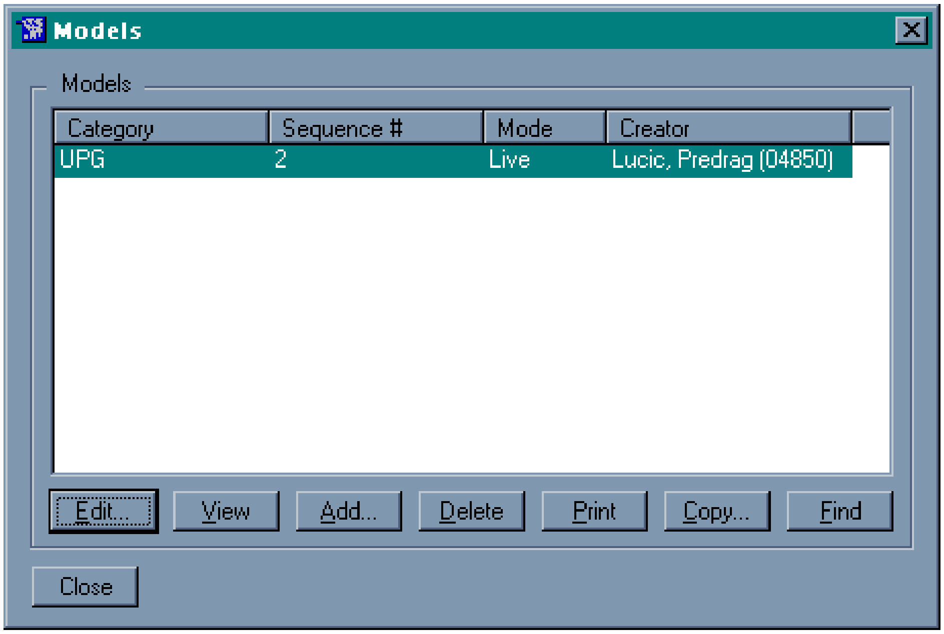


Figure 225. Models

This model will be assigned to a person who will implement this change as shown in Figure 226 on page 211. For example, he or she is an ITSO employee, therefore responsible for implementing this change is the ITSO organization. The cost center is ITSO organization, which is responsible for the costs of changes created from this model.

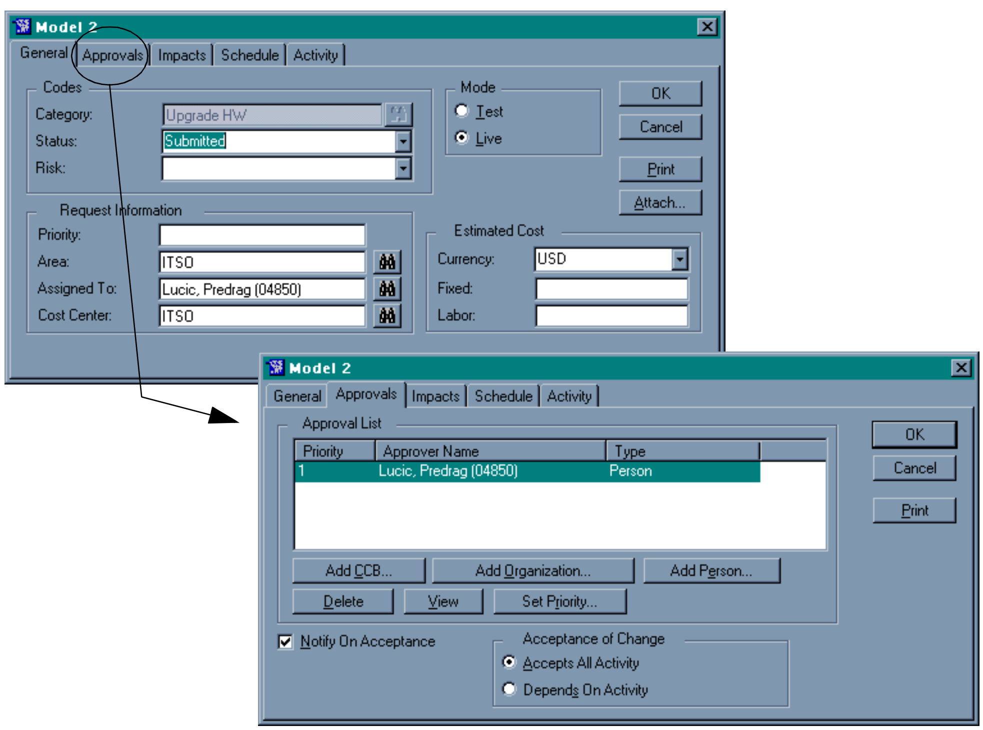


Figure 226. Change Model Definition

5.10.2 Change Submission

After a change is defined, the change requester submits the change. All approvers for the change are notified about the new change. When a change is submitted or when the approver accepts the change, all the time-independent rules, if relevant, are activated against the change.

The rules allow the facility to embed a unique business practice into the change management process. We define a rule that triggers when the cost of a change exceeds $50.00 USD as shown in Figure 227 on page 212. If this condition is met, we define the following actions:

- Change status of risk code to HIGH
- Notify a user
- Add an additional approver

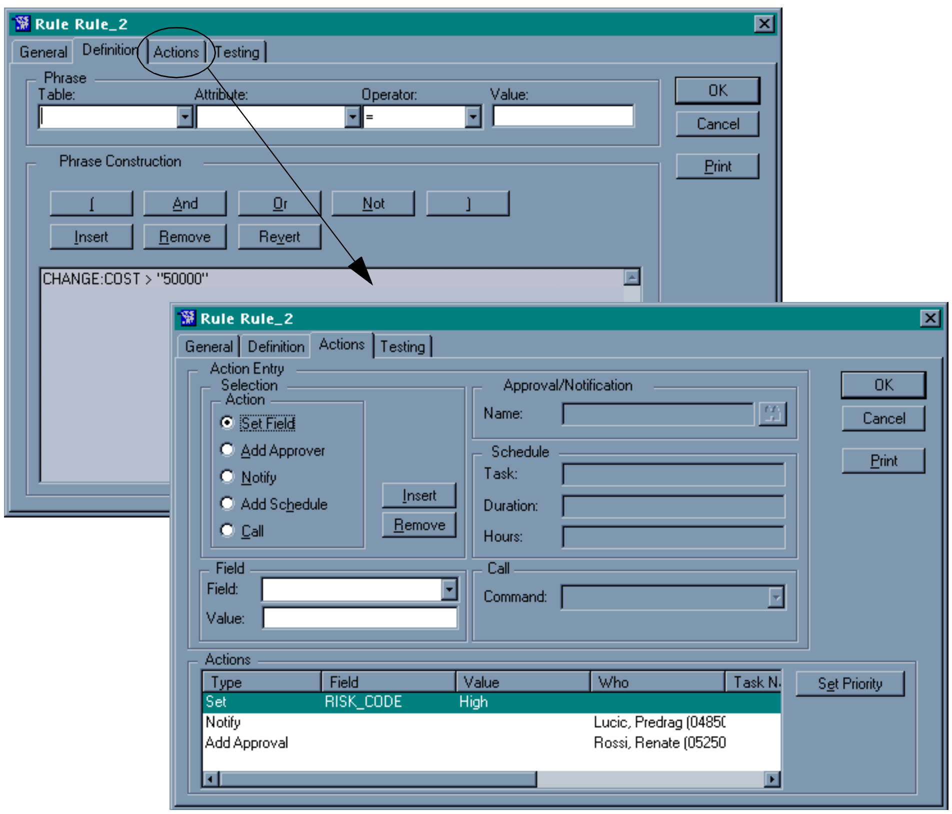


Figure 227. Rule Definitions

- Change approval

  Approvers review the change and decide to accept, reject or hold the change. Depending on this decision, status of change becomes accepted, rejected or held. Persons involved in this change are notified about the new status.

- Change implementation

  Affected personnel are notified of scheduled tasks.

- Change competition

  When all the tasks are complete, the change status may be altered to a completed status. When the change status is completed, all Tivoli Advisor assets associated with the change and the problem status codes of any attached problems are updated. The status codes of updated problems change to the value defined in the Tivoli Change Management application through the **Applications** menu. We define this to CLOSED as shown in Figure 228 on page 213.

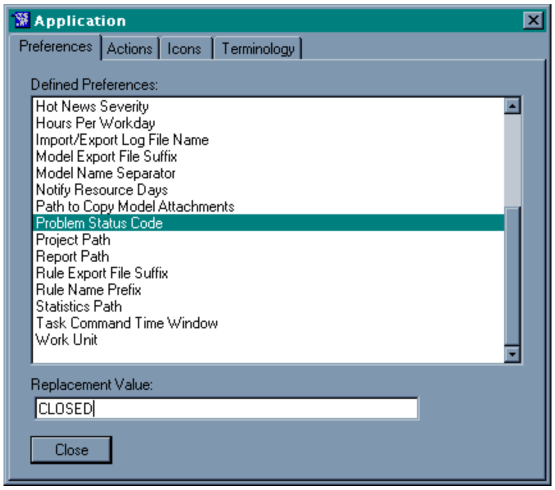


Figure 228. Definition of Problem Status Code after Change Is Completed

5.11 Tivoli Decision Support

This section shows using examples of some of the features of the new Tivoli decision support application. Here we used a pre-release of the Tivoli product so some screens may differ slightly from the GA code.

The TDS has the ability of integrate a problem management environment with existing systems and network management tools for reporting functions.

5.11.1 Administrator Setup

Before using the TDS we have to perform some tasks from the administrator's workstation.

As described in Chapter 4.5.3, "TDS Definitions" on page 50, the administrator's workstation will perform all the tasks related to cube definitions, do the connection to the Problem Management database and set the general options for TDS users and views. The cube definitions will be passed as input to the transformer module, and finally compiled into a compressed format (.MDC file).

We perform the following tasks:

- Configure ODBC drivers.
- Add decision support guides.
- Add and connect a data source.
- Set general options in the administrator module.
- Set specific parameters for building cubes.
- Schedule periodic automatic builds of cubes.

- Create new cubes and customize existing ones.

5.11.1.1 Configuring the ODBC Drivers

The initial prerequisite required is the 32-bit ODBC driver.

From the Windows Control Panel select the **ODBC** icon and ODBC Data Source Administration will allow you to add a new data source (see Figure 229 on page 214).

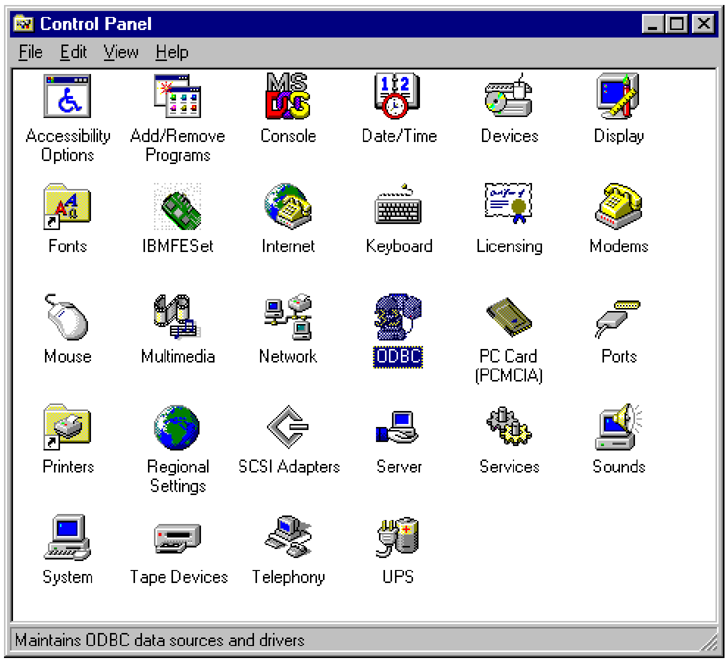


Figure 229. Starting the ODBC Configuration

Select the **INTERSOLV** driver then click on **Finish**.

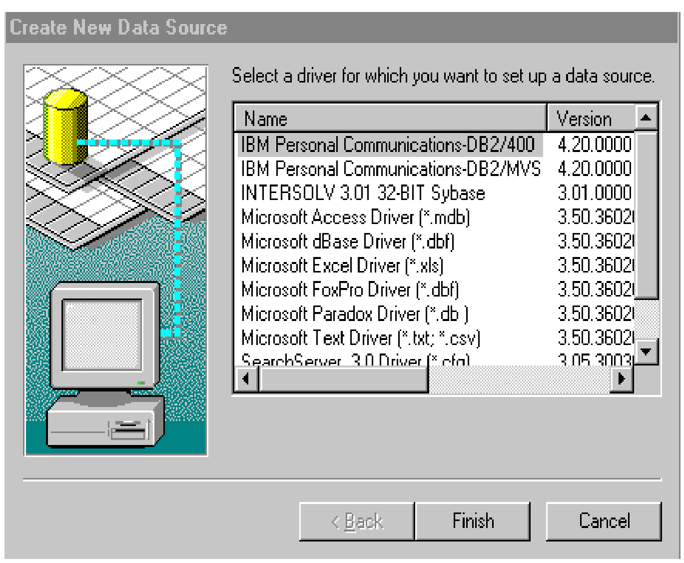


Figure 230. Adding a New ODBC Data Source (DSN)

When prompted for the Sybase server name, as contained in the $DSQUERY variable, we enter the database name or instance, which in our environment is ADVISOR. This is the problem management database name.

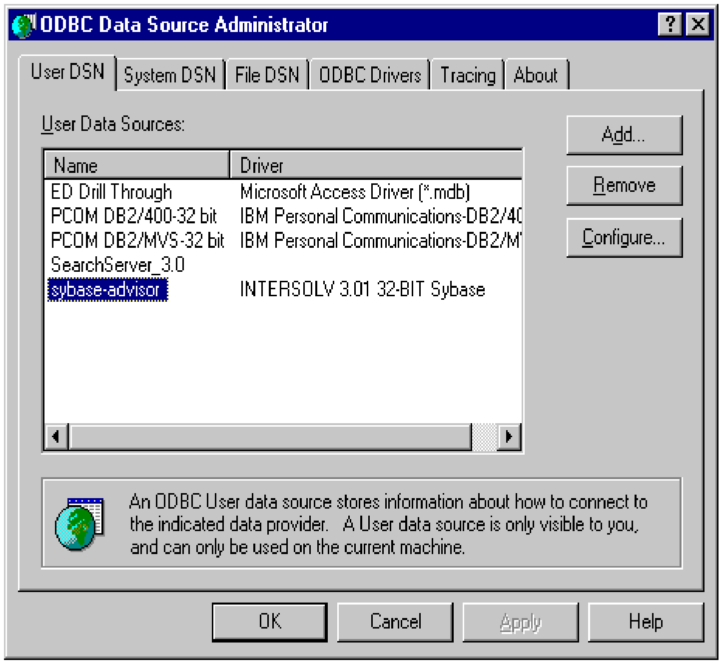


Figure 231. ODBC Sybase Driver Setup

For the general option fields we entered `rs600024` as the database server (see Figure 231 on page 215).

By selecting the **Use DNS** tab we can see the sybase-advisor information (see Figure 232 on page 216).

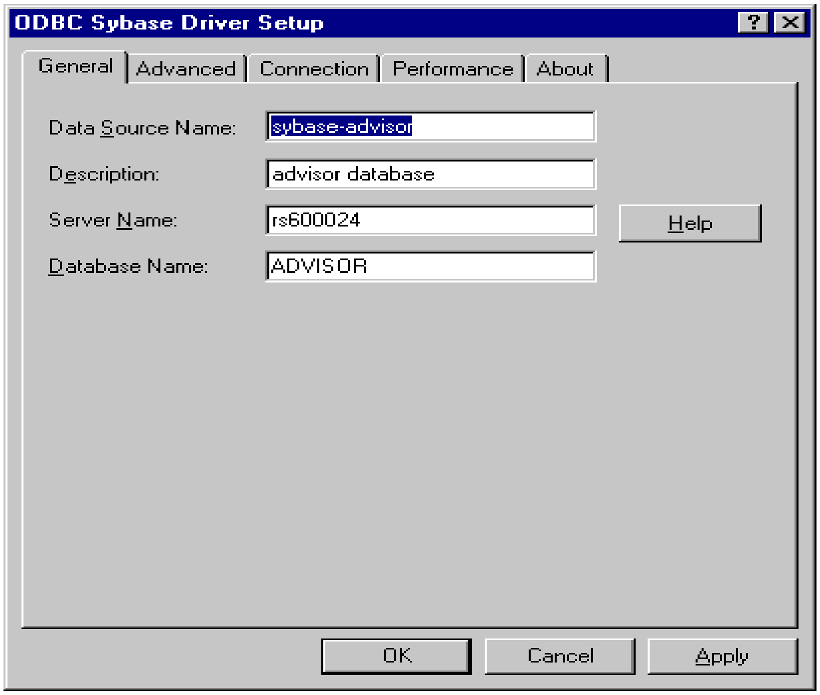


Figure 232. ODBC Driver Setup

5.11.1.2 Add Decision Support Guides

We launch the Discovery Administrator interface from our NT workstation by selecting **Start -> Programs -> Tivoli Decision Support -> Tivoli Discovery Administrator**.

The first time we used the administrator we were asked to import one installed discovery guide. Here we selected **Yes**.

Figure 233. First Time We Launch TDS Interface

TDS guides need to be linked to our environment. These guides contain pre-built algorithms, queries, reports and views. No customization is needed but to link them to our live data.

At this stage we decided on what kind of prospective we wanted to gain from our reports and which scope in data searches in configuring TDS guides.

We selected to import discovery guides that were installed automatically with the product. Figure 234 on page 217 allows us to add our own discovery guides.

Figure 234. Import Installed TDS Guides

Every decision support guide is a module that groups views around similar concepts, for example, call center management.

The application provides the ability to import the discovery guides at any time by selecting **Administrator Panel -> Decision Support Guides**.

Figure 235. Select to Import Discovery Guides

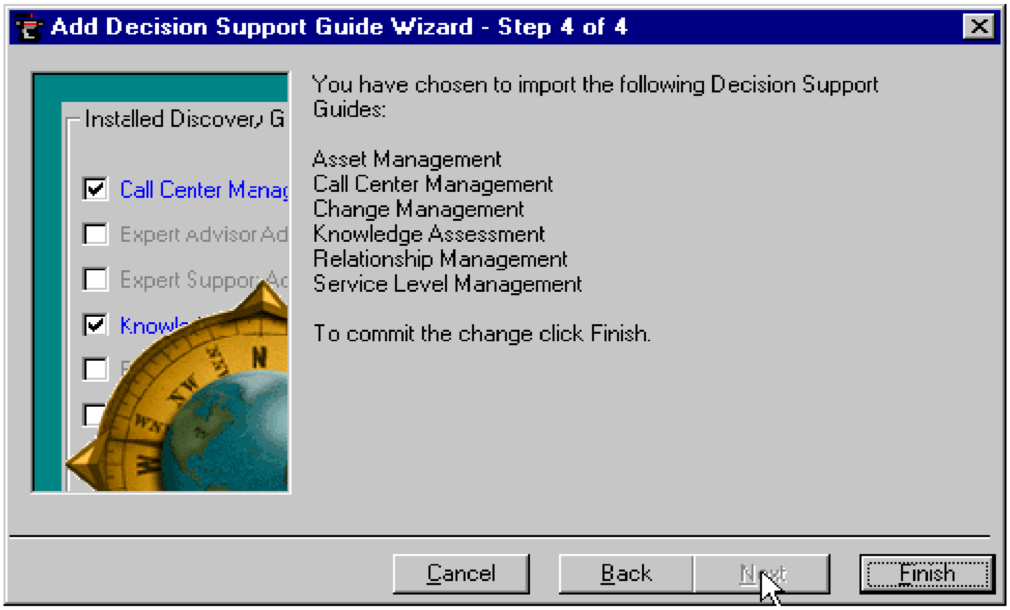


Figure 236. Add Decision Support Guide

We can get all the information for discovery guides once they are installed in our environment. From the Administrator window, we double-click on the **Discovery Guide** folder, then we select one of them, for example, the Call Center Guide, and using the right-hand mouse button, select **Properties** (see Figure 237 on page 218).

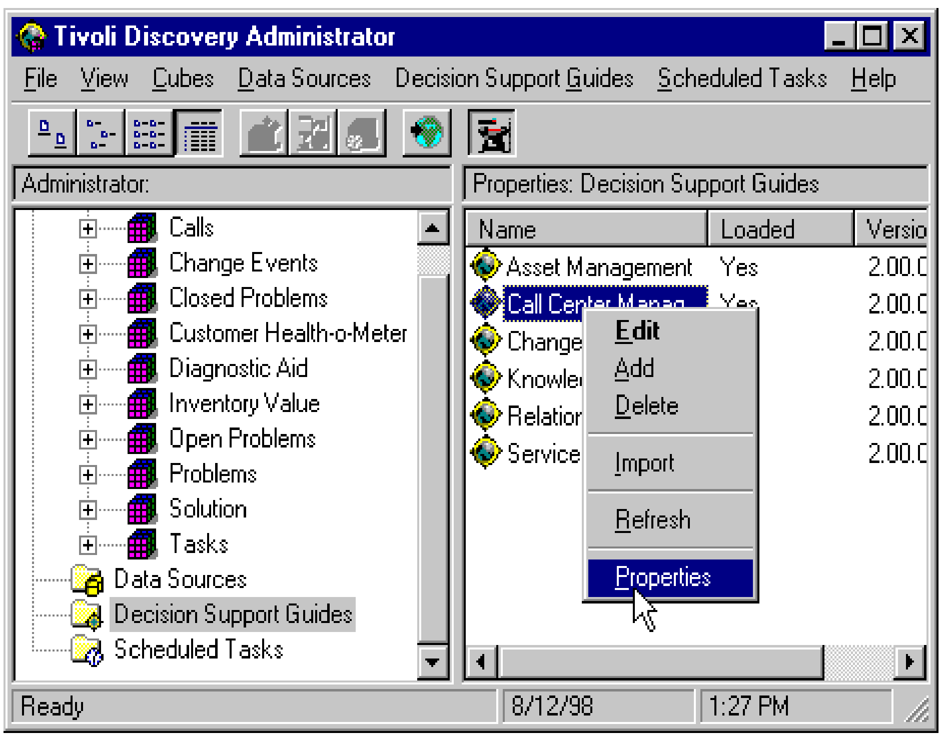


Figure 237. Call Center Discovery Guide Properties

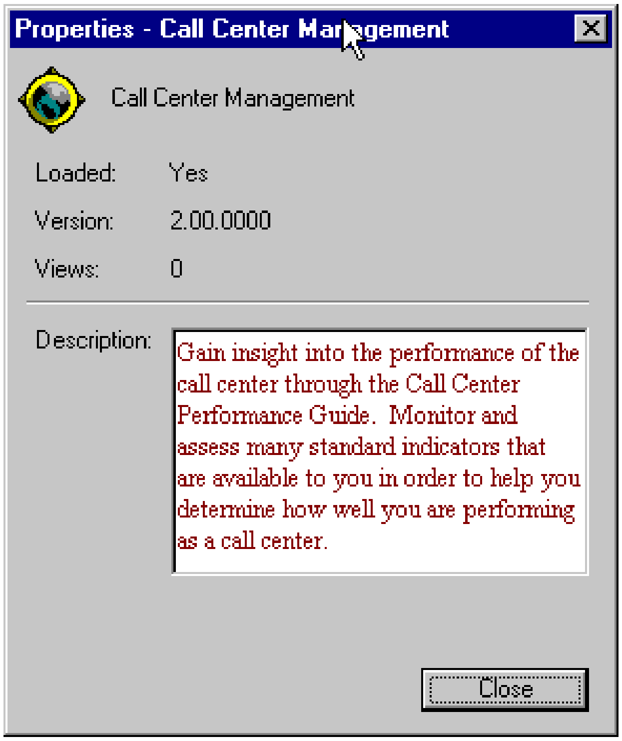


Figure 238. Call Center Properties

The call center has now been installed.

5.11.1.3 Add and Connect Data Source

The first time we launch the TDS Administrator program, we are asked, also, to perform a few tasks for connecting the PowerPlay cubes and views to live data.

Figure 239. Add the Data Source

The external databases are accessed via the defined ODBC drivers. Now we are going to point the drivers and open the connection to the database.

As you can see, comparing Figure 231 on page 215 with the following figure, we select the data source name (DSN) previously configured for the ADVISOR database.

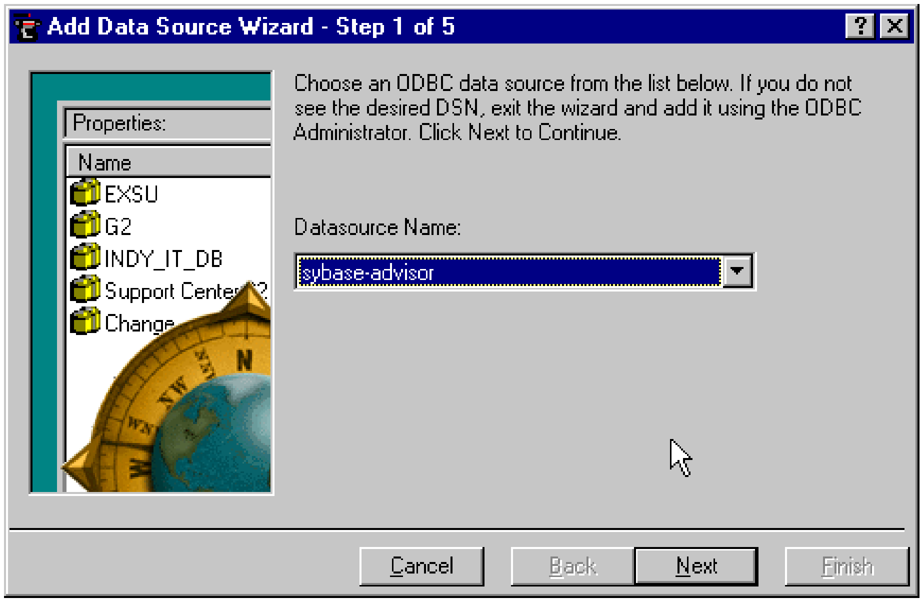


Figure 240. Selecting Advisor Data Source

We are now asked to specify the user name and the password for accessing tables in the ADVISOR database.

Figure 241. UID

Next we are asked for a qualifier that fully qualifies the tables we are accessing. The qualifier varies depending on the database platform. The Help menu pointed us to the setup of Application Software Expert (ASE). We verified our connection using the EA SQL Configuration Editor.

Note

Please be aware that even if you test the connectivity of your data source and the connection is successful, you still may have an incorrect qualifier for a data source. The Test Connectivity command does not test the qualifiers.

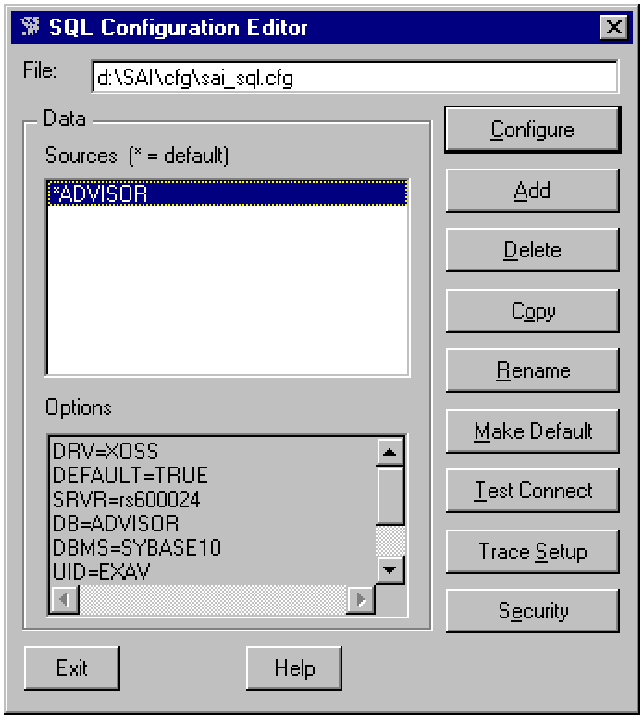


Figure 242. Checking ASE SQL Configuration for ADVISOR Qualifier

The data source definition is completed so we select **Finish**. We tested the connection from the Administrator's main panel by double-clicking on the **DataSources** folder, followed by right-clicking on **sybase-advisor** and **Test Connectivity**.

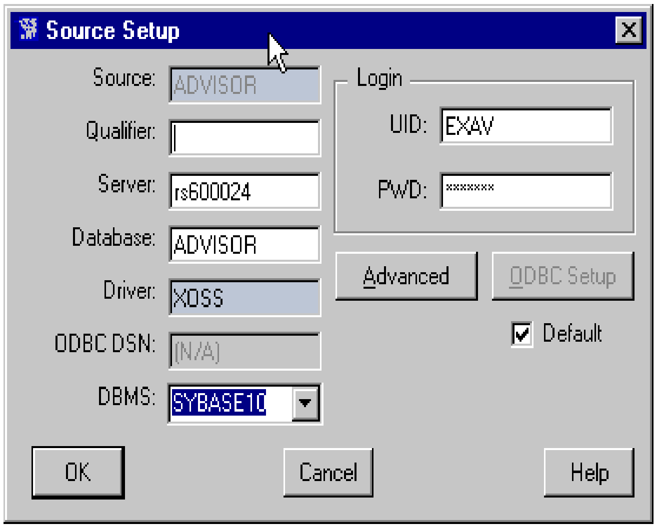


Figure 243. Source Definition

The Administrator's main panel allows us to modify data source properties, and eventually delete them and re-add them manually, basically with the same sequence we followed before.

5.11.2 Assigning a Data Source

From the main TDS administrator's screen you can assign the data source (see Figure 244 on page 222).

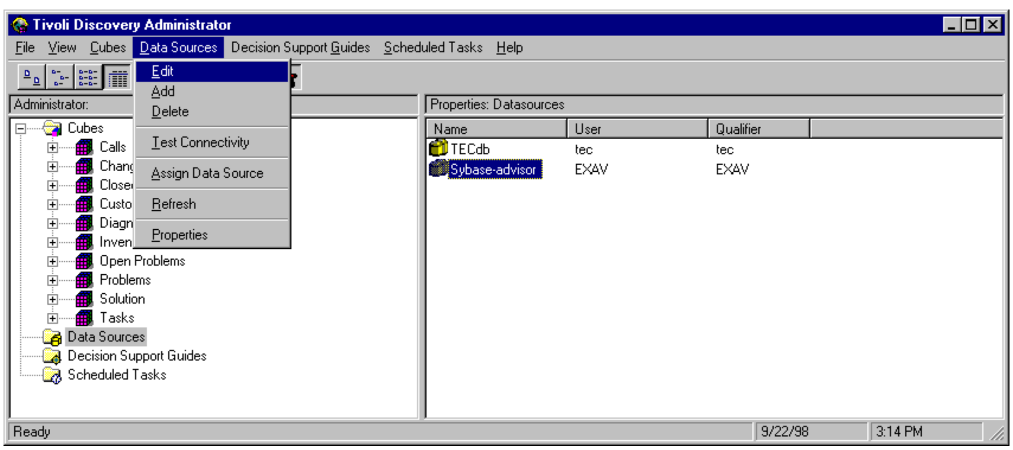


Figure 244. The Data Source Definition

Figure 245 on page 223 shows the first screen to add the data source.

Figure 245. Data Source Definition for ADVISOR Database

Tivoli Decision Support enables you to establish one data source for all your cubes or you can pick and choose the data source for your cubes. This is useful if you have several data sources in your enterprise that may be used to populate your cubes. Instead of relying on just one data source to fill all your cubes, you can use several sources depending on the type of data you want to review.

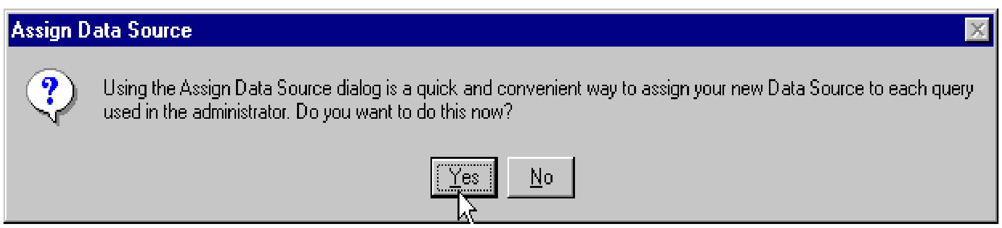


Figure 246. Assign Data Source

The Assign Data Source window appears as shown in Figure 246 on page 223. We select all the queries we want to assign a data source in the Query Name column.

You can see the cube to which a query belongs in the Cube Name column. Also, you can see the current, if any, data source assigned to a query in the Data Source column. From the Data Source drop-down list, select the data source for these queries, and then click on **OK**.

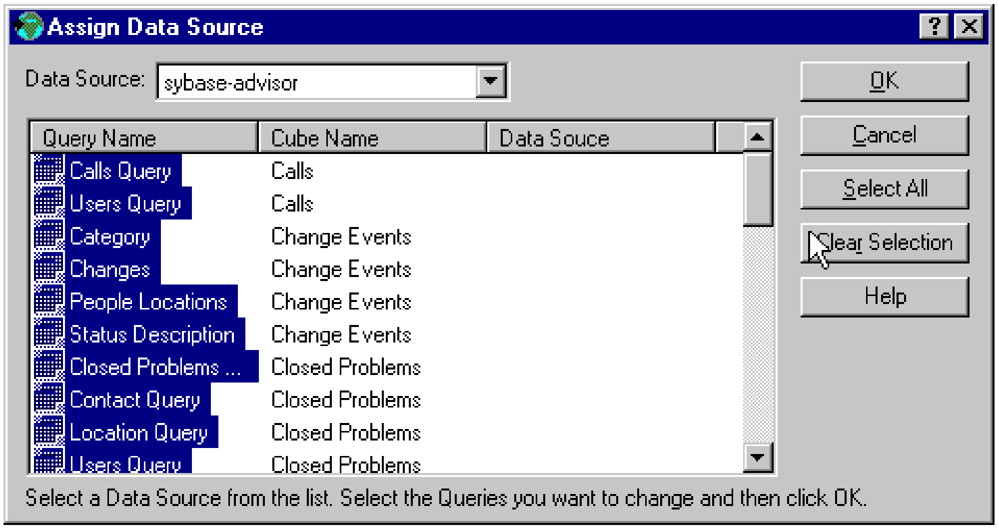


Figure 247. Assigning a Data Source

The Administrator's windows will be revealed (see Figure 248 on page 224).

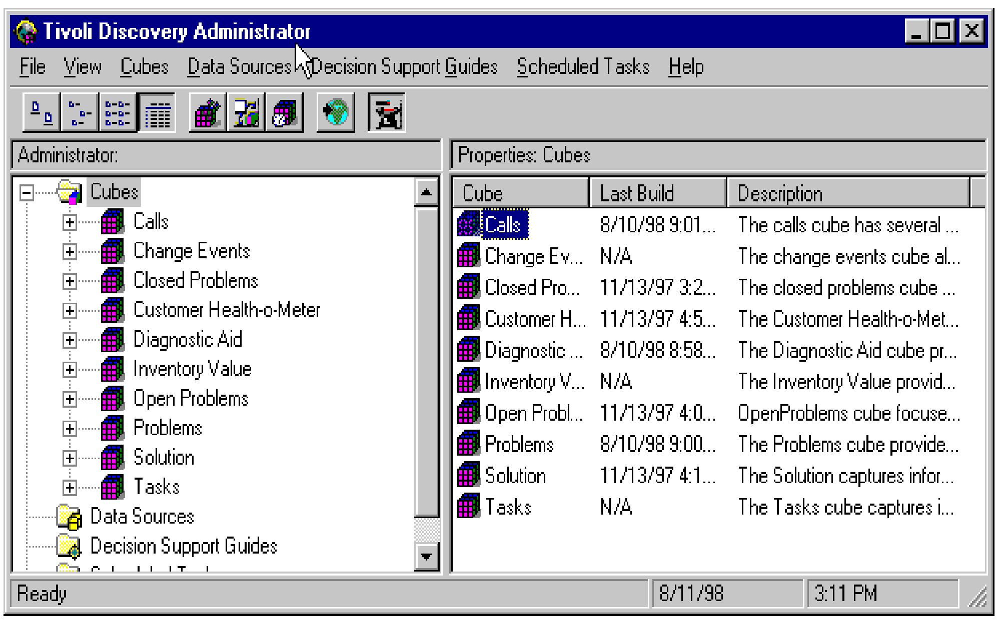


Figure 248. TDS Administrator's Window

5.11.2.1 Setting General Options

The administrator needs to know where to find certain files, for instance, the source files on the network configuration, so we select the **View** menu and **Options** and we put the location where the source files are. In the network Source path box we put the name of the path that contains all of the server content, meaning cubes, data and reports folders.

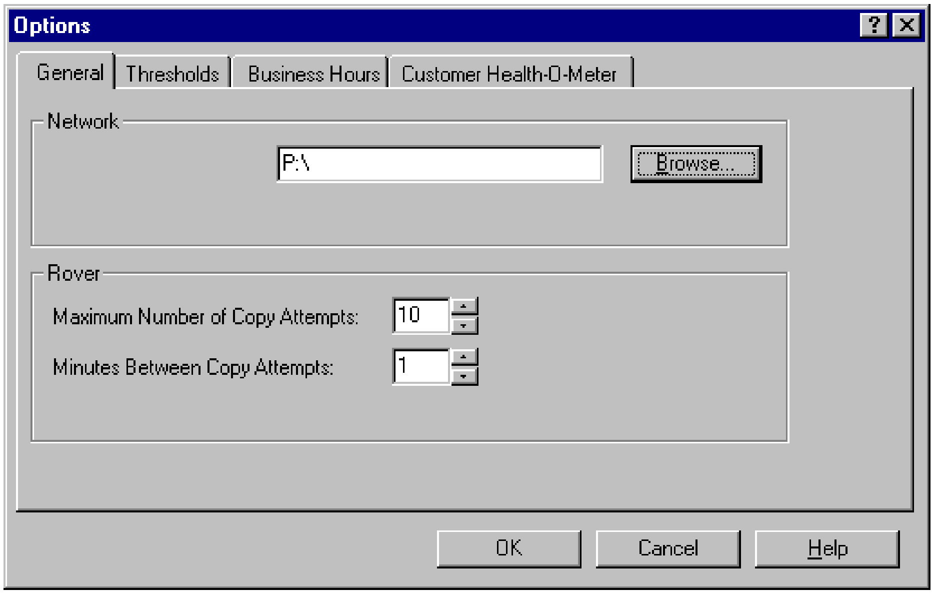


Figure 249. Setting Shared Source Path

Purely for reporting purposes, we are asked to specify the threshold levels for a given problem severity. Tivoli Decision Support uses this threshold in a number of views to calculate resolution times. For example, TDS can assume that for a given severity level the requests must be resolved in a certain amount of time, so calculating the difference, or the delay could be a measure in our reports.

This value will not impact any other applications but the reporting functions. It has to be as near as possible to the policies defined for our problem management environment.

Select the **Thresholds** tab (see Figure 250 on page 253).

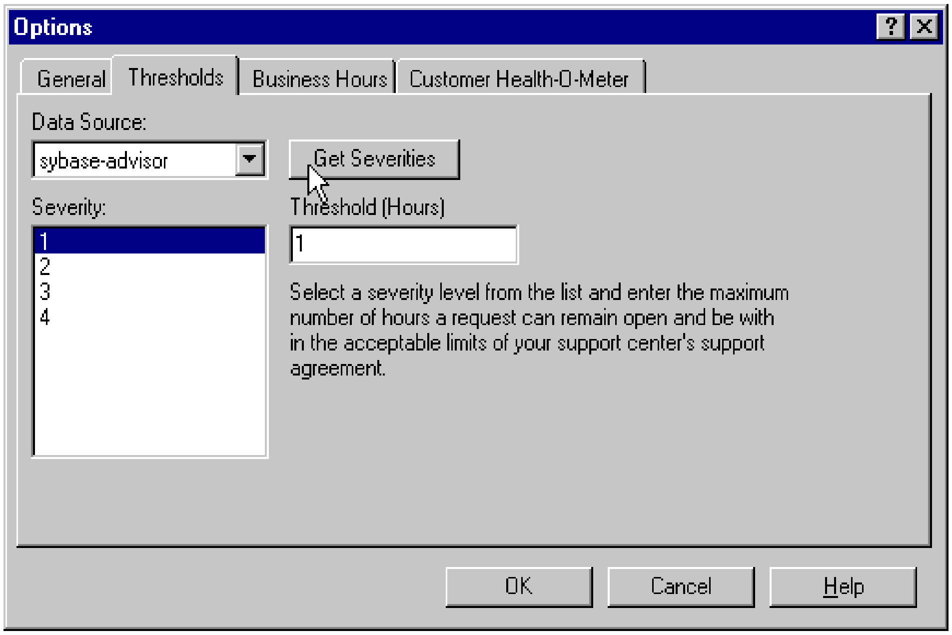


Figure 250. Options

All the calculations on delayed time are dependent on the business hour timeframe. We are not actually mapped with a company calendar, including holidays or vacation days, but we can specify the amount of working hours. Here you can click on the appropriate data source and then click on **Get Severities**. Then you can specify the hours for each level.

Select the **Business Hours** tab, update fields as requested and you get the following panel. Select the **Populate** button.

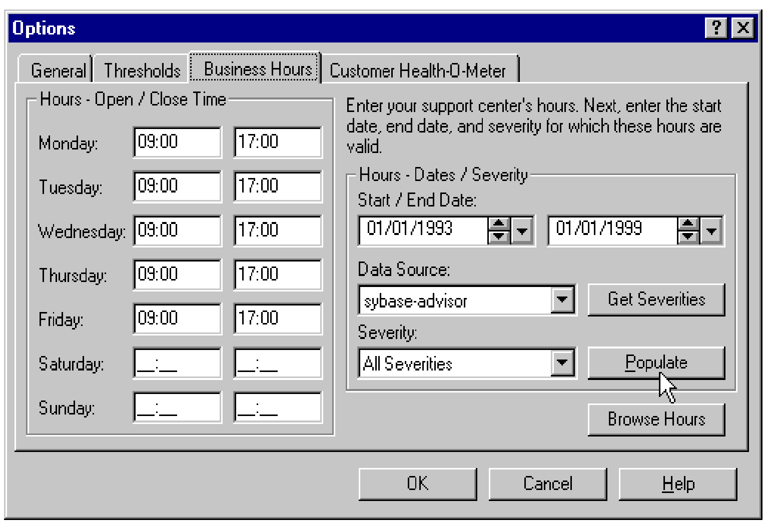


Figure 251. Setting Business Hours

Figure 251 on page 226 shows the business hours defined. To avoid confusion and to ensure that threshold-dependent views calculate correctly, we need to configure Expert Discovery for your support center's normal hours of operation the first time you run the Expert Discovery Administrator by selecting the **Business Hours** tab. You can also specify different business hours for each severity level.

The Customers Health-O-Meter tab lets you specify it you want information from locations that have call activity or specific locations (see Figure 252 on page 227).

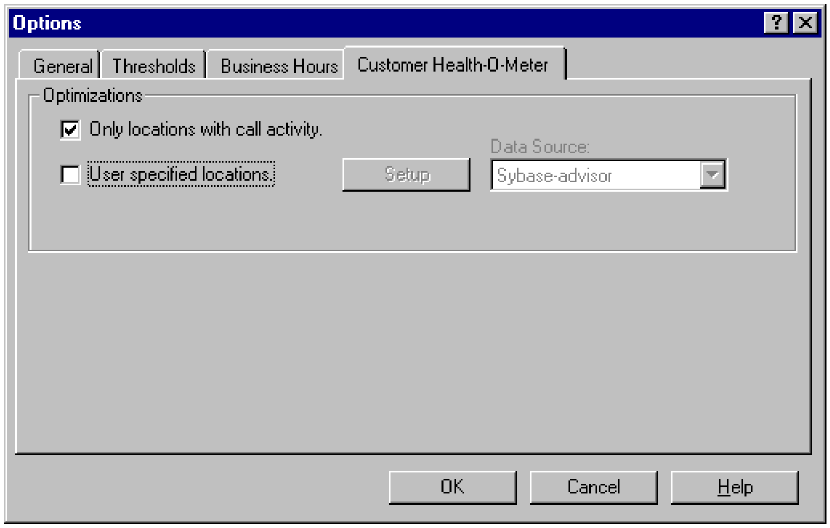


Figure 252. Health-O-Meter

5.11.2.2 Work with Cubes

First we take a look at the cube parameters. We are checking if the measured fields are going to be calculated in a reasonable range of time, due to our environment being new.

From the Administrator window, double-click on **Call Cube**; detailed queries and parameters appear. Select **Parameter**, on the Properties panel select **Data Range** using the right mouse button and select **Values** (see Figure 253 on page 228).

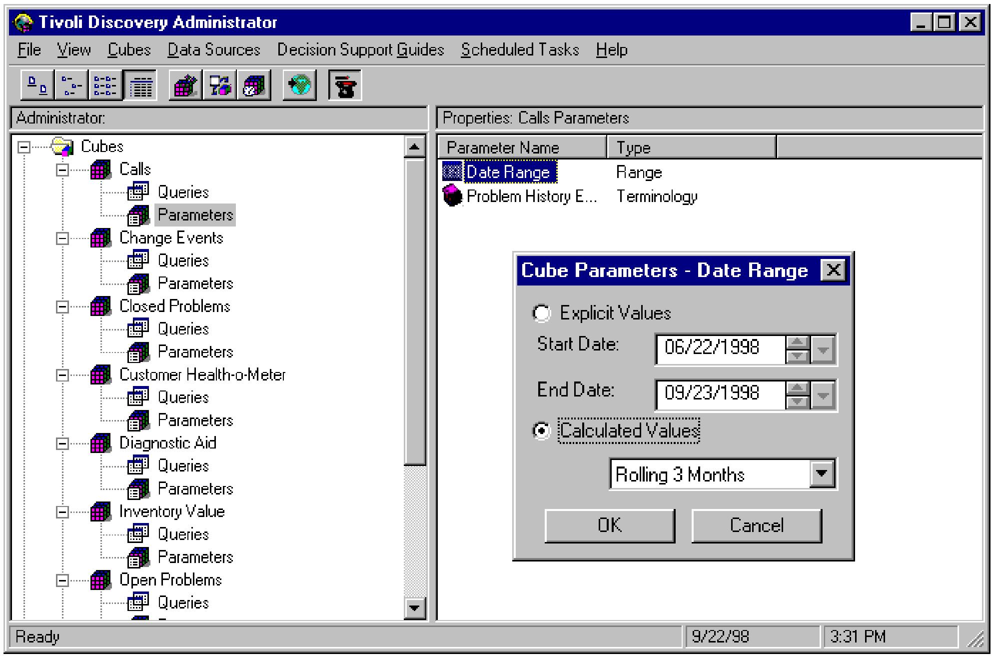


Figure 253. Selecting Data Range Values for Building Cubes

Note

The following steps for exploiting a query are optional. It is not required to perform this separately from a cube build.

From the Administrator's panel, select **Queries**. The queries that are used in the same cube will be listed on the right side of the window. Select **Call Query** and right-click **Edit**.

Listed are SQL statements that will extract values from the database. Clicking the **Calculated Columns** tab we can see the calculated values.

This step provides us with a better understanding of which values are extracted from the ADVISOR database.

If we decide to retrieve inventory information related to our calls, it is not a difficult step to personalize or modify the existing queries.

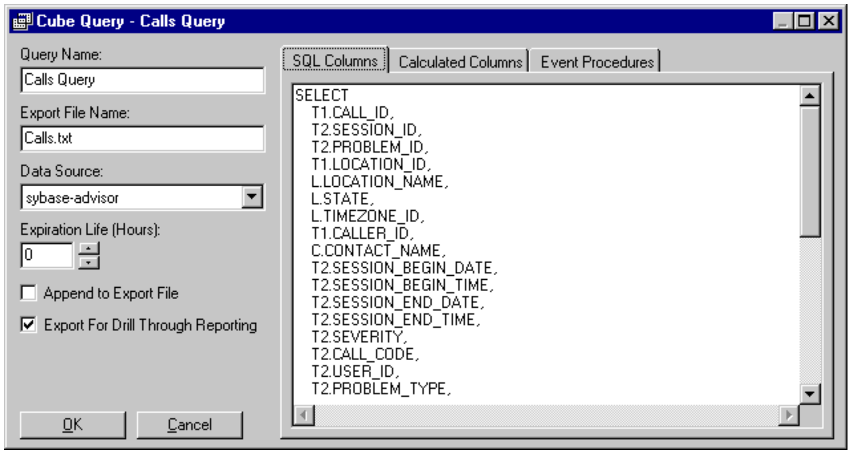


Figure 254. Editing Calls Query

Now we check manually if the connection to the database is correct. We want to see which fields the selected query is extracting from the database.

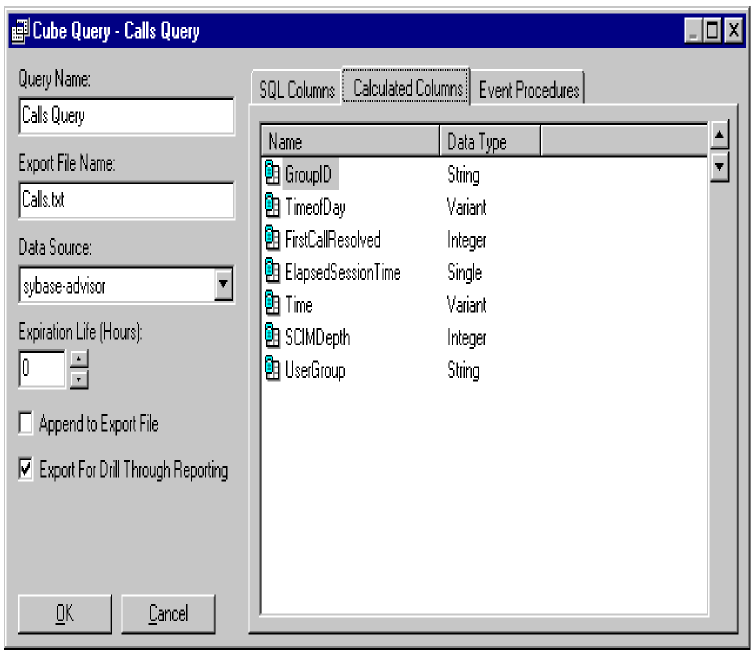


Figure 255. Cube Query

First we exported only field labels by right-clicking on **Export Fields Name Only**. The results are written in a text file, located on the file server directory $SHARENAME\Data\Export.

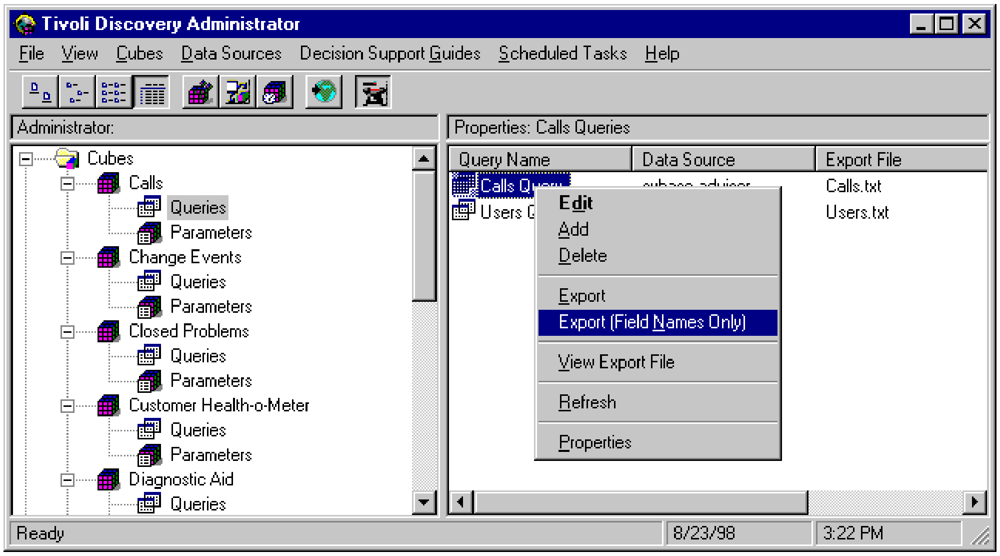


Figure 256. Viewing Exported Field Labels

Since the result of the simple query had no errors, we decided to run the complete query by right-clicking on **Export**. The result is a confirmation window appears.

Figure 257. Call Text - Notepad

This window shows the parameters of this query, including the time range previously defined. We select to view the export file to check that the data is in there.

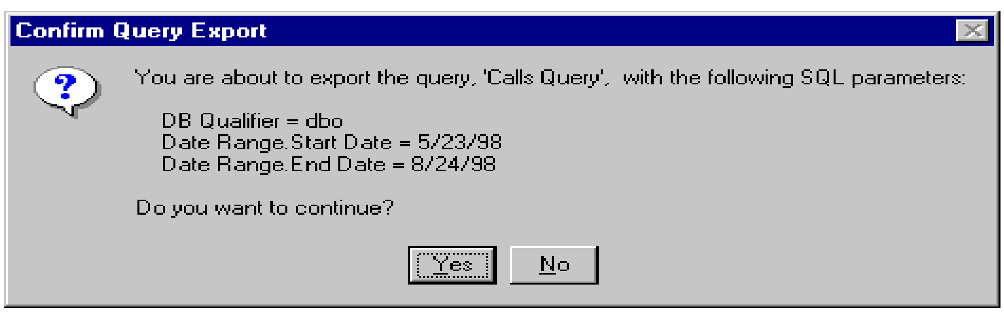


Figure 258. Confirm Query Report

Click on **Yes** to confirm.

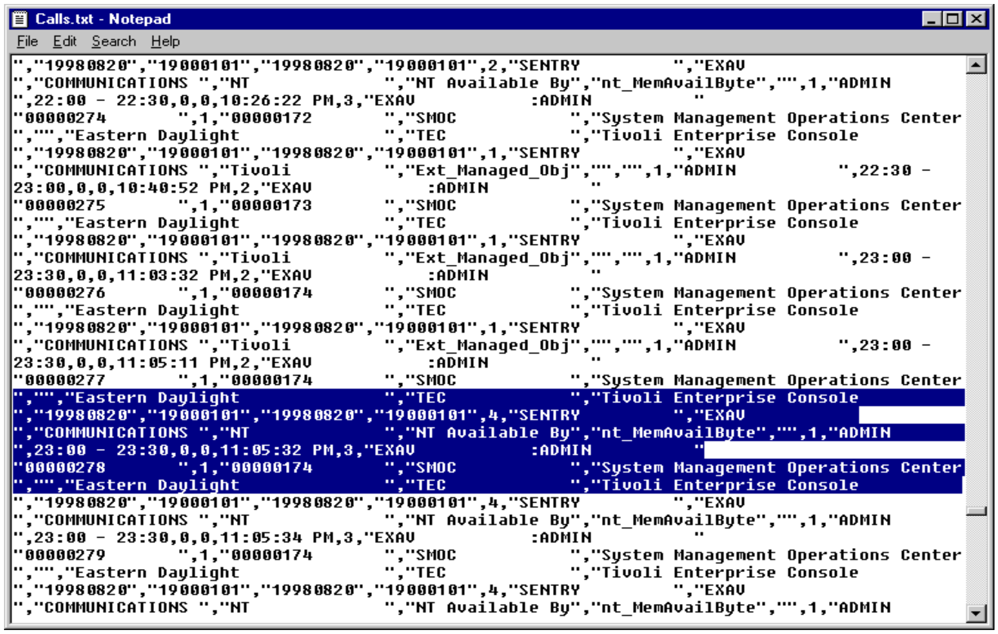


Figure 259. Result of Calls Query

The results are shown in Figure 259 on page 231.

You can also manually edit the results of a query. We used WordPad to view the file, as shown in Figure 260 on page 231.

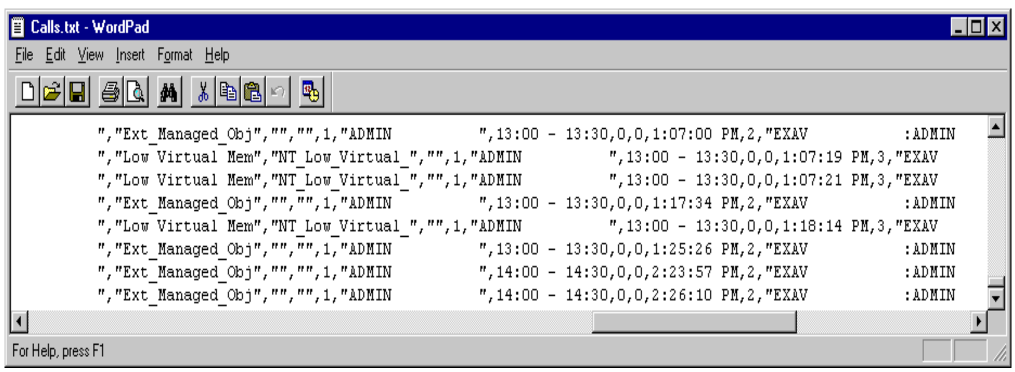


Figure 260. Manually Editing the Results of a Query

5.11.2.3 Building Cubes

The process of building cubes consists of three steps:

- Getting data from database (export queries)
- Calculating fields (measure)
- Compiling cubes using the transformer

Back in the Administrator's window, we select the **Calls** cube, right-click and select **Build**.

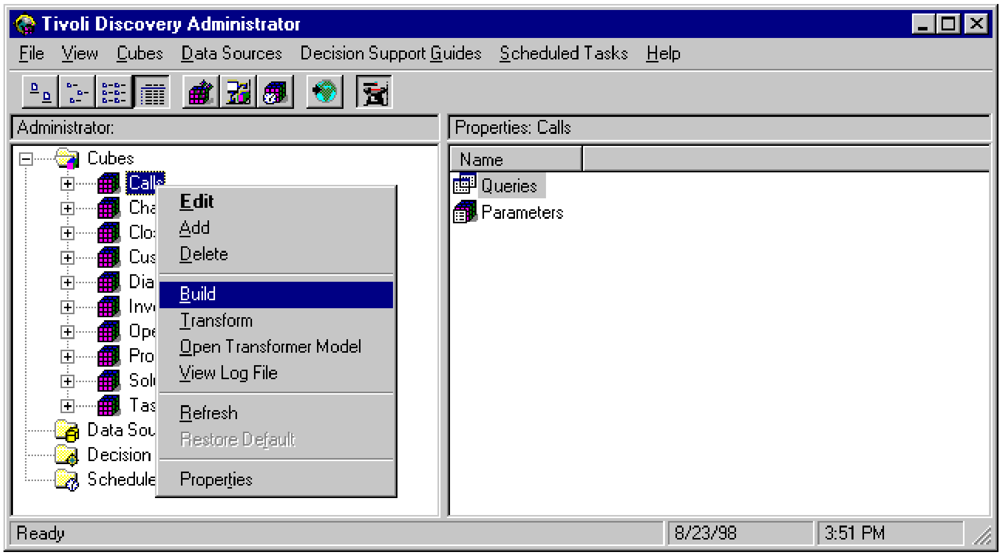


Figure 261. Manual Build of Cubes

Initially the output shows running queries. Then the Transformer Compiler window appears and data is collected from the export files, calculated in selected fields, defined in categories, precompiled in a metadata definition file, compiled in %SHARENAME%\Cubes\Temp and then copied to %SHARENAME%\Cubes.

Looking at the %SHARENAME%\Cubes directory you can see the Calls.mdc cube and Calls###.ppr PowerPlay reports.

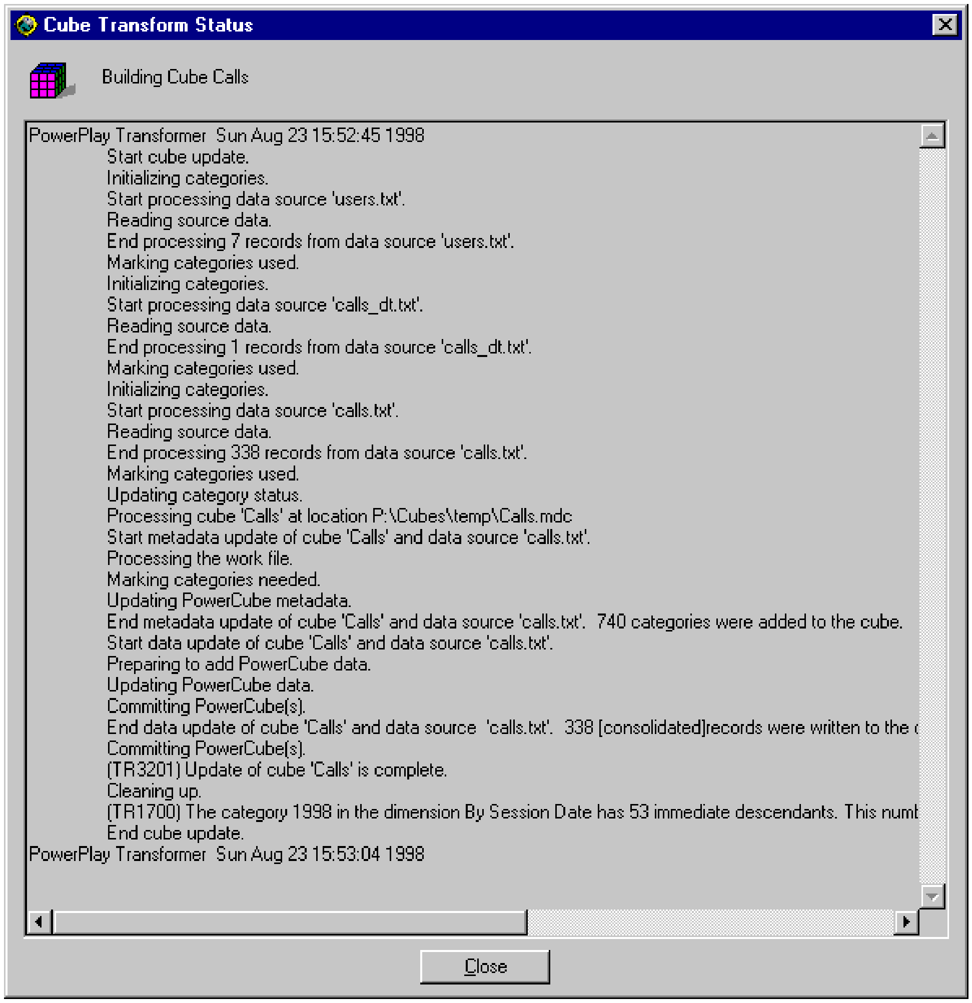


Figure 262. Output of Transformer Compiling Stage, Building a Cube

The output is shown in Figure 262 on page 233.

5.11.2.4 Schedule Automatic Build

As pointed out in Chapter 3.5.2, “Tivoli Decision Support Architecture” on page 48, cubes represent static data, extracted and compiled. This provides a large amount of data available for interrogation.

One of the roles of the Tivoli Decision Support administrator is scheduling an automatic build of cubes in predefined intervals. We are able to define a task for building each single cube, giving us the freedom to apply specific intervals to heterogeneous data. For example, the connection to an external DataSources for Inventory data could be rebuilt depending on updating on the inventory database, while the problem-related cube has to be rebuilt in frequent time intervals.

Remember that the building activity requests many resources from the Administrator workstation, and uses the network connection to the database intensively, so it is a good habit to schedule cube building at night.

For defining a recurrent building of a cube, from the Administrator’s window, highlight **Scheduled Task** and **Cube Build**. We are prompted for a task name.

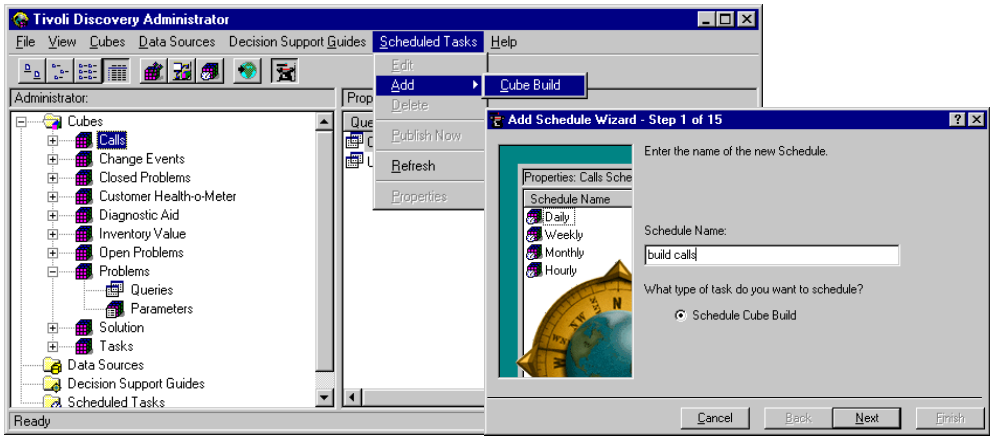


Figure 263. Adding a Scheduled Building of a Cube

As pointed out before, we can distinguish which cube has to be rebuilt by time, or, in other terms we have to define a task for every single cube building.

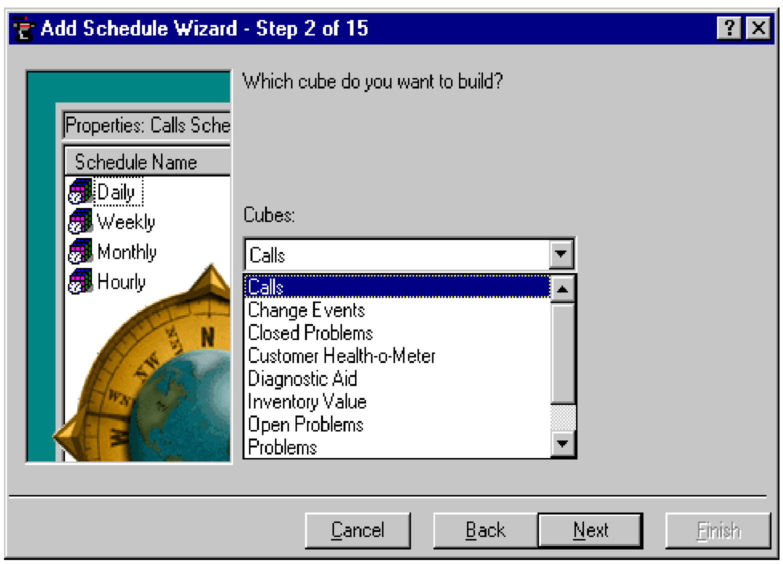


Figure 264. Selecting a Cube to Be Rebuilt in the Task

Time intervals and date range values for this specific task are the next questions we are prompted for. We selected a daily frequency that will be effective for one week. Although very tough to go through, this granularity allows us to put different definitions even for the same cube building.

Figure 265 on page 235 shows the definitions we gave to our scheduled task. Then we select **Finish**.

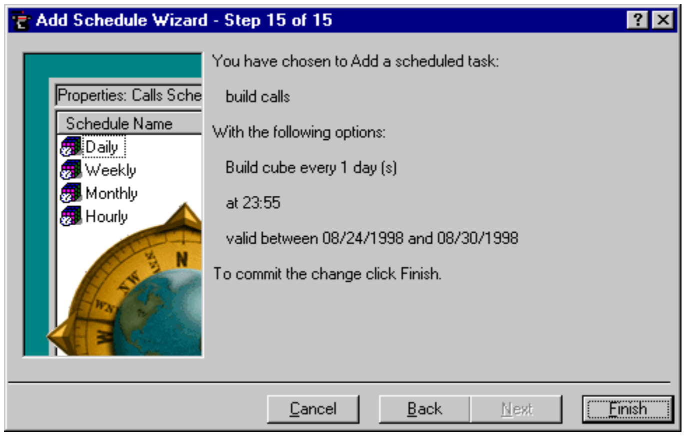


Figure 265. Defined Scheduled Task

Please be aware that building cubes is a resource-consuming activity.

5.12 Tivoli Service Desk Application Development

The TDS application can be customized to cover the needs of specific business processes, and it is possible to develop new applications using the TDS Application Software Expert capability.

In this book we don't need to change the application because for our environment there is enough customization flexibility provided by the applications.

Generally, to change the application look or behavior the TDS provides the Application Software Expert (ASE). This is a set of tools to develop new applications. We cannot cover in this book all the functionality of the ASE but we want at least to mention the main tools.

- The Script IDE

  The Integrated Development Environment (IDE) is designed to develop applications using a graphical interface environment rather than a command line environment. Using this tool it is possible to edit programs.

- Script programming language

  Programs are composed by using the script language. This is a structured programming language cross platform and cross RDBMS with a Pascal-like syntax. It provides a large set of functions and a rich set of data types useful in developing business applications and event-driven graphical user interfaces.

- Interface designer

  The Interface Designer is the environment to create graphical windows of the applications. It is based on graphical selections of items to put in the window, such as text fields or buttons and then allows you to quickly create new windows.

- Script debugger

  This is the tool used for debugging script applications. It provides typical debugging functions (such as break points) in a graphical interface environment.

• Script Parser

Translates the text files that contain script source code into binary format. It translates.kb files to.kbc files. The Parser is available in a command line interface and graphical interface version.

• Interface Generator

Processes the files generated by the Interface Designer. It is like a compiler of the files generated by the designer. The Interface Generator is available in a command line interface and graphical interface version.

• Script Interpreter

This is the run-time system of the script programs. It executes the expressions of a script program.

• Interface Viewer

This is used to view the components generated by the Interface Generator without starting the entire program.

5.13 Tivoli NetView Inventory Integration Setup

Integration between Tivoli NetView and Tivoli Inventory allows us to complement the data stored in the Tivoli Inventory configuration repository with data about the network gathered by NetView. Once the information is in the configuration repository, it can be used by other Tivoli applications, for example, to determine which systems are ready for software distribution.

Once the Tivoli/Inventory Integration Adapter is installed on the managed node that acts as the NetView server, and the Tivoli/Inventory Integration profile component is installed on all the TMR managed nodes, we have to perform the following steps to provide the Tivoli Inventory database with the new information that comes from a network discovery:

• Add new tables and views to the Tivoli Inventory RDBMS.

• Create the NetView Inventory query library.

• Scan the NetView database to import selected OVWDB fields.

5.13.1 Prerequisite

When building our environment, we assumed that Tivoli Inventory was already installed and configured. A valuable source of informations about Tivoli Inventory is *TME 10 Inventory 3.2: New Features and Database Support*, SG24-2135.

In our environment the Inventory RIM host is an AIX managed node, connected to a Sybase server. Inventory 3.6 is installed on all our managed nodes, including NetView server and TMA gateway, for distributing profiles to TMAs.

5.13.2 Adding New Tables and Views to Tivoli Inventory RDBMS

We are provided with different SQL scripts, one for each RDBMS vendor, named:

```
nvinv_<RDBMS-vendor>_schema.sql
```

These are located in directory $BINDIR/TMF/TIPN. These scripts are installed on the RIM host only.

We open a session on our Inventory RIM host, open an isql session and run the following script:

```
root@rs600024:/Tivoli/bin/aix4-r1/TMF/TIPN >: isql -U tivoli -P tivoli
```

```
>@nvinv_syb_schema.sql
```

The script installs views and tables in the same manner as the Tivoli Inventory schema. Please note that in the following lines extracted from the nvinv_syb_schema.sql script, it refers to OVWDB fields and that among those field definitions there are the newly added Tivoli fields by the new nvsniffer command.

Extract of nvinv_syb_create.sql

```
*** CREATE TABLE NV_NODES(
Object_Id integer NOT NULL,
        Selection_Name varchar(64) NOT NULL,
        IP_Hostname varchar(64) NULL,
        IP_Status varchar(64) NULL,
        isPrinter integer NULL,
        isIPRouter integer NULL,
        vendor varchar(64) NULL,
        isComputer integer NULL,
        isConnector integer NULL,
        isBridge integer NULL,
        isRouter integer NULL,
        isPC integer NULL,
        isIP integer NULL,
.........
isTMA integer NULL,
        isTME integer NULL,
        isTMAGateway integer NULL,
        isTMAEndpoint integer NULL,
        TMA_Gateway_Label varchar(64) NULL,
        TMA_Gateway_Port integer NULL,
        TMA_Endpoint_Label varchar(64) NULL,
        TMA_Endpoint_Port integer NULL,
        Tivoli_ManagedNode_Label varchar(64) NULL,
        Tivoli_ManagedNode_Port integer NULL,
        Tivoli_Interp varchar(64) NULL
)
GO
ALTER TABLE NV_NODES
        ADD     PRIMARY KEY (Object_Id,Selection_Name)
GO ***
```

5.13.3 Create NetView/Inventory Query Library

We are going to add a new query library to our policy region. First we check that QueryLibrary is a managed resource of the selected policy region, then we can run the script from any of the managed nodes where the NetView/Inventory Profile was installed. This script is located in $BINDIR/TMF/TIPN with the syntax of:

```
nvinv_create_queries.sh <Policy_Region_Name> [ <Query_Library_Name>].
```

Next we ran the command:

```
nvinv_create_queries.sh rs60008-region
```

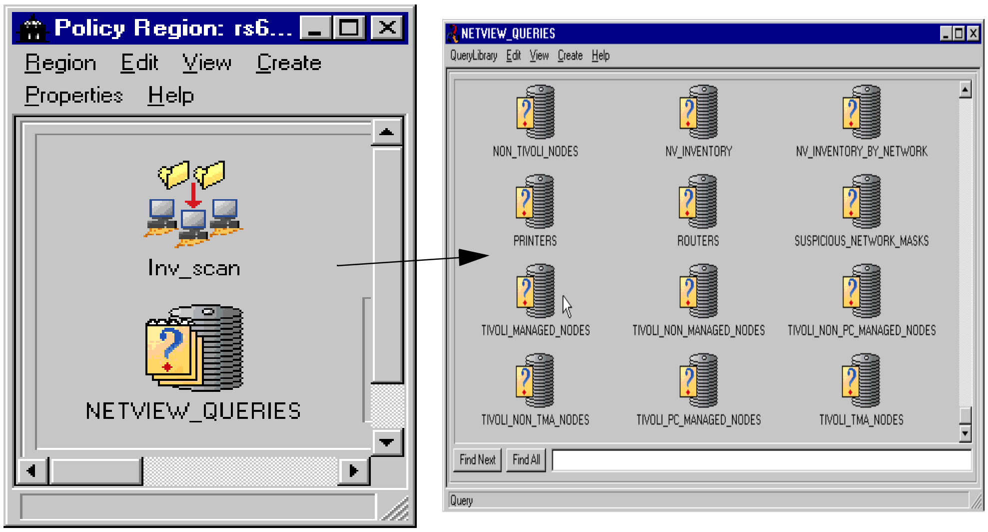


Figure 266. NetView Query Library

5.13.4 Scan NetView Database to Import Selected OVWDB Fields

All the information is gathered from the OVWDB using a standard inventory scanning mechanism. We defined a NetView/Inventory profile, selected which OVWDB field we were interested in, then scanned the NetView server by distributing the profile. The NetView/Inventory Integration adapter will export data from the NetView database.

5.13.4.1 Create a NetView/Inventory Profile

First we make sure we have sufficient roles for creating an inventory profile and that the policy region we are working with has the NetView/Inventory profile resource added.

First we added a new profile manager:

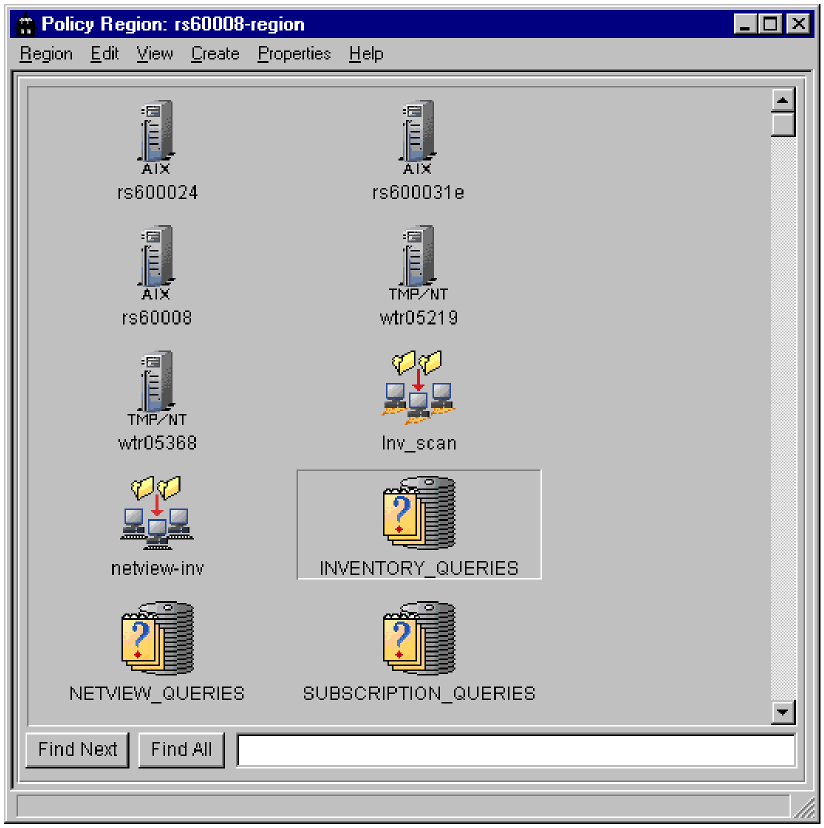


Figure 267. NetView-inv ProfileManager

Then we added a profile of type NetView/Inventory and added entries for each NetView object and its related fields and we distribute it to our NetView server.

The scanning engine will interface with the NetView adapter providing specifications for exporting data from the NetView database. Finally, the Inventory database will be updated with new information retrieved from the network view of our environment.

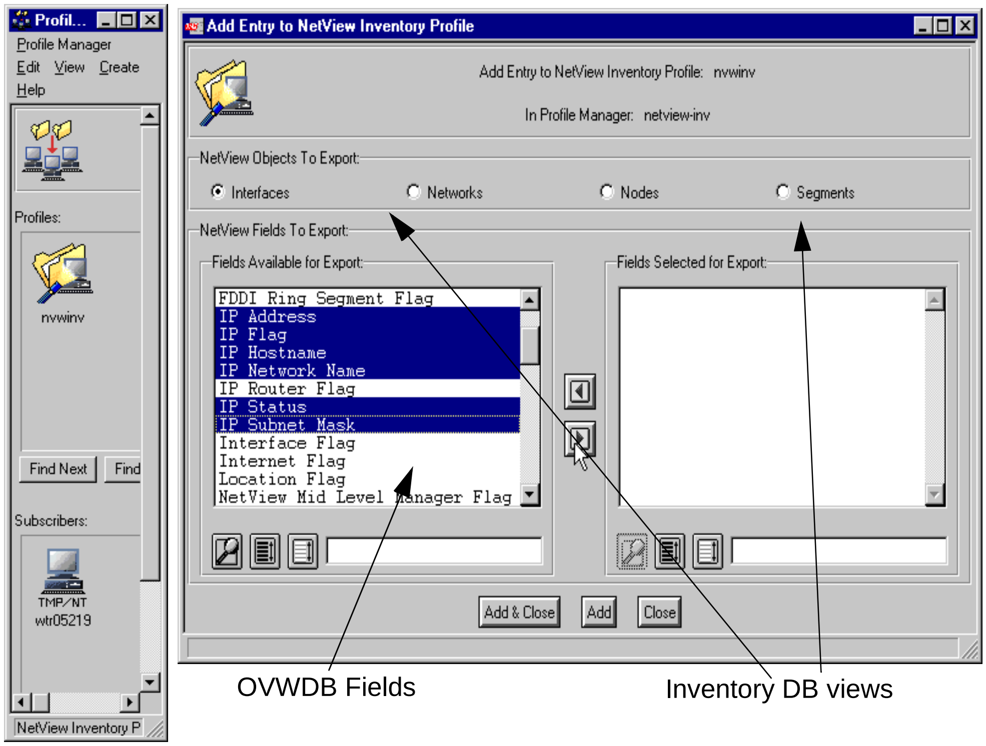


Figure 268. Editing NetView/Inventory Profile

Figure 268 on page 240 shows the editing of the inventory for NetView.

Chapter 6. Problem Management Scenario Using the TEC

This chapter describes how to integrate the Tivoli Enterprise Console and the Tivoli Help Desk suite.

At this stage we have the following components configured:

- TMR server, managed nodes and endpoints
- The TEC
- Tivoli Problem Management environment
- Tivoli NetView
- NT Logfile adapters
- Distributed Monitoring monitors

The main focus for this example is to show integration with the TEC and the problem management environment. Before we begin the example we list the types of information that would be required before starting the implementation. This information is listed below:

- The event relationship diagrams for the NT servers to provide the input for the TEC rule development.
- TEC rule policies for events sent from the NT servers.
- Help desk policies for problem tickets created by the automated process.
- The SCIM definition for the data mapping from the events to the problem management environment.

The environment used for the example is shown in Figure 269 on page 242.

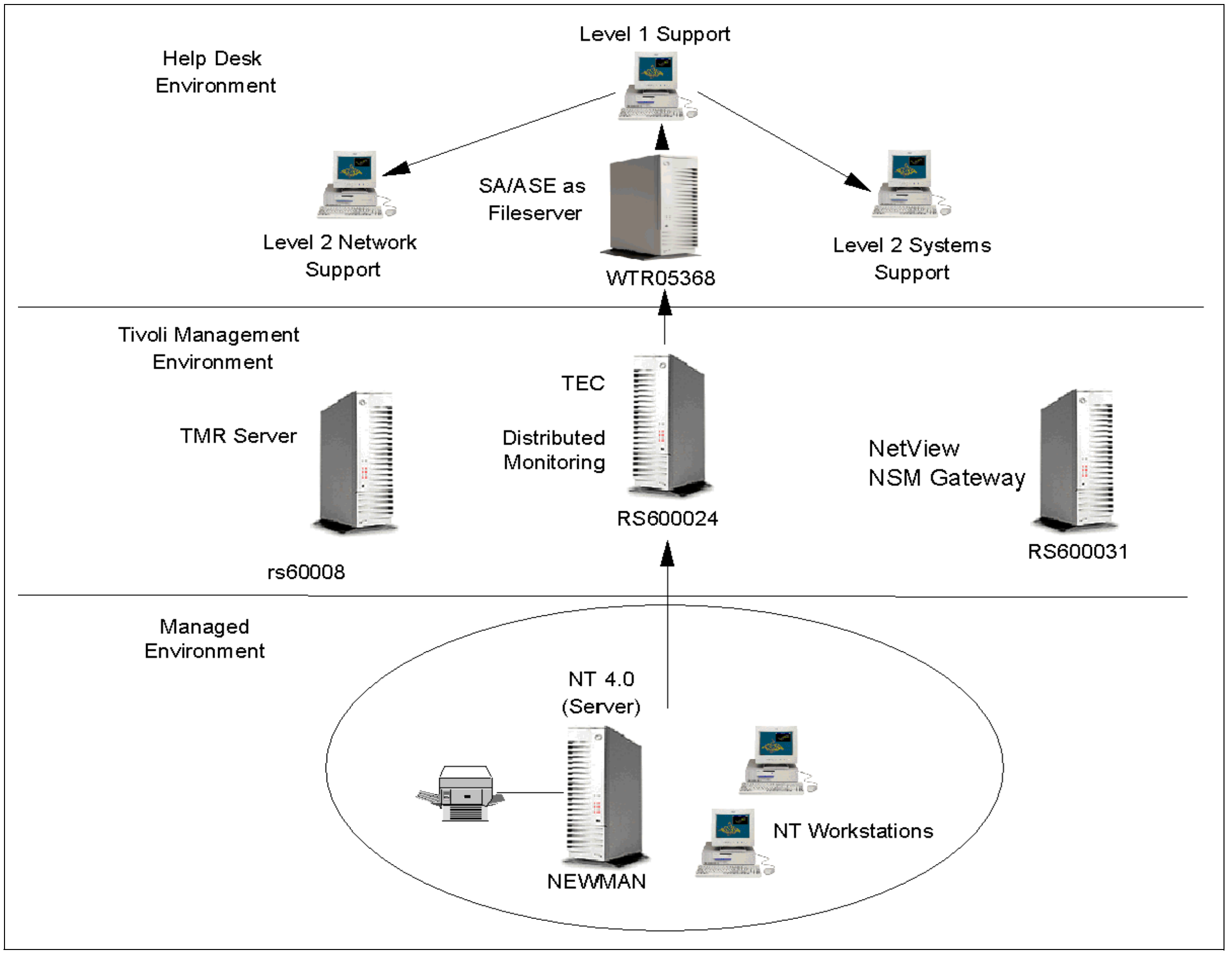


Figure 269. NT End-to-End Example

The example covers three levels of implementation. These are:

- The help desk environment
- The Tivoli managed environment
- The managed environment

The events will be generated from the managed environment, that is the NT server. The event will be generated using the NT logfile adapter and distributed monitoring. These events will be sent to the TEC for correlation. Based on certain policies only selected events will be forwarded to the help desk environment where they will be handled by the different levels of support. The problem will be resolved and the trouble ticket closed. This will close any TEC event in the event cache.

The tasks to perform this operation are in Table 17 on page 242.

Table 17. Example Task List

Task	Description
1	Planning stages
2	Data mapping design

Task	Description
3	TEC customization
4	Rule development
5	Problem management customization
6	Testing the rules

Note

The example contained in this chapter covers specifically the NT adapter and distributed monitoring. It is worth pointing out that the rule correlation examples contained in the redbook *Implementation Examples Using the TEC, SG24-5216* could also be used. These rules would have to be modified in order to map with the SCIM definition used in this publication. Also the script names that generate the trouble tickets would also have to be changed.

The types of planning information we would need for this example is a list of all the potential traps that could be sent from the managed devices.

The more planning that is done for the event correlation design the more efficient the integration from the TEC to the problem management system will be. This is due to the fact that the actual correlation will reduce the overall number of events that will generate trouble tickets.

As discussed in 2.2.2, “Event Management and Correlation Methodology” on page 14 we used the event design methodology that provides the event correlation diagrams that we can use to input into the rule development. In addition to these diagrams we also create flowcharts that show the rule flow.

Our main aim is to reduce the number of events that create trouble tickets. Next we look at the data mapping.

6.1 Data Mapping for the TEC/Problem Management Integration

One of the most important issues when integrating a management system and a help desk system is to have a good understanding of data mapping between these systems and defined policies for this data mapping.

In our environment there are three types of data sources. It is the data contained in these sources that have to be a mapped to each other. These data sources are:

- Events

  An event is an alert from a device that is managed by the Tivoli Management Environment. This could be a trap from a router, a logfile adapter alert or a distributed monitoring alert. This event is processed by the TEC.

- Calls

  A call is entered into the ADVISOR database when a user reports that something is wrong. The most common way of entering a call into the database is when a help desk analyst does it, but the call can also be generated by the TEC.

- Problems

  A problem is created from a call whenever the call cannot be associated with another problem or if the call cannot be solved at once. This means that if you have an open problem, calls can be attached to that problem. When the problem is solved the solution will be the same for all calls attached to that problem.

In the next section we show examples of how to integrate the Tivoli Advisor with Tivoli Enterprise Console following the process described in 2.3, “Data Mapping” on page 16.

6.1.1 Understanding the Problem Management Application

In this section we describe some characteristics of the Tivoli Advisor that are particularly important when integrating the TEC with Tivoli problem management.

6.1.1.1 Advisor Call Data

First of all we need to identify the typical data to be provided when submitting a new call. This data has to be provided when submitting a call automatically from the TEC. It is important to figure out the data field characteristics to be able to map data coming from a TEC event to a call record. For example, we have to make sure that it never happens that TEC attempts to submit a call by passing a field that exceeds the defined limit in the problem management database.

Typical characteristics of a field are:

- If the field is mandatory or optional
- What the maximum length of the field is
- The type of the field, for example numeric or alphanumeric

Figure 270 on page 245 shows the default Advisor window used to submit a new call. Of course, you may have a customized Advisor application containing other fields or having a different behavior. What we discuss here is an example for the default Advisor application.

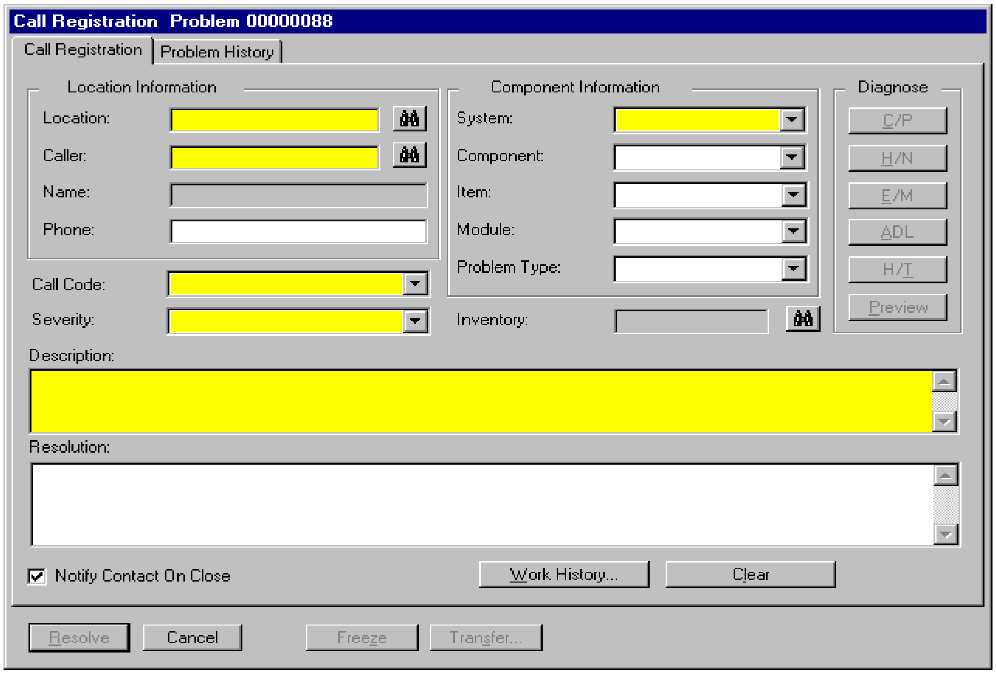


Figure 270. New Call Windows

Table 18 on page 245 shows the fields that we have to provide when submitting a new call. We have mapped these from the data contained in the Advisor panel. The table also shows whether the field is mandatory or not.

Table 18. Advisor Call Data Field Characteristics

Name	Mandatory?
Location	Yes
Caller	Yes
Call Code	Yes
Severity	Yes
Inventory	No
SCIM Fields	
System	Yes
Component	No
Item	No
Module	No
Description	Yes

For more details about the data fields refer to the Advisor entity relationship diagram provided with the product software package.

The remaining fields are automatically generated by the Advisor application based on the value of other fields. For example, the caller name is automatically defined by the caller ID field.

Later, we see that other data has to be taken in consideration to perform the integration between TEC and Advisor.

6.1.1.2 Interface between the TEC and Advisor

In this section we explain how the environment looks and how the different components communicate with each other.

The interface between TEC and Advisor is provided by the Tivoli ExpertView product and two main features are used:

- NSM Commands submit new calls from the TEC to Advisor.
- Gateway callbacks that send information back to the TEC**.**

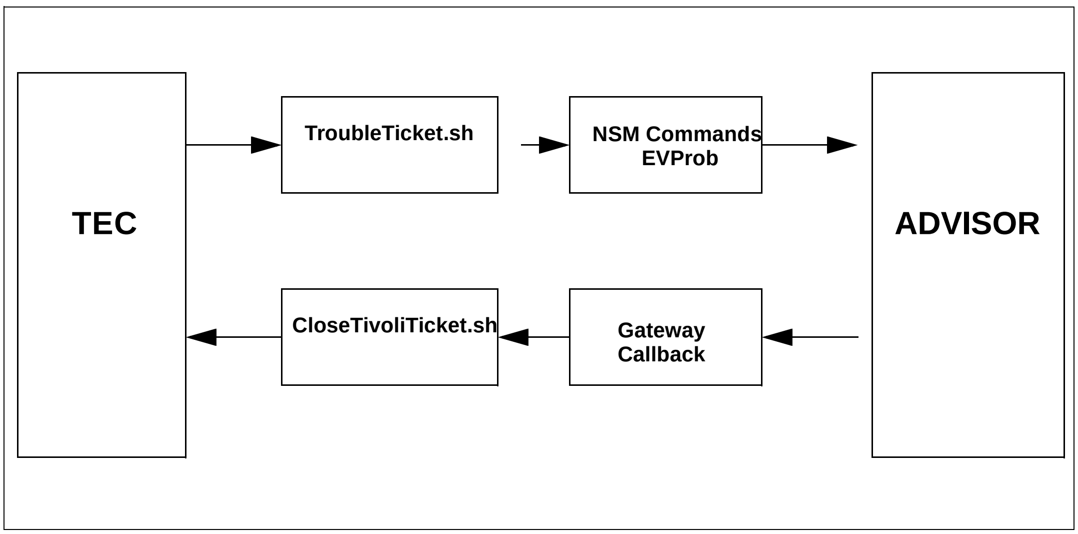


Figure 271. Interface Mechanism TEC - Advisor

The NSM Commands are part of the Tivoli NSM product and is installed on the system management platform. The NSM Commands communicate with the gateway using the command EVProb. This program is called by the system management application.

When an event comes in from any of the different elements that report events to TEC, the TEC rules engine decides if the event should generate a call in Tivoli Advisor (according to the event policies, see “Event Management and Correlation Methodology” on page 20).

The default script provided by the Tivoli Plus module is called sa_ticket.sh. We copied this script to TroubleTicket.sh script. This is the script we call from the rules we developed. The rules provided with the Plus module will call the script sa_ticket.sh. We copied the script so that we can use the same script when either calling from the rules or directly from the TEC GUI.

The rules will initialize the exec_program template in TEC to call the script TroubleTicket.sh passing a number of variables as arguments. This script is provided by the Tivoli PLUS module for Advisor. We need to modify the

TroubleTicket.sh and the variables that should be passed to it in order to achieve our goals.

If a problem for a managed object does not exist, then a new problem and call will be created. If a problem already exists for the managed object, then a call will be added into the already existing problem ticket depending on how the EVProb command is issued. The gateway will use a combination of the managed object (-n option) and the external problem ID (-x option) to determine the action. The following shows an example.

`EVProb -n joe -x 123` creates a new problem and call.

`EVProb -n joe -x 456` creates a new problem and call, since the combination of the -x and -n arguments are unique.

`EVProb -n joe -x 123` will update the first call.

A call can also be opened by an operator from the Event Console window by selecting **Trouble Ticket** from the pull-down menu. In this case the TEC calls the TroubleTicket.sh script passing the event information as environment variables.

Next the TroubleTicket.sh script executes the EVProb command with a set of arguments and flags that are generated within the script.

When the system management platform has initialized a call to the program EVProb that talks to the gateway, there is a call record created in the Tivoli Advisor application. This call can now be handled by the Tivoli Advisor and may be changed to a problem.

From the TECs point of view the responsibility of the event is passed to someone else, for instance the Advisor, then the TroubleTicket.sh script has to set the event status to ACK after the EVProb command has completed successfully.

When the problem is resolved in Advisor, a callback mechanism exists to change the event properties in the system management application. This means that you can close or update an event in the event repository when a problem is changed in Tivoli Advisor. Generally it is convenient to configure the callback mechanism to call a script CloseTivoliTicket.sh that performs a command such as wsetemsg to update the event status setting it to CLOSED.

Note that by these mechanisms we can implement a variety of solutions about what data and when has to be passed back and forward between TEC and Advisor. Of course, all of this depends on the policies defined for the event and problem management.

6.1.2 The EVProb Command

The EVProb command is used to perform the data mapping. The syntax is shown below:

```
EVProb -h <gateway host> -n <managed-obj> [-m <message name>] [-Sd]
[-x <ext-prob-id>] [-a <PROBLEM_VIEW-args>] [-u <user-group-notify>]
[-U <user-group-notify>] [-c <connection-type>] [-A]
```

The key arguments are outlined below. For additional information please refer to the *Network/System Administrator's Guide*.

The arguments are:

- -h <gateway host> is the name of the host were the Tivoli NSM Gateway is installed.
- -n <managed-obj> is the name of the managed object. The managed object is seen by the gateway as a failed node. In the Plus module for Tivoli Service Desk the default setting for the managed object is a combination of TEC event slots.

Flags:

- -x <ext-prob-id> is used to pass the external problem ID to the ADVISOR database. In the Plus module for Tivoli Service Desk the default setting is to send the event ID.
- -a <PROBLEM_VIEW-args> is used to set any argument in the PROBLEM_VIEW view in the Tivoli Advisor database (see the manual *Network/System Administrator's Guide*). For example, this can be used to set the System, Component or Item fields. The -a flag has its own syntax:

  ```
  -a "Column Name:Data;Column Name:Data"
  ```

  For example:

  ```
  -a "System:Hardware;Item:Monitor"
  ```

- -A is a flag that tells EVProb to read the PROBLEM_View arguments from standard input.
- -u <user-group-notify> and -U <user-group-notify> are used to send notifications to users and groups in Tivoli Advisor.

An example of this is shown below:

```
EVProb -h wtr05368 -n rs600024 -a\
"System:Hardware;Item:Server;Item:Disk;Module:C-Drive" -u EXAV
```

This will return the problem ticket ID. Our case is shown in Figure 272 on page 249.

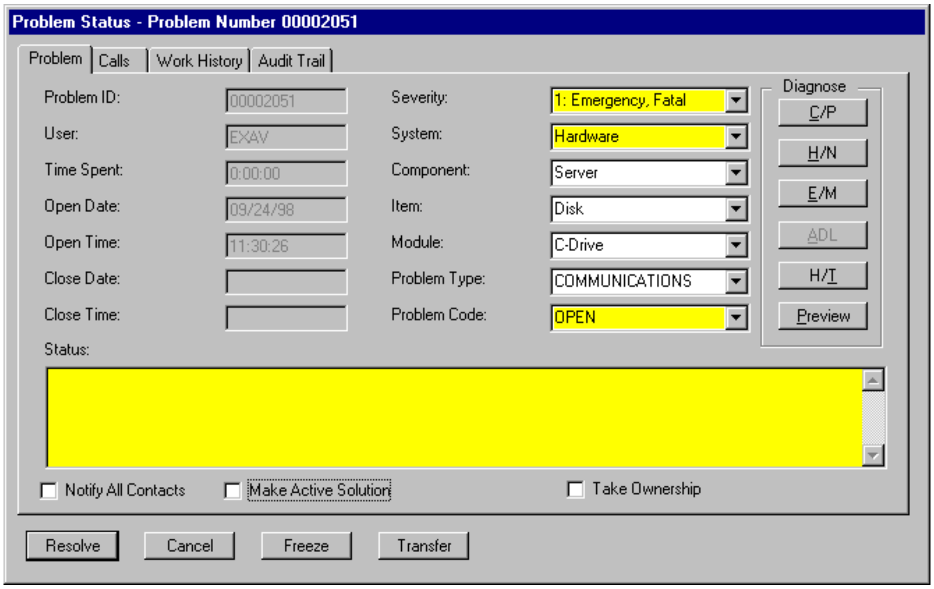


Figure 272. Trouble Ticket in Advisor

6.1.3 Managed Object

This is an important concept used by the gateway. Calls are opened by the EVProb command specifying a managed object. Calls with the same managed object are considered to be the same problem.

For example, suppose that a call is created by the EVProb command specifying a certain managed object, that call will also be a problem in Advisor. Then, suppose that new calls are opened specifying the same managed object as the first. Those calls do not open new problems but will be created as calls attached to the previous problem.

6.1.4 Critical Data

Before performing the mapping from event data to call fields we have to get into more detail about what the Advisor does with the data. Although some data fields could be optional, their presence provides added value to the problem management functionality. We have to figure out what the critical data elements in Advisor are, in other words the data the Advisor uses to perform specific functionality.

Looking at the Advisor functionality the SCIM data can be considered very critical; it provides problem categorization. The SCIM is important for the diagnostic aids as well as to know what skills are needed to work on a specific problem. Therefore, the better the data mapping is, the more efficient the Tivoli Advisor users can be when addressing the call.

Generally, for certain events, we could not have all the information corresponding to every SCIM field, but maybe we have information to fill just some of the SCIM fields. This will be a big added value because it provides problem categorization anyway. For example, a partial SCIM specification can help to identify whether the problem is hardware or software and whether it requires UNIX or NT skills to be solved. This information allows us to identify the most qualified people the problem has to be assigned to.

Other critical information we have to take care of is the managed object. In fact the managed object defines the gateway behavior when a new call is submitted (see “Managed Object” on page 249).

6.1.5 Mapping Event Slots to Call Fields

The mapping of the TEC event slots to corresponding fields to an Advisor call depends on the customization of the Advisor and on the policies adopted for the problem management process.

We want, at least, to show a general way to proceed in the mapping analysis and later show a real example of a TEC event becoming an Advisor call. For our discussion we assume that:

- The Advisor application has not been customized in terms of fields and behavior; it is the default Advisor configuration.
- The problem management organization has two levels. Level 1 gets calls from users and provides front-end support. Level 2 answers problems not solved by Level 1. The Level 2 technicians are very skilled people.
- The calls open by TEC are managed directly by Level 2.

Under these assumptions we describe a possible way to map event slots into call fields, and we discuss the reasons for our choices. Of course, because the mapping is strongly dependent on a real problem management solution, what we describe here is just a possible solution.

The following table summarizes our mapping decisions. Later we explain each row of the table. The table contains three columns. The first is the Advisor field name. The second lists the slots we use to get information for that field. The third explains what manipulation we need to do on the slots to map them in the Advisor call fields.

Table 19. Mapping TEC Events to Advisor Calls

Call Field	Event Slots	Description
ManagedObject	Cause Event ID	This must be the event ID of the first event issuing the call to Advisor (see later).
External Problem ID	Event ID	As is.
Location	none	Default to SMOC.
Caller	none	Default to the caller name TEC.
Call Code	source	As is. Needed to create new call code in Advisor, with just the names of the event source involved as is.
Severity	severity	Map event severity to call severity values
System	source	Map source to the platform name. LOGFILE=Unix; NV390*=OS390, NT=NT. For SENTRY more info is needed such as the class name, to identify the platform. Or map just to HW, SW, NET etc.
Component	sub_source / source	It is source only if sub_source=NULL.

Call Field	Event Slots	Description
Item	class name	As is. Create new SCIM entries to include items equals to the names of those classes we deal with.
Module	none or hostname	Could be the hostname if you need this as a separated field.
Inventory	hostname	We map the hostname into Object ID, because we use Object ID for asset tag in Tivoli Asset Management.
Description	hostname, msg, origin, class specific slots, any other slots	Concatenate strings SLOTNAME_1=slotvalue_1 .... SLOTNAME_n=slotvalue_n

Table 19 on page 250 defines some general rules for the data mapping and identifies a mapping common to all the TEC event classes. In fact the table shows only slots of the EVENT root class and doesn't take in consideration possible slots specified by other derived classes.

In a specific environment under specific policies, the mapping should be performed considering the specific events and their classes and considering the meaning of their slots. So, in that case the mapping will be performed with some exceptions in respect to the table above.

We show in the next sections how to customize the products to implement the data mapping described in the Table 19 on page 250. Here after we explain the reasons of the mapping defined in the table.

6.1.5.1 The Managed Object

We assume that all the correlation activity is done at the TEC level. The TEC rules decide when to open a new problem and when to submit calls attached to a previously created problem. So we use the event ID as the managed object and if we have to submit calls attached to a previous problem the event ID must be that of the event that first created the problem. To understand what this means, let's look at the following example:

1. The TEC receives a primary event and opens a new call, then a problem, specifying the event ID as the managed object.
2. A secondary event is received and is linked to the primary (that is, the cause event). When the secondary event is received the TEC may issue a new call to Advisor but will specify the managed object, the event ID of the primary event.

This way only one problem is created into the Advisor database, and the secondary event just opens a call attached to the previously created problem.

6.1.5.2 The External Problem ID

The external problem ID is the field that links a TEC event with the Advisor problem it generates. When a problem is closed the system can automatically close the event that generated that problem.

We used the TEC unique event identifier as the external problem ID.

6.1.5.3 Location and Caller

These items are the ID of the caller and in what location the caller is reporting the problem from.

Because of the fact that all events that are generated will come from TEC in this environment, we decided to set the caller to TEC and the location to be the System Management Operations Center (SMOC).

6.1.5.4 Call Code

The call code is used within Tivoli Advisor to specify what source the call comes from. Therefore, we thought it would be natural to set the different sources that exist in TEC as the call codes.

6.1.5.5 Severity

Mapping the different severities in TEC to the severities in Advisor is a fairly easy task. We chose to do it as shown in Table 20 on page 252:

Table 20. Mapping Severities between TEC and Advisor

Severity in Advisor	Description in Advisor	Severity in TEC
1	Emergency	Fatal
2	Important	Critical
3	Medium	Minor
4	Low	Warning
4	Low	Harmless / Unknown

6.1.5.6 System

The system is the higher level of abstraction in the SCIM structure. We assume that for a TEC event the higher level of abstraction is the source. Many events with different class names and slot values can come from the same source.

For system we do not use the source value as is, but we map that value to the specific platform or IT area. For example, LOGFILE will be UNIX or a source such as NV390* will be mapped on an OS390 value. NT will be NT and a source such as NV6K could be mapped to network. For Sentry other information is necessary such as the class name to identify the platform that is impacted.

Depending on the problem management policies and on how the support department is organized, the source could also be used just to identify whether the problem is hardware or software-related.

When a problem is opened from TEC the system field immediately categorizes the problem by identifying which skill is required, and the problem can be assigned to the correct support department.

6.1.5.7 Component

Since we assumed source as the system, it is natural for us to assume sub_source as the component. It happens sometimes that the sub_source slot is empty so we cannot use this information. In that case we just use the event source but other values could be used depending on the specific environment.

6.1.5.8 Item

Generally this item identifies a specific object. In our case the most specific information about a problem represented by the event is the class name.

We use the class name to map the item and then we need to create new SCIM entries for all the classes managed. An alternative could be to map the class names to items predefined in the SCIM.

Note that the item can only be 15 characters long. Because in our mapping the class name could be more then 15 characters, in the TroubleTicket.sh script we must truncate the class name to the first 15 characters. A more appropriate solution may be to customize the Advisor application to extend the field length.

6.1.5.9 Module

This can be left empty. If you have a specific need to have a hostname on a separate field (for example, for reporting purposes), we suggest you to put the value of the hostname slot here.

6.1.5.10 Inventory

We use the Tivoli object ID corresponding to the hostname to populate this field, as this is defined as an asset tag for the effected object in the migration procedure described in “Installation of Tivoli Inventory Integration Module” on page 76.

In order to map this field in the Tivoli Advisor call, we need to customize the default Tivoli NSM Gateway application. We describe the necessary steps for customization later in “Inventory Field Customization” on page 259.

6.1.5.11 Description

The description contains slots useful to describe the problem, especially those slots not mapped to any other field. So, the description could be like:

```
T/EC event info = HOSTNAME=XXXXX CLASS=XXXXX ORIGIN=XXXXX MSG=XXXXX
```

Where XXXXX represents the values of the corresponding TEC event slots.

We think that it is also good to have the description slots already mapped to other fields since adaptive learning uses the description field to learn about solutions (see “Diagnostic Aids” on page 121).

6.1.6 Program Configuration

In this section we show how to customize the products to practically achieve the integration and the data mapping described above. First we take a look at the Advisor and ExpertView Gateway customization.

6.1.6.1 Location and Caller

We have to create the location of the System Management Operations Center (SMOC) and to create the different contacts in that location:

In the Tivoli Advisor startup window, choose **Edit** -> **Locations**.

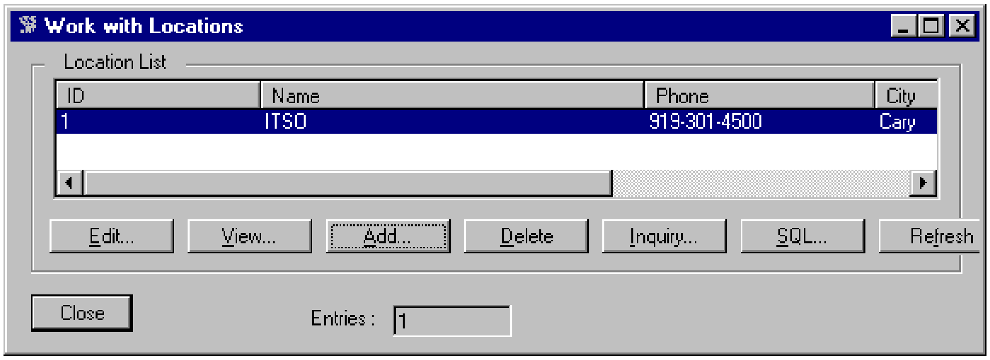


Figure 273. Work with Locations

To add the new location click on **Add**.

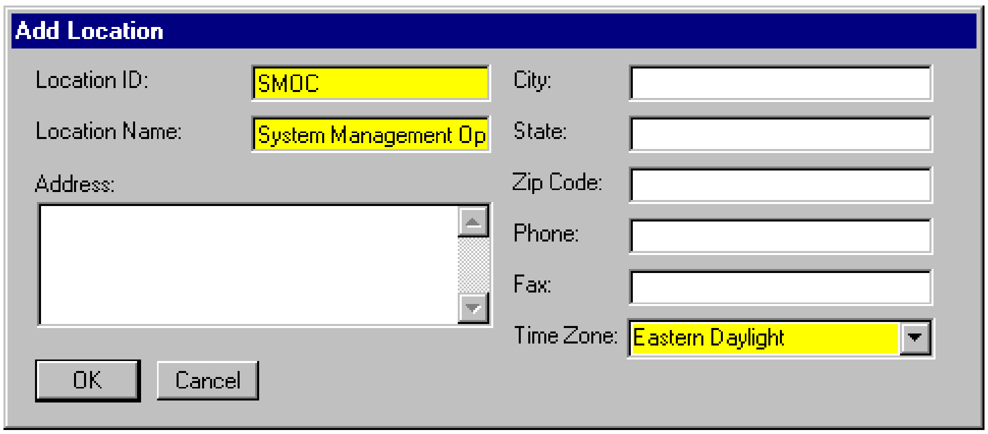


Figure 274. Add Location Window

We completed the required fields as shown in Figure 274 on page 254 then clicked on **OK**. The new entry has now appeared in the list of locations in the Work with Locations window.

To add a contact for this location we marked the entry and clicked on **Edit**.

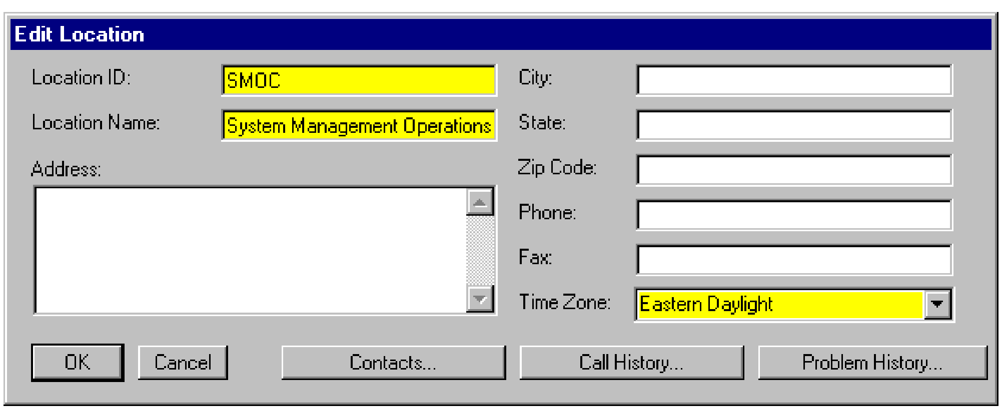


Figure 275. Edit Location

You can see that the difference between the Add Location window and the Edit Location window is that three buttons are added to the Edit Location window.

To add a contact for this location, click on **Contacts**.

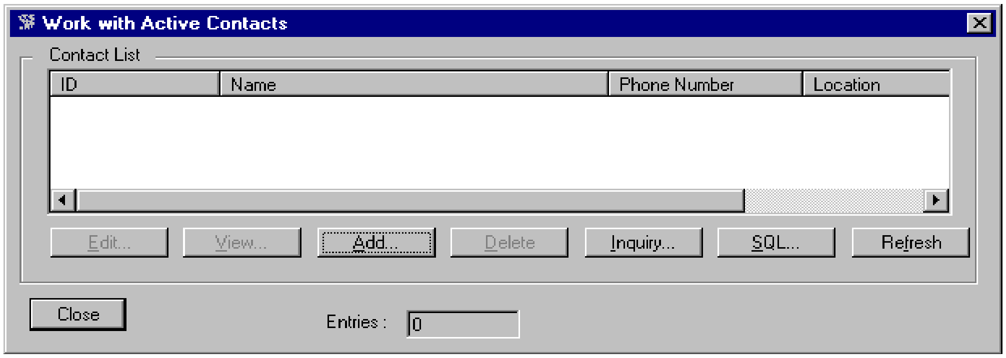


Figure 276. Work with Active Contacts

As you can see from Figure 276 on page 255 there is no contact entry yet. To create one, click on **Add**.

.

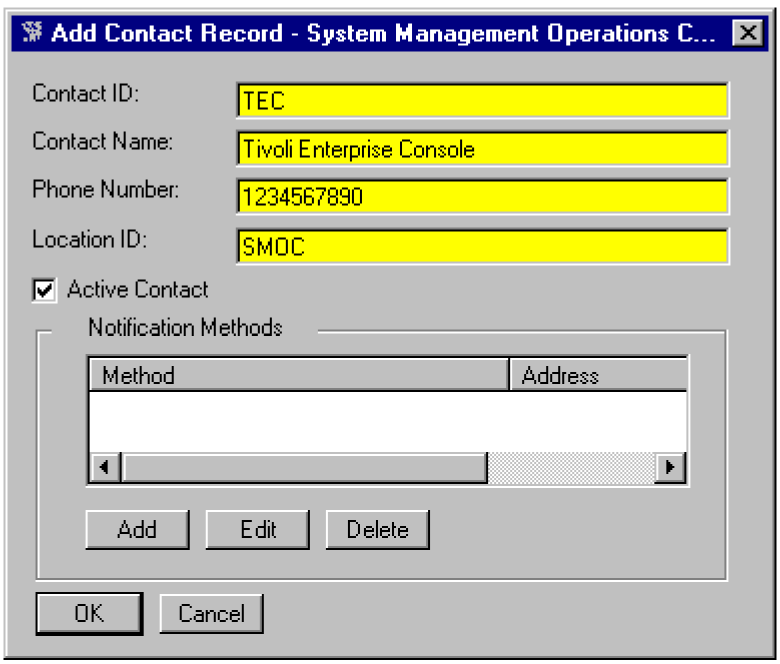


Figure 277. Add Contact Record - System Management Operations Center

We completed the required information. Notice that here you can configure a contact notification method by clicking on **Add** in the Notification Methods box. Since this user is a fictional one, we did not need to do this.

We now have the location and the caller that should be registered for all the calls generated from TEC. Now we can inform the Tivoli NSM Gateway to do this for all the TEC events.

To do this start the Tivoli NSM Gateway and log in.

From the startup window choose **Edit - System Configuration**. From the System Configuration window choose the **Open** tab.

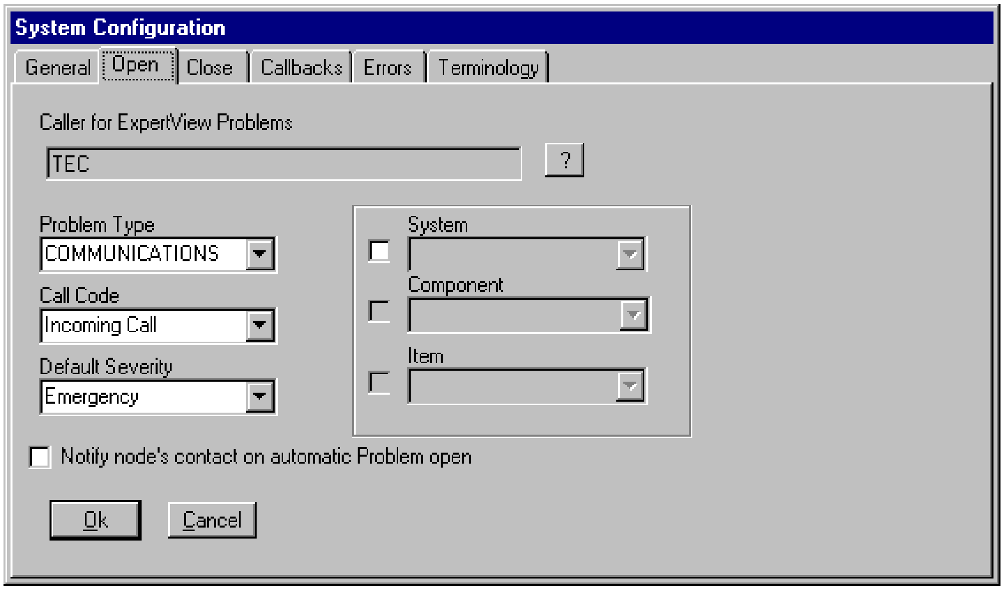


Figure 278. System Configuration

In Figure 278 on page 256 you can see that we have entered the user name that we previously defined in the caller for ExpertView Problems dialog. The gateway will now create all calls into Advisor with this caller name.

6.1.6.2 Call Code

To change, add or remove a call code in Tivoli Advisor, choose **Options -> Call/Problem** in the Advisor startup window. Then select the **Call Code** tab (see Figure 279 on page 256).

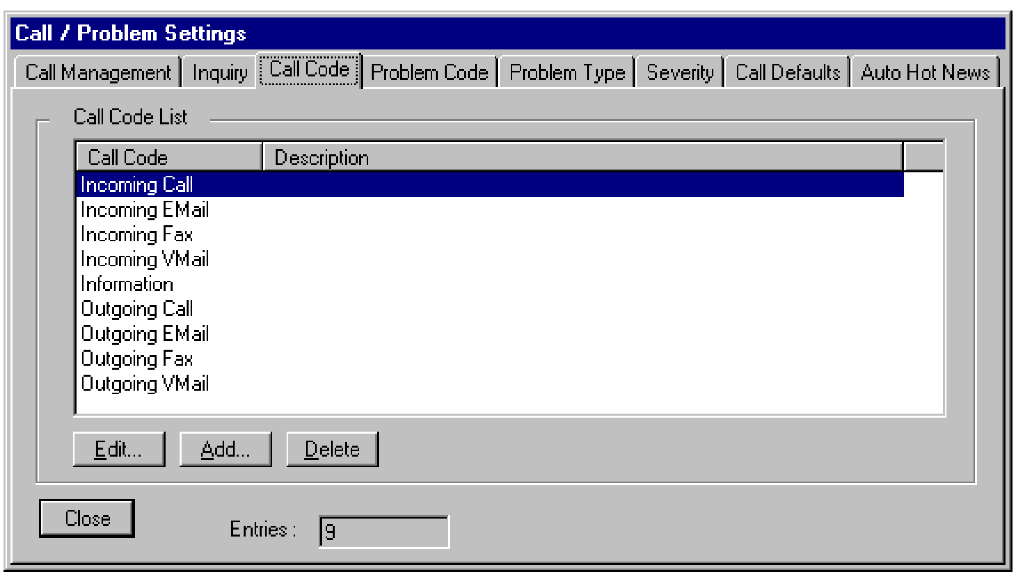


Figure 279. Call/Problem Settings, Call Code

To add a new call code, click on **Add**.

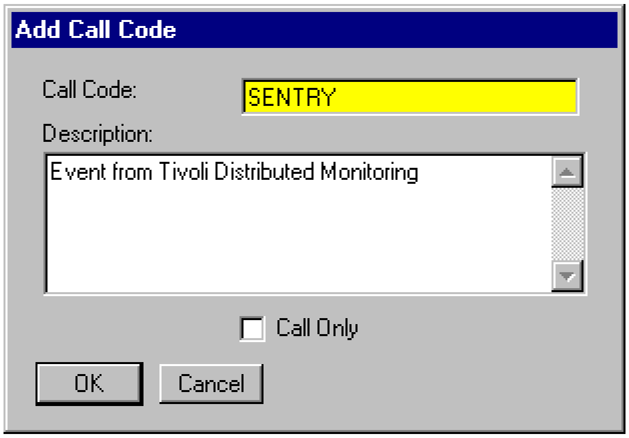


Figure 280. Add Call Code

Add SENTRY for the call code and click on **OK**. The new call codes definition are shown in Figure 281 on page 257.

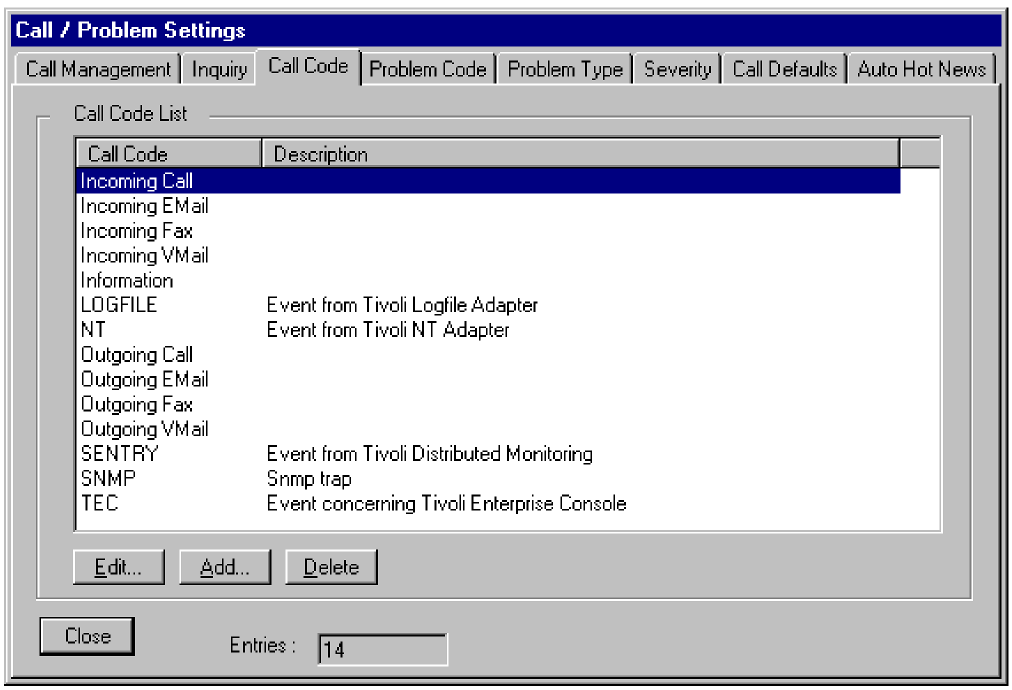


Figure 281. Call/Problem Settings with New Call Codes

6.1.6.3 Severity

We have already defined how the severity should be mapped. To clarify this we decided to add the severity from TEC to the description of the severities in Advisor.

From the Advisor window select **Options -> Call/Problem** and then click on the **Severity** tab (see Figure 282 on page 258).

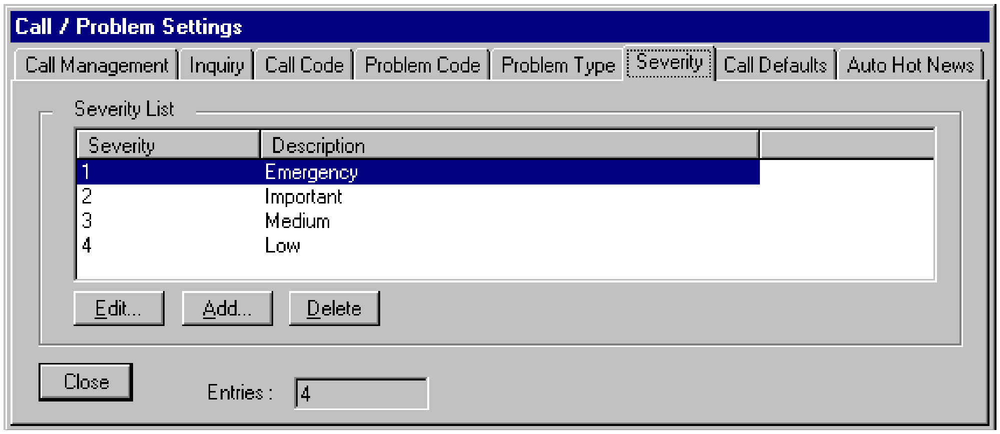


Figure 282. Call/Problem Settings, Severity

To edit the description, click on **Edit** (see Figure 283 on page 258).

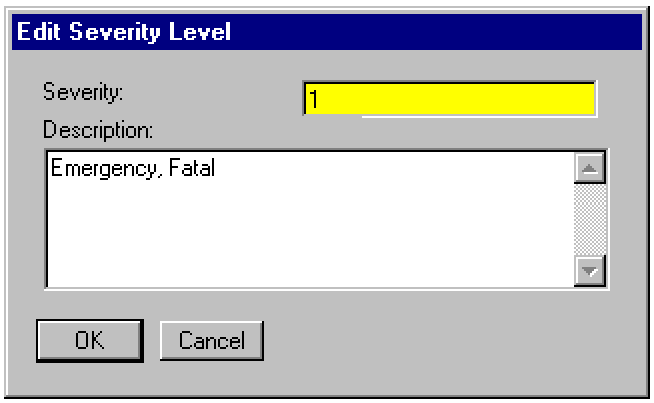


Figure 283. Edit Severity Level Window

As you can see here, we added the TEC severity after the original severity.

The result of all the changes are shown in Figure 284 on page 258.

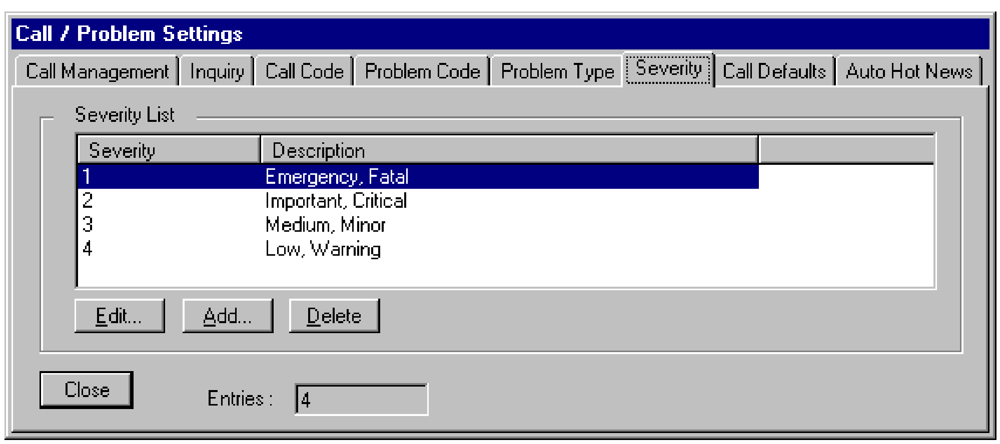


Figure 284. Call/Problem Settings, Severity with New Severity Descriptions

To get the correct mapping of the severity, we had to modify the severity.info file which is located in the installation directory of the PLUS module. This is how the file looked after the modifications were made:

```
Emergency,Fatal=1
Important,Critical=2
Medium,Minor=3
Low,Warning=4
Harmless,Unknown=4
```

Figure 285. Modified Severity.Info File

The problems will now be opened with the correct severity.

6.1.7 SCIM

The new SCIM entries have to be created for every value of system, component and item required by the mapping described earlier.

For example, suppose that we expect to open an Advisor call from an event having the source name of NV6K. This source value is mapped to network, so we have to create a SCIM entry when system is network. Of course we have to do the same for component, item and module if required.

For information on how to add new SCIM entries refer to “Defining a SCIM structure” on page 117.

6.1.8 Inventory Field Customization

As described in 6.1.1, “Understanding the Problem Management Application” on page 244, the EVProb command can use the -a argument for optional parameters which can be used to assign values to any column in the PROBLEM_VIEW view in the Tivoli ADVISOR database.

The problem we face here is in the fact that the PROBLEM_VIEW view also contains the column INVENTORY_ID from the SESSION table which links the problem to the asset in Tivoli Asset Management. This column cannot be used with the -a option by default.

We manage to solve this problem by changing the evobject.kb file on the NSM gateway machine wtr05368. This file provides routines for opening, closing and modifying problems in the Tivoli Advisor database based upon data supplied from the EVProb command. We modified the EV_UpdateProblemView procedure in this file to accept also INVENTORY_ID column values supplied by the EVProb command. We added the following lines in the procedure as shown in Figure 286 on page 260.

```
.
.
ELSWHEN 'OPEN_DATE' THEN
    probRec.open_date := {arg}:DATE;
    probRecModified := TRUE;
ELSWHEN 'OPEN_TIME' THEN
    probRec.open_time := {arg}:TIME;
    probRecModified := TRUE;
ELSWHEN 'INVENTORY_ID' THEN
     sesionRec.inventory_id := arg;
     sessionRecModified := TRUE;
--
--  Customization Point : Insert other Problem_View columns
--  here, if appropriate.
--NOTE TO CUSTOMIZERS - When adding a new column to
--this "switch" statement, you must assign the
--argument value into the appropriate PROBLEM,
--SESION, or CALL record ** AND ** set the
--appropriate "xxxRecModified" flag variable.
--
```

Figure 286. Added Lines in File vobject.kb

After changing the file we parsed the Tivoli Advisor in order for the changes to take effect.

6.1.9 Additional Customization

Throughout this publication we have assumed that the Advisor application has not been modified from its original version. Actually, it could be necessary to modify some Advisor attributes to perform appropriate data mapping. For example, to map the event class name to an Item field we cut the class name. A better solution could be to extend the Item field length. This could be the case of many fields.

It could also happen that slots presented in your customized classes, and relevant for your specific environment, do not have corresponding fields in the Advisor call record. In this case you may want to add new fields to the Advisor call structure to support these extra slots.

6.1.10 Script Customization

Data mapping is performed by the script TroubleTicket.sh. When this script is invoked, specific parameters have to be passed with a specific format.

Although the TroubleTicket.sh script is provided by the PLUS module, we need to modify this to implement the data mapping described in “Mapping Event Slots to Call Fields” on page 250.

We strip the values by using colons for separators for example, sed -e 's/://g'‘.

If any of the fields are blank, then we must assign a value. This will be when the script is called directly from the TEC screen.

The managed object is defined as:

```
MANAGEDOBJECT=$EV_KEY (The Event ID).
```

SYSTEM is mapped to the operating system platform.

The SOURCE variable is defined as:

```
LOGFILE    SYSTEM=Unix ;;
NV6K       SYSTEM=Network ;;
NV390*     SYSTEM=OS390 ;;
NT*        SYSTEM=NT ;;
SENTRY     $SOURCE
```

Set the COMPONENT to equal the SUN_SOURCE. If the SUB_SOURCE is NULL, then set COMPONENT to equal SOURCE:

```
COMPONENT=$SUB_SOURCE
```

The class name will be the ITEM. This must be cut to 15 characters only.

```
ITEM=$CLASS_NAME

ITEM=‘echo $ITEM | dd ibs=15 count=1 2>/dev/null‘
```

Inventory is set to the managed node ID. From the hostname we can resolve the object ID:

```
INVENTORY_ID=‘wlookup -ar ManagedNode | grep $HOSTNAME | cut -f 2‘
```

Set the Advisor DESCRIPTION field.

```
SHORT_MSG=‘echo $MSG | dd ibs=127 count=1 2>/dev/null‘

DESCRIPTION="\’T/EC event info = STATUS=$STATUS HOSTNAME=$HOSTNAME CLASS=$CLASS
ORIGIN=$ORIGIN MSG=$SHORT_MSG\’"
```

Finally we can call the EVProb to create the trouble ticket.

```
echo "EVProb -h $GATEWAY  -n $MANAGEDOBJECT $UNIQUE_ID -a
\"CALL_CODE:$CALL_CODE;${SEV}SYSTEM:$SYSTEM;COMPONENT:$COMPONENT;ITEM:$ITEM;IN
VENTORY_ID:$INVENTORY_ID;${MOD}DESCRIPTION:$DESCRIPTION\""  >> $WINFILE
```

We verify whether the ticket has been created successfully. If so, we need to look for the console name so that we can set the status of the event on the TEC to acknowledged, (ACK).

```
if [ x"$STATUS" = x"OPEN" ] ; then

ALI=‘wlookup TME_server‘

PRINCIPAL=‘objcall 0.0.0 o_get_principal‘

ADMIN=‘objcall $ALI get_principal_admin "$PRINCIPAL"‘

wsetemsg -t ACK pms_triggered=1 "@$EC" $EV_KEY
```

The slot values are shown in Figure 287 on page 262.

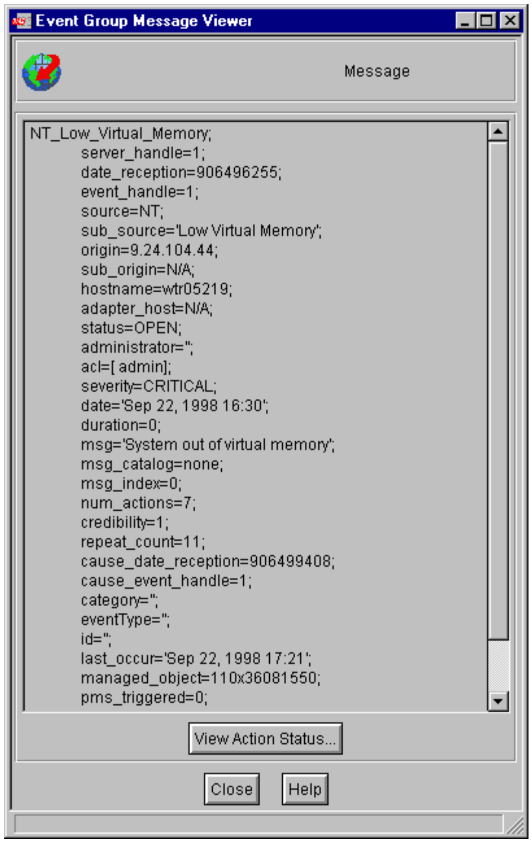


Figure 287. TEC Event Slot Definitions

Figure 287 on page 262 shows an example of the TEC slot definitions from the event message display.

Please refer to “The TroubleTicket.sh Script” on page 345 for more details about the script and its syntax.

Note that if you need to customize the TroubleTicket.sh script for your specific needs, there are comments in the scripts that show how to add your own customization code.

To close the TEC event when the corresponding Advisor problem is closed, the CloseTivoliTicket.sh is used (see “The CloseTivoliTicket.sh Script” on page 355). This script is invoked by the callback described in the next section.

6.1.11 Callbacks

When a problem generated by a TEC event is closed, we want the corresponding event to be closed too. To achieve this function the callback mechanism provided by the ExpertView Gateway can be used.

The gateway has the capability to call specific routines when a specific event happens. In our case we want to invoke the CloseTivoliTicket.sh script when a problem is closed. That script will close the corresponding cause event or events.

Figure 288 on page 263 shows how to customize the gateway to call the script when a problem is closed.

First log in to the gateway from the startup window, choose **Edit -> System Configuration,** and select **Callbacks.**

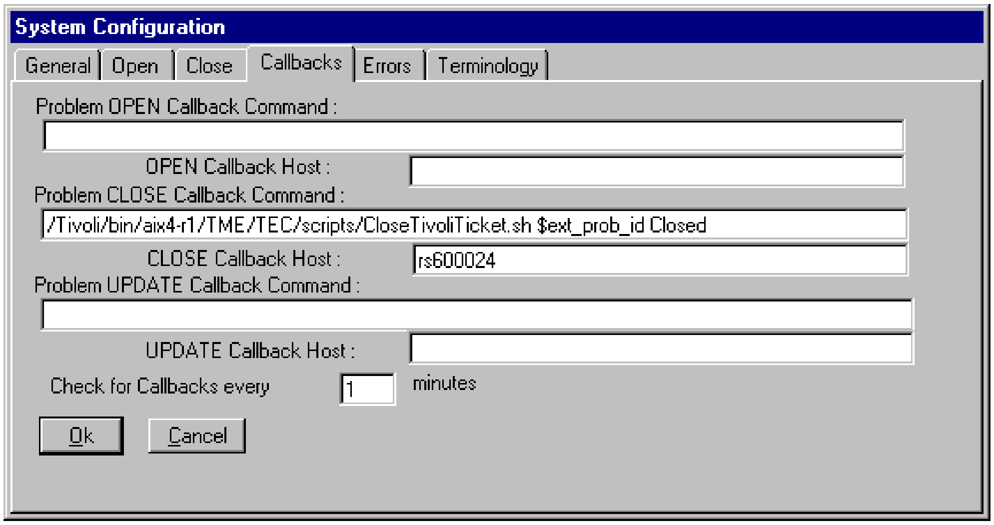


Figure 288. Setting the Callbacks

In the Problem CLOSE Callback Command field specify the full path of the CloseTivoliTicket.sh script passing $ext_prob_id as the input parameter. A second parameter is passed informing you that the event status is to be set. The script sets everything different from acknowledged to CLOSED.

The ext_prob_id parameter is the TEC event ID (see “The External Problem ID” on page 251). The script uses that parameter to issue the wsetemsg command to close the TEC event.

In the CLOSE Callback Host field specify the host where you want the script to run. The host machine must have the Application Software Expert installed. In our case the host is the same as the TEC server host (rs600024).

In the Check for Callbacks every field, we specify the time interval the gateway checks for callbacks to be issued. Note that the value of 0 minutes means no callbacks are to be invoked.

6.2 TEC Customization

In this section we describe all the TEC customization we had to perform in order to implement the scenario.

We do not describe the setting up of the profile managers, managed resources or policy regions.

First we define the problem we want to perform event correlation on. Then when a set of events occurs the TEC will create a problem ticket or call based on our defined policies.

6.2.1 The Problem

Windows NT uses the same concept of memory hierarchies as all modern operating systems do. It consists of three levels:

- Internal/external CPU cache memory
- External main memory (RAM)
- Paging area

The external main memory and the disk paging area is the virtual memory.

During the project we were very often confronted with the following system application pop-up produced by Windows NT, so we decided to create our problem scenario on that issue:

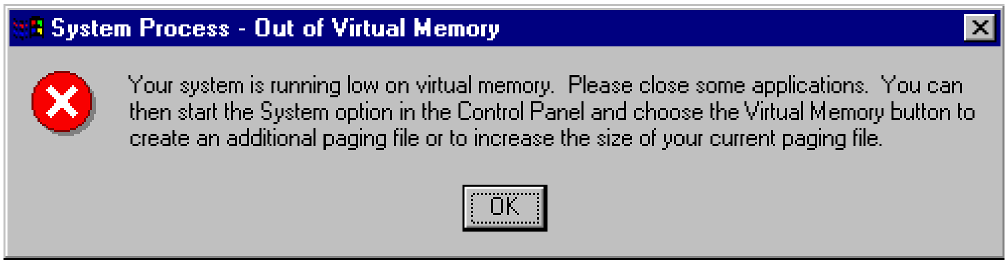


Figure 289. Application Pop-Up Message

On a server this can lead to serious problems. Windows NT starts allocation of memory in its physically available RAM as long as there is a certain minimum amount is available. However, it reserves an amount of about 4 MB for its own use and tries to keep this memory free.

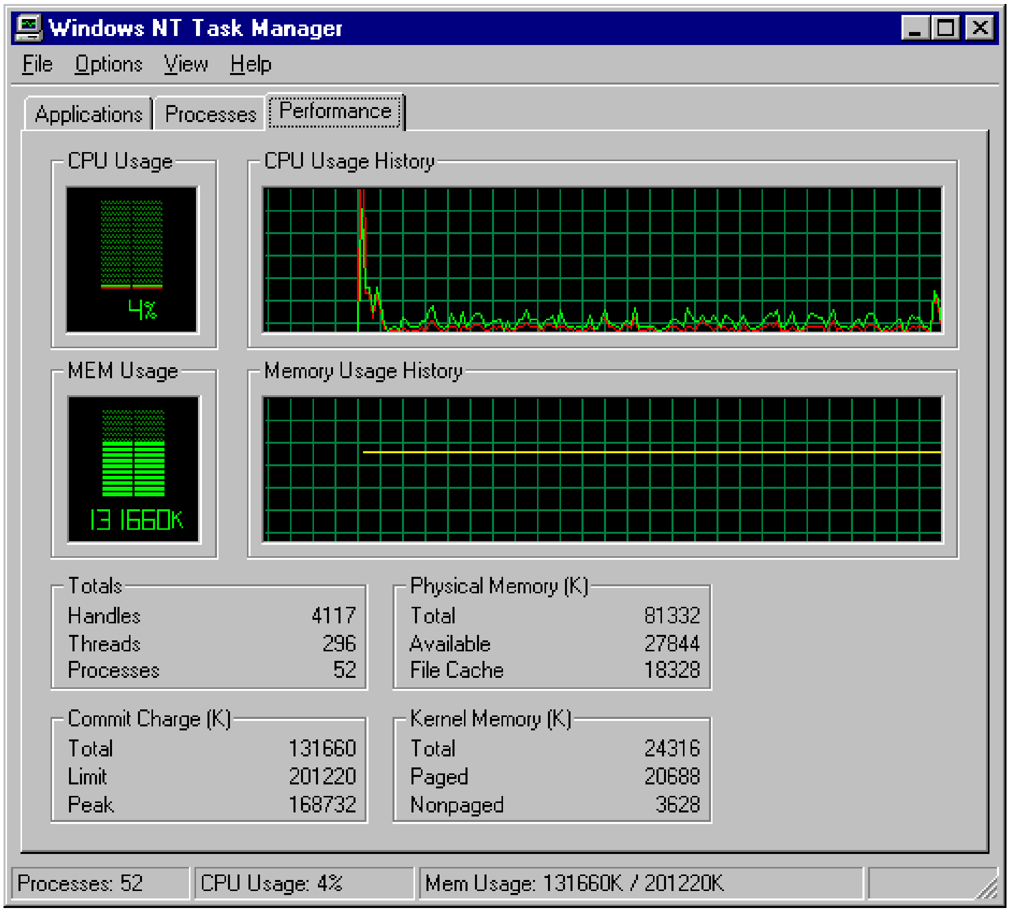


Figure 290. Windows NT Task Manager - Memory Usage

As a result the not used pages in memory get paged out to the disk area and the Commit Charge Total value starts to increase.

If this process continues, the value eventually touches the Commit Charge Limit resulting in the mentioned pop-up shown in Figure 289 on page 264.

Windows NT then rises this limit in steps to allow for more virtual memory that can be allocated. This can temporarily ease the problem, but still points to an amount of RAM that is too small to serve all the processes that request memory. The results when hitting the memory limit are unpredictable. The application that caused the RAM shortage may or may not start; other processes may or may not fail. The whole environment can be described as unstable.

As the amount of memory is the primary problem and the increase in used paging memory area is the consequence, we implement a standard monitor of the NT_memory collection, called Available Bytes and the TEC adapter for Windows NT to create TEC events in case this problem shows up.

To be able to make this scenario work, we need to bring the Windows NT Server into unattended mode. If this is not done, every pop-up window needs to be acknowledged to be entered in to the log, so that the TEC adapter can actually see it. This is done by editing the following registry value:

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Windows\ErrorMode

We changed this value to equal the value 2 in hexadecimal notation.

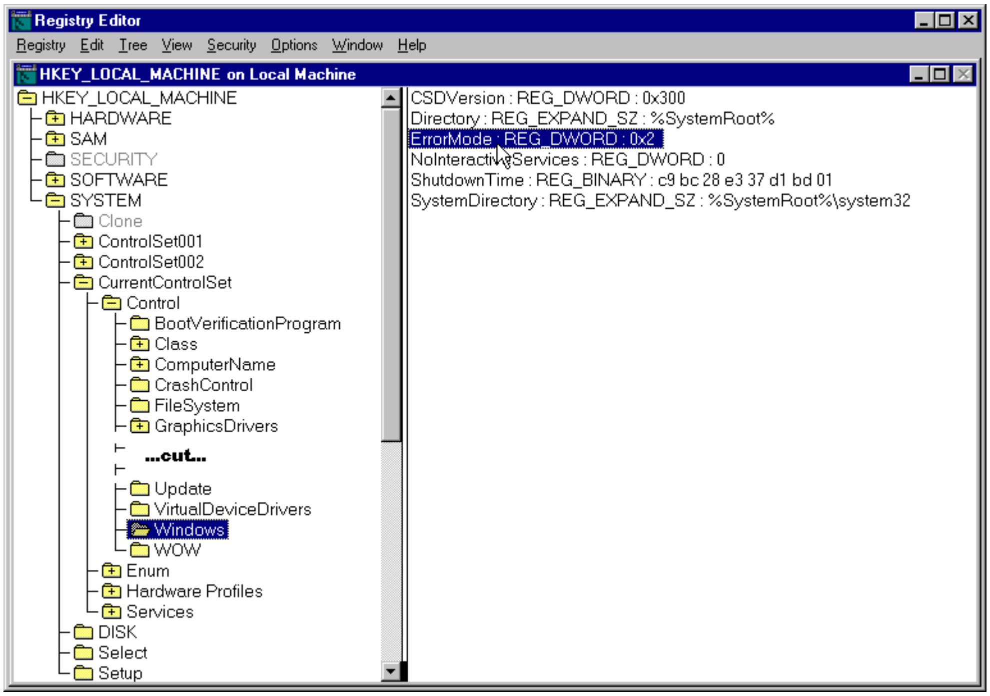


Figure 291. ErrorMode Configuration for an Unattended Windows NT Server

Figure 291 on page 266 shows the modification performed.

6.2.2 The Eyes

As the amount of memory is the primary problem and the increase in used paging memory area is the consequence, we will implement a standard monitor of the NT_memory collection, Available Bytes and the TEC adapter for Windows NT to create TEC events in case this problem shows up.

We configured distributed monitoring for NT memory monitors as shown in the following section.

6.2.2.1 Configuration for Distributed Monitoring

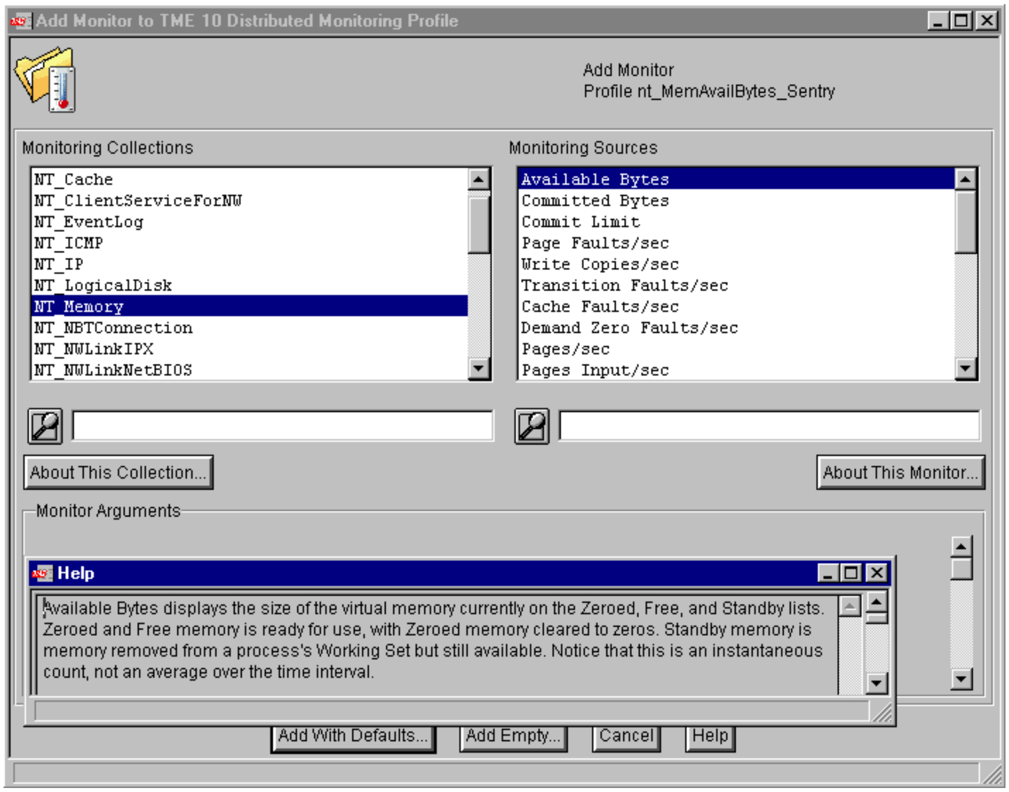


Figure 292. Monitor for the Available Bytes

The monitor is configured for measuring every 5 minutes. If the amount of available bytes drops below 5 MB, it sends an event of severity WARNING to the TEC. If the value rises above 12 MB (which we found out to be a reasonable value for our environment), it sends an event of severity HARMLESS to the TEC.

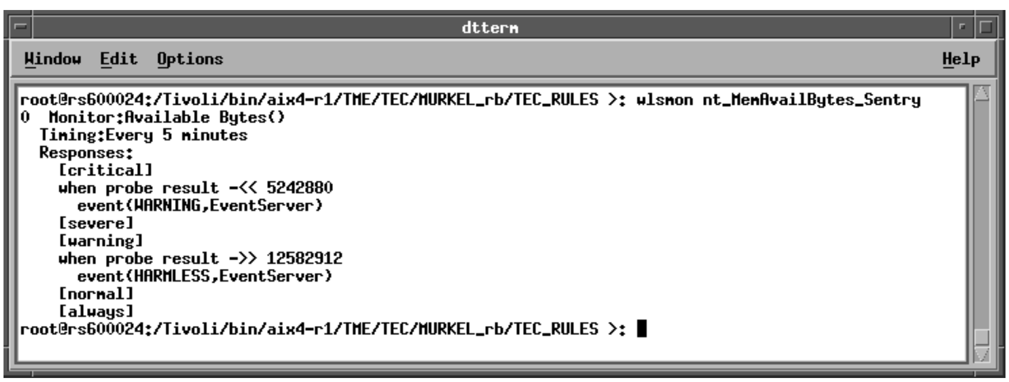


Figure 293. nt_MemAvailBytes Monitor Configuration

6.2.2.2 Configuration of the TEC Adapter for Windows NT

Note

To deploy the TEC adapter for Windows NT, all the Tivoli Management gateways need to have the Enterprise Console Adapter Configuration Facility installed, while the.fmt file (in our case the tecad_nt.fmt file) that gets distributed is the one on the TMR server.

The application pop-up in Figure 289 on page 264 creates an informational entry in Windows NT's event logs (system, security and application). By configuring the TEC adapter, we can extract the information from this log and forward it to TEC.

Note

Make sure to configure the Windows NT event logs to overwrite messages as needed. If the event log fills up, no events are written to the logfile anymore; therefore no error messages are routed to the TEC. Choose an appropriate size for your environment. In our environment we did not need to change the default size of 512 KB per log.

Create a new profile of type ACP and open it.

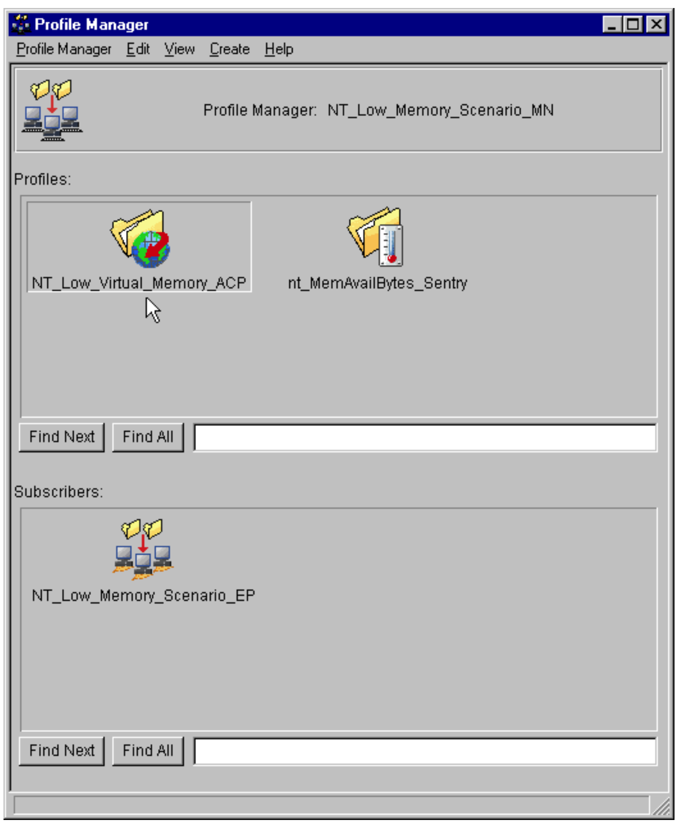


Figure 294. Create an ACP Profile

Open the profile and add the type tecad_nt.

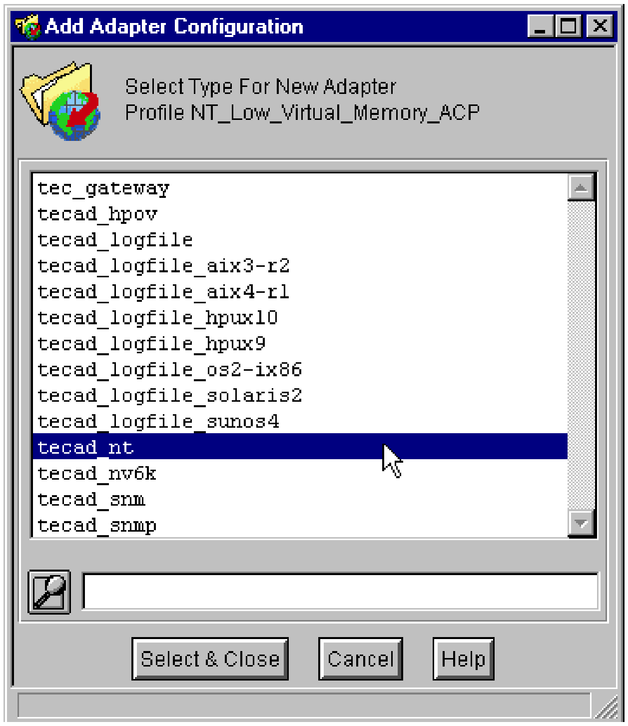


Figure 295. Adapter Configuration Facility

Figure 296 on page 269 is displayed.

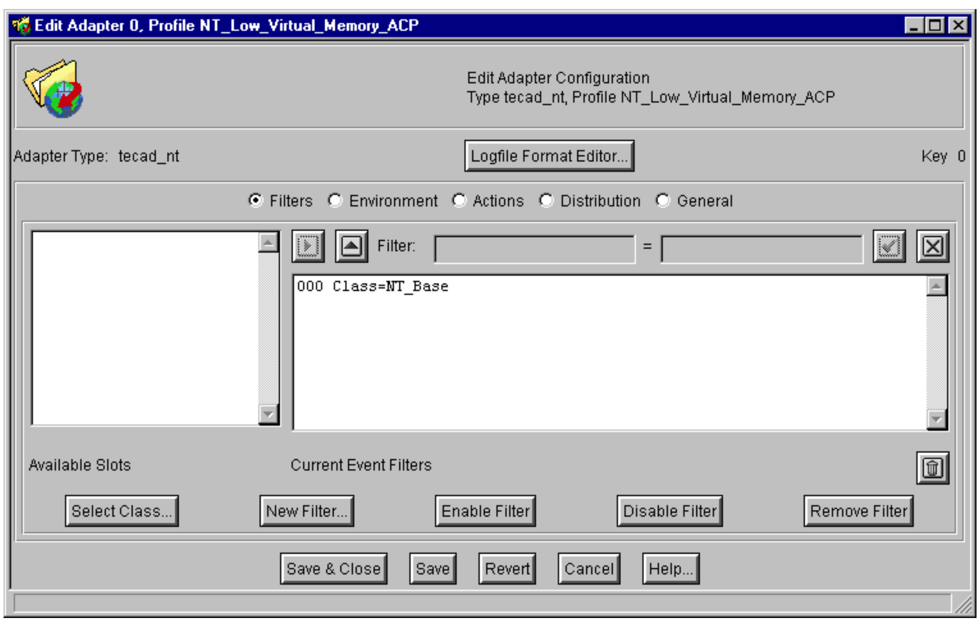


Figure 296. Edit Adapter Configuration

For testing purposes it is useful to disable the NT_Base class filter, so you can see every event that is routed to the TEC, even if the format is not correctly parsed. This can happen when the format file tecad_nt.fmt does not fit the message format. We experienced this particular problem with the

NT_Low_Virtual_Memory class. The application pop-up telling you about the system being low on virtual memory is sent as this class type.

Initially the events were sent to the TEC in the NT_Base class. This happened because of the.fmt file, which has the wrong base to parse the Windows NT event log.

The following extracts from the tecad_nt.fmt file show the changes we had to do to actually get the NT message in the correct class.

The original entry in the.fmt file looked like this:

```
FORMAT NT_Low_Virtual_Memory FOLLOWS NT_Base
%t %s %s %s %s %s %s Out of Virtual Memory - Your system is running low on virtual memory.
Please close some applications. You can then start the System option in the Control Panel
and choose the Virtual Memory button to create an additional paging file or to increase
the size of your current paging file.
END
```

Figure 297. Extract from the Original tecad_nt.fmt File

We changed the entry to the following statement:

```
FORMAT NT_Low_Virtual_Memory FOLLOWS NT_Base
%t %s %s %s %s %s Popup %s Application popup: System Process - Out of Virtual
Memory : %s+
END
```

Figure 298. Changed Entry for NT_Low_Virtual_Memory

In addition we had to modify the type of the slot ID in the class definition file tecad_nt.baroc because after the events finally arrived in the correct class their parsing failed because the data type of ID is actually of type STRING while INT32 was expected, (see Figure 211 on page 214)

6.2.3 The Events

Table 21 on page 271 shows the events we used with their associated classes and descriptions. Be aware that the event in the nt_MemAvailBytes class serves as the primary event (sent to TEC in severity WARNING) as well as the clearing event (sent to TEC in severity HARMLESS). Both are generated from the same monitor. It is of course possible to create an event in its own class, for example nt_MemAvailBytes_cleared, but the intention of this book is to show the TEC integration into a problem management solution; therefore we did not focus on creating our own event classes or custom monitors. To get information on how to create a custom monitor, refer to the redbook *Creating Custom Monitors for Tivoli Distributed Monitoring,* SG24-5211.

Table 21. Event Listing

TEC Event Class Name	Event Description
nt_MemAvailBytes (Severity WARNING)	Indicates that there is very little free RAM memory left on the NT server.
NT_Low_Virtual_Memory	Indicates that the paging space on the NT server has reached its limit.
nt_MemAvailBytes (Severity HARMLESS)	Indicates that the problem on the RAM memory is cleared.

For a more detailed description of when the events are sent, refer to 6.2.1, “The Problem” on page 263.

6.2.3.1 The Event Relationship

The following event relationship diagram shows the causal relationship of the three events. It does not show the incoming order of the events nor does it imply that the cause event must have an effect event or vice versa.

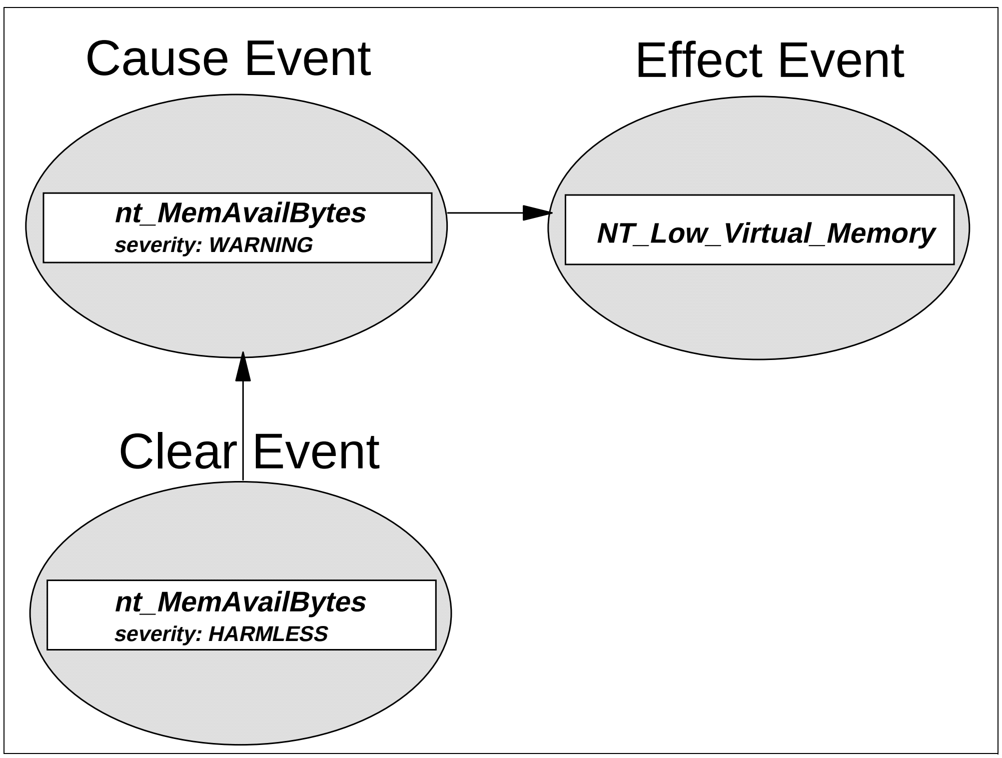


Figure 299. Event Relationship Diagram NT Memory Shortage

Depending on how the timing of the monitor probes is chosen, it may happen that one or more NT_Low_Virtual_Memory events show up, but no nt_MemAvailBytes event is triggered by Tivoli Distributed Monitoring. In contrast it may happen that the amount of available physical RAM memory is below 5 MB and stays there for a certain time, but the paging file never hits its limit.

All these uncertainties make for a good example to show the TEC in such an environment showing the capabilities of the rules engine.

To make sure that the problem management Level 1 and Level 2 support people only get the really important and critical events, correlation needs to be done. Actually there are multiple levels of correlation in most implementations. The first level mostly is NetView or another SNMP manager. The second one is usually the TEC and in our environment we even have a third level of correlation, which is done in Software Artistry's ExpertAdvisor. This will become clear in the course of this chapter.

6.2.4 Tivoli NetView Events

The redbook *TEC Implemention Examples*, SG24-5210 contains a number of rules and event relationship diagrams for SNMP generated events sent from TIvoli NetView. The rules contained in this redbook can be modified in order to fit into our environment and create problem management tickets. The options that require change would be:

- Class definition and slot values
- Replace the trouble ticket scripts called with the names referenced in this redbook

With little effort the examples could be integrated with this environment.

6.2.5 The Rule Policies

The first thing you need to code rules in TEC is a straight catalog of requirements (policies) that result from the output of the event methodology.

Under consideration of the unpredictable behavior that results from the timing issues with the probes and the Windows NT memory management, we decided on a list of policies, that take into account that memory usage peaks can be caused by a faulty process or a pure coincidence, but do not necessarily point to a serious problem. However, if there is a persistent problem, the repeat counters will rise and the events will be triggered. Furthermore we decided not to implement timers as this kind of problem is not necessarily clear in a few moments, but usually takes some hours, though we have also seen it arising in less than 5 minutes.

The policies for our sample event scenario are the following:

- nt_MemAvailBytes (WARNING) is considered to be the primary event.
- NT_Low_Virtual_Memory is considered to be the secondary event.
- nt_MemAvailBytes (HARMLESS) is considered to be the clear event.
- Update the severity of the primary event from WARNING to CRITICAL if two or more secondary events have been received.
- Create a trouble ticket, when:
  - Three or more secondary events and at least one primary has been received.
  - SIx or more secondary events and no primary has been received.
  - Three or more primary events has been received (whether or not secondaries exist).

- One clear event closes the primary and secondary events.
- Do not show the HARMLESS clear event.
- Make sure, that no matter what event comes first, only one trouble ticket gets created and significant changes are appended to this ticket.
- TEC is *not* allowed to change the status of a trouble ticket; it only creates or updates it.
- Closing of a trouble ticket should close the corresponding TEC events.
- Inform the TEC if a trouble ticket has been opened and if the creation was successful.
- No duplicate events are ever shown on the TEC console.

The number of repetitions of the NT_Low_Virtual_Memory event is highly dependent on the environment. For our environment (with no more than two or three users per NT server), the chosen thresholds were fine, but we have seen these errors showing up in system logs of other NT servers in fairly large amounts.

To keep the amount of updates that are sent to the trouble ticket down to a reasonable amount (they should warn you, but not keep the TEC or the problem management system busy with counting the duplicates or inserting updates), a closer look at the repeat_count comparisons of NT_Low_Virtual_Memory might be necessary.

6.2.6 The Event Class Definitions

The integration with the Advisor requires a unique identifier to enable it to correlate calls either to the cause problem (updated or closed events). Expert Advisor refers to the managed object as the identifier for a trouble ticket (problem). We added two new slots to the two baroc files that were affected by the change. The two files are ntMemory.baroc and tecad_nt.baroc.

```
*********** ntMemory.baroc start***************************
TEC_CLASS :
NT_Memory ISA Sentry3_5_Base
DEFINES {
managed_object: STRING;
last_occur: STRING;
};
END
TEC_CLASS :
nt_MemAvailBytes ISA NT_Memory;
END


*********** ntMemory.baroc end ****************************

*********** tecad_nt.baroc start **************************
TEC_CLASS :
NT_Base ISA EVENT
        DEFINES {
                source: default= "NT";
                sub_source: default= "NT";
                sub_origin: default= "N/A";
hostname:
dup_detect=YES;
                adapter_host: default= "N/A";
                msg_catalog: default= "none";
                msg_index: default= 0;
                repeat_count: default= 0;
category:STRING;
eventType:STRING;
sid:STRING;
#idINT32;
id STRING;#see
                managed_object:STRING;
last_occur: STRING;
};
END

...cut...
*********** tecad_nt.baroc end ****************************
```

Figure 300. Excerpts of ntMemory.baroc and tecad_nt.baroc Files

The event slot managed_object will contain the unique problem identifier (called external identifier in ExpertAdvisor) which is a simple concatenation of the slots event_handle, server_handle and date_reception.

Note that the slot managed_object is of data type STRING, despite the slots event_handle, server_handle and date_reception being defined as integer slots (INT32 or INT). This is because the bo_set_slotval template is used to set the managed_object slot.

As this template just concatenates the integer values to a string, it will not work if the data type of the managed_object slot is INT or INT32. However, this does not influence the conversion of the hexadecimal noted date_reception slot to the decimal value that finally shows up in ExpertAdvisor as the external ID.

The slot last_occur will be used to capture the date of an incoming duplicate event.

Major implementations call for major changes, therefore we added an additional event slot called pms_triggered of data type INT to the base event in the root.baroc file. This is necessary because all events that are routed from the TEC to ExpertAdvisor need to have this slot to be able to determine whether a problem ticket on their behalf has been opened or not.

6.3 The Rule Flowcharts and Rules

The three rule files developed are called:

- NT_Low_Virtual_Memory.rls
- NT_Mem_AvailBytes.rls
- NT_MemAvailBytes_Clear.rls

All the code for the rules are contained in Appendix A.2, “Rulesets” on page 356.

6.3.1 Ruleset NT_Low_Virtual_Memory.rls

The flowcharts for this rule are shown in Figure 301 on page 275.

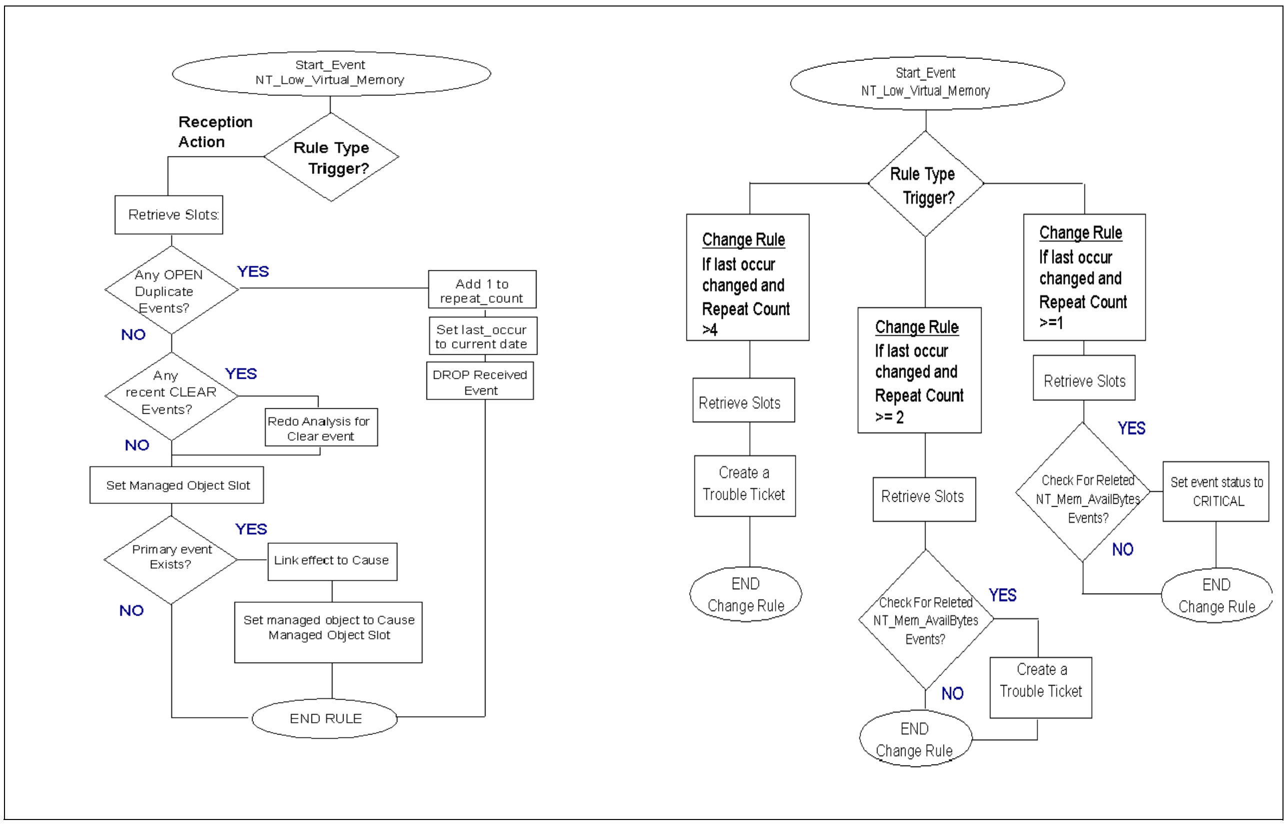


Figure 301. NT_Low_Virtual_Memory.rls Flowchart

This ruleset is comprised of four rules, one regular rule and three change rules. It handles all events of the class NT_Low_Virtual_Memory.

The regular rule is outlined below. If an event of this class is received, it will:

- Check for a duplicate and if it finds one:
  - It adds to its repeat_count, places a change request for its slot last_occur according to the date of the received event, drops the received event and quits all actions in this rule.
- Check for an instance of a recent clear event (60 seconds past/future) and if it finds one:
  - It triggers the re-analysis of this event.

- Set the managed_object slot to a concatenation of event_handle, server_handle and date_reception.
- Check for an instance of the primary event nt_MemAvailBytes, severity outside HARMLESS and if it finds one:
  - It links the received event to the found instance event.
  - Places a change request for its slot managed_object according to the managed_object value of the first_instance event.
  - Sets the status of the received event to the status of the first_instance event.

Change rules are:

1. If an event of this class is received that contains a change request to slot last_occur, the repeat counter is equal to or greater than 1 and has a status outside CLOSED or ACKNOWLEDGED it:
   - Checks for an instance of the primary event nt_MemAvailBytes, status outside CLOSED and a severity of WARNING and if it finds one:
     - Sets the first_instance of the event severity to CRITICAL.
2. If an event of this class is received that contains a change request to slot last_occur, the repeat counter is equal or greater than 2 and has a status outside CLOSED or ACKNOWLEDGED it:
   - Checks for an instance of the primary event nt_MemAvailBytes, status outside CLOSED and a severity of WARNING and if it finds one:
     - Creates a trouble ticket from the first_instance event.
3. If an event of this class is received that contains a change request to slot last_occur, the repeat counter is equal or greater than 5 and has a status outside CLOSED or ACKNOWLEDGED, it does *not* look for a primary event, but instead will:
   - Set its own severity to CRITICAL.
   - Send its own data off as a trouble ticket.

6.3.2 Ruleset NT_MemAvailBytes.rls

This ruleset contains four rules. The flowcharts for these rules are shown in Figure 302 on page 277.

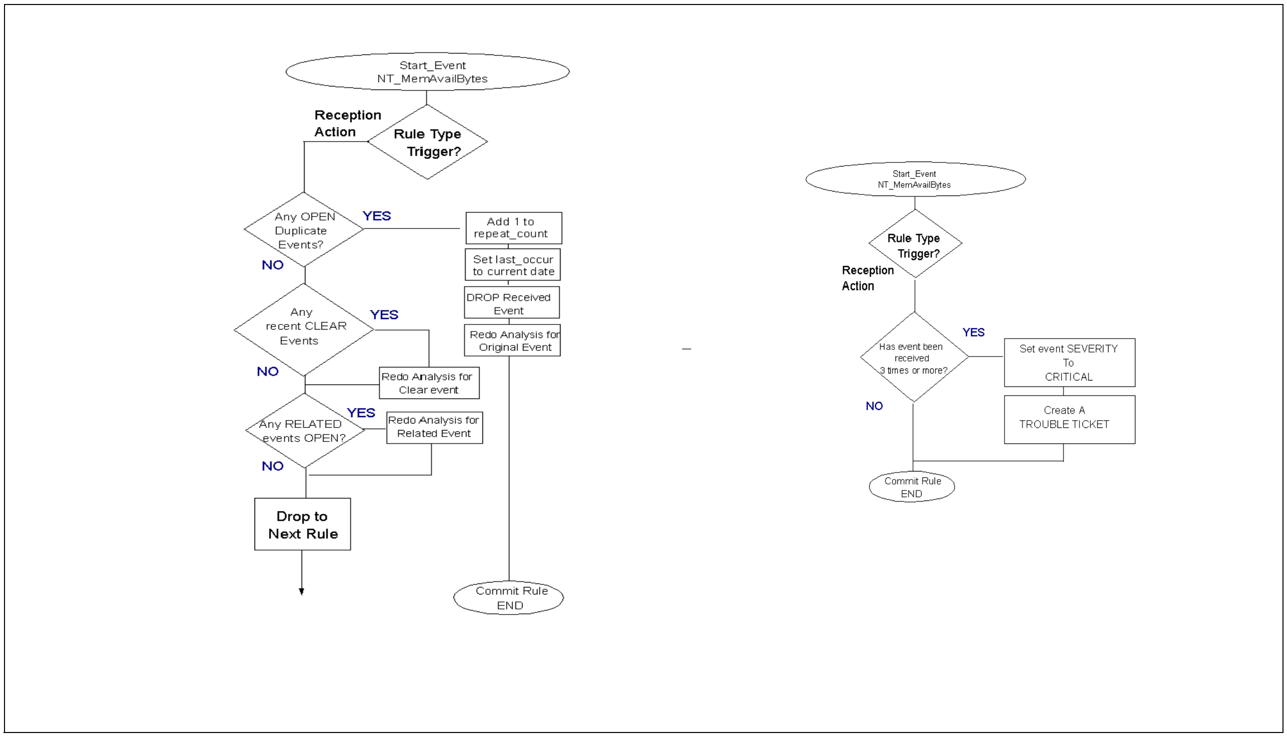


Figure 302. Flowchart Showing the Rules for MemAvailBytes.rls

The change rules are shown in Figure 303 on page 277.

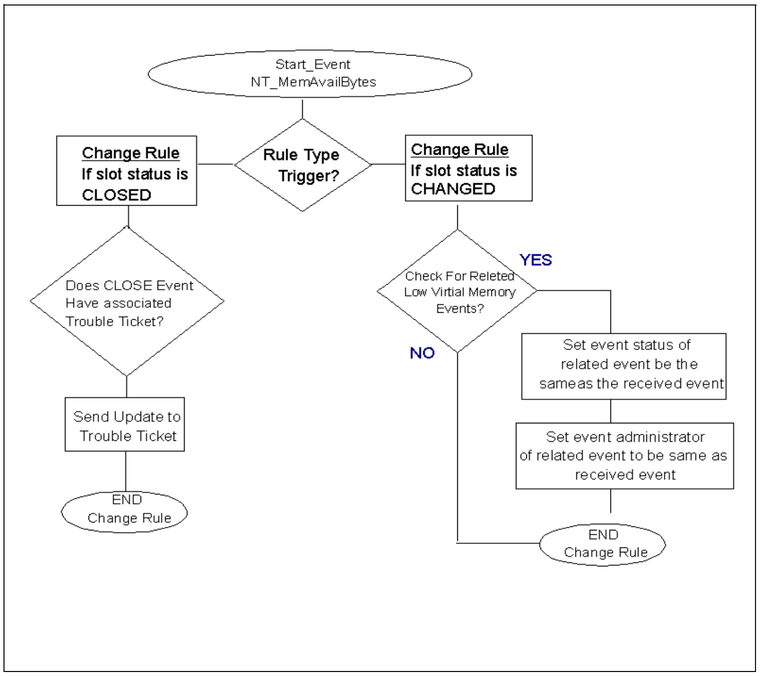


Figure 303. Flowchart Showing the Change Rules for Mem_AvailBytes.rls

This ruleset consists of four rules, two normal rules and two change rules.The rules handle all events of the class nt_MemAvailBytes with a severity outside HARMLESS.

1. If an event of this class is received, it will:
   - Check for a duplicate and if it finds one, it will:
     - Add to its repeat_count.
     - Place a change request for its slot last_occur according to the date of the received event.
     - Request a re-analysis of the found duplicate event (to check its repeat_count in the next rule).
     - Quit all remaining actions in the rule.
   - Check for an instance of a recent clear event (60 seconds past/future) and if it finds one, it will trigger the re-analysis of this event.
   - Checks for an instance of the secondary event NT_Low_Virtual_Memory and if it finds one, it will trigger the re-analysis of this event.
   - Set the managed_object slot to a concatenation of event_handle, server_handle and date_reception.

The second rule will check for:

2. If an event of this class is received, it will:
   - Check the repeat_count and if it equals 2 or greater create a trouble ticket.

Change rules are:

1. If an event of this class is received, that contains a change request to slot status and has a status outside HARMLESS, it will:
   - Check for an instance of the secondary event NT_Low_Virtual_Memory and if it finds one:
     - Set the event's severity of the first_instance event according to the received events' severity.
     - Set the event's administrator of the first_instance event according to the received events' administrator.
2. If an event of this class is received that contains a change request for slot status to CLOSED, the slot pms_triggered equals 1 (meaning: it has a corresponding trouble ticket), has a status outside CLOSED and a severity outside HARMLESS, it will update the trouble ticket. (It does *not cl*ose the trouble ticket by policy.)

6.3.2.1 Flowchart for Nt_MemAvailBytes_clear.rls

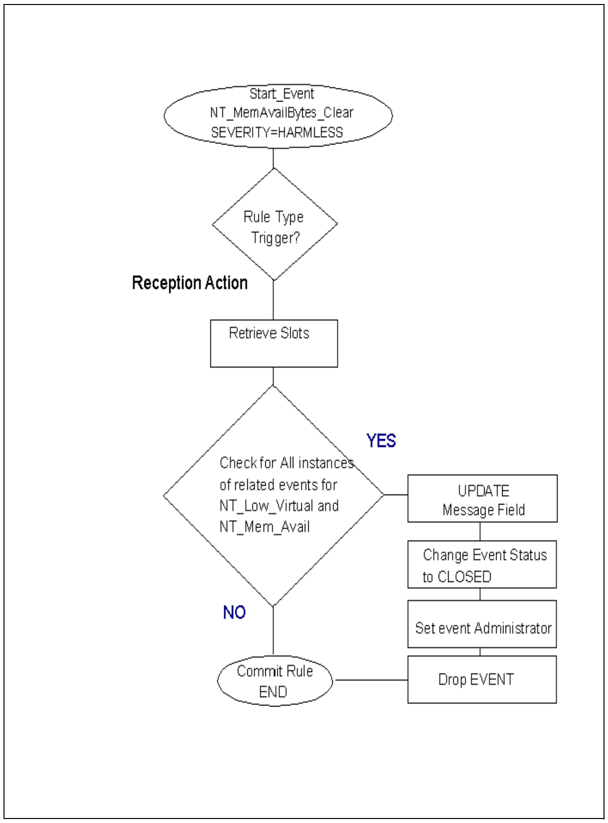


Figure 304. Flowchart for NT_MemAvailBytes_Clear.rls

This ruleset contains one rule that will handle all events of the class nt_MemAvailBytes in severity HARMLESS.

If an event of this class is received, it will:

- Check for all instances of the classes nt_MemAvailBytes and NT_Low_Virtual_Memory with a status outside of CLOSED and a severity outside HARMLESS and if it succeeds, it will:
  - Concatenate the message of the incoming event and the original message and sets the words Problem resolved in front of the term.
  - Place a change request to change the status slot value to CLOSED (to trigger the change-rule that keeps the secondary events in track).
  - Set the administrator to be a non-existent administrator to make it obvious that it has been closed by TEC.
  - Drop the received event.

6.3.3 Compiling the Rules

We imported the rules into our current rulebase using the command wimprbrules. Our rule_base file is shown below:

```
rule_set('NT_Low_Virtual_Memory', 'NT_Low_Virtual_Memory.rls', active).

rule_set('nt_MemAvailBytes', 'nt_MemAvailBytes.rls', active).

rule_set('nt_MemAvailBytes_clear', 'nt_MemAvailBytes_clear.rls', active).

rule_set('saexpertise_monitors', 'saexpertise_monitors.rls', active).

rule_set('sa', 'sa.rls', active).
```

The additional rulebases saexpertise_monitors and sa.rls are provided with the problem management application.

We used the command shown below to compile the new rule base files:

```
wcomprules -t NT_Low_Virtual_Memory.rls MURKEL
```

We ran the wcomprbrules command for all the defined rulebases:

```
wloadrb -u MURKEL
```

6.3.4 Testing the Rules

We created the policy region called 2511-Scenario with the resources shown in Figure 305 on page 280.

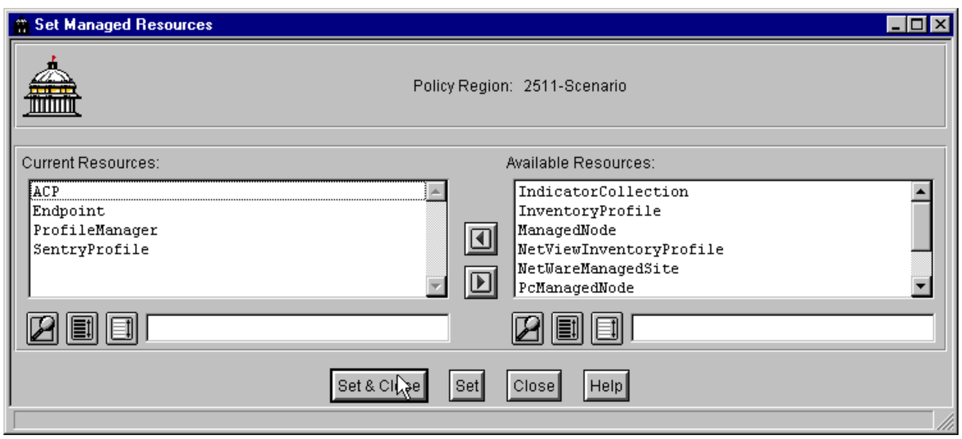


Figure 305. Resources of the Region 2511-Scenario

In this region, we created two profile managers. We only describe the setup of the profile manager NT_Low_Memory_Scenario_EP.

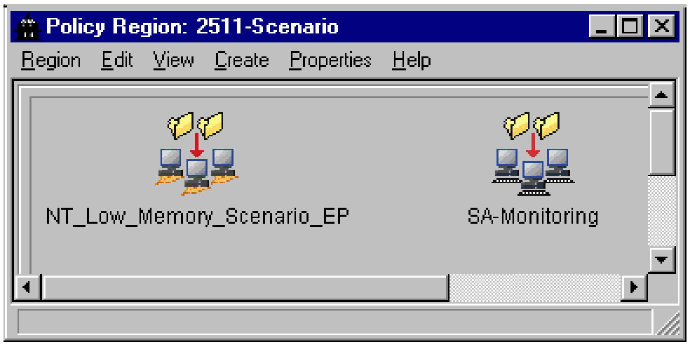


Figure 306. Contents of the Policy Region 2511-Scenario

As we wanted to use the new Tivoli Version 3.6 features, we created it in dataless endpoint mode.

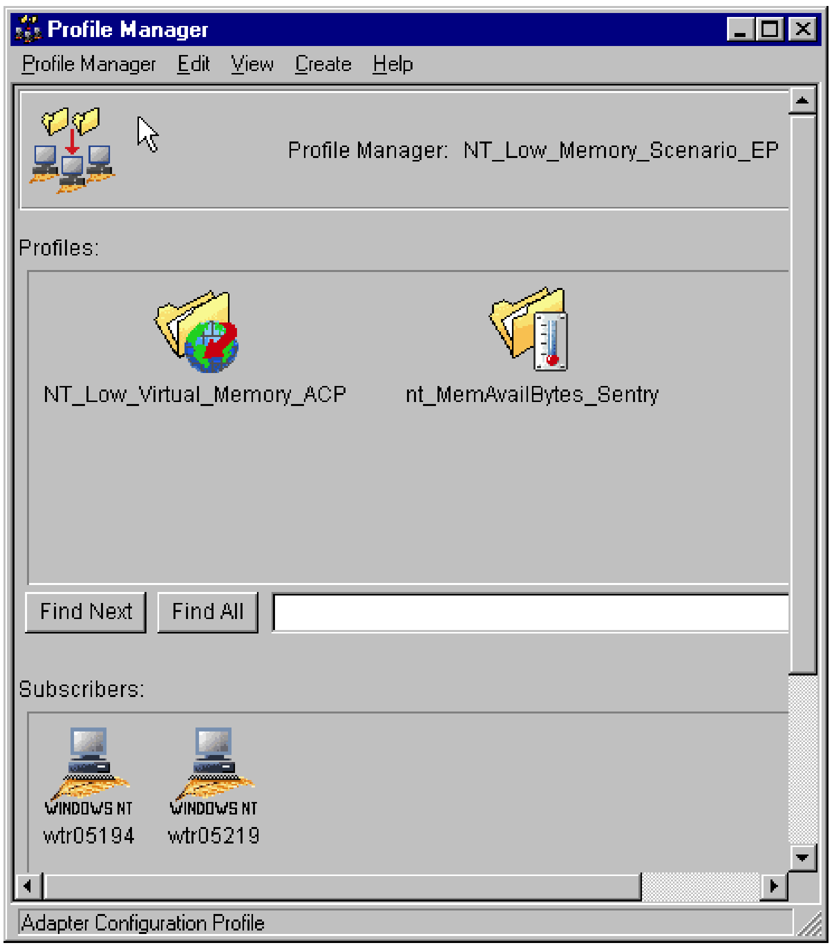


Figure 307. Contents of Profile Manager NT_Low_Memory_Scenario_EP

We then created the ACP profile, the Distributed Monitoring profile and its monitors according to the specifications described in Chapter 6.2.2, “The Eyes” page 266 and distributed it to the endpoints.

We experienced some minor problems with the distribution of the nt_MemAvailBytes_Sentry profile to the endpoints, but usually they could be solved by issuing the command a wclreng -z <endpoint-name>. You only need to add the parameter -z if there is a name conflict between a managed node and an endpoint (which was the case in our environment).

More serious problems showed up when after distributing the profiles, we received the message that the TECNTAdapter had been shut down, which of course would prevent a thorough testing of our environment. We propose to see such effects as related secondary events to the primary nt_MemAvailBytes event and to implement appropriate procedures to prevent an undetected loss of an adapter.

Our test events are listed below:

```
wpostemsg -r WARNING -m "System out of virtual memory" hostname=wtr05219
sub_source="Low Virtual Memory" NT_Low_Virtual_Memory NT

wpostemsg -r WARNING -m "Available Bytes below 5MB" hostname=wtr05219
sub_source="NT Available Bytes" nt_MemAvailBytes SENTRY

wpostemsg -r HARMLESS -m "Available Bytes back up again above 12MB"
hostname=wtr05219 sub_source="NT Available Bytes" nt_MemAvailBytes SENT
```

We monitored the TEC event log (wtdumprl) and the rule trace to confirm that our events are being sent to the TEC as per our rule policies.

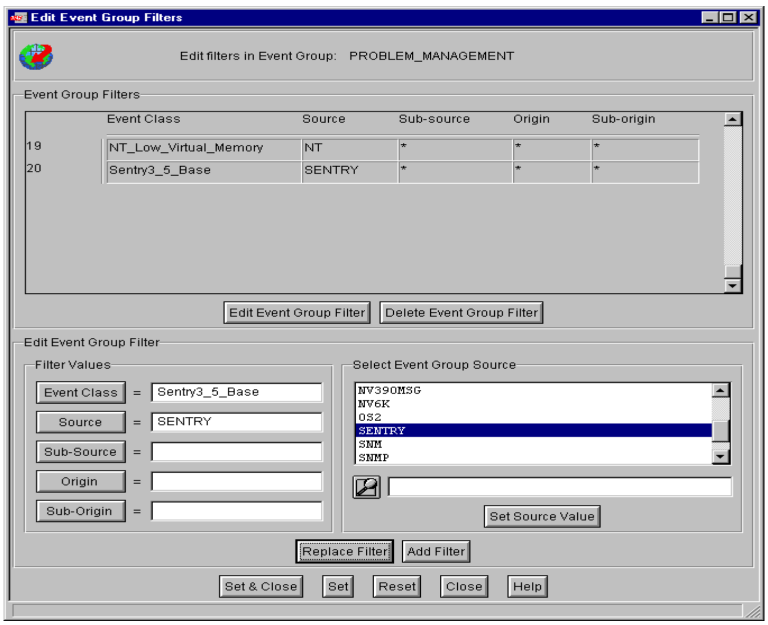


Figure 308. Creating the Event Filters

Figure 308 on page 282 shows the filtering that we added for the group definitions and to show the tested events arriving at the TEC console.

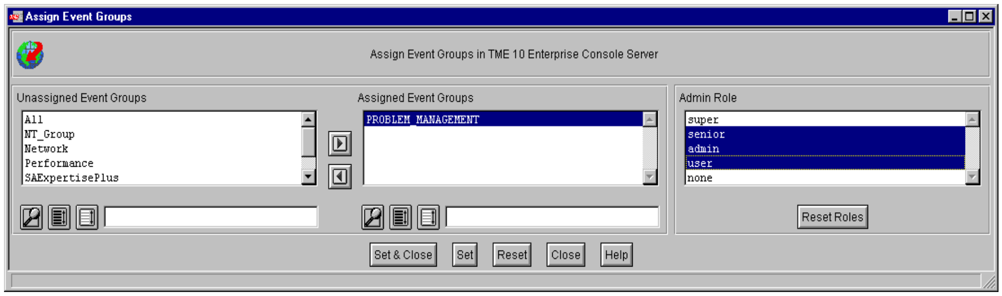


Figure 309. Assigning the Event Groups

Figure 309 on page 283 shows the event group definitions.

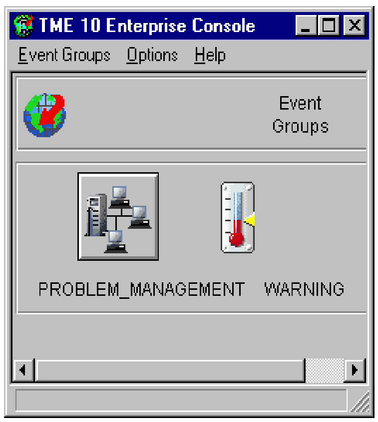


Figure 310. TEC PROBLEM_MANAGEMENT Screen

To start the TEC console for the PROBLEM_MANAGEMENT group click on the icon shown in Figure 310 on page 283.

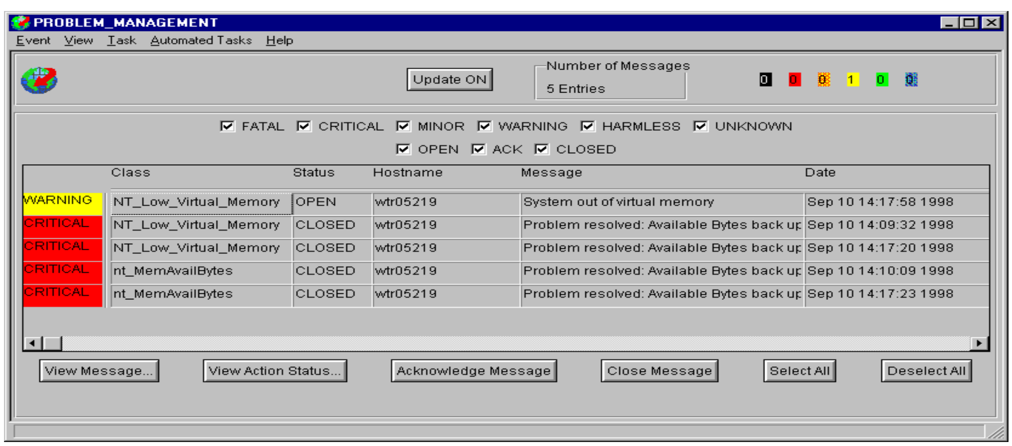


Figure 311. TEC Events for TEST

The TEC events appear in the event window shown in Figure 311 on page 283.

The information sent to the advisor is shown in Figure 312 on page 284.

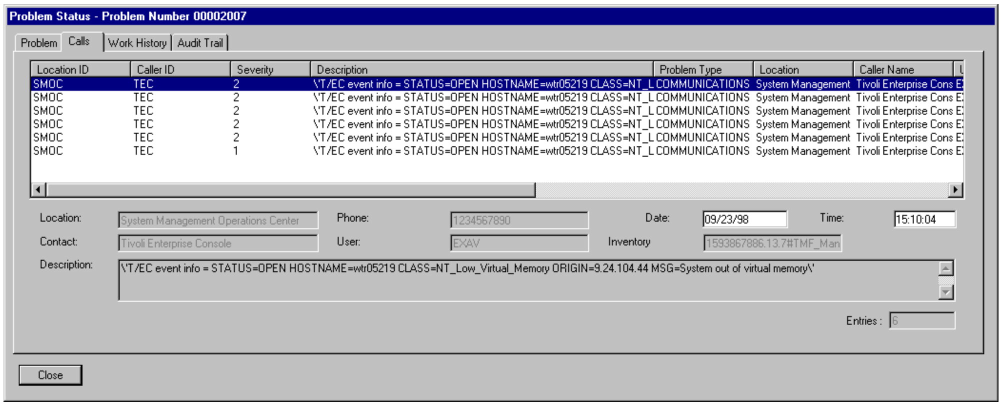


Figure 312. Problem Record

The problem status window shown in Figure 313 on page 284 shows the SCIM definition for the NT event.

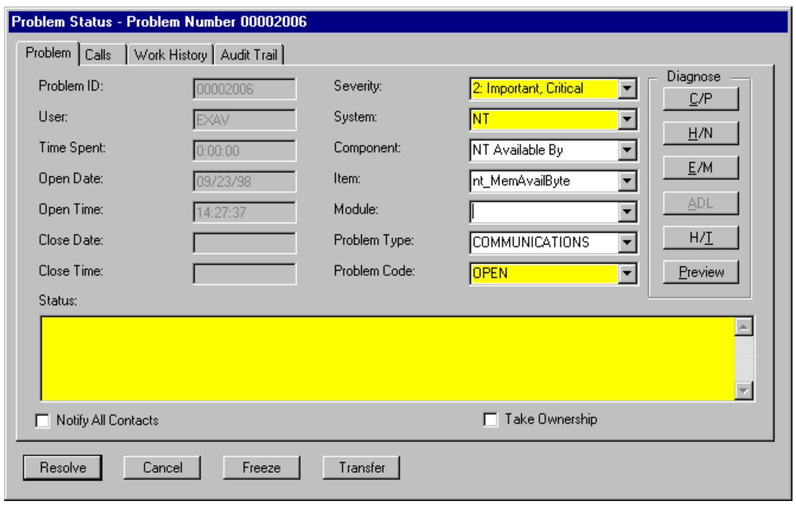


Figure 313. SCIM Definition for NT Event

Now the call can be handled using the Advisor. The next step would be to click on **Preview** to use the problem diagnostic tools.

6.3.5 Creating a Trouble Ticket from the TEC

The creation of a trouble ticket from the TEC can be valuable when there is a critical event in the TEC event window, which has been recognized as a serious problem that should create a trouble ticket immediately.

To create a trouble ticket from the TEC console using the same script as defined for the previous example, select **Edit** followed by **Trouble Ticket** as shown in Figure 314 on page 285**.**

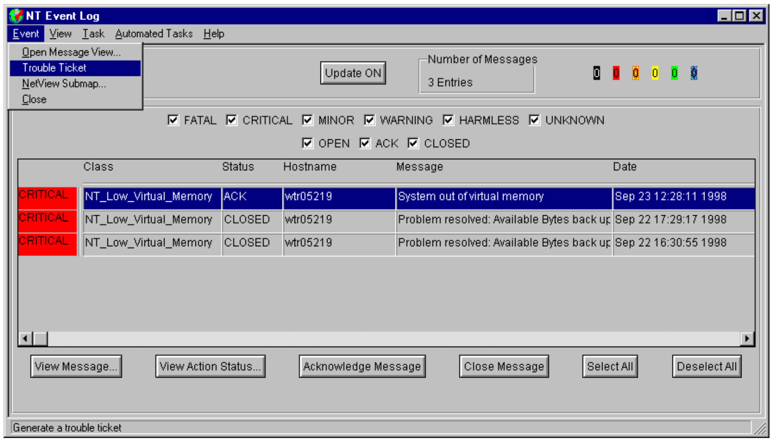


Figure 314. Generating the Trouble Ticket from the TEC Console

This calls the same script as the script discussed in the rule example. The main difference is that the arguments required to be parsed to the EVProb command are generated from the environment variables. If any of these values do not get resolved the script will assign a value.

To verify the trouble ticket has been sent you can click **View Action Status** (see Figure 315 on page 285).

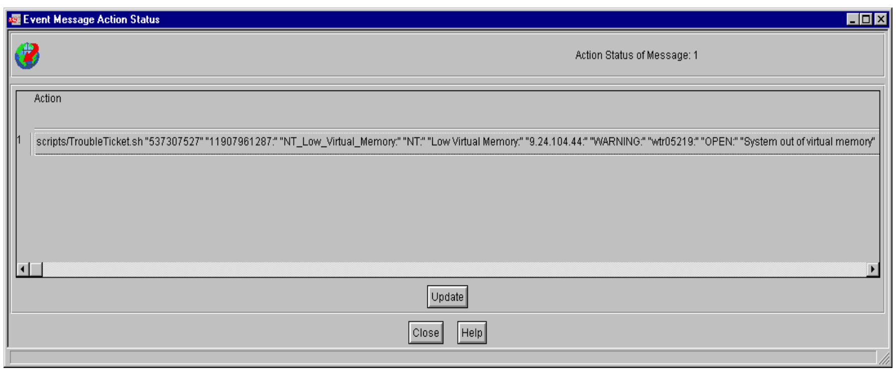


Figure 315. Verify the Trouble Ticker Has Been Created

The message for this event is shown in Figure 316 on page 286.

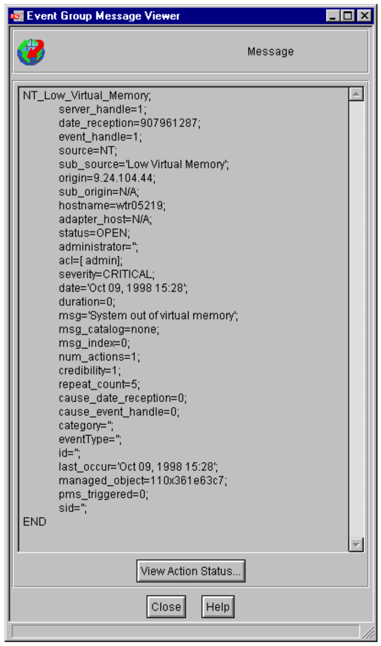


Figure 316. TEC Action Status

This information is mapped to the advisor as shown in Figure 317 on page 286.

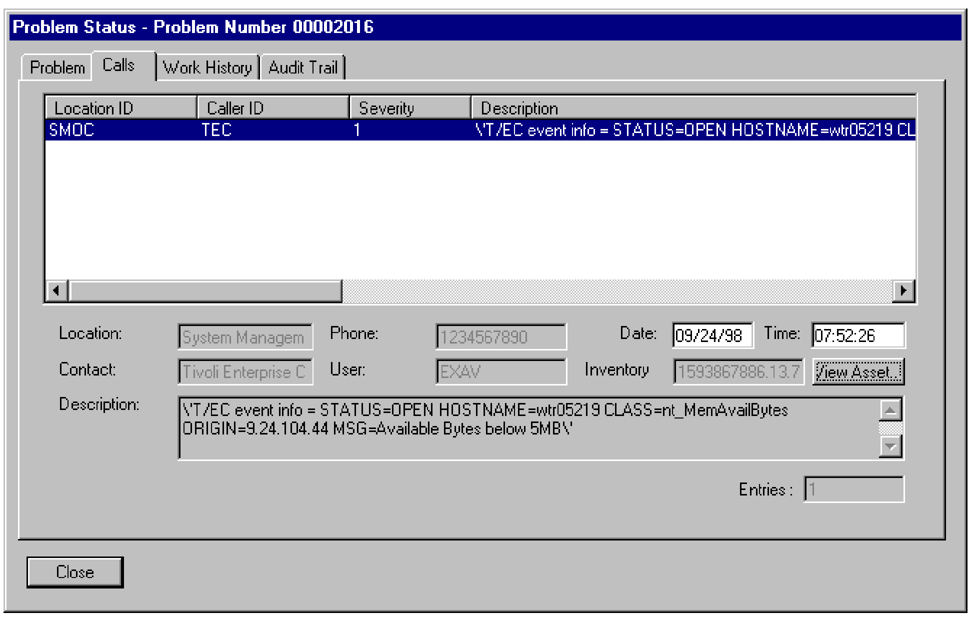


Figure 317. Advisor Call

The data mapping for the SCIM and the description was performed by the TroubleTicket.sh script.

Chapter 7. Using the Problem Management Applications

This chapter explains, by using examples, how the problem management solution can be used. The examples we cover in this section are outlined below:

- A user calls the help desk to register a problem. This problem is tracked and sent to second level support. Using the network tools the problem is then resolved.
- A TEC reported problem that will automatically create a trouble ticket. This problem will activate a change request. Also Tivoli Asset Management is used to determine the current hardware configuration of this machine.
- Using the Knowledge Pak to resolve a simple problem to show the benefit of having a populated knowledge database.

First we describe the process flow for the examples.

7.1 The Process

Figure 318 on page 288 shows the process from a call submission to problem resolution. It shows the relationships of the Tivoli Advisor with the other Tivoli applications: TEC, Foundation Manager and Evolution Manager.

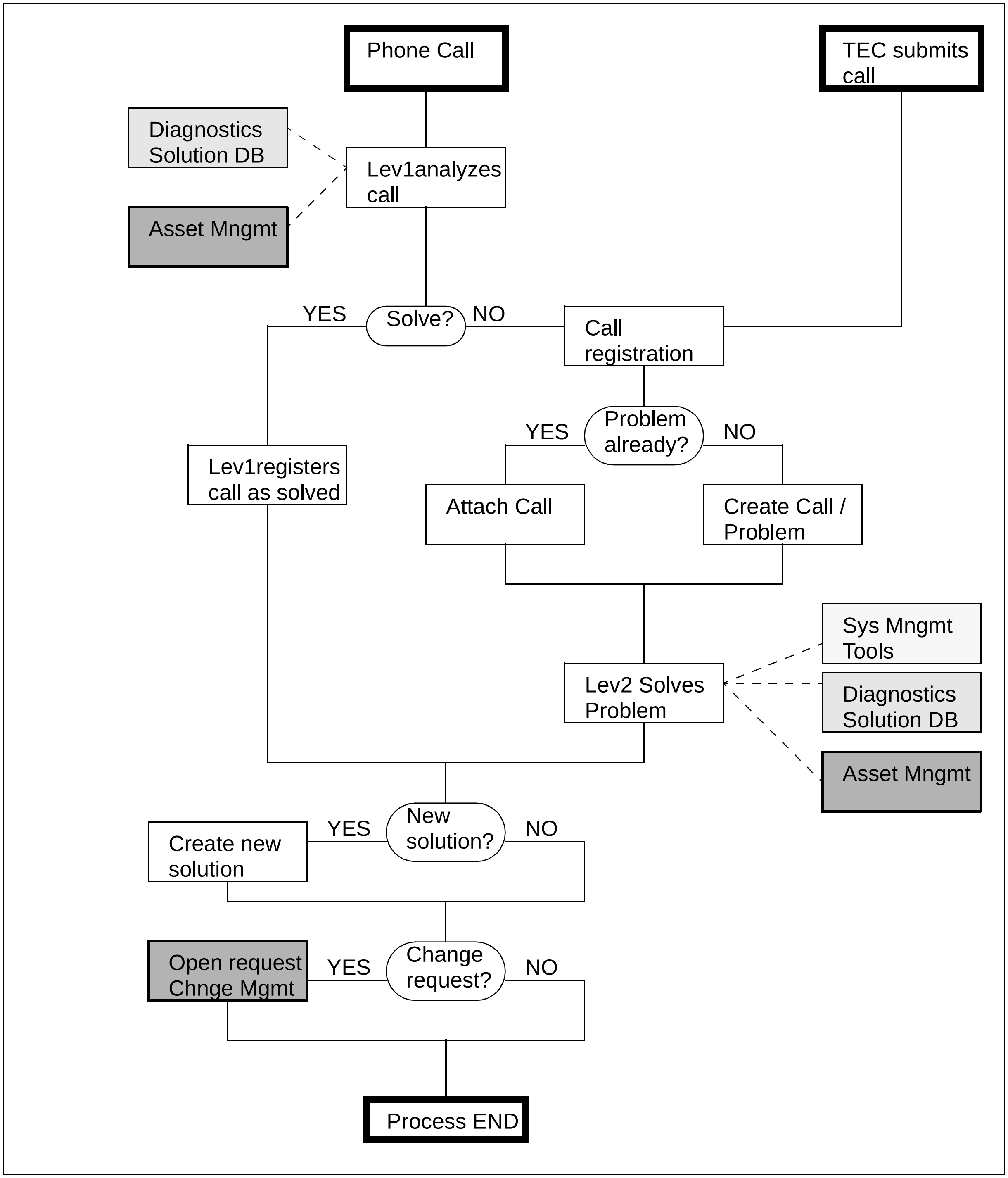


Figure 318. Problem Management Process

This is an example of a problem management process. We want to explain the functionality and power of the Tivoli Desk Service products. We assume that the help desk is made up of two levels. Level 1 will answer the calls and provide the front end support solving the problems that do not require the specific skills of the level 2 support personal.

Problems that are not solved by Level 1 are then passed to the Level 2 technicians that will work to provide the solution.

The following points explain the main steps of the problem management flow.

- When a call is received by the Level 1 help desk operator they analyze the problem reported by the caller and then try to answer the call immediately. To analyze the problem the operator can use the Tivoli Service Desk diagnostic tools and other Tivoli products available. In particular the Asset Manager provides quick access to the characteristics of the failing system.
- If the problem is solved the help desk operator registers the problem as already resolved. If the help desk operator cannot solve the problem immediately, he or she can register the call after filling in all the useful information that the caller provides. In the call registration, the SCIM has to be specified, which provides the categorization of the problem to be analyzed.
- When the help desk operator submits the call, he or she will check if a problem regarding the same dysfunction is already open. In this case the call will be attached to that problem; otherwise it will become a new problem.
- After registration the problem will be considered under the responsibility of the Level 2 technicians.
- The call can also be submitted by the TEC as explained in Chapter 6, “Problem Management Scenario” on page 185. Also in this case the call will be categorized by the SCIM specification. The call could also be submitted as attached to a previous TEC open problem or as a new problem.
- We assume that problems created by TEC are not analyzed by Level 1 but go directly to Level 2 attention.
- Once a problem has been opened it can be assigned to the technicians group with the skills necessary for that problem. The skills necessary are easily determined by the SCIM values.
- The important changes to the problems are sent to the right people by the notification function. The elapsed time is monitored by the escalation function.
- The Level 2 technicians analyze the problem using all the tools they have available (TEC, NV/6000, Diagnostic tools, Asset Manager etc.).
- The solution for a problem could be a solution already present in the solution database or a new one. If it is new, it has to be added to the solution database so that it will be a knowledge asset to be reused later.
- In order to resolve a problem, it could be necessary to perform a change on the impacted systems for example, change a part or upgrade the system memory. In this case a change request has to be submitted by using the change management product. When the change has been approved and applied, the problem can be closed.

As we have seen above, the integration between the TEC and Advisor as well as the existing integration between problem, change and asset management products, is a key point to perform efficient management of a network environment.

The integration between those products exploits the functionality of each one and makes it possible to manage IT problems controlling the flow of the activities necessary to maintain the IT environment.

7.2 Example 1 - Phone Call

This section shows an example of a phone call received by a Level 1 operator. The problem is that the caller's printer is not working

Figure 319 on page 290 shows the flow of that call and how it is managed.

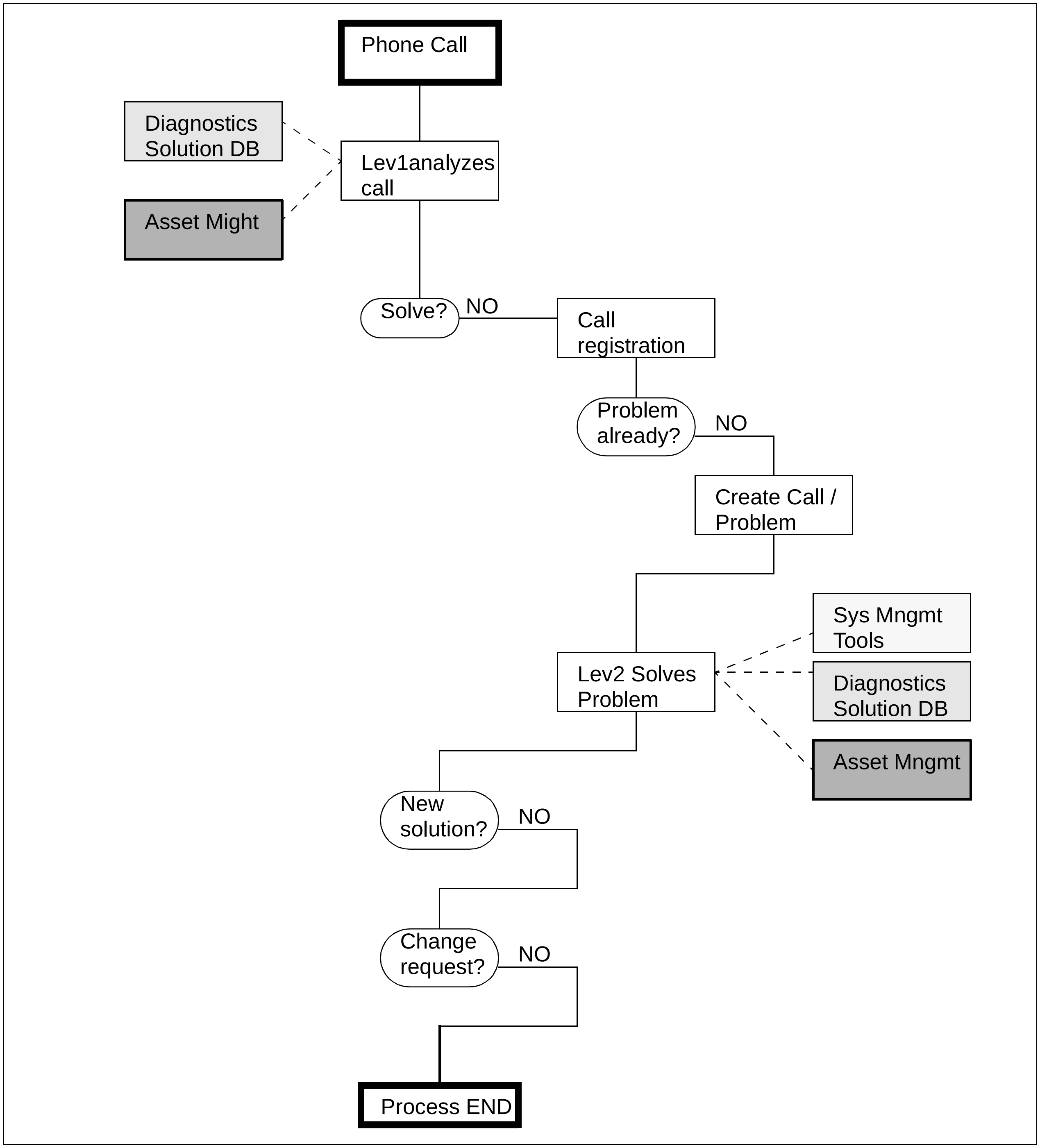


Figure 319. Phone Call Problem Process

We show the steps that the operations perform from the caller's phone call to the resolution of the problem.

The caller cannot print anything and phones the help desk service. The Level 1 operator answers the call and obtains information from the caller.

The operator asks the caller a number of questions and realizes that the printer involved is a network printer.

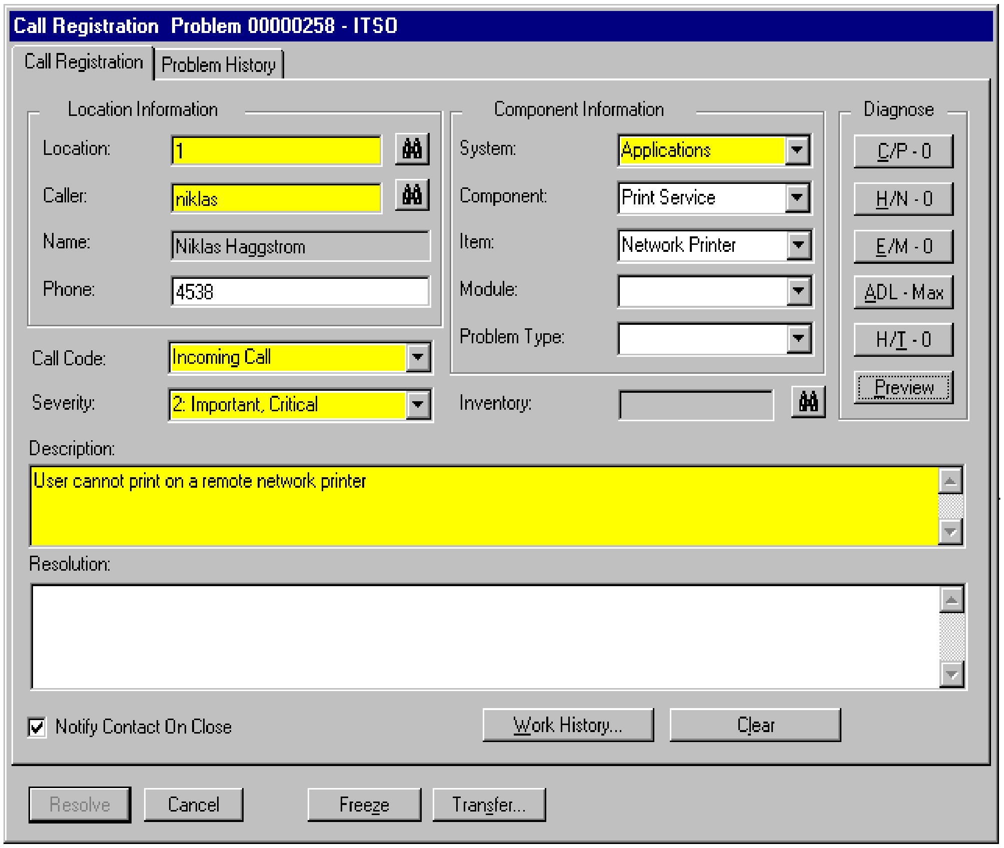
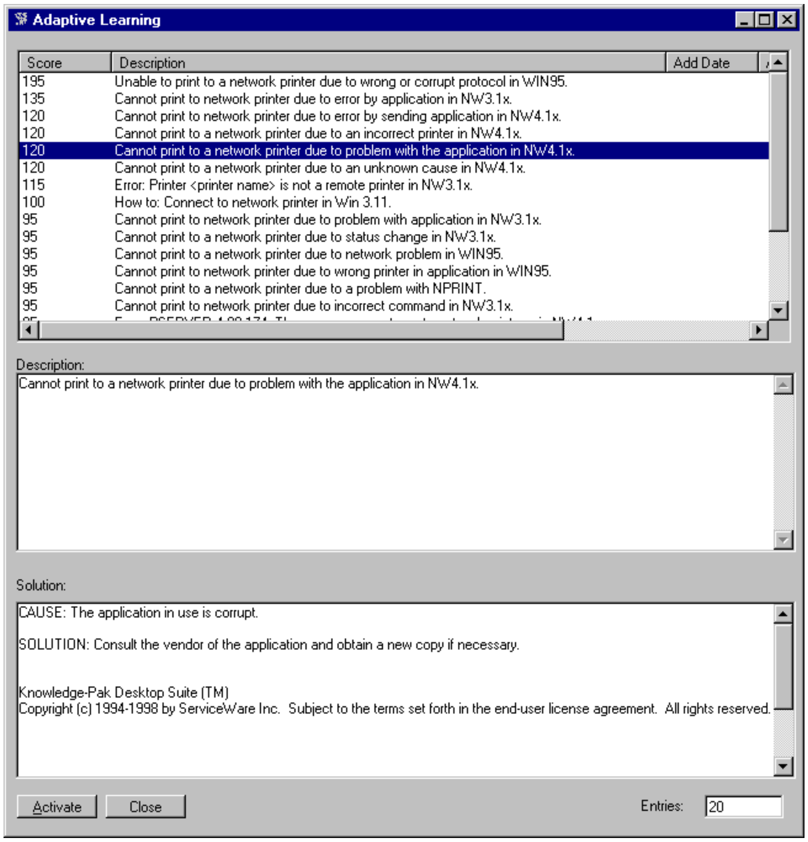


Figure 320. Call Registration

Level 1 sets the SCIM and registers all the information pertaining to the problem, (see Figure 320 on page 291) and looks for possible solutions. Although the ADL returns many possible solutions none of these work for the current case. Everything seems correctly configured on the caller computer.

The ADL output is shown in Figure 321 on page 292.



1

Figure 321. ADL Not Solving the Problem

The Level 1 operator opens a new problem for the Level 2 support and transfers it to a group of Level 2 technicians that have specific skills for that problem.

To transfer the call a group user has to be specified as in Figure 322 on page 292.

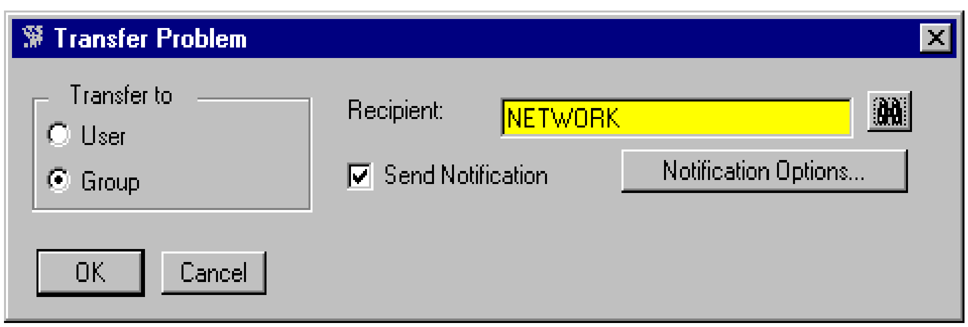


Figure 322. Call Transfer

The members of the group NETWORK specified above as Recipient, will receive an alarm notifying them that a new problem has been assigned to them. The alarm will be like in the following figure.

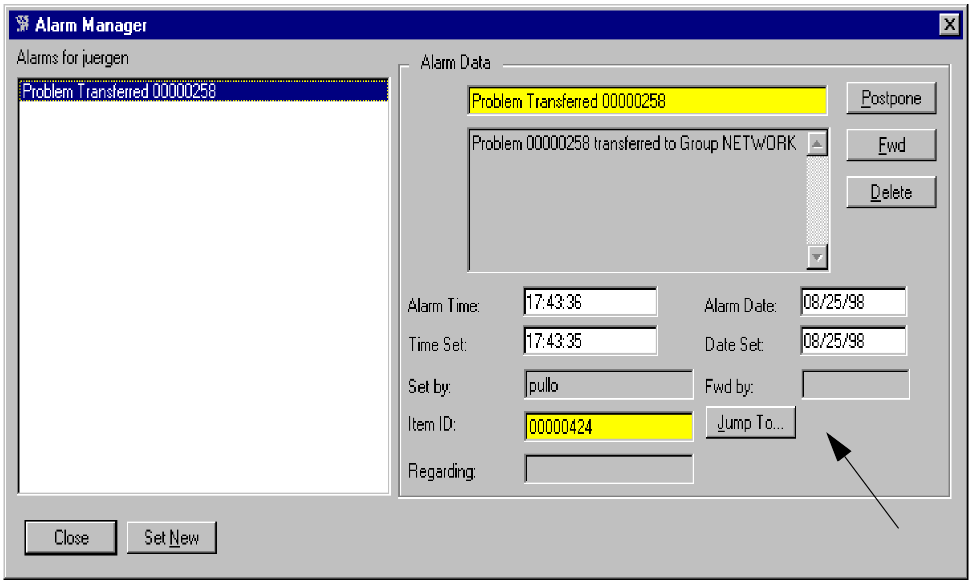


Figure 323. Alarm for a New Assigned Problem

By clicking on **Jump To** the Level 2 user will display the problem as shown in Figure 324 on page 293 and then they will click on the **Take Ownership** box to take the ownership of the problem.

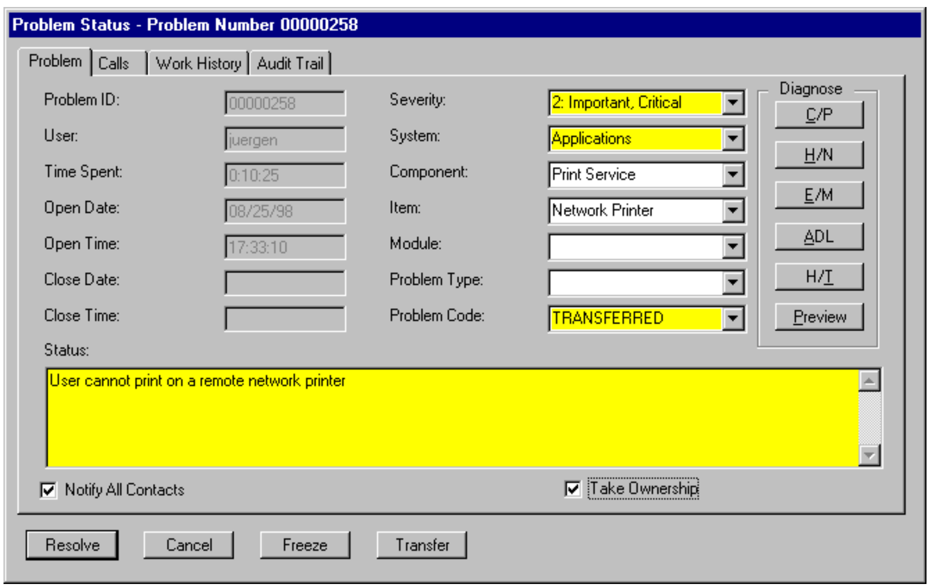


Figure 324. Level 2 Takes the Ownership of the Problem

After the Level 2 technician freezes the problem this will be in open status and the Level 2 support becomes the owner.

Using TME NetView the Level 2 technician sees that there are some problems on a switch connecting the user workstation to the printer.

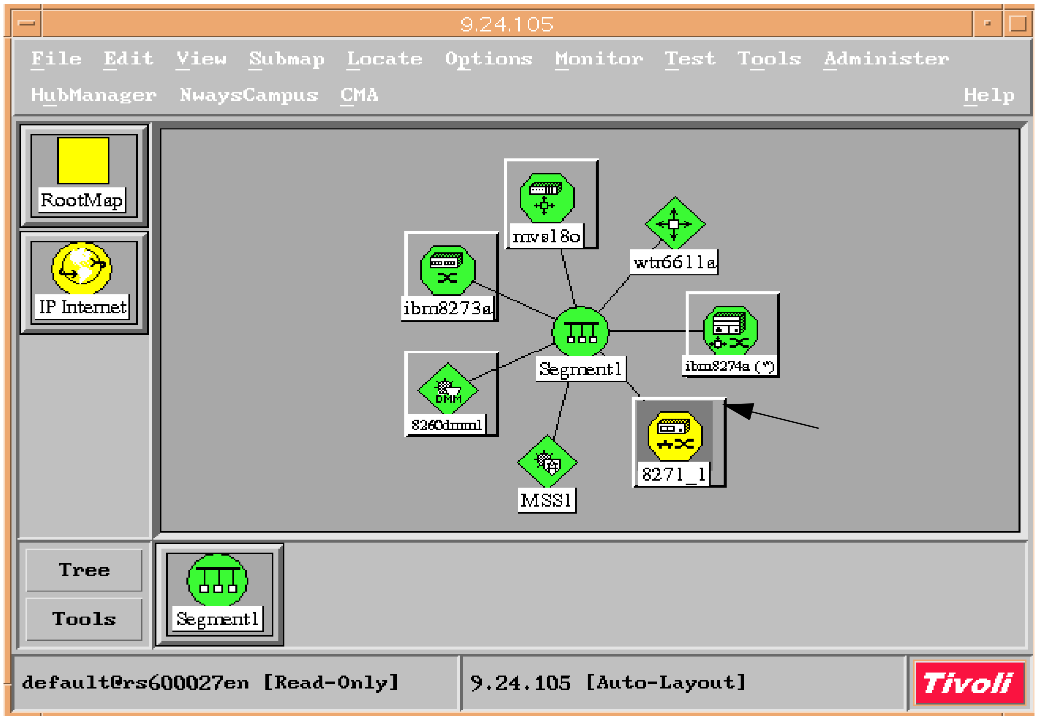


Figure 325. Switch with Marginal Status

Next the technician looks in more detail at the problem and finds the switch port is down (see Figure 326 on page 294).

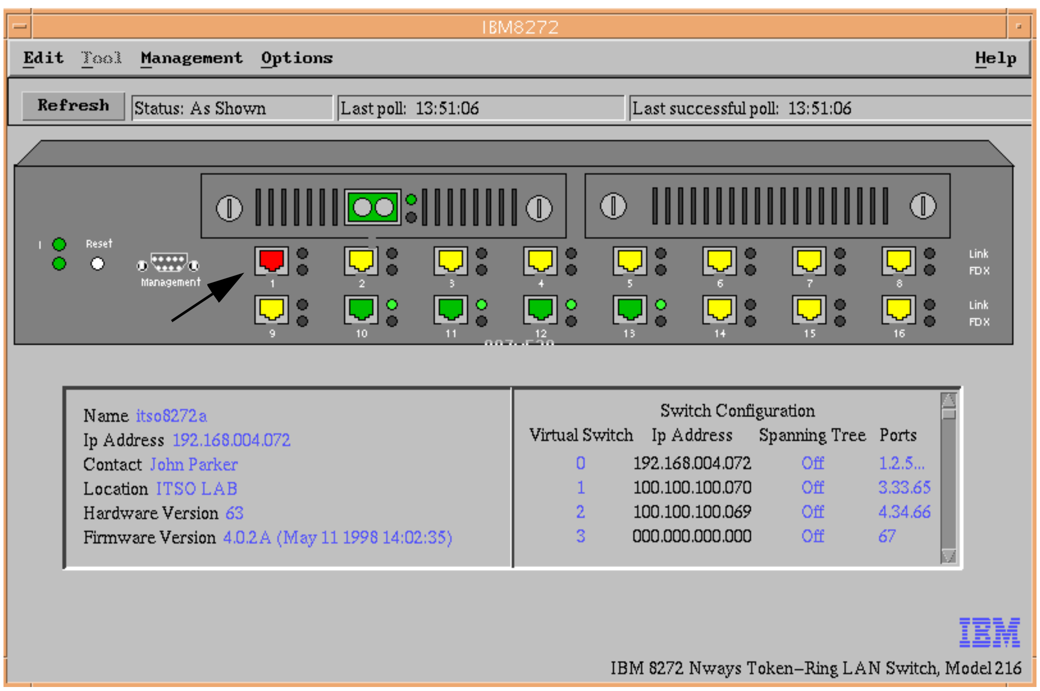


Figure 326. Port 1 Is Down

This is the cause for the printer problem.

The Level 2 technician resets the switch, checks that the problem is solved and then updates the Advisor problem. The status field is updated leading to a resolution for the problem (see Figure 327).

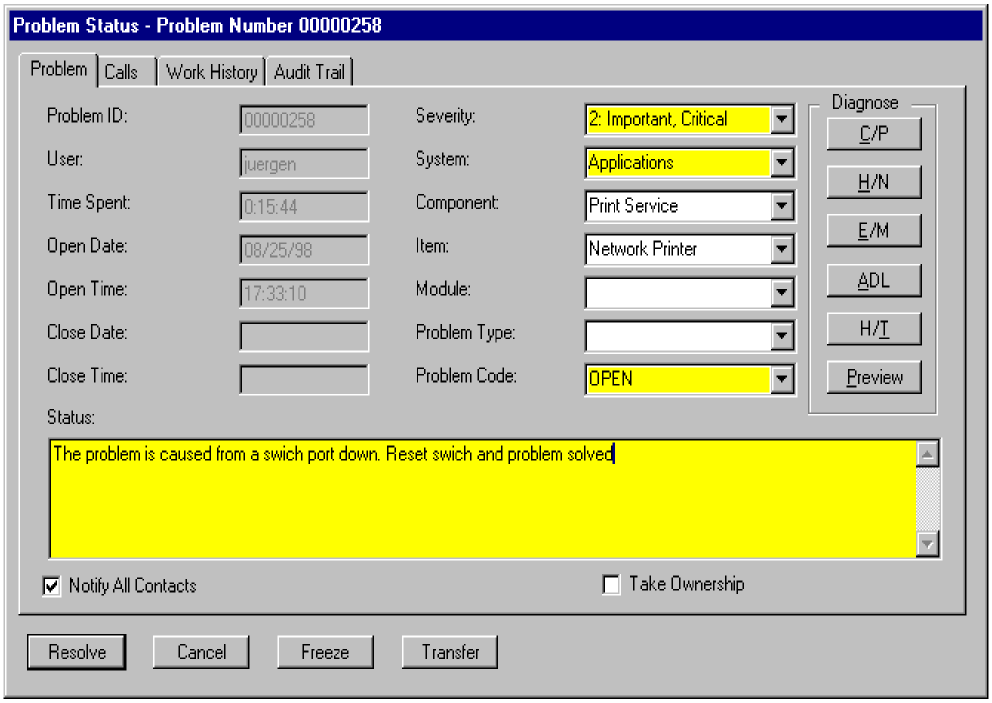


Figure 327. Problem Resolution

The problem will be in CLOSED status.

7.2.1 Consideration

In the above example we have seen a problem caused by a network device. The Level 2 technician needed to use TME NetView to analyze the cause of the problem.

This is an example with no integration between TME NetView and TEC products. We want to underline here that the integration between those two products provides a big benefit to IT support.

In fact, when the NetView adapter is configured, it will send an event to TEC reporting that a network device down. In this case the Level 2 technician immediately knows that information just by viewing the TEC events.

Moreover, if TEC is customized to open a call to Advisor when a switch has a problem, the scenario described in our example will be different. When the caller calls, then the problem is already open in Advisor and maybe a Level 2 technician is already working on it.

Level 1 receives the call and just looking at the Hot News can see that there is a switch problem and is under the control of the Level 2 technicians. So, Level 1 can give this information to the caller and ask him or her to try to print at a later time.

7.3 Example 2 - TEC Reported Problem

This section shows an example of a call that is generated by the TEC. The diagram in Figure 328 on page 296 shows the flow of the problem and how it is managed. We describe the steps performed to track and solve the problem.

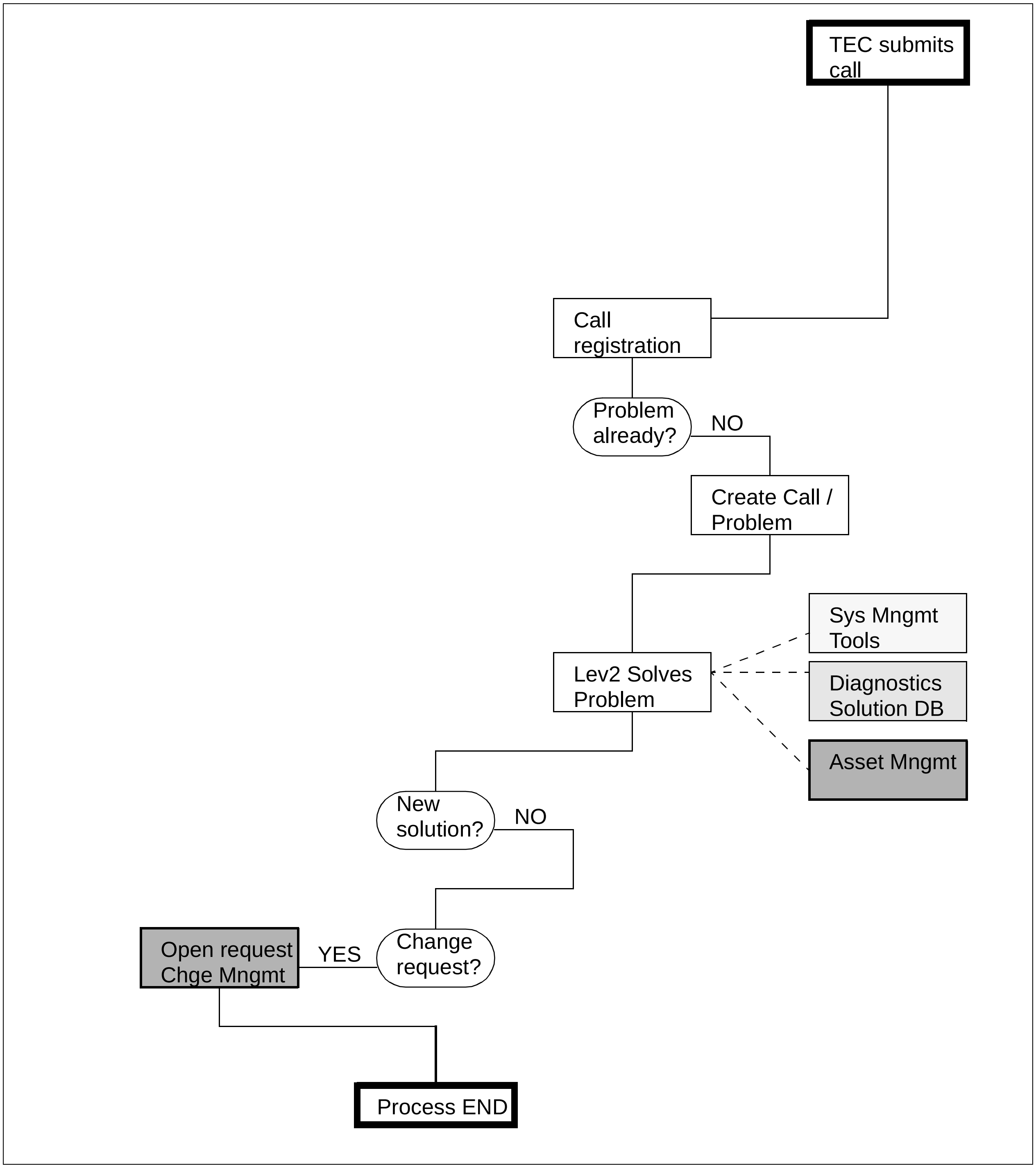


Figure 328. TEC Reported Problem

Using the rules, which we describe in Chapter 6.3, “The Rule Flowcharts and Rules” on page 275, opens a problem in Tivoli Advisor. The initial owner of this problem is the user EXAV, which we define in the Tivoli NSM Gateway setup (see Figure 5.7 on page 198).

Notifying users about this open problem is not an easy task to perform, as it requires changes in procedures and functions provided as default with the Tivoli Service Desk suite.

We decided that the second level support will monitor any problems that are opened by the user EXAV. When a problem is noticed as shown in Figure 330 on page 298, we then transfer the problem to the help desk group of users assigned to this problem type.

Users in this group are notified about this problem and the first user who opens this problem becomes the owner.

The first notification of the TEC problem is when the the alarm appears on the Advisor screen for the user EXAV, as shown in Figure 329 on page 297.

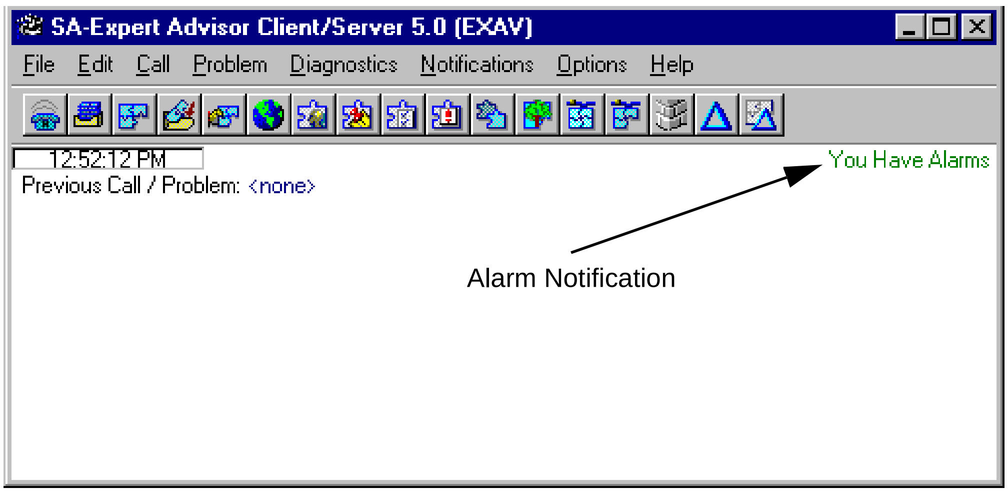


Figure 329. Alarm Notification

From the main Advisor screen select **Problem** followed by **Work with Problem**. We selected the problem shown in Figure 330 on page 298.

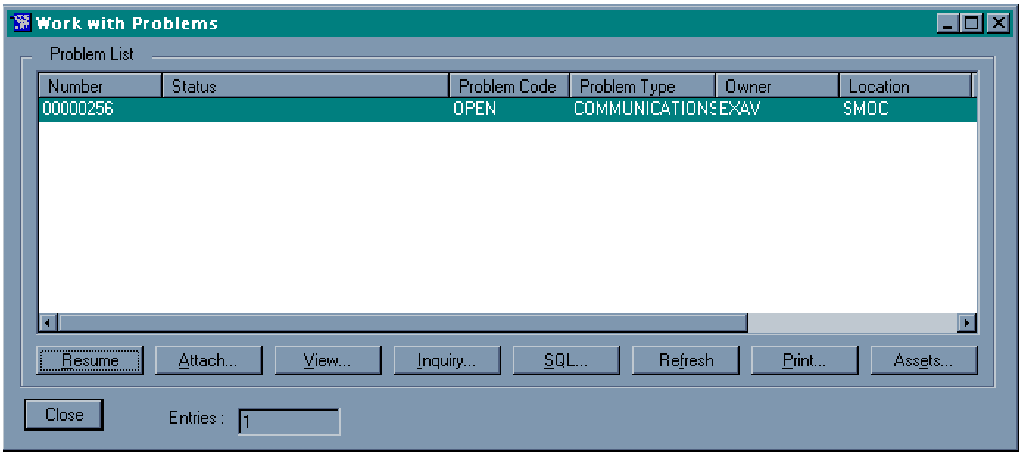


Figure 330. Problem Opened by the TEC

Level 1 support will receive the call. First they will check for any solutions available for the problem. In the Calls window is the description of the problem and to what asset this problem is associated as shown in Figure 331 on page 298.

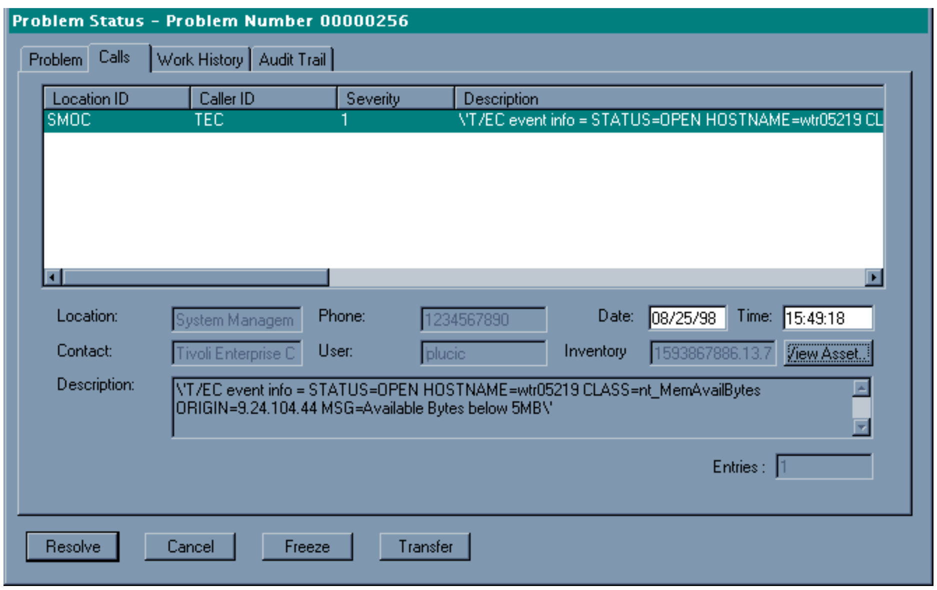


Figure 331. Status of the Open Problem

At this point the problem can be passed to Level 2 support by selecting **Transfer** (see Figure 332 on page 299).

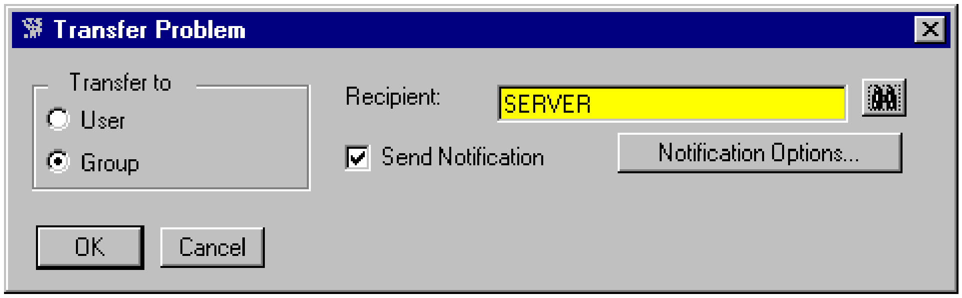


Figure 332. Transfer to Server Group

Confirmation is shown in Figure 333 on page 299.

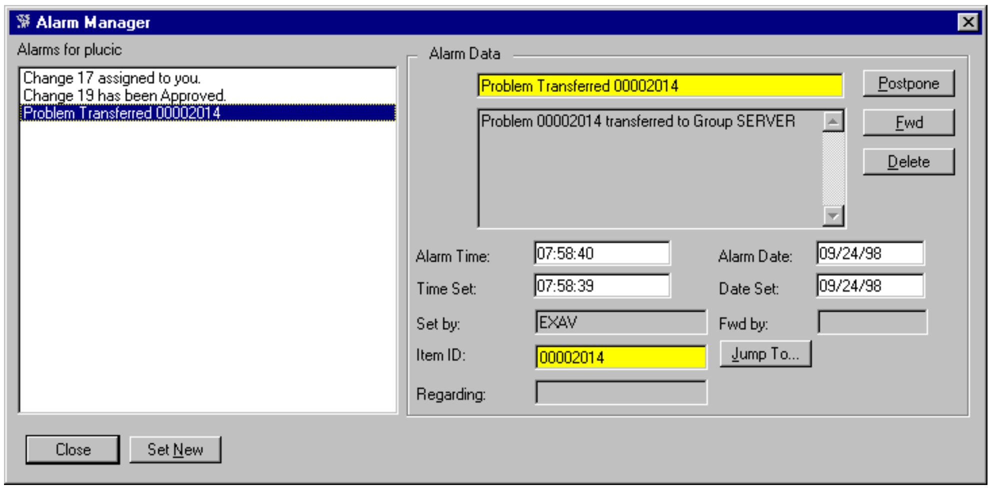


Figure 333. Transfer of Problem

Based on the description, the help desk user suspects a problem with the server's memory, so he or she selects **View Asset** to see Inventory details about this machine as shown in Figure 334 on page 300.

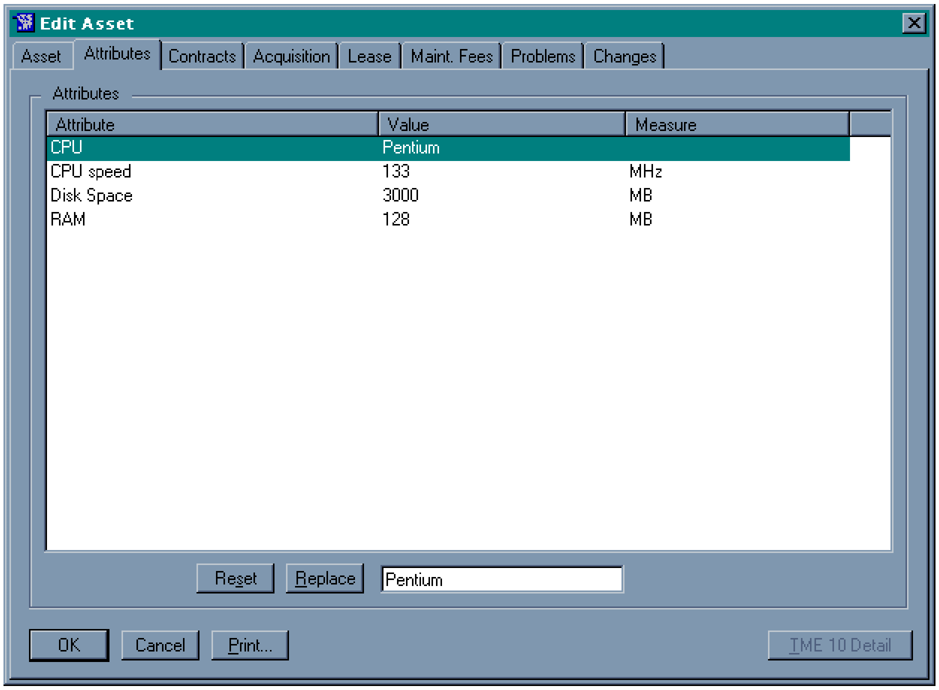


Figure 334. Inventory Attributes

Collected information points to a memory problem so the help desk user decides to request the change for upgrading the affected asset. The user first freezes the problem, so that nobody can edit it in the meantime. When freezing the problem the user also changes the status to OPEN_CHG as shown in Figure 335 on page 300.

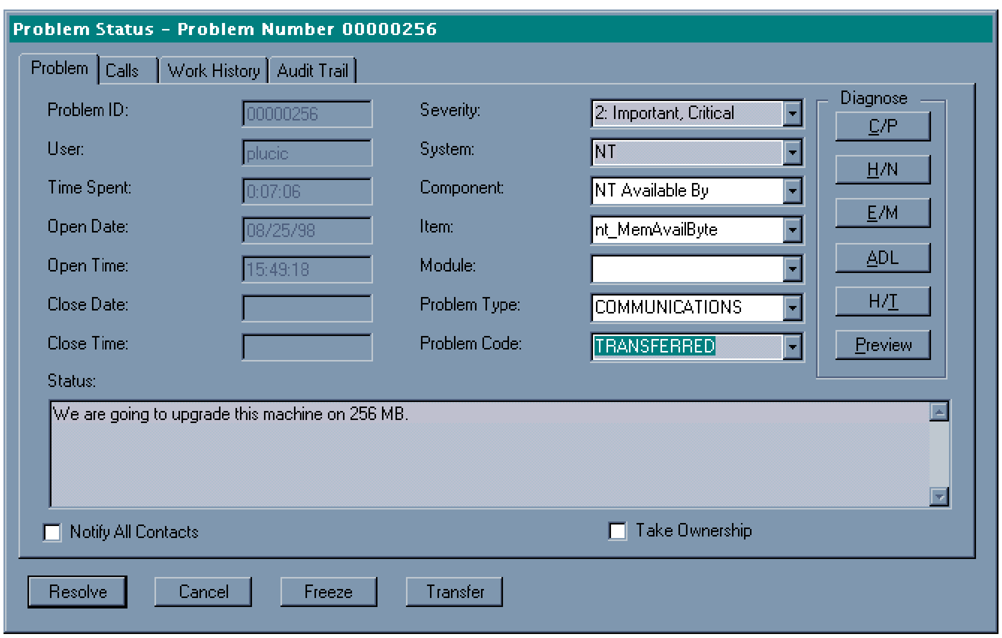


Figure 335. Suspending the Problem

While the problem is still open, the user can click on **Enter a new change from problem** and define a category and status of the change as shown in Figure 337 on page 301.

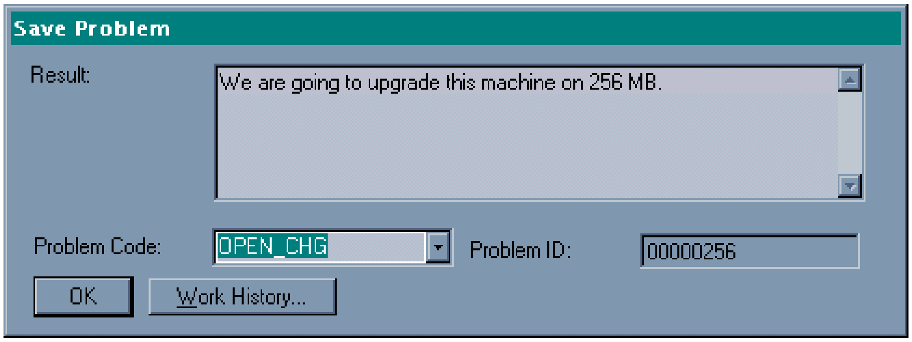


Figure 336. Upgrade Request

The upgrade request is shown in Figure 336 on page 301.

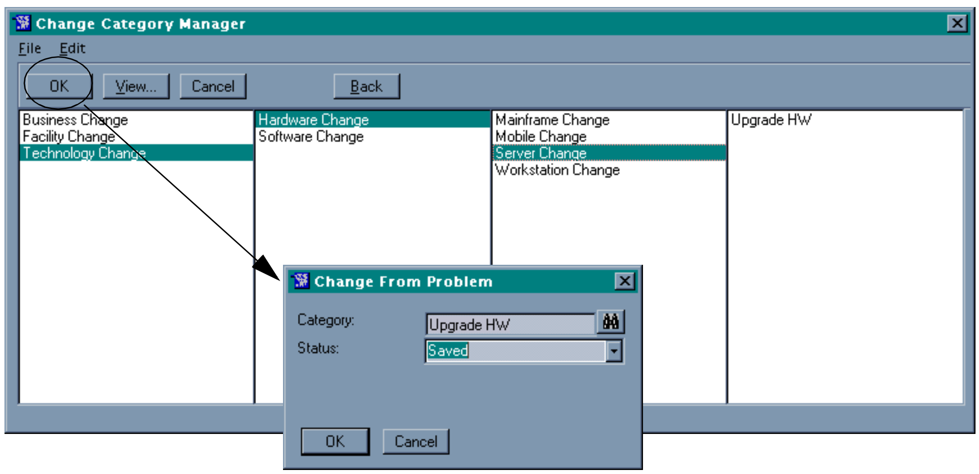


Figure 337. Creating the Change from an Opened Problem

When the change window opens, more fields are already populated as shown in Figure 338 on page 302. This is the case, because we associate change in this category to a model we define in 5.10.1, “Change Request” on page 209.

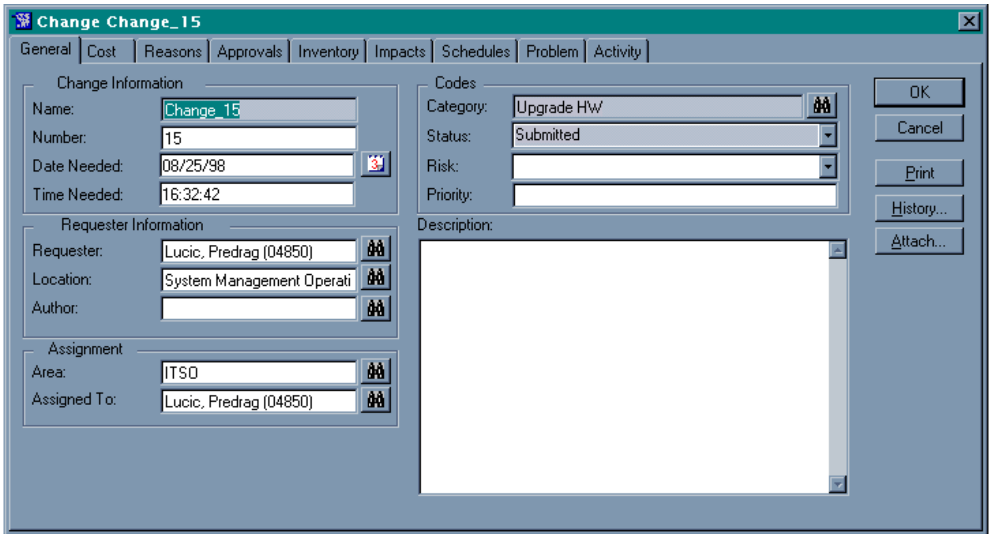


Figure 338. Change Window

By clicking on **Cost** we can enter the actual amount (see Figure 337 on page 303).

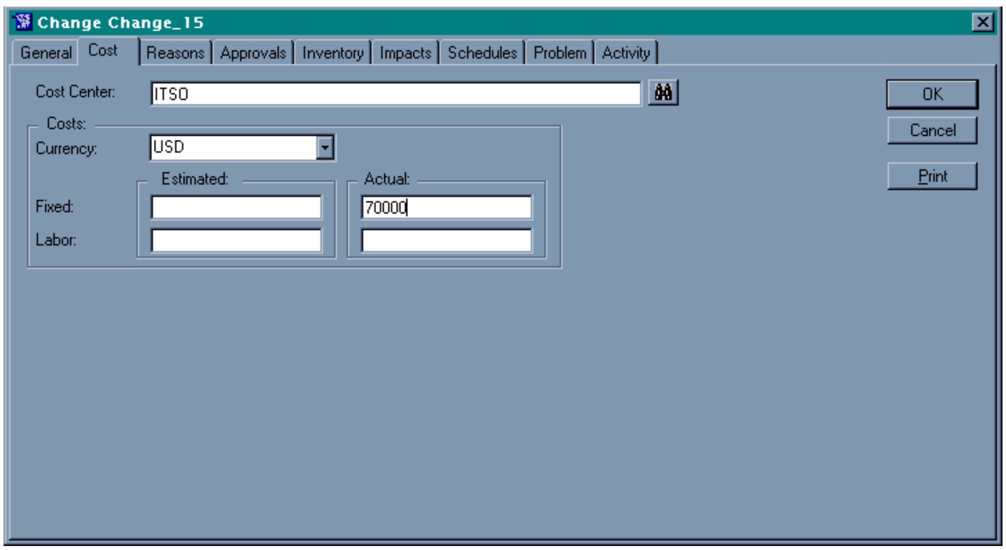


Figure 339. Adding the Cost of the Change

The cost of the change is greater than 50000. This change also affects the associated asset so the user defines which attribute changes and what value as shown in Figure 340 on page 303.

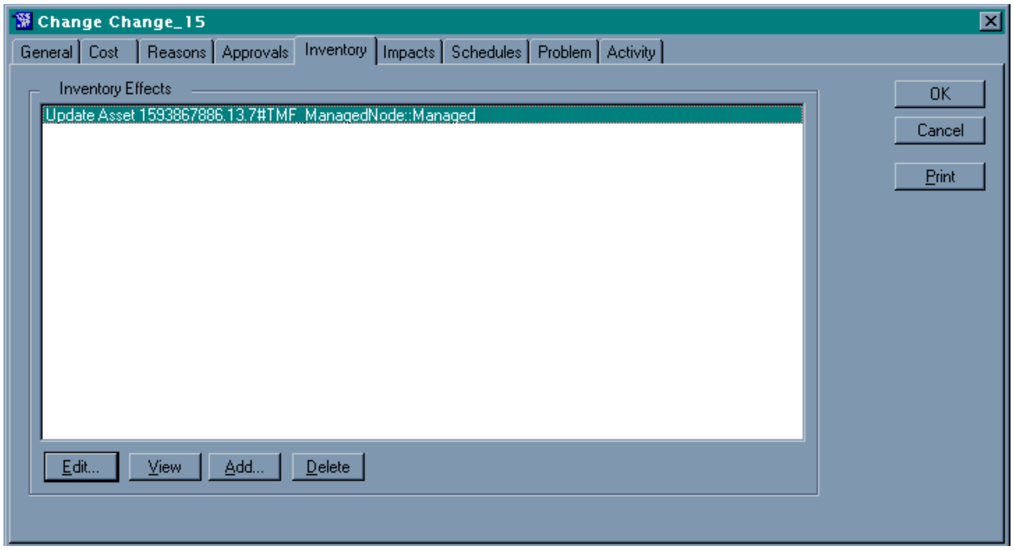


Figure 340. Defining Change Effect on Affected Asset

Select **Edit** to see the assets to be updated (see Figure 341 on page 303).

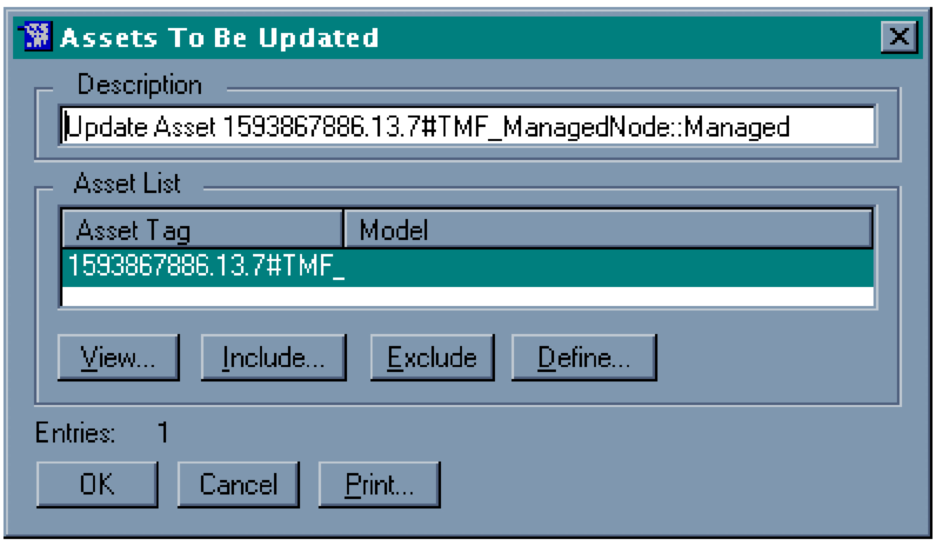


Figure 341. Assets To Be Updated

Next select **Define** to see the Group Maintenance screen shown in Figure 342 on page 304.

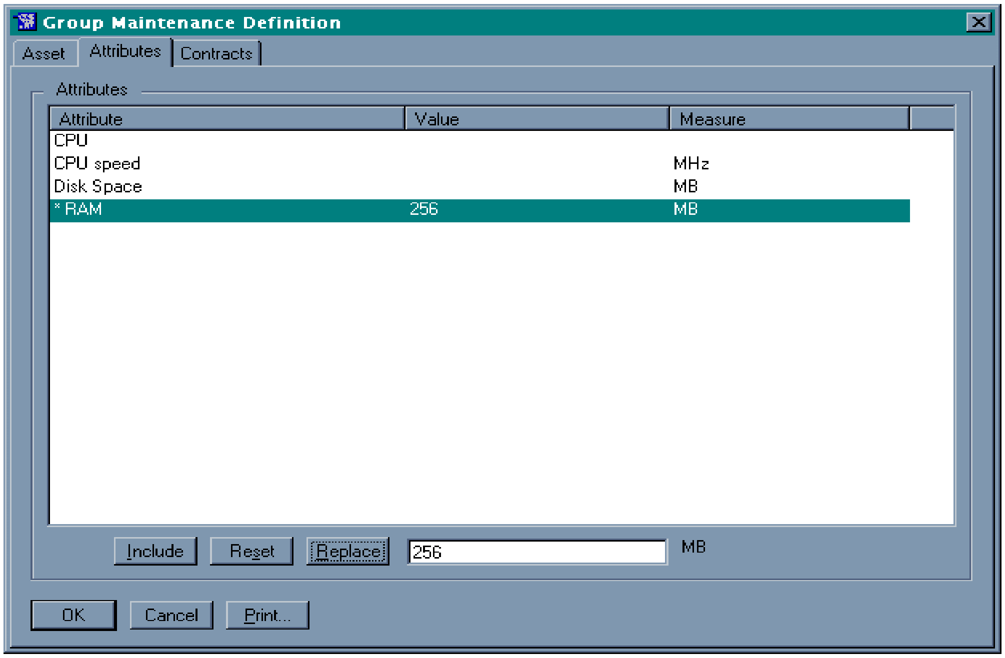


Figure 342. Group Maintenance

The user now submits the requested change. The change is automatically checked against the live rules. In our case it is affected by our define rule.

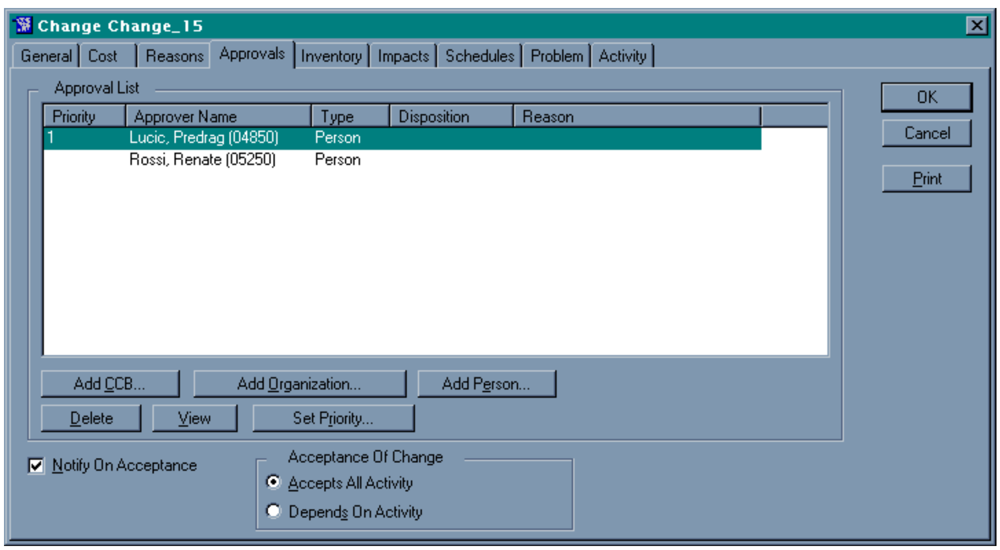


Figure 343. Rule Adds Additional Approver

All approvers are notified by the arriving request for approval. Approvers approve the request by entering the Approval window and accepting the change as shown in Figure 344 on page 305. This action sets the status of the change to Approved.

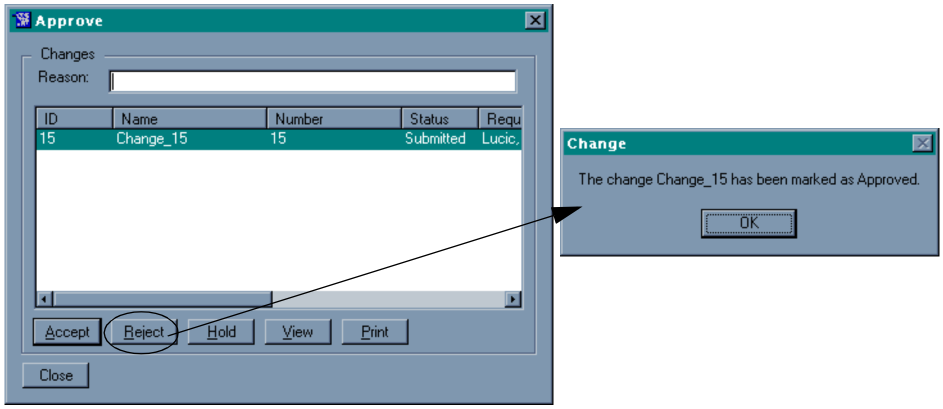


Figure 344. Approving the Change

When all the actions about the change are complete the person the change is assigned to can complete the change by setting the status to Completed as shown in Figure 345 on page 305.

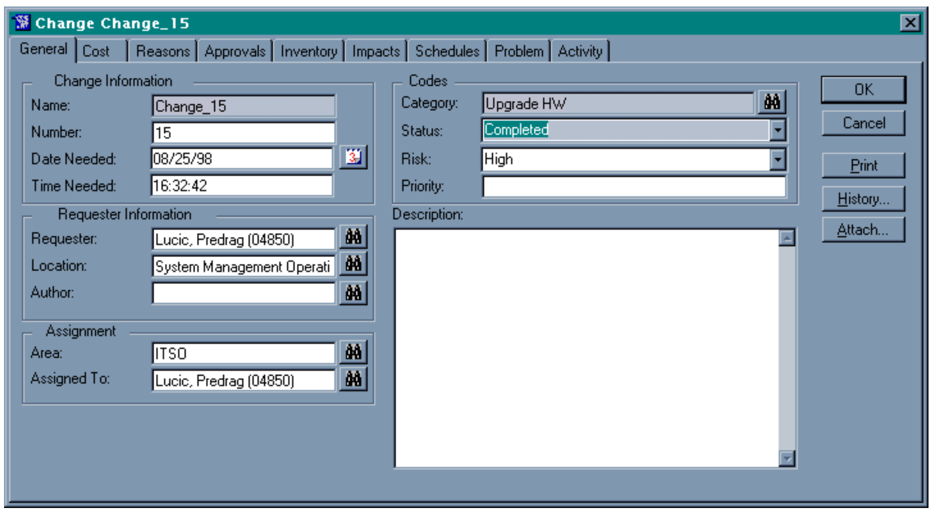


Figure 345. Complete the Change

When the change is complete the inventory attributes of the affected assets automatically change to the value defined in the change request as shown in Figure 346 on page 306.

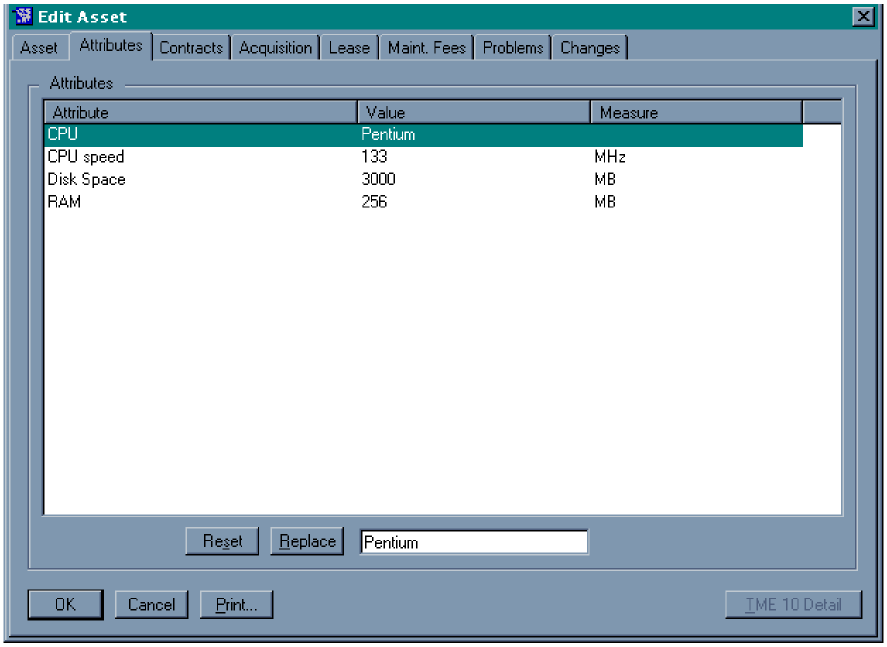


Figure 346. Inventory Attributes Changes

When the help desk user this change is assigned to completes the change the problem status of all the problems associated to this change is set to CLOSED. The problem is now closed as shown in Figure 347 on page 306.

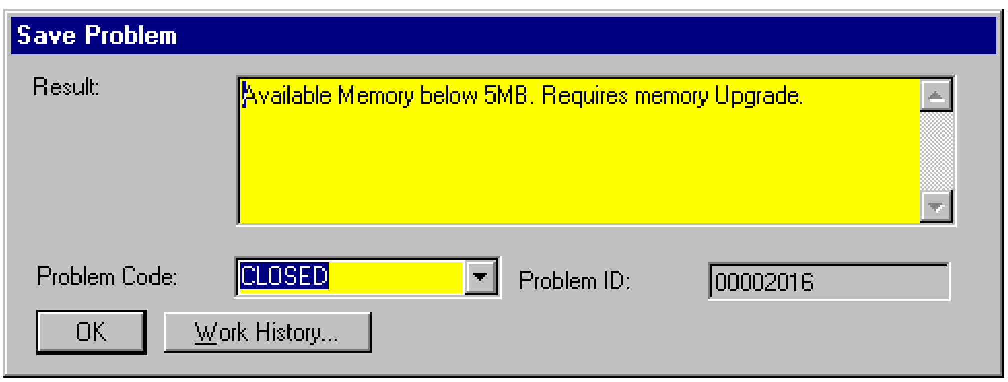


Figure 347. Save Problem

Click on **OK**. Finally select **Resolve** from the screen shown in Figure 348 on page 307 to close the problem.

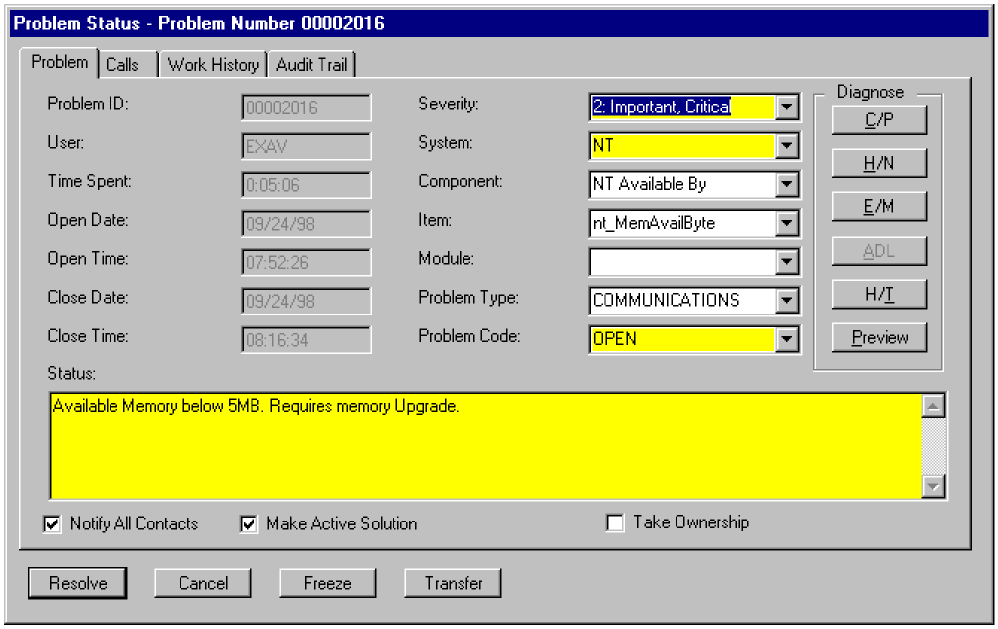


Figure 348. Resolve the Problem

The problem is now resolved. This causes the closing of the associated TEC events as shown in Figure 349 on page 307.

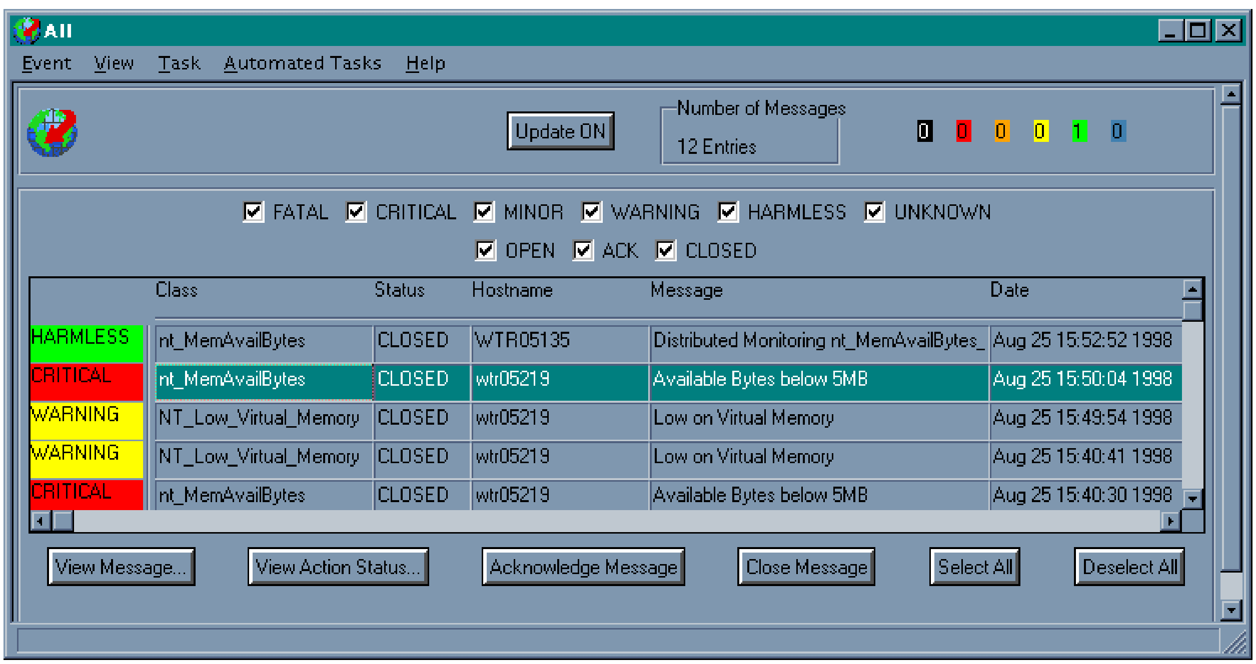


Figure 349. Closing the Problem Closes the TEC Event

Select **View Action Status** on the closed event, as shown in Figure 350 on page 308.

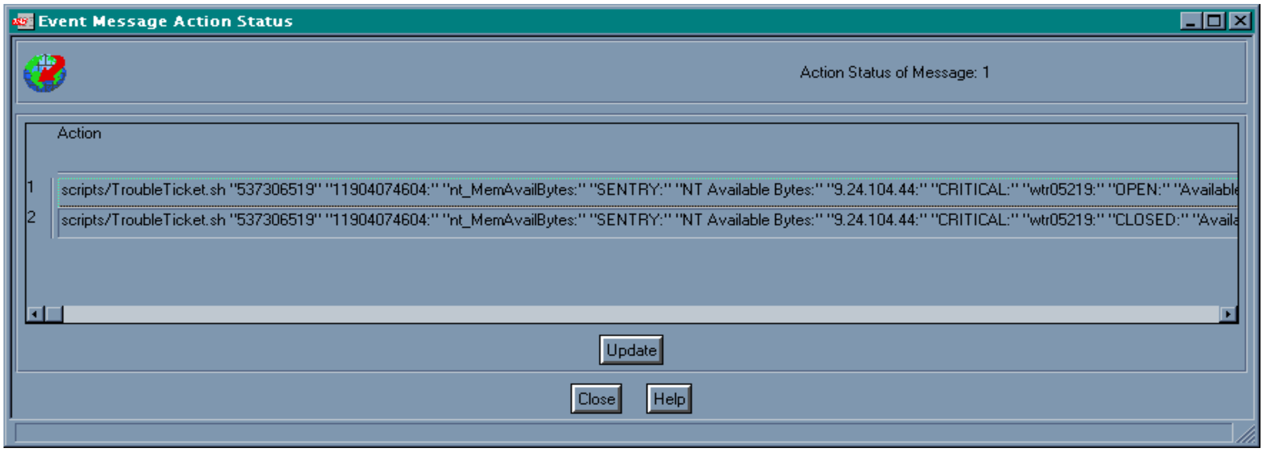


Figure 350. TEC Information

This shows the actions performed on the TEC event.

7.3.1 Diagnostic Configuration Options for the TEC Event

We entered the information for the common problem as shown in Figure 351 on page 308.

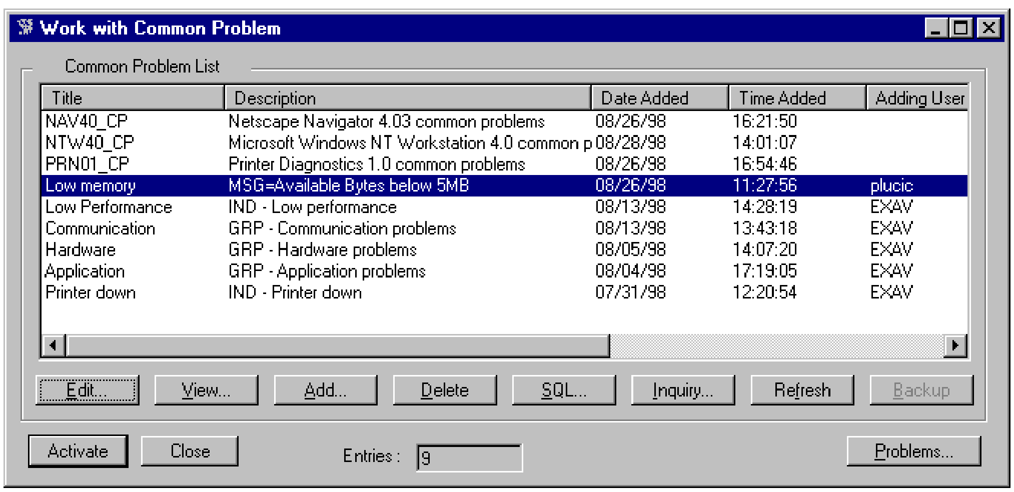


Figure 351. Common Problem

The Common Problem entry screen is shown in Figure 352 on page 309.

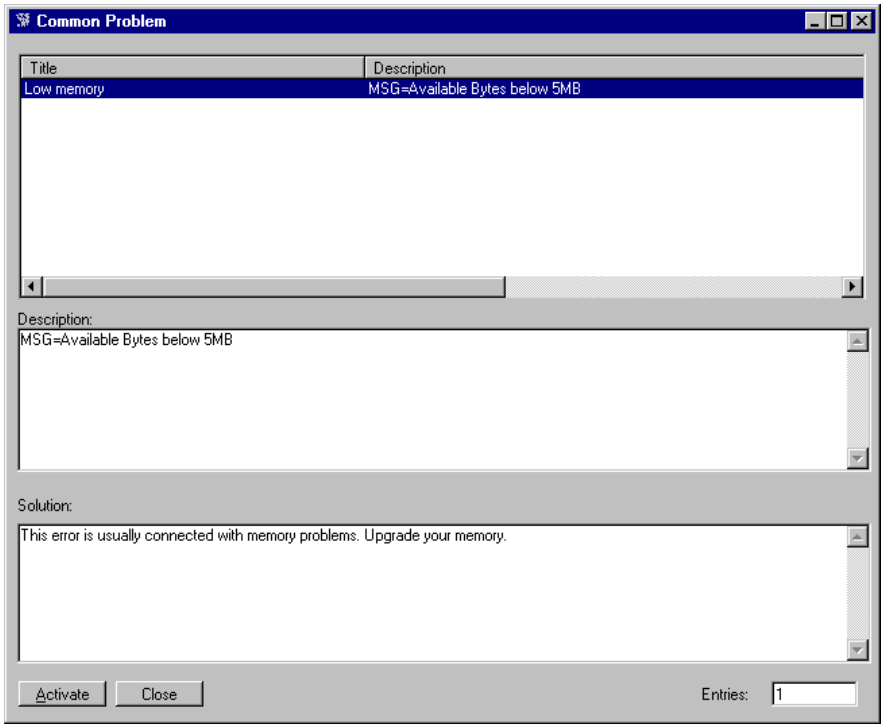


Figure 352. Common Problem Window

Adding a Hot News item is shown in Figure 353 on page 309.

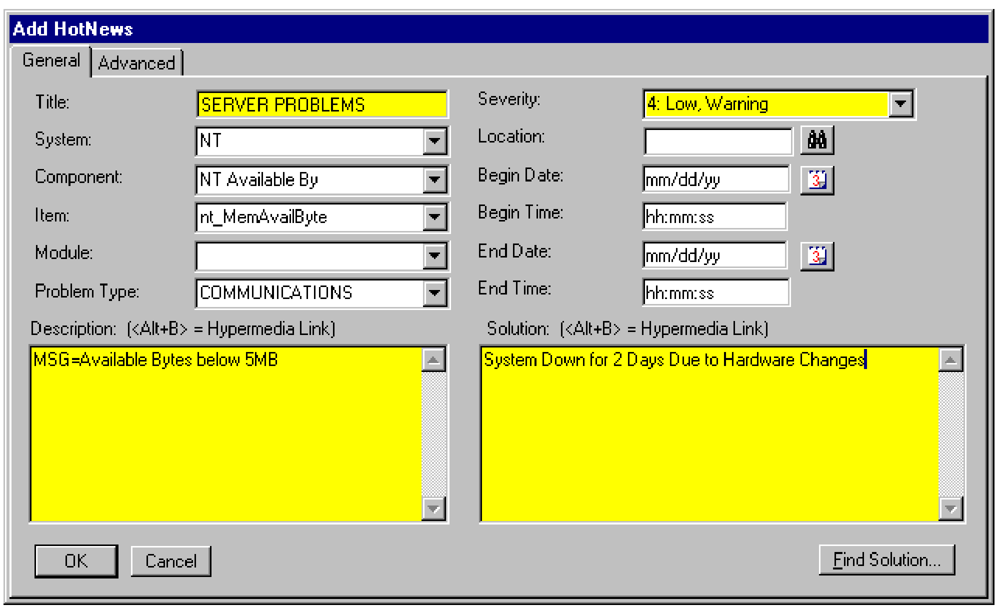


Figure 353. Hot News Item

The change could be added to the system Hot News, to provide information regarding the changes to the server (see Figure 354 on page 310).

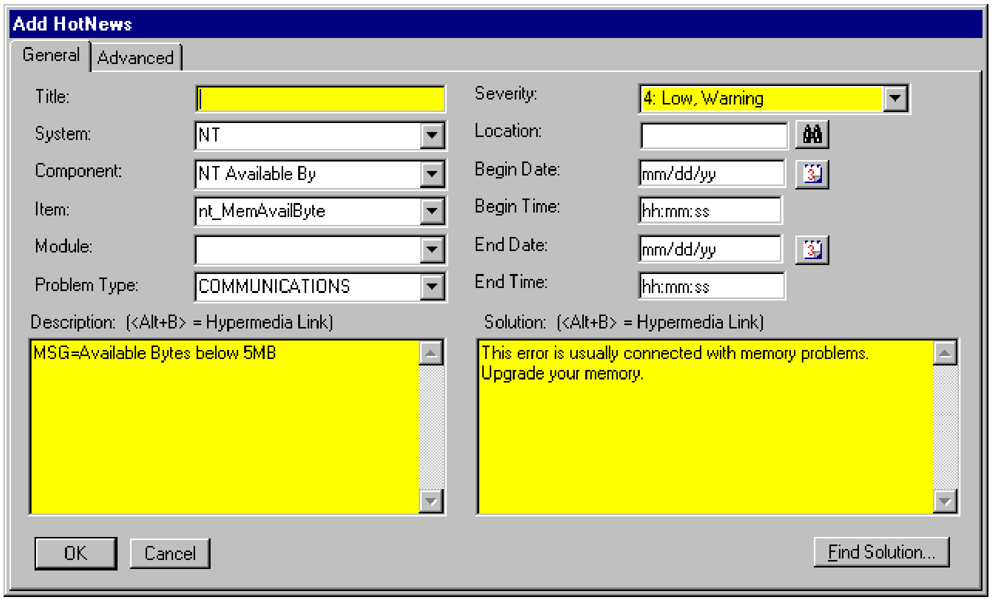


Figure 354. Add Hot News Item

The solution can contain the information shown in Figure 355 on page 310.

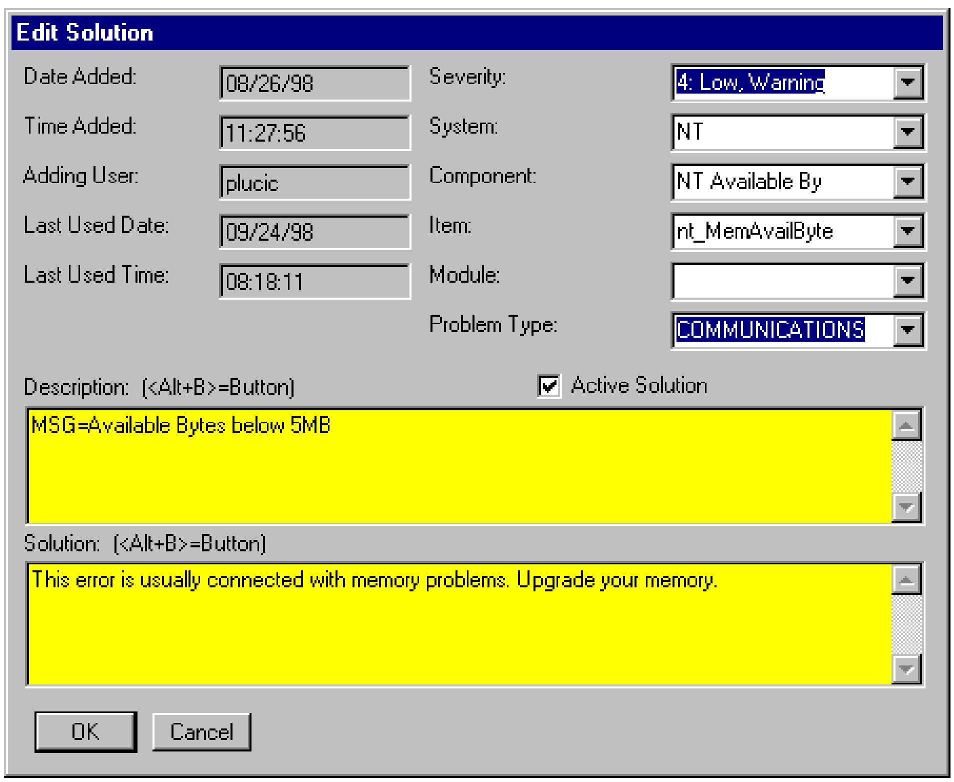


Figure 355. Solution to the Problem

The next time this event is registered as a call the help desk can resolve the problem faster and more efficiently using the diagnostic tools.

7.4 Knowledge Pak Example - 1

Here we show how powerful the Knowledge Pak can be. We have installed the desktop power pak.

This example shows how to use the information contained in the Knowledge Paks to resolve problems. The help desk operator enters the call details as shown in Figure 356 on page 311.

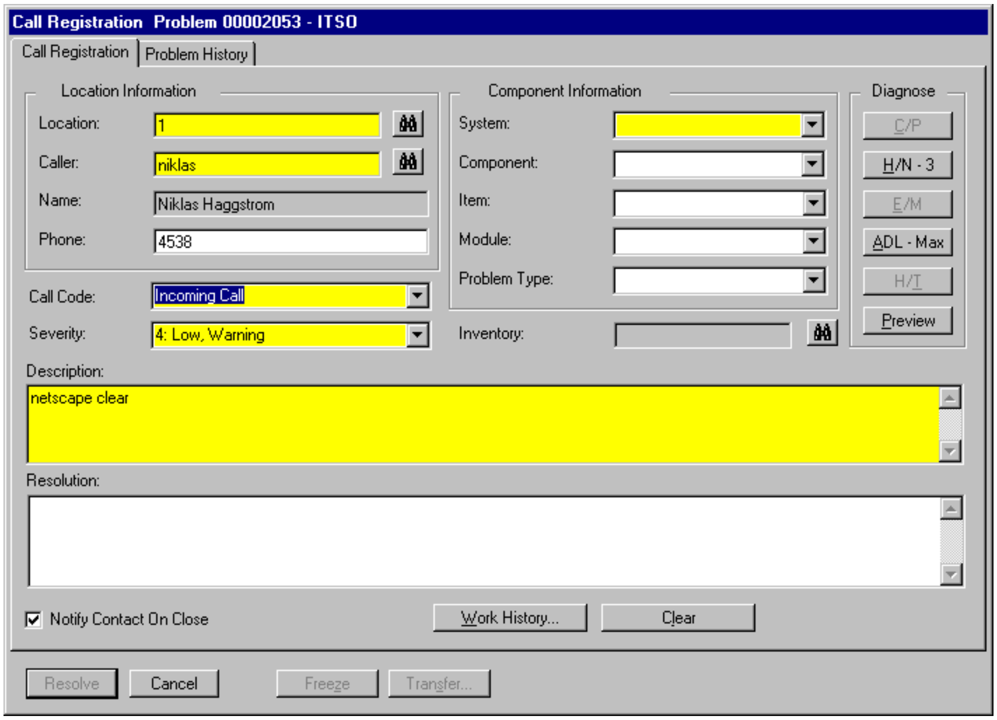


Figure 356. Entering the Call Data

Once the information has been entered the help desk operator clicks **Preview** This will reveal the screen shown in Figure 356 on page 311.

Due to the Description field containing the required parameters the ADL button can now be selected (see Figure 357 on page 312).

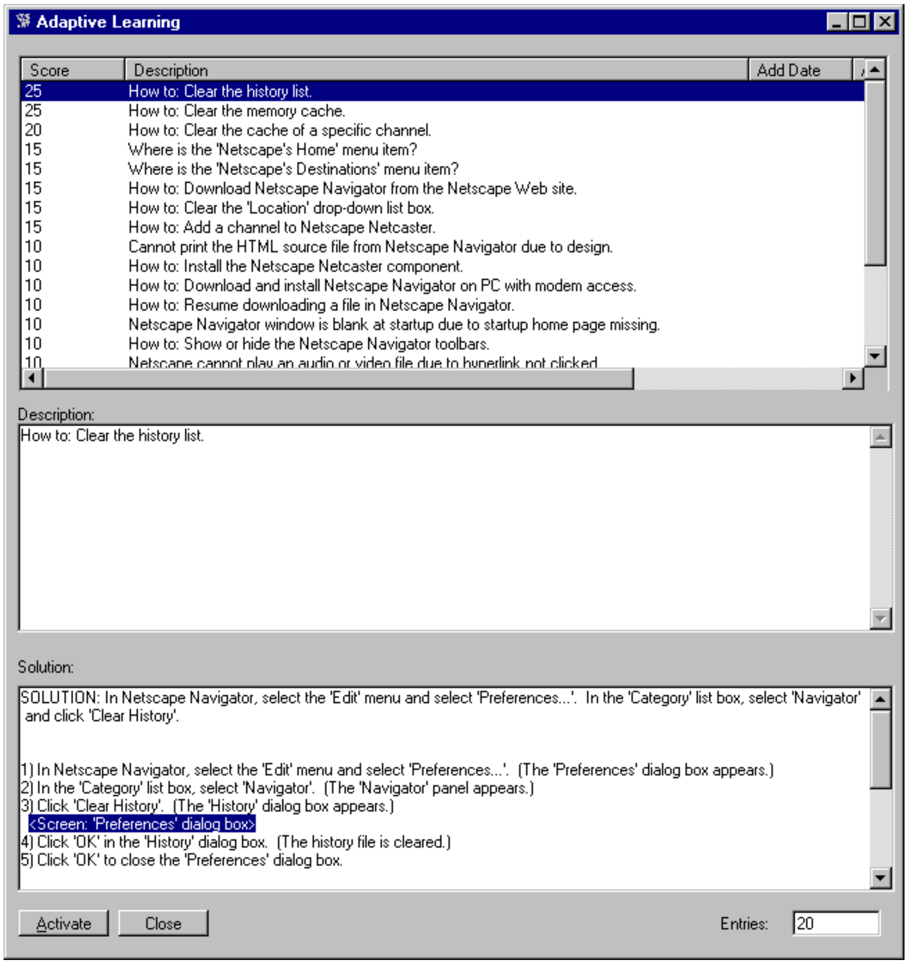


Figure 357. Adaptive Learning Screen

Also the other diagnostic tools contain data items populated by the Knowledge Pak, for instance, the common problems and error codes.

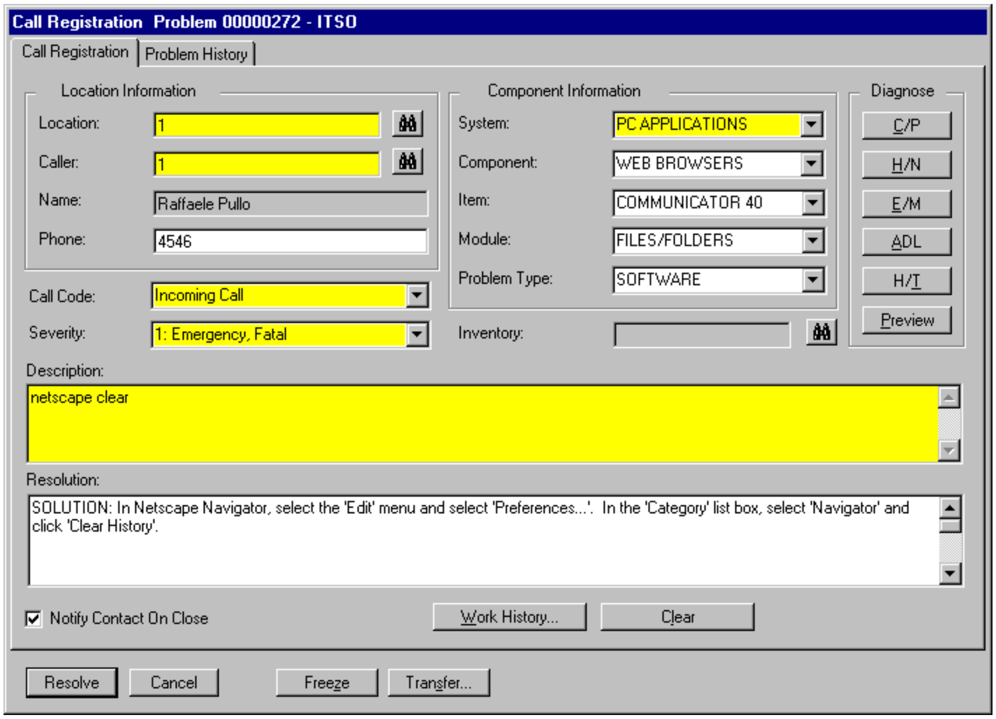


Figure 358. Call Resolution

Finally the call can be resolved by clicking on **Resolve** from the screen shown in Figure 358 on page 313.

7.5 Knowledge Pak - Example 2

The call is entered with a user that cannot find out how to undelete a file using Windows Explorer. The call is entered as shown in Figure 358 on page 313.

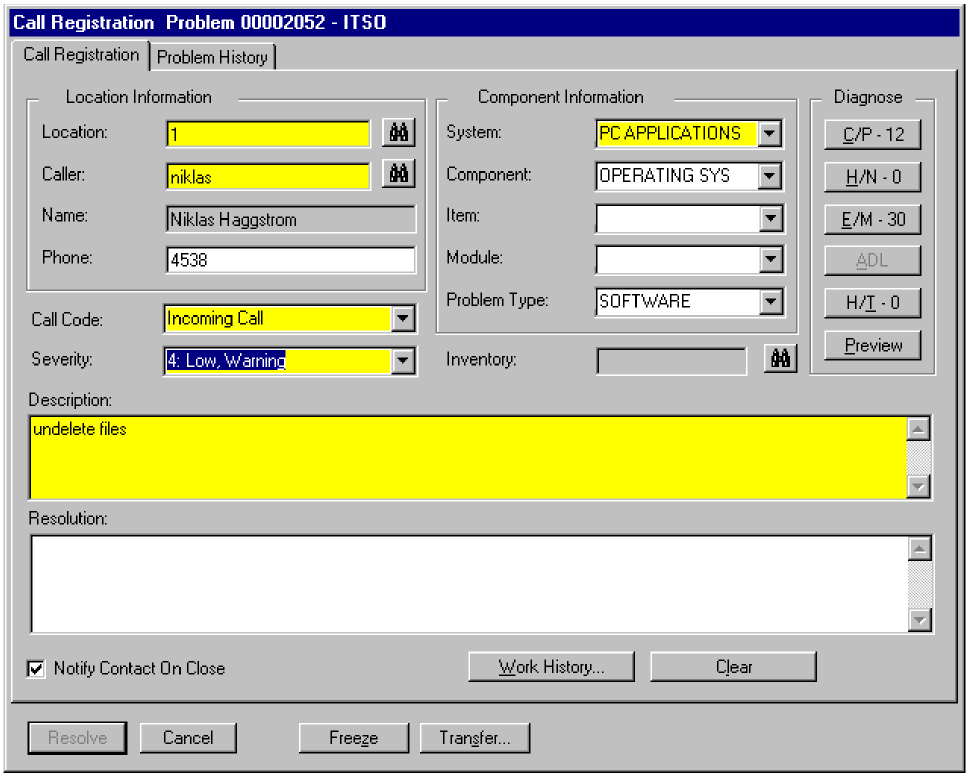


Figure 359. Call Registration

The description entered reads undelete files. By clicking on the **Preview** button we can see the values for the diagnostic aids appear.

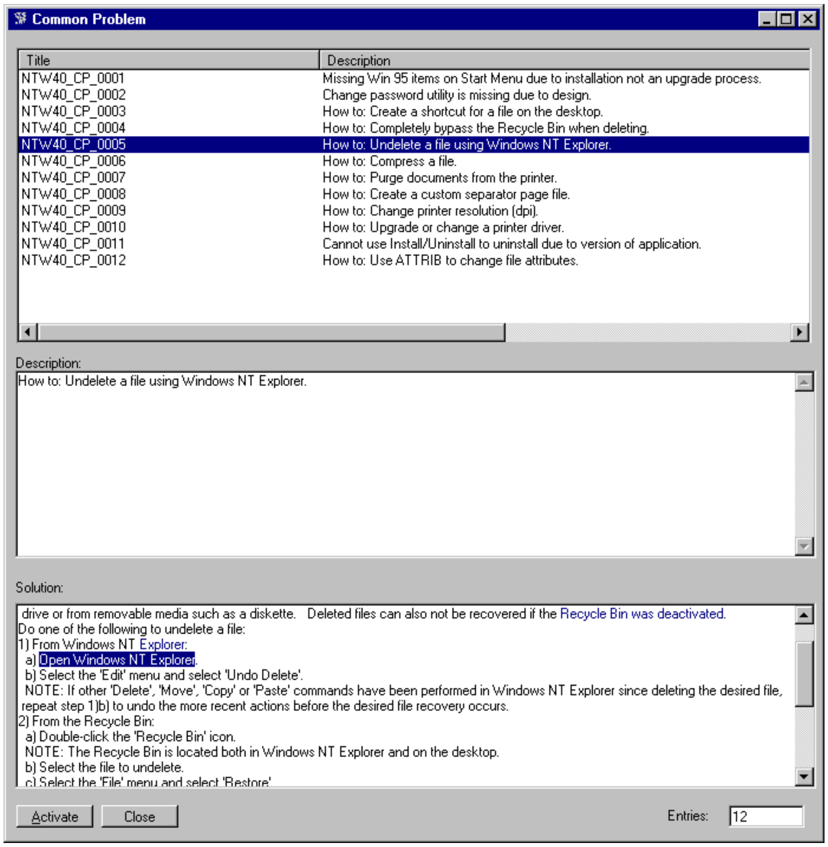


Figure 360. Common Problems

We selected the common problems(C/P). By browsing down the list of problems we located the one we were looking for (see Figure 360 on page 315). From here we selected **Activate**.

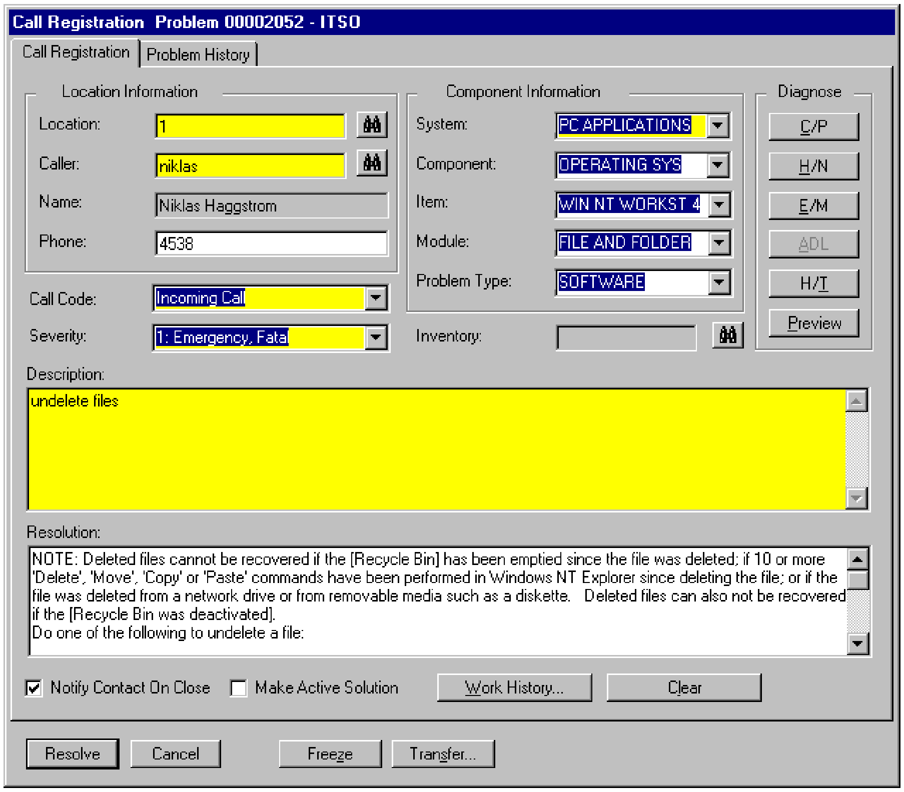


Figure 361. Call After Activate

Figure 361 on page 316 shows the new fields populated. The Resolution field informs the operator of the instructions to tell the user what to do.

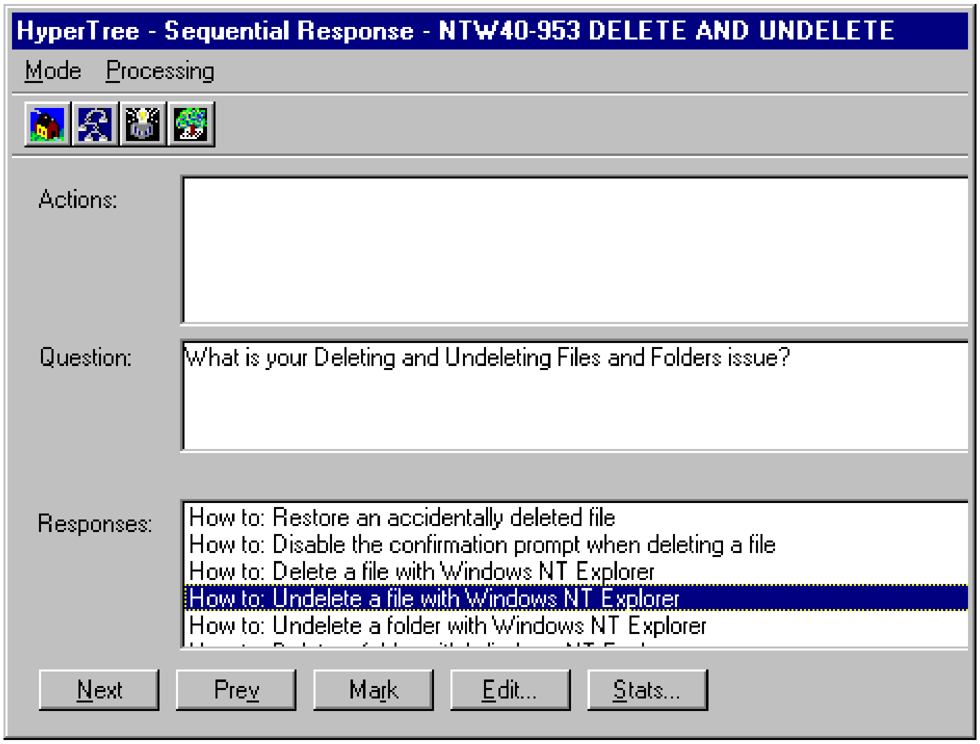


Figure 362. HyperTree for Undelete

Alternatively they could have used the HyperTree as shown in Figure 362 on page 317.

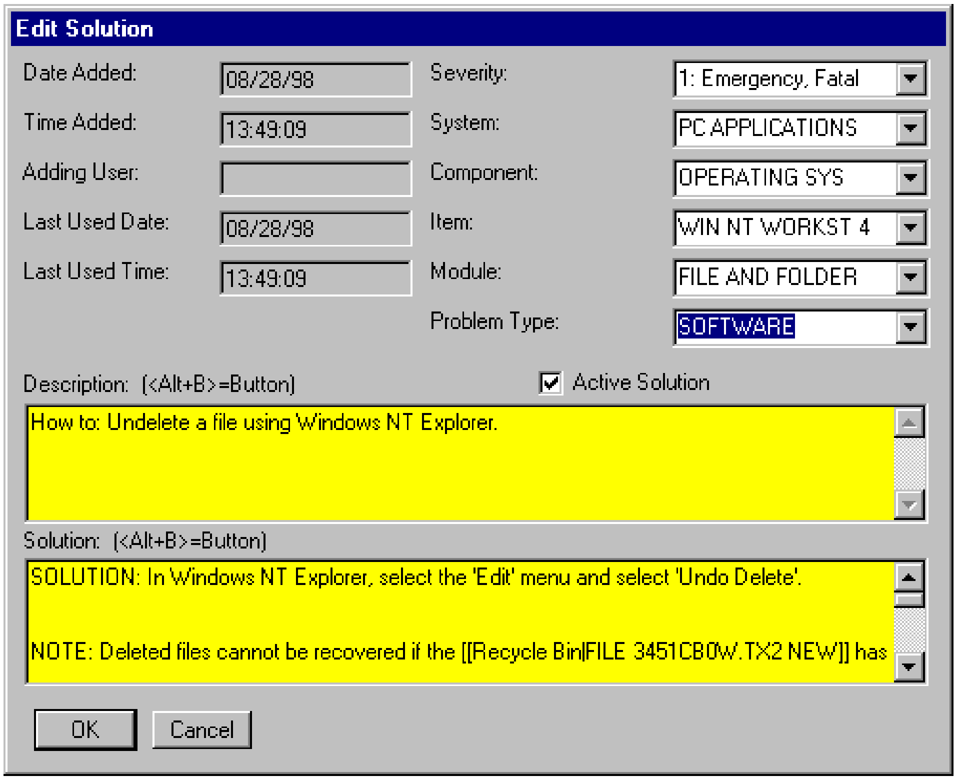


Figure 363. Edit Solution for the Delete Files

The solution is shown in Figure 363 on page 318.

7.6 Reporting with Tivoli Decision Support

This section provides an introduction to the capabilities of Tivoli Decision support. We show how to report the problems that were shown in the examples shown in this chapter.

One of the steps involved in effective problem management is analyzing the data contained in the problem management database. This can provide an overall high-level view of what is going on in our environment. This analysis can be made from an IT manager, with the focus on technical problems, or can be focused on how the help desk is performing.

Here we are going to show the technical point of view, looking for what happened and where. We also show the types of information that is contained in the problem management database.

7.6.1 Available Data

To report on the technical analysis data we need information about events, causes of events, sources, frequency, resolutions and automatic recovering procedures. Getting such information in a TDS environment means selecting the appropriate fields in a database table and correlating them by dimensions or criteria.

We are faced with two different environments: the TEC and the Advisor. The SCIM is the basis for most of the view when reporting on problems generated in the ADVISOR database. Analyzing TEC events is completely different from analyzing calls and problems trends.

To show the TEC events we had to map the Tivoli object_id to the inventory call field. We analyzed the flow of an event and the correlation with other related events, all of this at done at the TEC rule level.

7.6.2 Using the TDS

The first time the user interface is started you are asked to provide the location of the server's components and files.

From the Start menu select **Programs -> Tivoli Decision Support -> Tivoli Discovery Interface**.

Enter the shared source path. In our environment we linked to $SHAREDPATH on the file server machine. We left the default values for the database option.

From the Topic map panel we select the **Role** tab and define which role we will have for retrieving the reports we want to see (see Figure 364 on page 319).

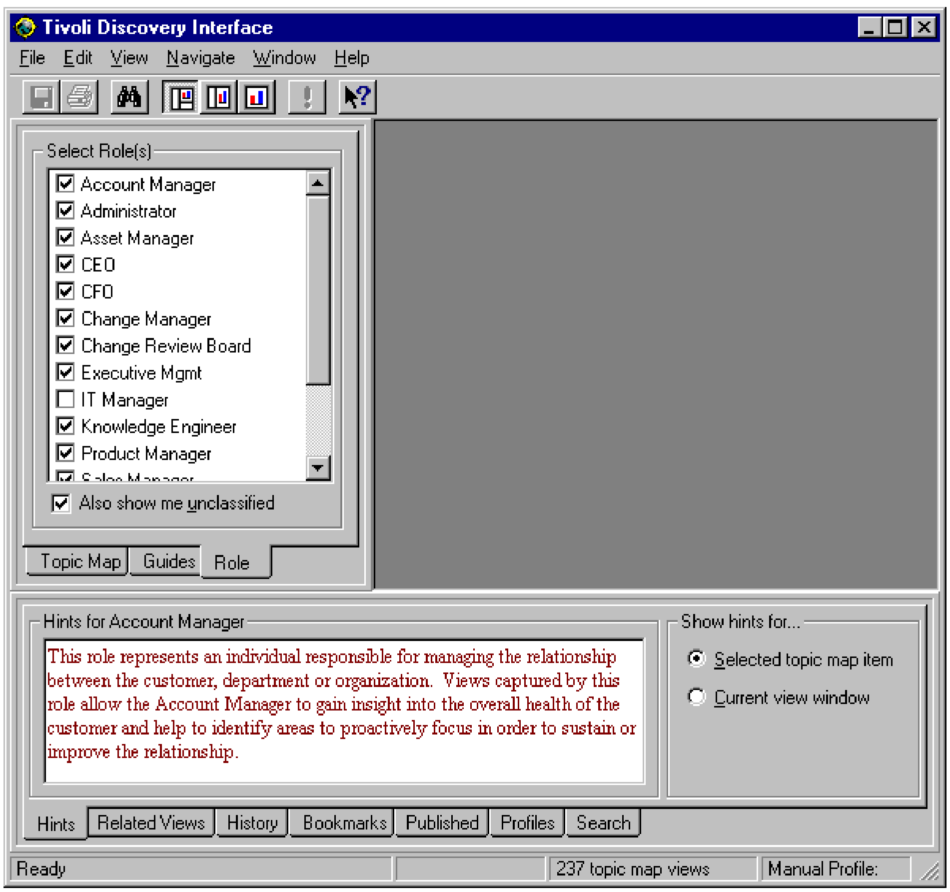


Figure 364. TIvoli Discovery Roles

The guides are also selected from this screen by clicking on the **Guide** tab. The TDI screen is shown in Figure 365 on page 320.

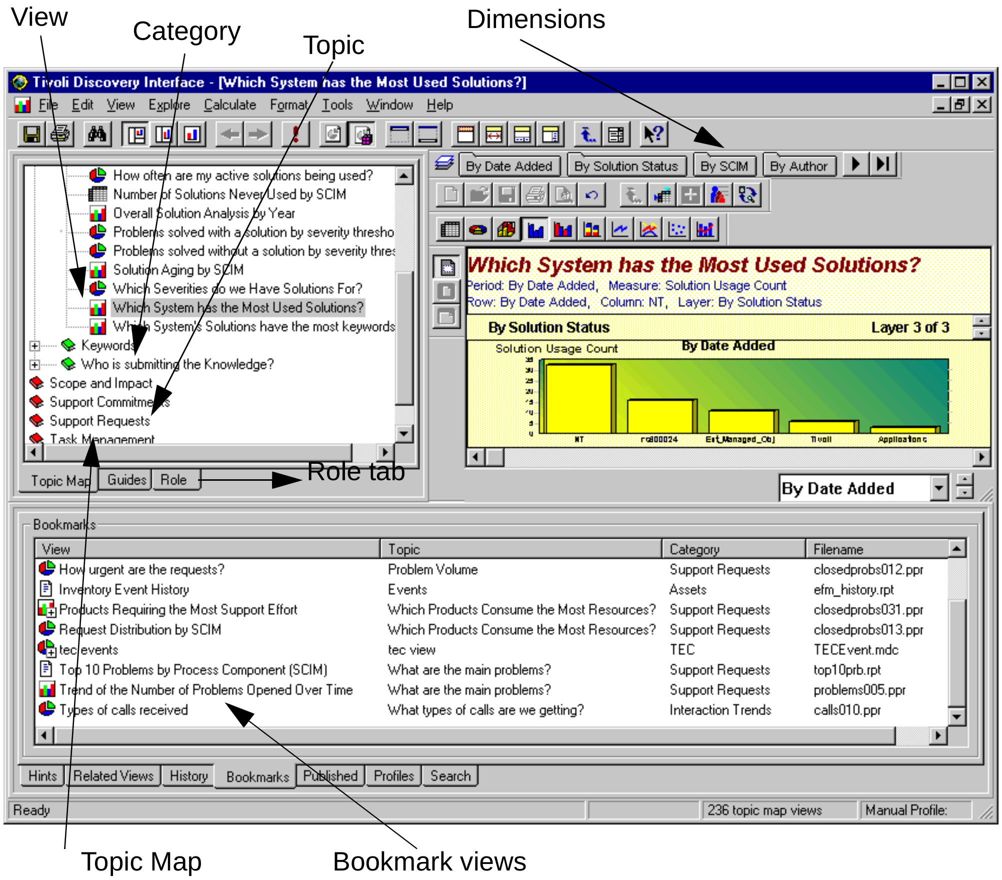


Figure 365. Tivoli Discovery Interface

In the Topic Map you can see the views, or reports you are going to use grouped by topic and category. All of these are organized and defined when the discovery guide is installed.

The View panel contains opened reports, detailed views, multidimensional views and PowerPlay reports.

When we open a PowerPlay report we can enable the option of viewing dimensions, power bars and displays. All of which can be done by choosing the **View** pull-down menu. Selecting a slice of a pie report for instance and then dragging the dimension's folder over it will allow us to slice and dice into different scopes for one single report.

The Bookmarks panel allows us to locate the most significant reports. The Bookmarks tab in the Hints panel shows favorite user-defined views and reports.

Please refer to *Tivoli Discovery Interface User Guide* for a complete description of the interface options.

7.6.3 Reports of Advisor problems

Here we look at the problems received by the help desk. From the Topic map select the **Support Request** category. Next select the **Which Products**

Consume the Most Resources topic. Finally double-click on **Request Distribution by SCIM**.

Then from the Interaction Trends category and the What type of calls we are getting topic, double-click on **Types of calls received**.

Selecting the **Full Screen** tab, and resizing the opened reports, you will get the following picture. As you can see now, the mapping data between TEC and Advisor, and the SCIM structure definition are the main criteria around which the data scope is defined.

In the following picture calls are measured and organized by the system level of the SCIM hierarchy in the lower part, and by the SCIM component in the upper report, being a component mapped to a problem type.

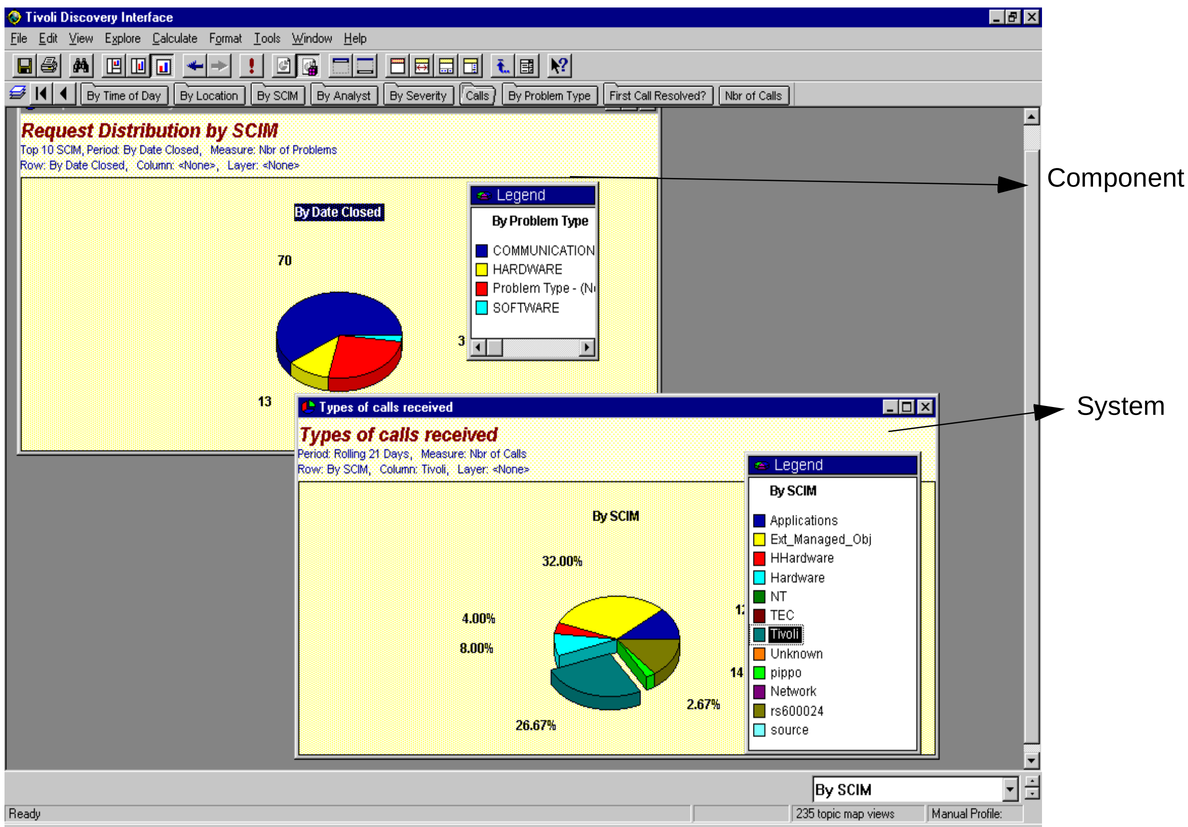


Figure 366. SCIM views of calls and problems

Now we want to better understand if our problem management environment is effectively configured and if the problems are closed using the diagnostic aids provided with Advisor.

From the Diagnostic category and the How Effective are our Diagnostics topic, click on the **Diagnostic Aid usage by SCIM** view.

Figure 367 on page 322 shows that the diagnostic tools are very effective due to most of the NT problems being solved using the diagnostic tools.

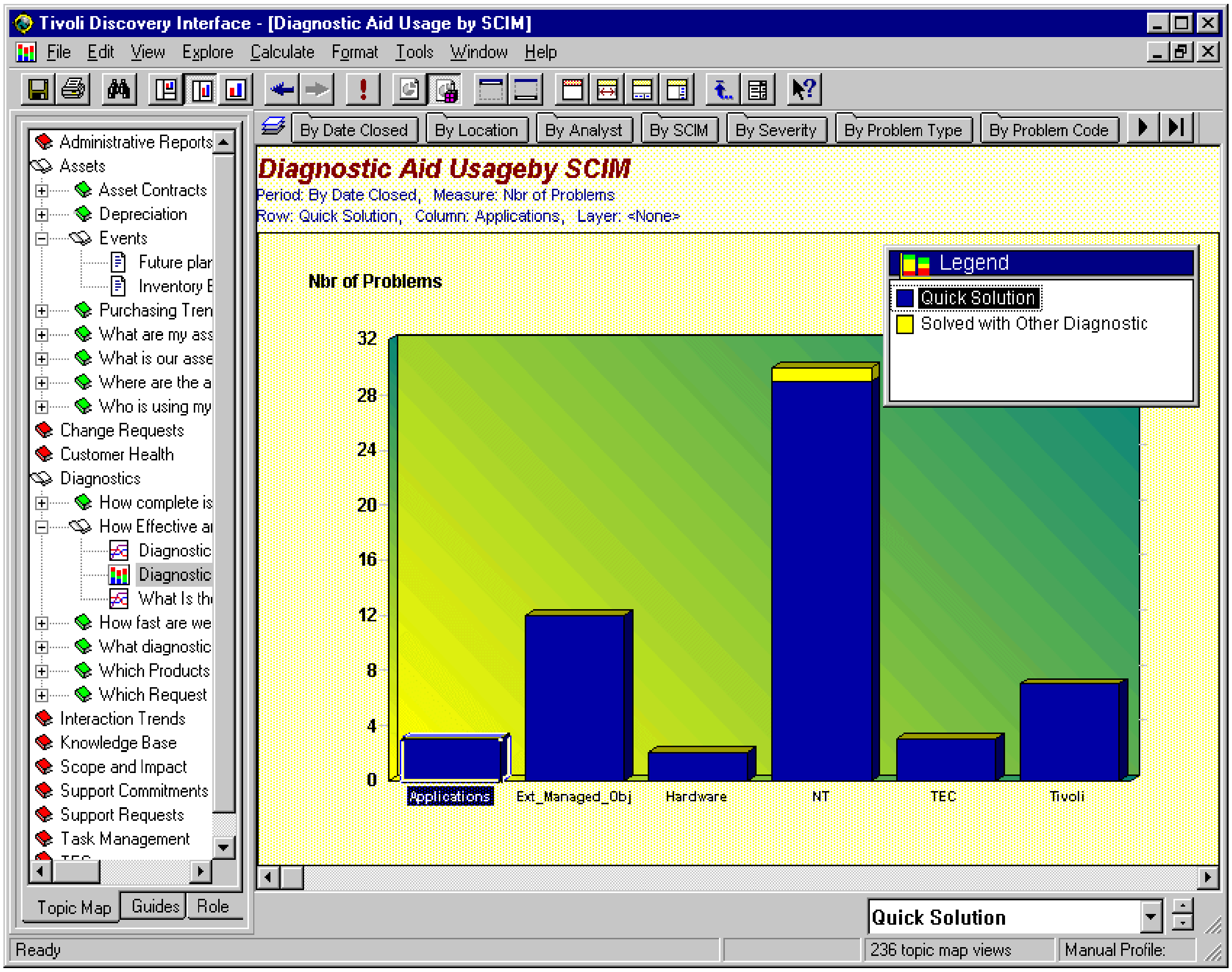


Figure 367. How Effective Is Our Diagnostic?

Other views of our data are shown in the following screens. The selection portion of the screen is shown in Figure 368 on page 323.

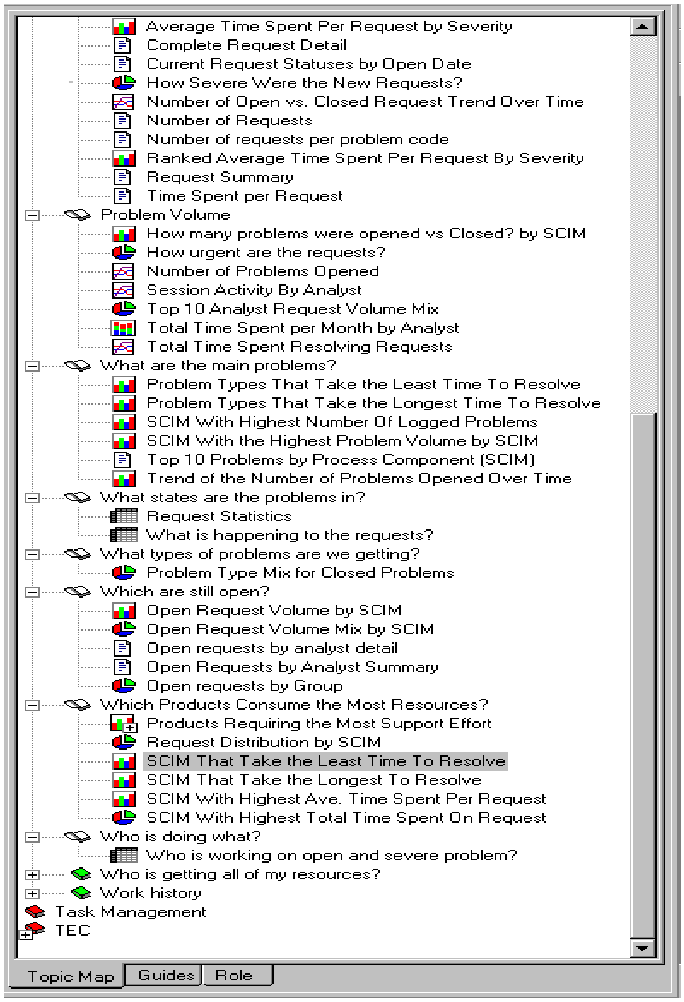


Figure 368. Report Options

The time to resolve results is shown in Figure 369 on page 324.

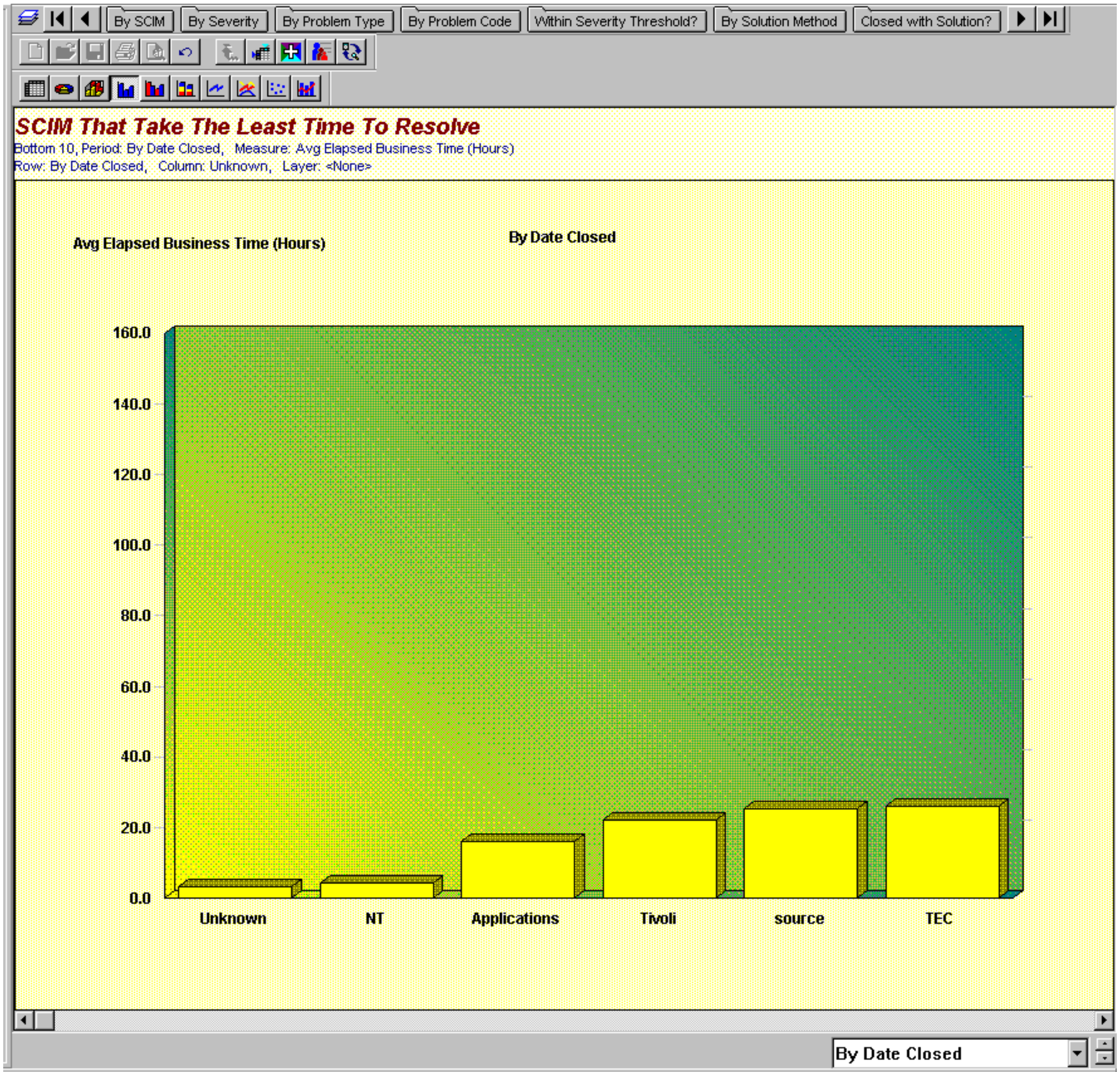


Figure 369. Resolution TIme Spent on Calls

The number of CLOSED problems is shown in Figure 370 on page 325.

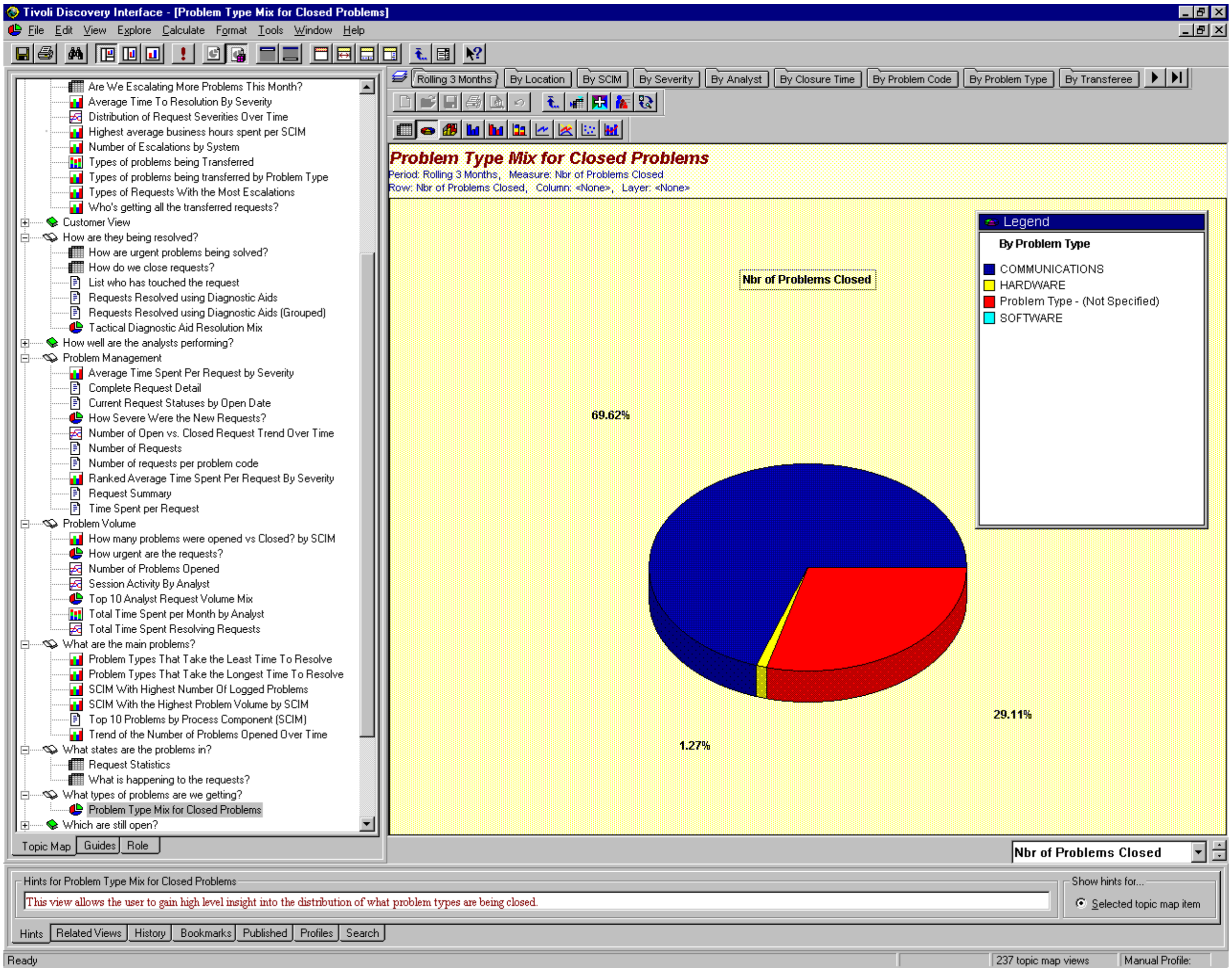


Figure 370. Closed Problems

The actual time spent on problems is shown in Figure 371 on page 326.

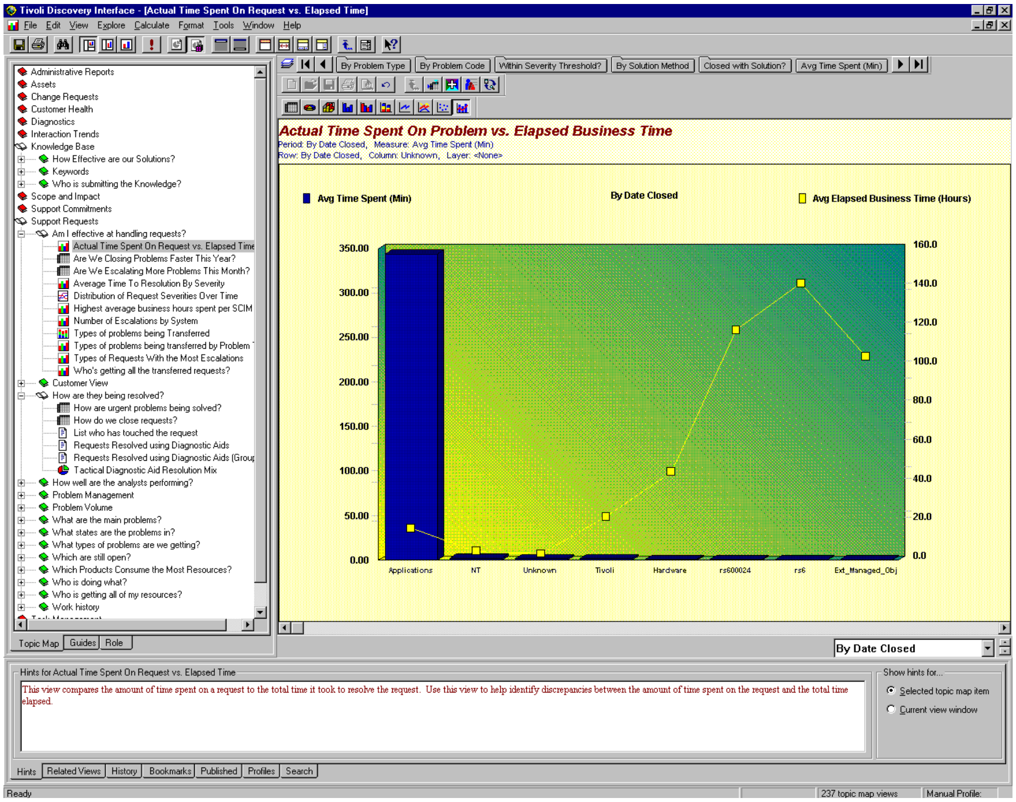


Figure 371. Actual Time Spent on Problems

The output for which systems used most solutions is shown in Figure 372 on page 327.

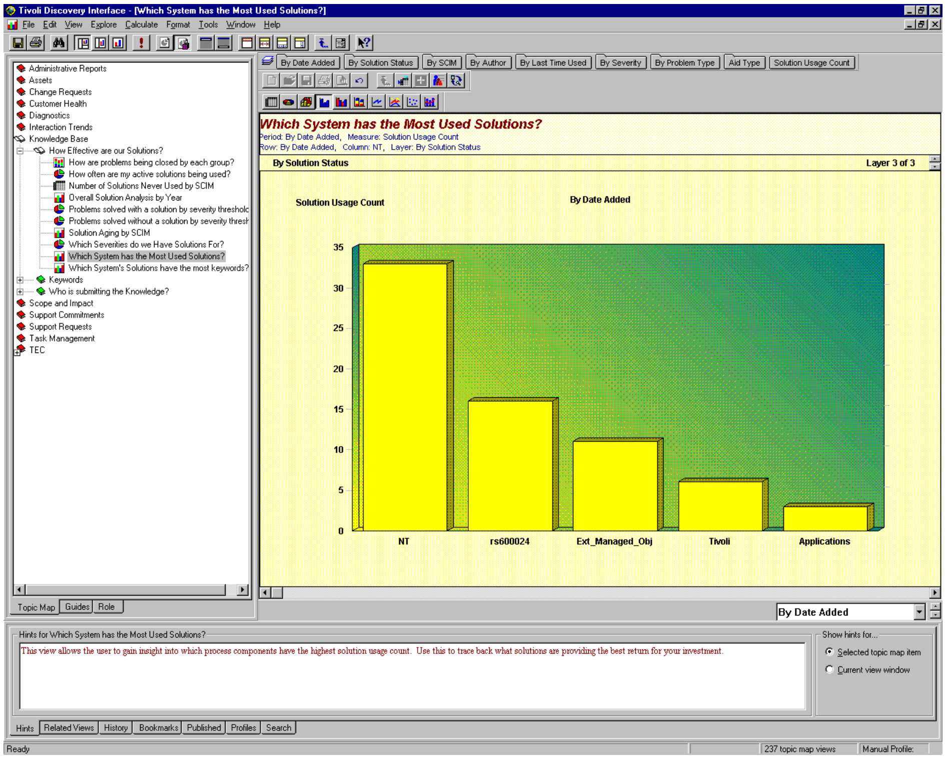


Figure 372. Systems with Most Solutions

The next step is to publish the data.

7.6.4 Publishing Reports

The reports can be generated in HTML format for publishing internally or externally on the Web. From the Tivoli Discovery Interface window select F**ile-> Publish -> to HTML**.

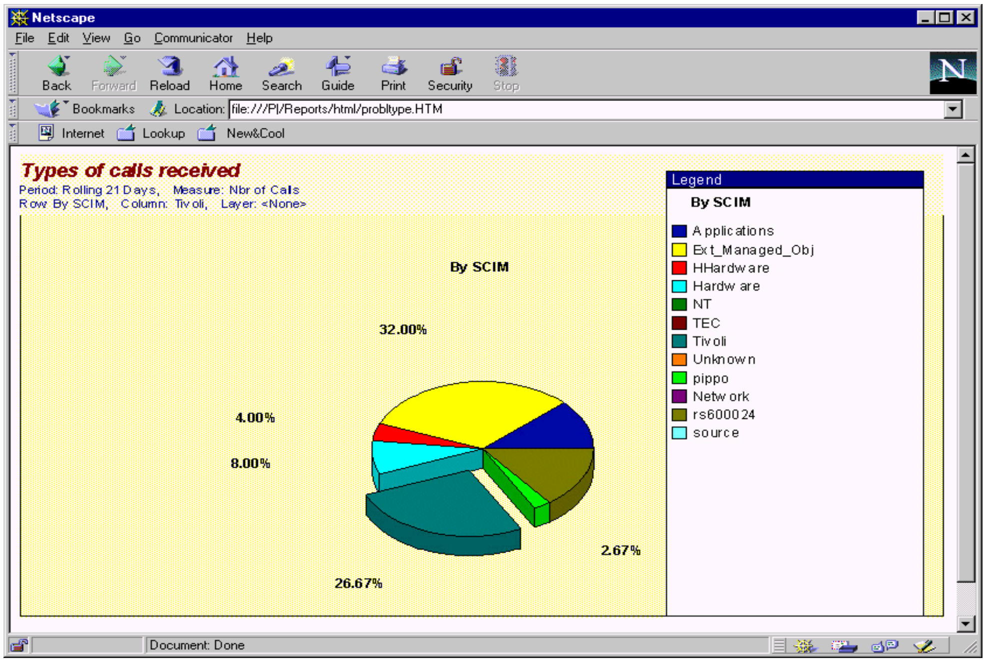
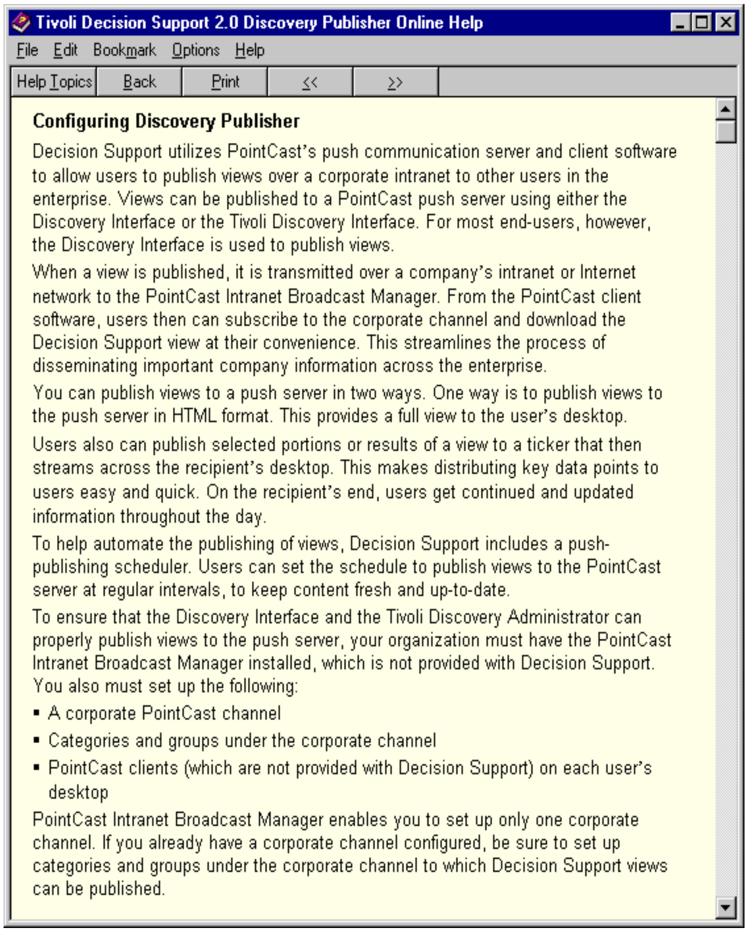


Figure 373. Report Saved As HTM File

Actually Tivoli Decision Support can also publish on a push server similar to PointCast.

For configuring the publishing feature from your Window panel run **Start -> Programs -> Tivoli Decision Support -> Tivoli Discovery Publisher**.

Figure 374 on page 329 shows an overview of this feature.


Tivoli Decision Support 2.0 Discovery Publisher Online Help

File Edit Bookmark Options Help

Help Topics | Back | Print | << | >>

Configuring Discovery Publisher

Decision Support utilizes PointCast's push communication server and client software to allow users to publish views over a corporate intranet to other users in the enterprise. Views can be published to a PointCast push server using either the Discovery Interface or the Tivoli Discovery Interface. For most end-users, however, the Discovery Interface is used to publish views.

When a view is published, it is transmitted over a company's intranet or Internet network to the PointCast Intranet Broadcast Manager. From the PointCast client software, users then can subscribe to the corporate channel and download the Decision Support view at their convenience. This streamlines the process of disseminating important company information across the enterprise.

You can publish views to a push server in two ways. One way is to publish views to the push server in HTML format. This provides a full view to the user's desktop.

Users also can publish selected portions or results of a view to a ticker that then streams across the recipient's desktop. This makes distributing key data points to users easy and quick. On the recipient's end, users get continued and updated information throughout the day.

To help automate the publishing of views, Decision Support includes a push-publishing scheduler. Users can set the schedule to publish views to the PointCast server at regular intervals, to keep content fresh and up-to-date.

To ensure that the Discovery Interface and the Tivoli Discovery Administrator can properly publish views to the push server, your organization must have the PointCast Intranet Broadcast Manager installed, which is not provided with Decision Support. You also must set up the following:

- A corporate PointCast channel
- Categories and groups under the corporate channel
- PointCast clients (which are not provided with Decision Support) on each user's desktop

PointCast Intranet Broadcast Manager enables you to set up only one corporate channel. If you already have a corporate channel configured, be sure to set up categories and groups under the corporate channel to which Decision Support views can be published.

Figure 374. Discovery Publisher

7.6.5 Reports of Events Received by TEC

Here we wanted to show how the TEC Discovery Guide will work when it becomes available. We show this using the beta release of the code.

For the TEC-related views we can see how effective our monitoring system is. Figure 375 on page 330 shows reports based on events by event_class. These are decided into severity levels measured by the number of events received in one week.

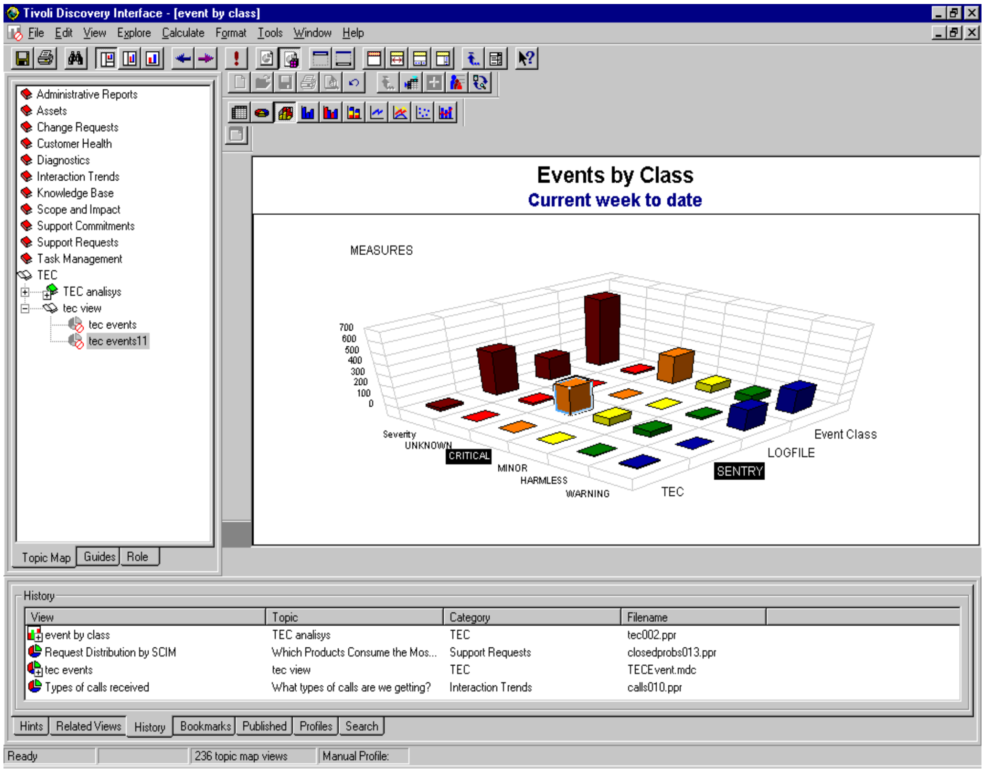


Figure 375. TEC Events by Class

We can see that our Distributed Monitoring monitors are sending a large number of events. It may be considered too high. One option is to configure Distributed Monitoring to start performing automatic based on threshold levels. We may be able to resolve problems at the source. Also by receiving so many critical events we may want to review our policies.

Next we take a look at our critical servers, what servers are sending events and how many (see Figure 376 on page 331).

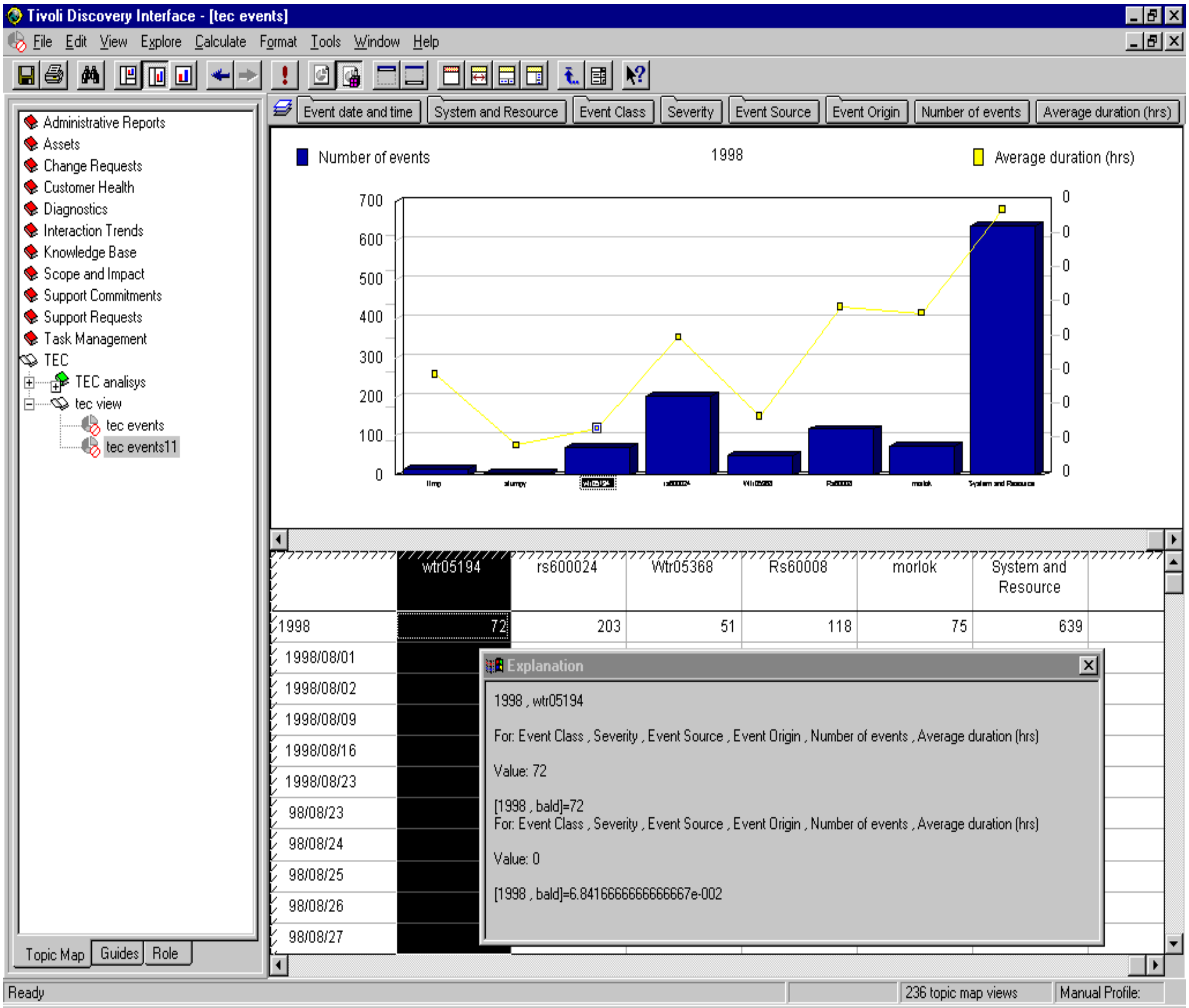


Figure 376. TEC Events by Hostname

Within a TEC analysis the number of events has to be considered carefully. It should be related to the type of source that is sending the event. With this type of information we can see that more configuring, filtering and thresholding is required. For instance, more rules need to be developed to provide more correlation.

Chapter 8. Problem Determination

This chapter briefly discusses what areas of the management environment you may want to monitor. When providing a complete end-to-end problem management solution involves a large number of applications including databases, network and systems management tools and application daemons. Some of the areas that need to be monitored include:

- Monitoring the databases
- Monitoring the application
- Reporting of specific problems

We have documented some areas that should be used as a guide to what can be monitored on each server within our management framework. Some considerations are as follows:

- What management tools do we need for perform this management?
- What logfiles can we see?
- What additional Plus modules can we use?

Next we list the areas that can be monitored.

8.1 Elements and Resources

Table 22 on page 333 shows what elements need to managed.

Table 22. Monitored Resources

Hardware Element	Resource	Method	Notification Mechanism
Problem Manager Server	Disk/Memory	Distributed Monitoring	TEC/Notices
(Change/Asset Manager)	Daemons	Tivoli PLUS Module	TEC
	SQL Connection	PLUS Module	TEC
NT Problem management Clients	SQL Connection	Scripts	TEC/Notices
	Disk	NT Adapter	TEC
TMR Server	Disk/Memory	Logfile Adapter	TEC/Notices
	oserv Daemon	Logfile Adapter	TEC/Notices
	Paging Space	Distributed Monitoring	TEC/Notices
NSM Gateway	Disk/Memory	Distributed Monitoring	TEC/Notices
	Daemons	Logfile Adapter/Scripts	TEC/Notices
TEC Server	Disk/Memory	Distributed Monitoring	TEC/Notices
	TEC Daemons	Logfile Adapter/Scripts	Notices/Pop-Up
	RIM Connection	Logfile Adapter/Scripts	Notices/Pop-Up
Inventory	Disk /Memory	Distributed Monitoring	TEC/Notices
	RIM Connection	Scripts	Notices/Pop-Up

Hardware Element	Resource	Method	Notification Mechanism
NetView Server	NetView Daemons	NetView Adapter/Scripts	TEC/Notices
	Disk/Memory	Distributed Monitoring	TEC/Notices
RDBMS SQL Server	Daemons	Sybase/Oracle Plus Module	TEC/Notices
	SQL Logfile	Logfile Adapter	TEC/Notices
Network Connections	Availability	NetView	TEC/NetView

The scripts referred to are shell scripts that could use commands such as `df` to monitor the server resources. Then by using wpostemsg or other mechanisms the errors can be reported to the systems manager responsible for the availability of the management platforms.

One point to remember is depending on where the error occurs the notification mechanism may be failing. This is why we suggest pop-up windows to notify you that the servers have failed.

8.2 Using the Problem Management Tivoli Plus Module

The Tivoli Plus module provides a number of monitoring tools for the environment.

If the communications fail between our servers, then either we are relying on redundant servers or we may have a serious problem. This section focuses on the SA-EXPERTISE Plus module.

Figure 377 on page 334 shows the main screen for the Plus module.

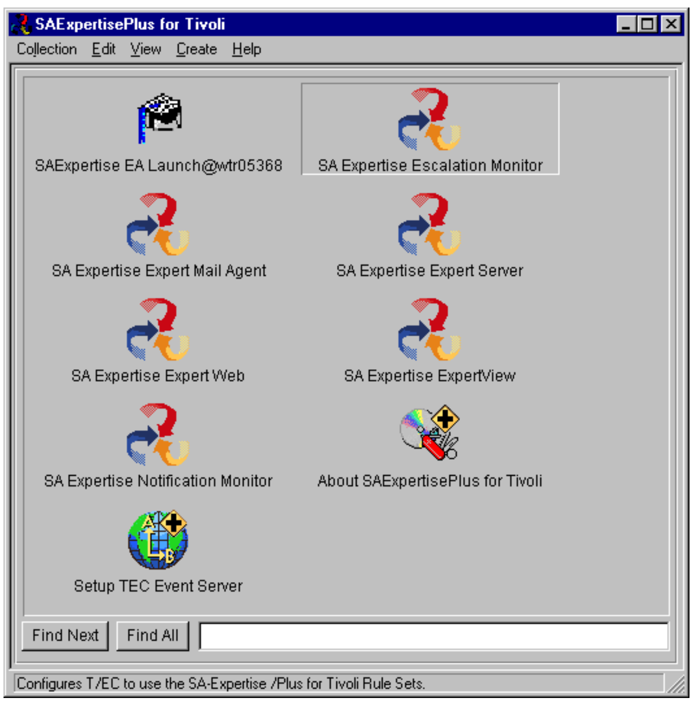


Figure 377. SAExpertise Plus Module

This is the main collection that contains the tasks, configuration programs and monitors for the problem management environment. To define the event group for the SA monitors for the TEC add the group shown in Figure 378 on page 335.

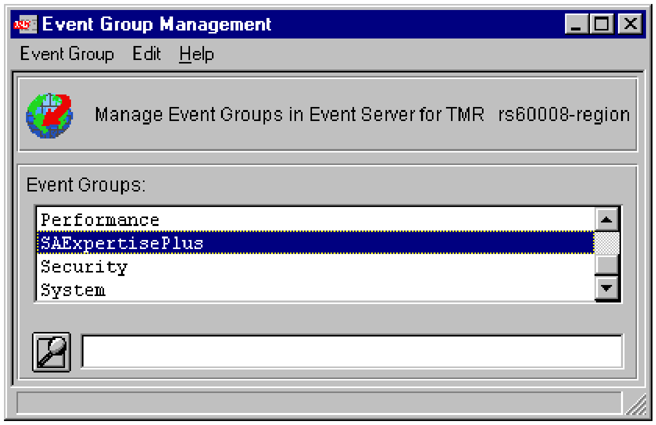


Figure 378. Event Query

Now we can see these specific events from the TEC console. To access the export server functions click on the SA Expertise Server. The expert server functions are shown in Figure 379 on page 335.

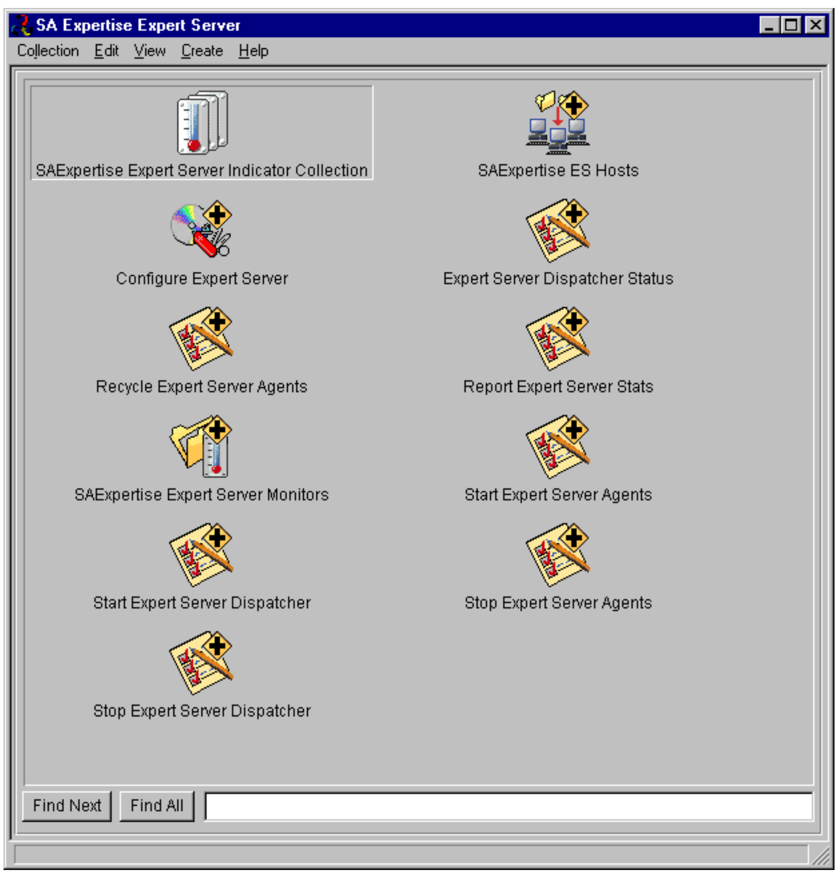


Figure 379. Expert Server

The monitors for the expert server are shown in Figure 380 on page 336.

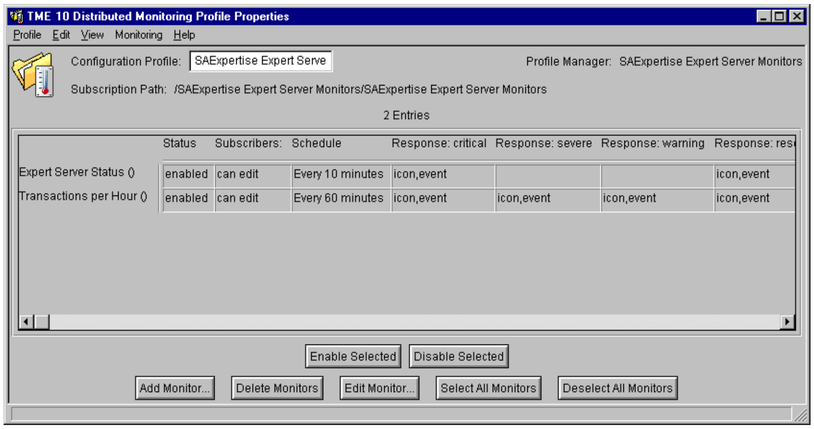


Figure 380. Expert Server Monitors

The example shows the monitor for transactions per hour. When this exceeds 500 a critical event will be sent to the TEC.

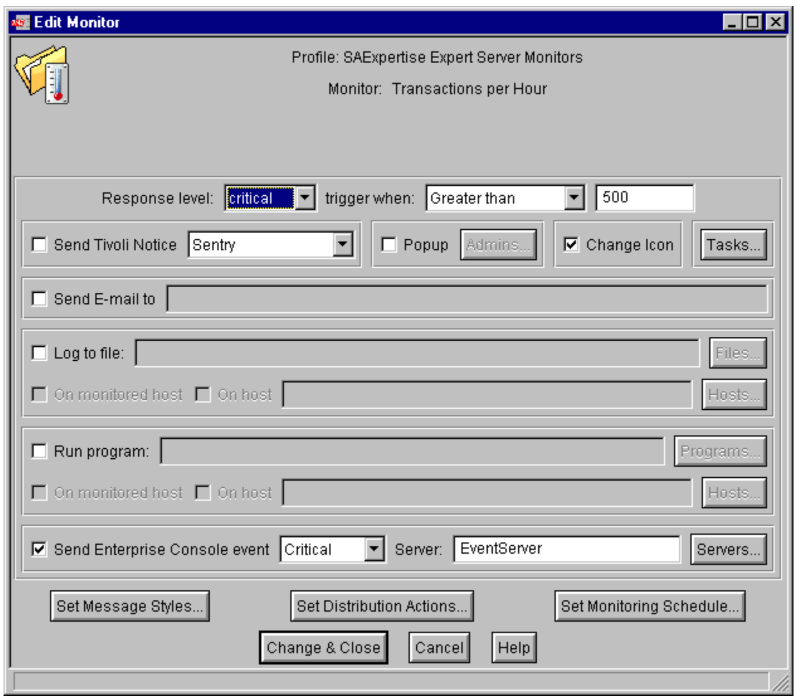


Figure 381. Edit Monitors

For a complete list of available monitors see Figure 382 on page 337.

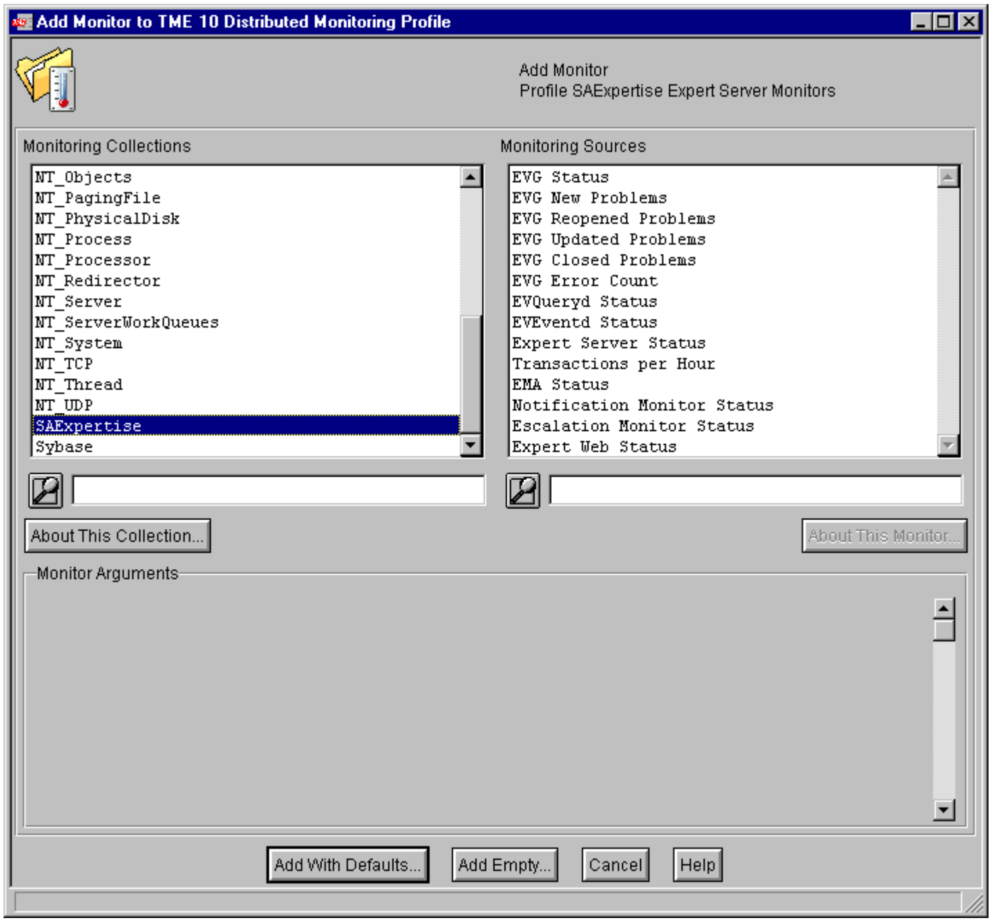


Figure 382. List of Available Monitors

To define the subscribers see Figure 383 on page 337.

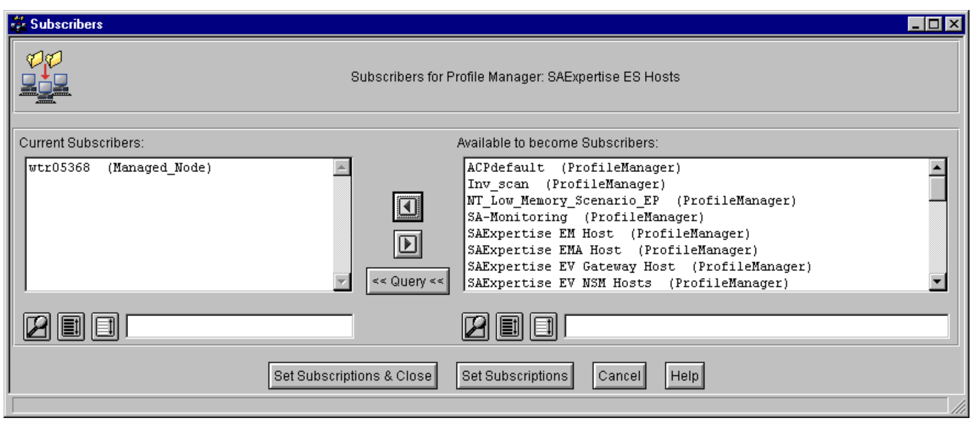


Figure 383. Setting Up the Subscribers

The ExpertView functions are shown in Figure 384 on page 338.

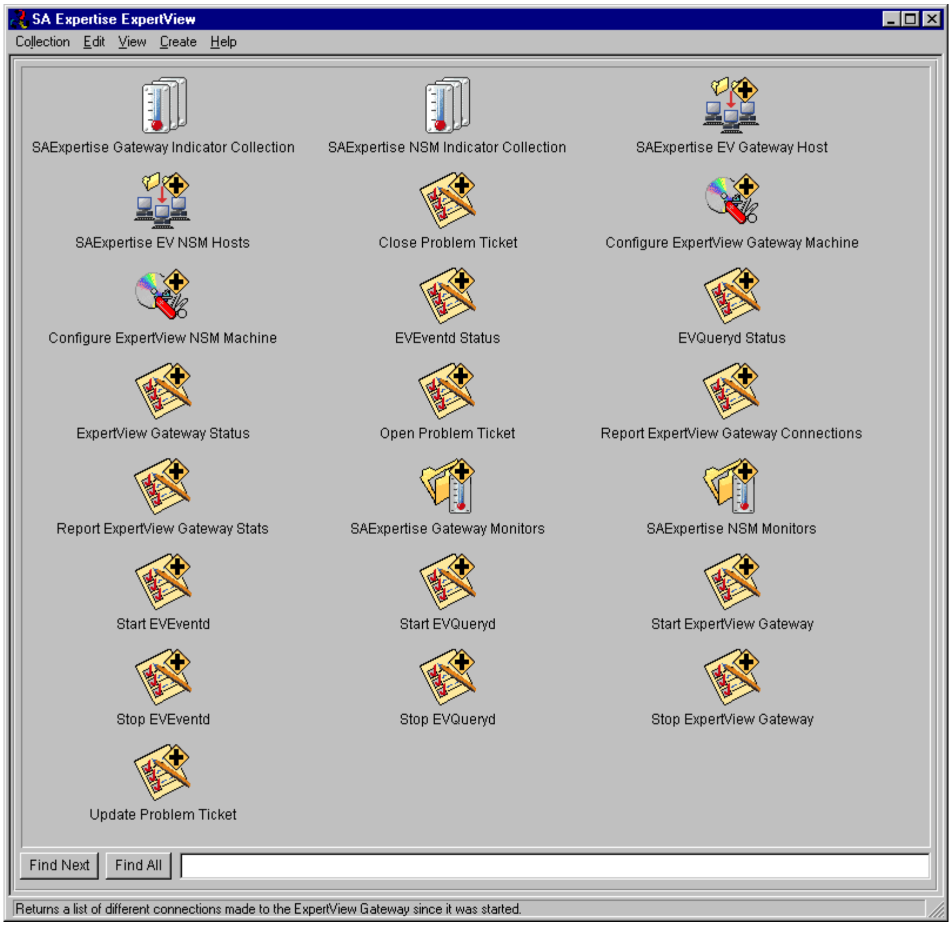


Figure 384. ExpertView Functions

The options here provide more than just monitoring of the environment. They also include opening, updating and closing problem management tickets.

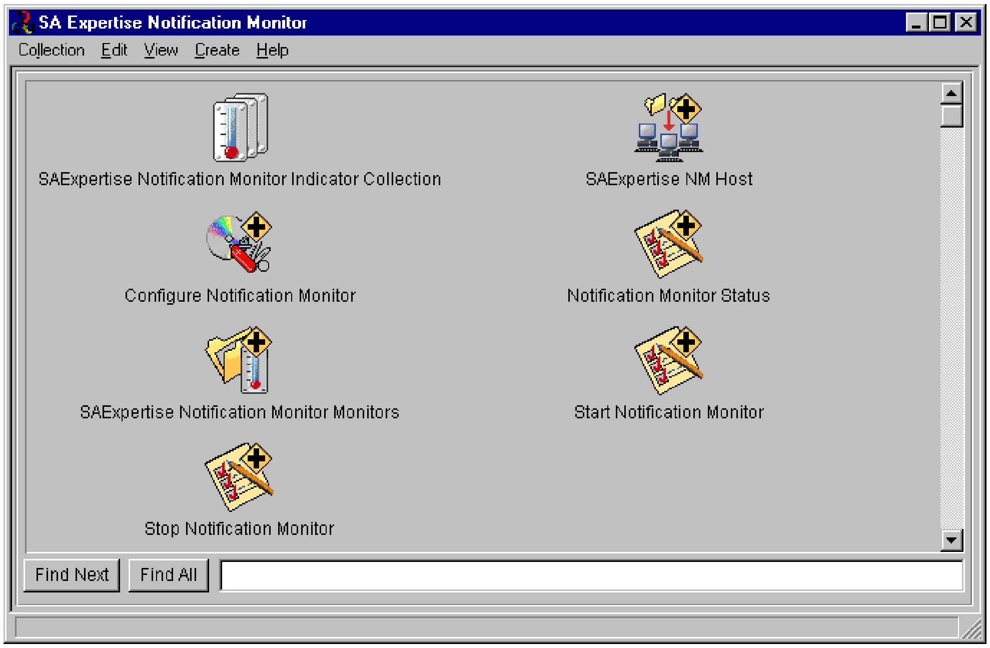


Figure 385. Notification Monitors

Figure 385 on page 339 shows the notification server monitors.

The Plus module is provided with a set of TEC rules. These rules can be used in conjunction with the monitors. The next section explains the rules.

8.3 Monitoring Overview

With the Tivoli/Enterprise Console, SA-EXPERTISE/Plus for Tivoli provides a set of filters to identify events. SA-EXPERTISE/Plus for Tivoli also includes a set of predefined correlation rules to automate the task of responding to specific events. An event is any significant change in the state of system resources or an application. In the case of SA-EXPERTISE/Plus for Tivoli, an event is a change in a monitored resource that affects applications in the EXPERTISE suite. Examples of events are:

- Starting and stopping of a process
- An ESM application becoming available
- A threshold being met or exceeded

Event management provides predefined or automated responses to specific events. Thus, potential problems are identified and responded to before they cause system down time. For example, if a process fails, the Tivoli/Enterprise Console can automatically re-run the process. The Tivoli/Enterprise Console can also notify the system administrator of repeated process failures that may indicate a more severe problem with an application or the network. For some events, there is no automated response except that they are displayed in the TEC.

8.3.1 Events and Rules

If the selected event does not contain the managed_object event slot or this event slot is empty, then a combination of the source and the origin is used when opening or updating an Expert Advisor problem. For example, if an event's source is NT and the event's origin is10.32.58.6, then the ExpertView managed object would be NT_10.32.58.6.

The Tivoli/Enterprise Console event ID is tracked to allow Expert Advisor to perform callbacks. The Event Message field and other event information is stored in the EA_Problem_Description field.

If the selected event does not contain the managed_object event slot or this event slot is empty, then a combination of the source and the origin is used when opening or updating an Expert Advisor problem.

The Event Message field and other event information is stored in the EA_Problem_Description field. For more information on managed objects, callbacks, and the EVProb program, see the *SA-ExpertView Network Administrator's Guide*. This table describes the events and rules that Tivoli/Distributed Monitoring may use when monitoring EXPERTISE resources. The table is organized according to the notification that Tivoli/Distributed Monitoring sends to the Tivoli/Enterprise Console. Tivoli/Distributed Monitoring Event Messages "

8.3.2 TEC Events and Rules from the Plus Module Configuration

With the supplied rules the following list details what events can be sent using the monitors with the Plus module.

Table 23. Rules to Monitor the Problem Management Environment

Event Class	Severity	Correlation	Message
SA_ES_TransPerHour	CRITICAL	Send a critical event and close related events	ExpertView Gateway is down
SA_ExpertView_Gateway_Status	FATAL	If >2 times a day, send notification to "Sentry-Urgent" notification group - Re-open original event and alter Msg box on the TEC Desktop to reflect this - Increment original event's repeat count	ExpertView Gateway is up
SA_ExpertView_Gateway_Status	HARMLESS	Close related fatal event and auto-acknowledge this event after 2 minutes	ExpertView number of problems opened per hour is less than 40.
SA_EVG_OpenedProblems	CRITICAL	Send a critical event and close related events.	ExpertView number of problems closed per hour is less than 40.
SA_EVG_OpenedProblems	CRITICAL	Send a critical event and close related events.	ExpertView number of problems closed per hour is less than 40.

Event Class	Severity	Correlation	Message
SA_EVG_ClosedProblems	CRITICAL	Send a critical event and close related events.	ExpertView number of problems updated per hour is less than 40.
SA_EVG_ReopenedProblems	CRITICAL	Send a critical event and close related events.	ExpertView number of errors per hour is less than 1.
SA_EVG_ErrorC	CRITICAL	Send a critical event and close related events.	Notification Monitor is down.
SA_ExpertView_EVEventd_Status	FATAL	If >2 times a day, send notification to Sentry-Urgent notification group- Re-open original event and alter Msg box on the- T/EC Desktop to reflect this - Increment original event's repeat count	EVEventd is up.
SA_ExpertView_EVEventd_Status	HARMLESS	Close related fatal event and auto-acknowledge this event after 2 minutes	EVQueryd is down.
SA_ExpertView_EVQueryd_Status	FATAL	If >2 times a day, send notification to "Sentry-Urgent"- notification group- Re-open original event- and alter Msg box on the- T/EC Desktop to reflect this- Increment original event's repeat count	EVQueryd is up.

These rules are contained in the file:

$BINDIR/generic_unix/TME/PLUS/SAEXPERTISE/saexpertise_monitors.rls.

8.4 ExpertView Gateway

The gateway is very much involved in the integration between the TEC and the TDS products. It could be useful to start the gateway in debugging mode when experiencing problems. To do this use the parameters -D1 -D2.

In our environment we have the gateway running on an NT system. To run the gateway in debugging mode we configured a shortcut invoking the following command:

```
C:\SAI\WINASE32\kml.exe evgate -D1 -D2
```

For more details on those parameters refer to the product manual *Gateway Administrator's Guide.*

In this way you will get more information the Gateway Error Display window.

8.5 TroubleTicket.sh Script

Implementing the integration between TEC and TDS products, the TroubleTicket.sh script is very important because it is the script that actually invokes the EVProb command (see "TEC/Advisor Integration and Data Mapping" on page 185). So, we provide here some suggestions on how to perform problem determination using that script.

Note that this script logs messages, error messages or just informational messages, in the file:

```
/tmp/SAExpertise.log
```

This file could contain information useful during problem determination with TroubleTicket.sh script.

A good solution to monitor that file is to configure the TEC Log File Adapter for the SAExpertise.log file. The adapter will generate a TEC event when the TroubleTicket.sh script logs an error message into the file.

If you have problems submitting a call, we suggest you to use the following approach to debug the problem.

In the TroubleTicket.sh script before the statement that calls the EVProb command redirect the EVProb syntax to a text file. For example, on a UNIX system:

```
echo "EVProb -h $GATEWAY  -n $MANAGEDOBJECT $UNIQUE_ID -a
CALL_CODE:$CALL_CODE;${SEV}SYSTEM:$SYSTEM;COMPONENT:$COMPONENT;ITEM:$ITEM;INVE
NTORY_ID:$INVENTORY_ID;${MOD}DESCRIPTION:$DESCRIPTION" > /tmp/cmd.out
```

This statement must be on a single line.

The file /tmp/cmd.out will contain the EVProb command and parameters invoked to submit a new call. Looking at the /tmp/cmd.out file you can check the command syntax.

You may also invoke the EVProb command in /tmp/cmd.out. On a UNIX system you can do it this using the command:

```
ksh -x /tmp/cmd.out
```

Use the -x parameter to debug the calling of /tmp/cmd.out.

In this way you can directly call the EVProb command as the TroubleTicket.sh script does and you have more control on what is happening.

8.6 ExpertAdvisor and Sybase

Make sure that the Sybase ADVISOR database is configured with the option trunc log on chkpt.

If this option is not set, then the ExpertAdvisor starts up slowly. You can select Work with problems and you are presented the list of the problems, but as soon as you select one for Resume or View the ExpertAdvisor cannot display the next mask and will hang. There will be no error message display.

To check if this option is turned on, go to isql with dbo permissions (sa) and issue the following commands.

```
root@rs600024:/Tivoli/bin/aix4-r1/TME/TEC/MURKEL_rb/TEC_RULES >: isql -Usa
Password:
1> sp_helpdb ADVISOR
2> go
 name                        db_size       owner                    dbid
         created
         status

 ------------------------ ------------- ------------------------ ------
        --------------
        --------------------------------------------------------------------
---------------------------------
 ADVISOR                     156.0 MB EXAV                          6
         Jul 15, 1998
         trunc log on chkpt


 device_fragments                size          usage                free kbyte
s
 ------------------------------ ------------- -------------------- ----------
-
 ADVISOR1_DATA                   98.0 MB       data only                  6192
0
 ADVISOR1_LOG                    20.0 MB       log only                   1995
2
 ADVISOR1_NDX                    39.0 MB       data only                  3993
6

(return status = 0)
1>
```

Figure 386. Checking the Options for the ADVISOR Database

Should the option not be set after the creation of the database, use isql with dbo permissions (sa) and issue the following commands:

```
root@rs600024:/Tivoli/bin/aix4-r1/TME/TEC/MURKEL_rb/TEC_RULES >: isql -Usa
Password:
1> sp_dboption ADVISOR, 'trunc log on chkpt',TRUE
2> go
Database option 'trunc log on chkpt' turned ON for database 'ADVISOR'.
Run the CHECKPOINT command in the database that was changed.
(return status = 0)
1> use ADVISOR
2> go
1> checkpoint
2> go
1>
```

Figure 387. Setting Sybase Database Options

8.6.1 Client Connectivity with the Sybase Server

There are some occurrences where we could not connect to the server from the SQL client. The initial test is to run the SYBPING command on the client as shown in Figure 388 on page 344.

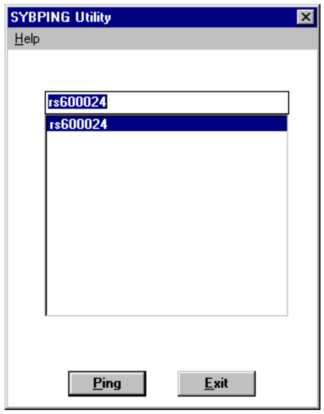


Figure 388. SYBPING Command

The output is shown in Figure 389 on page 344.

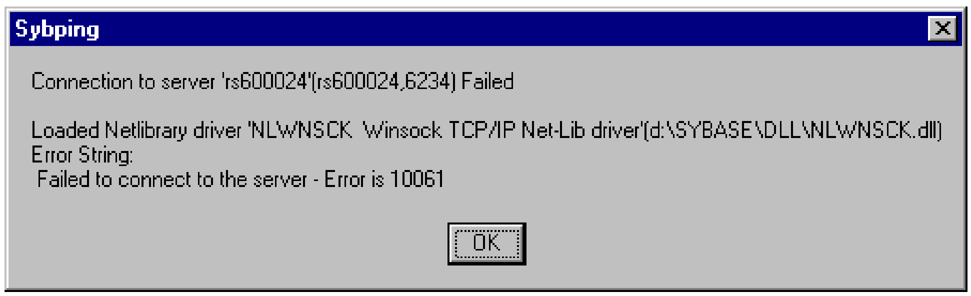


Figure 389. Failed Connection to The Server

Also we encountered this error on the workstation. This was due to the sybase server not running.

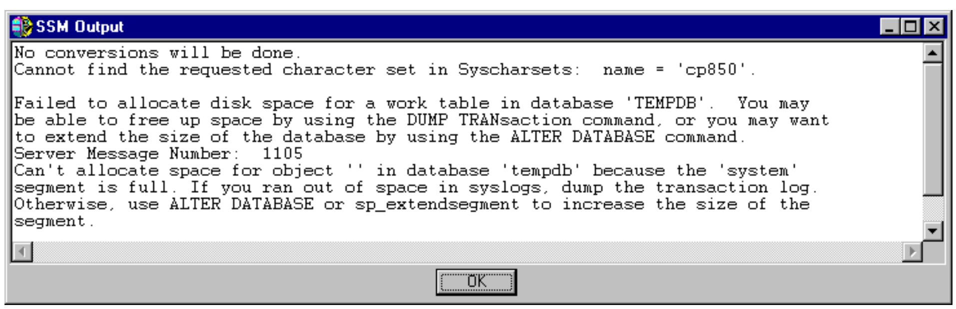


Figure 390. Problem with Database Size

We resolved this problem by issuing the sql command:

```
dump tran tempdb with no_log
```

Appendix A. Scripts and Rulesets

This appendix contains the scripts developed during the project.

A.1 The TroubleTicket.sh Script

The TroubleTicket.sh script is used to open problem records in the Advisor database from the TEC events. This script has been modified from the existing script delivered with the Tivoli Plus module for Advisor.

This script is located in the directory $BINDIR/TME/TEC/scripts. There is also a symbolic link to the TEC install directory. The script is called using the TEC rule template exec_program.

The script calls the EVProb command to submit a call to Advisor and then use the wsetemsg command to set the event status to ACK and to set the slot pms_triggered=1. This slot is used by the TEC rules to know that an event has already generated an Advisor call.

A.1.1 Command Syntax

The TroubleTicket.sh can be invoked with no parameters. In this case the script will read the event slots values using the environment variables. This is the case of invoking the scripts from the Tivoli Console, choosing **File -> Trouble Ticket** from the TEC console window.

The script can also be invoked passing event slots as command line parameters. In this case the parameters to be passed are positional and must be terminated by a colon "**:**" with the exclusion of MSG. This must be the last parameter so that colons in the message do not cause parsing problems. Having the parameter terminated by colon is necessary to avoid parameter position shift when a parameter is null.

The parameters have to be passed in the following order:

1. MANAGED_OBJECT
2. EV_ID
3. CLASS_NAME
4. SOURCE
5. SUB_SOURCE
6. ORIGIN
7. SEVERITY
8. HOSTNAME
9. STATUS
10. MSG

These are the parameters the script currently supports. To add more parameters, the script has to be changed to parse these new parameters.

A.1.2 The Script Source

In the scripts source there are comments and examples guiding you to add your own specific code for your specific needs.

```
#!/bin/sh
#
# Product:SAEXPERTISEPlus for Tivoli
#
# Description:Main Module Install script
#
# (C) COPYRIGHT Tivoli Systems, Inc. 1996, 1997
# All Rights Reserved
# Licensed Material - Property of Tivoli Systems, Inc.
#
# Set the module version number so Link knows who's calling
MODULE_VERSION=2.0;export MODULE_VERSION

#
#       Set failure status: it will be resetted when a call is succesfully submitted
#
FAILURE_STATUS=YES

#
#       The standard directory and the /Plus override script
#
ETC=/etc/Tivoli
SCRIPT=PLUSsetup_env.sh

#
#       If this is NT, then etc is in a different location
#
if [ x\"$OS\" = x\"Windows_NT\" ] ; then
     ETC=$SystemRoot/system32/drivers/etc/Tivoli
fi

#
#       Look for our override script, if not there, then use the
#               standard one.
#
if [ ! -f $ETC/$SCRIPT ]; then
        SCRIPT=setup_env.sh
fi

. $ETC/$SCRIPT


#
#       Set up some popular environment variables.
```

Figure 391. TroubleTicket Script (1 of 9)

```
#

OSERV=‘objcall 0.0.0 get_oserv‘
export OSERV
INST_DIR=‘objcall $OSERV query install_dir‘
INST_DIR=‘echo $INST_DIR | tr "\\\\\\\\" "/"‘
export INST_DIR
REGION=‘objcall $OSERV query whoami | awk ’{print $1}’‘

LIBRARY=‘wlookup Library‘

#
#       Get the region name to avoid collisions
#
IRO=‘wlookup InterRegion‘
IRONAME=‘idlattr -t -g $IRO name string‘
IRONAME=‘eval echo $IRONAME‘

DEFSEN=\"TivoliSentryDefaults#$IRONAME\"

MNOID=‘objcall 0.0.0 get_host_location‘
eval MNNAME=‘idlattr -t -g $MNOID label string‘
export MNNAME

PLUSDIR=$INST_DIR/generic_unix/TME/PLUS
LINKDIR=$PLUSDIR/LINK
PRODDIR=$PLUSDIR/SAEXPERTISE
PATH=$LINKDIR:$PRODDIR:$PATH
export PATH

cd $PRODDIR

. ./PLUSproduct-info.sh

#
#       Debugging code: by creating a directory called .expertiseplusdebug in
#               the appropriate TEMP directory, the output of scripts is
#               redirected to a file in TEMP by the same name as the script.
#
TEMP=/tmp
if [ x"$OS" = x"Windows_NT" ]; then
        TEMP=$DBDIR/tmp
else
        TEMP=/tmp
fi
```

Figure 392. TroubleTicket Script (2 of 9)

```
#
DEBUGDIR=$TEMP/.expertiseplusdebug

SCRIPTNAME=‘echo $0 | tr "\\\\\\\\" "/"‘
SCRIPTNAME=‘basename $SCRIPTNAME‘

[ -d "$DEBUGDIR" ] && exec > $DEBUGDIR/$SCRIPTNAME && set -xv
[ -f "$DEBUGDIR" ] && set -xv

#!/bin/sh
#
# Product:SAEXPERTISEPlus for Tivoli
#
# Description:Install Options retrieval
#
# (C) COPYRIGHT Tivoli Systems, Inc. 1996, 1997
# All Rights Reserved
# Licensed Material - Property of Tivoli Systems, Inc.
#

#
#       If there are install options, bring them in!
#
if [ -n "$INSTALL_OPTS" ]; then
       . $LINKDIR/PLUSget_install_opts.sh
fi

################################################################################
################################################################################
#
# Insert your application specific code here
#
################################################################################
################################################################################
ERRORFILE=/tmp/sa_problem_ticket.out
errmsg() {
    errorhandler "$0" "$*"
}



if [ ! -d /tmp ]
   then mkdir /tmp
fi
```

Figure 393. TroubleTicket Script (3 of 9)

```
WINFILE=/tmp/PLUS-open-trouble-ticket$$.cmd
OUTFILE=/tmp/PLUS-open-trouble-ticket$$.out
SEVFILE=$PRODDIR/severity.info
ENV_FILE=$PRODDIR/ev_gateway.info


if [ -f "$ENV_FILE" ] ; then
   . $ENV_FILE
else
   errmsg "ERROR: $ENV_FILE Does not exist, Please run the \"Configure ExpertView\" Task to create
this file" DATE
   exit 1
fi

###################################################################
# Get Variables
###################################################################

if [ $# -gt 1 ] ; then
        # There are some arguments being passed
        if [ $# -ne 10 ] ; then
                #Wrong number of arguments
                echo $* > $TEMP/sa_ticket.parms
                $PRODDIR/errorhandler "$0" "Wrong number of arguments [$#] to $0
 "
                exit 1
        else
                MANAGEDOBJECT=‘echo $1 | sed -e ’s/://g’‘
                EV_KEY=‘echo $2 | sed -e ’s/://g’‘
                CLASS_NAME=‘echo $3 | sed -e ’s/://g’‘
                SOURCE=‘echo $4 | sed -e ’s/://g’‘
                SUB_SOURCE=‘echo $5 | sed -e ’s/://g’‘
                ORIGIN=‘echo $6 | sed -e ’s/://g’‘
                SEVERITY=‘echo $7 | sed -e ’s/://g’‘
                HOSTNAME=‘echo $8 | sed -e ’s/://g’‘
                STATUS=‘echo $9 | sed -e ’s/://g’‘
                MSG=${10}   # all the rest will be MSG, put MSG at the end
               # Submission by TEC rule
        fi
fi

if [ -n "$SOURCE" ] ; then
```

Figure 394. TroubleTicket Script (4 of 9)

```
CLASS=$CLASS_NAME
   SOURCE=$SOURCE
   ORIGIN=$ORIGIN
   STATUS=$STATUS

else
   errmsg "SOURCE Variable does not exist"
fi

######################################################################
######################################################################
#    Do the mapping
######################################################################
######################################################################

######################################################################
#  Event ID --> UNIQUE_ID (Advisor extern-prob-id)
######################################################################
UNIQUE_ID=$EV_KEY
if [ -n "$UNIQUE_ID" ] ; then
   UNIQUE_ID_DISP=$UNIQUE_ID
   UNIQUE_ID="-x $UNIQUE_ID"
fi
######################################################################
# Define MANAGED_OBJECT
######################################################################
if [ -z "$MANAGEDOBJECT" ] ; then
   MANAGEDOBJECT=$EV_KEY
fi

######################################################################
#  Severity --> SEVERITY map TEC severity to Advisor severity
######################################################################
SEVERITY=$SEVERITY
if [ -n "$SEVERITY" ]; then
    if [ -f "$SEVFILE" ] ; then
       SEVERITY=‘grep -i $SEVERITY $SEVFILE | cut -d"=" -f2‘
       if [ -n "$SEVERITY" ]; then
          SEV=SEVERITY:$SEVERITY\;  # Ready for EVProb command
       else
          SEV=""
          errmsg "ERROR: Severity is not Valid, letting gateway define severity"
       fi
    else
```

Figure 395. TroubleTicket Script (5 of 9)

```
SEV=""
      errmsg "ERROR: Severity File does not exist, letting gateway define severity"
    fi
fi

######################################################################
#  Source --> CALL_CODE
######################################################################
CALL_CODE=$SOURCE
######################################################################
#  Source --> SYSTEM  remap Event source to Operating System Platform
######################################################################
#  !!!!!!!!!!!! Put here  your platform mapping !!!!!!!!!!!!!!!!!!!
######################################################################
case "$SOURCE" in
    LOGFILE)      SYSTEM=Unix ;;
    NV6K)      SYSTEM=Network ;;
    NV390*)        SYSTEM=OS390 ;;
    NT)        SYSTEM=NT ;;
    NT*)        SYSTEM=NT ;;
    SENTRY)
                      ##############################################
                      # !!!!! Customize here your SENTRY source !!!
                      ##############################################
                      case "$CLASS" in
                             NT*)    SYSTEM=NT ;;
                             nt*)    SYSTEM=NT ;;
                             *)      SYSTEM=Unknown ;;
                      esac
                      ;;
    *)                  SYSTEM=$SOURCE ;;
esac

######################################################################
#  Sub_source/Source --> COMPONENT
#  if SUB_SOURCE is  NULL  just COMPONENT = SOURCE
######################################################################
if [ -n "$SUB_SOURCE" ] ; then
   COMPONENT=$SUB_SOURCE
else
   COMPONENT=$SOURCE
fi

######################################################################
```

Figure 396. TroubleTicket Script (6 of 9)

```
#  Class Name  --> ITEM
######################################################################
ITEM=$CLASS_NAME
#cut to 15 chracters. Default Advisor application supports 15 char
ITEM=‘echo $ITEM | dd ibs=15 count=1 2>/dev/null‘

######################################################################
#  Hostname  --> Module
# !!!!!!!!!!!!!!  Just remove the comments if you want to map this
######################################################################
#MODULE=$HOSTNAME
#MOD=MODULE:$MODULE\;

######################################################################
#  From hostname get object_id  --> InventoryID
######################################################################
INVENTORY_ID=‘wlookup -ar ManagedNode | grep $HOSTNAME | cut -f 2‘
if [ ! -n "$INVENTORY_ID" ] ; then
    =‘wlookup -ar Endpoint | grep $HOSTNAME | cut -f 2‘
fi
INVENTORY_ID=‘echo $INVENTORY_ID | dd ibs=40 count=1 2>/dev/null‘


#!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!#
#       ADD here your specific slots to map     #
#!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!#
#-->


######################################################################
#  Make Advisor DESCRIPTION
######################################################################
MSG=$MSG
if [ -z "$MSG" ] ; then
       MSG="$CLASS_NAME"
fi
SHORT_MSG=‘echo $MSG | dd ibs=127 count=1 2>/dev/null‘
DESCRIPTION="\’T/EC event info = STATUS=$STATUS HOSTNAME=$HOSTNAME CLASS=$CLASS ORIGIN=$ORIGIN
MSG=$SHORT_MSG\’"


EVPATH=‘echo "$EVBASE;" | sed ’s/\//\\\/g’‘
PATH=$PATH:$EVPATH

. ./ExpertiseEnv.sh
```

Figure 397. TroubleTicket Script (7 of 9)

```
#Check for ev_gateway.info file
GWFcreated='ls -R $PRODDIR | grep -i "ev_gateway.info"'
if [ -z "$GWFcreated" ]; then
  errmsg "ERROR: \"ev_gateway.info\" Not Found. Please run the \"Configure ExpertView\" Task to
create this file"
  exit 1
fi


#Check for EVProb existance
EVProbinstalled='ls -R $EVBASE | grep -i "EVProb"'
if [ -z "$EVProbinstalled" ]; then
  errmsg "ERROR: \"EVProb\" Not Found."
  errmsg "ExpertView NSM_CMDS not installed on this Node"
  exit 1
fi

######################################################################
# Run EVProb Command
######################################################################

#!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!#
# Note: SEV and MODE already has the ';' at the end and are           #
#       in EVProb format. This just because the could be NOT present  #
#!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!#

if [ x\"$OS\" = x\"Windows_NT\" ] ; then
    echo "EVProb -h $GATEWAY  -n $MANAGEDOBJECT $UNIQUE_ID -a
\"CALL_CODE:$CALL_CODE;${SEV}SYSTEM:$SYSTEM;COMPONENT:$COMPONENT;ITEM:$ITEM;INVENTORY_ID:$INVENTO
RY_ID;${MOD}DESCRIPTION:$DESCRIPTION\""  >> $WINFILE
    $WINFILE > /dev/null 2>$OUTFILE
   rc=$?
else
    EVProb -h $GATEWAY  -n $MANAGEDOBJECT $UNIQUE_ID -a
"CALL_CODE:$CALL_CODE;${SEV}SYSTEM:$SYSTEM;COMPONENT:$COMPONENT;ITEM:$ITEM;INVENTORY_ID:$INVENTOR
Y_ID;${MOD}DESCRIPTION:$DESCRIPTION"  > /tmp/EVProb.out 2>$OUTFILE
   rc=$?
fi

if [ $rc -ne 0 ] ; then
   errmsg "ERROR: Failed to open a trouble ticket for managed object=$MANAGEDOBJECT, External
ID=$UNIQUE_ID_DISP, RC=$rc" >> $OUTFILE
else
```

Figure 398. TroubleTicket Script (8 of 9)

```
EA_PROB_ID=‘grep -i "ID" $OUTFILE | cut -d"=" -f2‘
   errmsg "Trouble Ticket Number = $EA_PROB_ID, ${EAACTION}ED for managed object=$MANAGEDOBJECT,
External ID = $UNIQUE_ID_DISP"
   FAILURE_STATUS=NO
fi
rm $OUTFILE
######################################################################
# If Ticket created OK, set TEC Event status to ACK
######################################################################
#!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
# Specify your console name here
#!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
# Let’a look for the Console name
if [ $rc -eq 0 ] ; then
        if [ x"$STATUS" = x"OPEN" ] ; then
                ALI=‘wlookup TME_server‘
                PRINCIPAL=‘objcall 0.0.0 o_get_principal‘
                ADMIN=‘objcall $ALI get_principal_admin "$PRINCIPAL"‘
                #We have to find the event console to use

                for ec in ‘wlookup -r EnterpriseClient -a | awk -F\\t ’{print $2}’‘
                do
                        FOUND_IT=‘idlattr -t -g $ec collections SysAdmin::CollectionList | grep
"$ADMIN"‘
                        if [ ! -z "$FOUND_IT" ] ; then
                                eval EC=‘idlattr -t -g $ec label string‘
                                break
                        fi
                done
                if [ -z "$EC" ] ; then
                        echo "Unable to find an Event Console for Administrator ’$PRINCIPAL’" >& 2
                        exit 1
                fi
                wsetemsg -t ACK pms_triggered=1 "@$EC" $EV_KEY
        fi
fi

if [ x"$FAILURE_STATUS" = x"YES" ] ; then
        wpostemsg -r FATAL -m "TT subm. failed!  Check SA Connection!"   SA_Base TEC
fi
#rm $WINFILE
exit rc$
```

Figure 399. TroubleTicket Script (9 of 9)

8.7 The CloseTivoliTicket.sh Script

The following script is used to close a TEC event when an Advisor problem is closed. It has been extracted by the Tivoli Plus module for Advisor and some customization has been performed.

This script is located under the directory $BINDIR/scripts.

The syntax of this script is:

```
CloseTivoliTicket.sh <event_id> <status>
```

where:

<event_id> is the ID of the event you want to update and <status> is the event status you want to set. The source code is shown in Figure 400 on page 356.

A.1.3 Script Source

```
!/bin/sh
# Component Name: SAEXPERTISEPlus for Tivoli
# $Revision: 1.5 $
# Description:
# (C) COPYRIGHT Tivoli Systems, Inc. 1995
# Unpublished Work
# All Rights Reserved
# Licensed Material - Property of Tivoli Systems, Inc.
# Script: CloseTivoliTicket
#         This script closes a T/EC event when a SA-Expert Advisor Ticket is
#        closed.
#
# Arguments passed in from Software Artistry:
#      $1 == T/EC Event ID
#      $2 == new status

PATH=/bin:/usr/bin:/usr/ucb:$PATH
export PATH

. /etc/Tivoli/setup_env.sh
unset CHILD_OF_OSERV
ALI=‘wlookup TME_server‘
PRINCIPAL=‘objcall 0.0.0 o_get_principal‘
ADMIN=‘objcall $ALI get_principal_admin "$PRINCIPAL"‘
#We have to find the event console to use
for ec in ‘wlookup -r EnterpriseClient -a | awk -F\\t ’{print $2}’‘
do
        FOUND_IT=‘idlattr -t -g $ec collections SysAdmin::CollectionList | grep "$ADMIN"‘
        if [ ! -z "$FOUND_IT" ] ; then
                eval EC=‘idlattr -t -g $ec label string‘
                break
        fi
done

if [ -z "$EC" ] ; then
        echo "Unable to find an Event Console for Administrator ’$PRINCIPAL’" >&2
        exit 1
fi

if [ x"$2" != x"Acknowledged" ] ; then
        NEWSTATE=CLOSED
else
        NEWSTATE=ACK
fi

wsetemsg -t $NEWSTATE "@$EC" $1
```

Figure 400. ClostTivoliTivket.sh

A.2 Rulesets

This section contains the rules used in the examples contained in this redbook.

A.2.1 Ruleset nt_MemAvailBytes.rls

```
/* Ruleset nt_MemAvailBytes.rls

These rules handle incoming nt_MemAvailBytes under the following aspects:

Rule    'nt_MemAvailBytes_Handling'

 Reception Action 1.  Duplicate Detection
 Reception Action 2.  Is there a recent (1 minute past/future) clear event of
               type nt_MemAvailByte with severity HARMLESS and trigger
               its re-analysis
 Reception Action 3.  Check for a related NT_Low_Virtual_Memory event and trigger
               its re-analysis
 Reception Action 4.  Setting the managed_object slot to a unique ID just in case we need it


Rule    'Multiple_nt_MemAvailBytes_Create_Trouble_Ticket'

 Action 1.Create or update a Trouble Ticket, if  3 or more nt_MemAvailBytes have been received


Change Rule    'Keep_Linked_Event_In_Track_with_The_Cause_Event'

 Action 1.     It keeps the linked NT_Low_Virtual_Memory event in track with the cause
               nt_MemAvailBytes event


Change-Rule    'Sent_Update_to_Trouble_Ticket_if_Event_was_closed_from_within_TEC'

 Action 1.     If event was closed from inside TEC, send an update call to SA

*/

                       ...continued on the next page
```

Figure 401. MwmAvailBytes.rls (Part 1 of 2)

```
change_rule:
'Keep_Linked_Event_In_Track_with_The_Cause_Event':
( event: _event of_class 'nt_MemAvailBytes'
                where [
               severity:                            outside ['HARMLESS'],
               hostname: _hostname2                 equals _hostname,
               administrator: _administrator,
               event_handle: _event_handle2,
               date_reception: _date_reception2
               ],
       slot:status set_to _status,
action:
'Follow_Me...':
( first_instance(event: _linked_event of_class 'NT_Low_Virtual_Memory'
                where [
               hostname:                            equals _hostname2 ,
               cause_date_reception:                equals _date_reception2,
               cause_event_handle:                  equals _event_handle2
               ]),

        set_event_status(_linked_event, _status),
        set_event_administrator(_linked_event,_administrator)
)
).

change_rule:
'Sent_Update_to_Trouble_Ticket_if_Event_was_closed_from_within_TEC':
( event: _event of_class 'nt_MemAvailBytes'
                where [
               pms_triggered: _pms_triggered        equals 1,
               status: _status2                     outside ['CLOSED'],
               severity: _severity2                 outside ['HARMLESS'],
               hostname: _hostname2                 equals _hostname,
               date: _date,
               event_handle: _event_handle2,
               server_handle: _server_handle2,
               date_reception: _date_reception2,
               managed_object: _managed_object2,
               msg: _msg2,
               source: _source2,
               sub_source: _sub_source2,
               origin: _origin2
               ],
       slot:status set_to _status equals 'CLOSED',

action:
'Send_relief_update...':
(
       exec_program(_event,'scripts/TroubleTicket.sh','"%ld" "%ld%ld%ld:" "%s:" "%s:" "%s:" "%s:"
       "%s:" "%s:" "%s:" "%s"',
       [_managed_object2,_event_handle2,_server_handle2,_date_reception2,'nt_MemAvailBytes',_sour
       ce2,_sub_source2,_origin2,
       _severity2,_hostname2,'CLOSED',_msg2],'YES')
)
).
```

Figure 402. nt_MemAvailBytes.rls (Part 2 of 2)

A.2.2 Ruleset NT_Low_Virtual_Memory.rls

```
/* Ruleset NT_Low_Virtual_Memory.rls
The ruleset handles incoming NT_Low_Virtual_Memory_Events under the following aspects:
Rule
Reception Action 1.   Duplicate Detection
 Reception Action 2.  Is there a recent (1 minute past/future) clear event of type nt_MemAvailByte
with severity HARMLESS?
 Reception Action 3.  Setting the managed_object slot to a unique ID just in case we need it
       Action 4.Link the NT_Low_Virtual_Memory event to a possibly available nt_MemAvailByte event

Change Rules
1st change_rule:It updates the severity of an related nt_MemAvailBytes event to CRITICAL
               after having received 2 NT_Low_Virtual_Memory events

 2nd change_rule:In case there is only 1 nt_MemAvailBytes event, but 3 NT_Low_Virtual_Memory
               events have already been received, it's time to create a TroubleTicket...
               on nt_MemAvailBytes

 3rd change_rule:In case, we see 5 NT_Low_Virtual_Memory events without 1 nt_MemAvailBytes
               just calls for a Trouble Ticket...
               on NT_Low_Virtual_Memory
*/
rule:
'NT_Low_Virtual_Memory_Handling':
(
   event: _event of_class 'NT_Low_Virtual_Memory'
               where [
                       hostname: _hostname,
                       severity: _severity,
                       event_handle: _event_handle,
                       server_handle: _server_handle,
                       date_reception: _date_reception,
                       date: _date,
                       msg: _msg
                       ],
reception_action:
'Check_For_NT_Low_Virtual_Memory_Duplicate_Events':
  (
       first_duplicate(_event,
                       event: _dup_ev
                       where [
                               status: outside ['CLOSED']
                               ]),

       add_to_repeat_count(_dup_ev,1),
       place_change_request(_dup_ev,last_occur,_date),
       drop_received_event,
       commit_action
    ),
```

Figure 403. NT_Low_Virtual_Memory.rls (Part 1 of 4)

```
reception_action:
'Check_For_NT_Low_Virtual_Memory_Duplicate_Events':
  (
       first_duplicate(_event,
                      event: _dup_ev
                      where [
                             status: outside ['CLOSED']
                             ]),

       add_to_repeat_count(_dup_ev,1),
       place_change_request(_dup_ev,last_occur,_date),
       drop_received_event,
       commit_action
    ),

reception_action:
'Check_For_A_Recent_nt_MemAvailBytes_Clear_Event':
  (
       first_instance(event: _clear_ev of_class 'nt_MemAvailBytes'
              where [
                      hostname:             equals _hostname,
                      severity:      equals 'HARMLESS',
                      status:        outside ['CLOSED']
                     ],

              _event - 60 - 60),
       redo_analysis(_clear_ev)
  ),

reception_action:
'Setting_The_Managed_Object_Slot_Of_The_Incoming_NT_Low_Virtual_Memory_Event':
  (
   atomconcat([_event_handle,_server_handle],_temp_message),
   atomconcat([_temp_message,_date_reception],_final_message),
   bo_set_slotval(_event,managed_object,_final_message)
  ),

action:
'NT_Low_Virtual_Memory_Link_Effect_To_Cause_nt_MemAvailBytes':
  (
       first_instance(event: _cause_event of_class 'nt_MemAvailBytes'
              where [
                      hostname:             equals       _hostname,
                      severity:      outside ['HARMLESS'],
                      status: _status       outside['CLOSED'],
                      managed_object: _managed_object
                     ]),

       link_effect_to_cause(_event,_cause_event),
       place_change_request(_event,managed_object,_managed_object),
       set_event_status(_event,_status)
  )
).
```

Figure 404. NT_Low_Virtual_Memory.rls (Part 2 of 4)

```
change_rule:
'Update_Severity_of_nt_MemAvailBytes_when_2_NT_Low_Virtual_Memory':
(
   event: _event of_class 'NT_Low_Virtual_Memory'
              where [
                      repeat_count: _repeat_count greater_or_equals1,
                      status:              outside       ['CLOSED','ACK'],
                      hostname: _hostname,
                      date: _date,
                      msg: _msg
                      ],
              slot: last_occur set_to _new_last_occur,

action:
  'Set_Event_Severity_of_nt_MemAvailBytes_to_CRITICAL_after_2nd_Low_Virtual_Memory':

 (
        first_instance(event: _first_instance_event of_class 'nt_MemAvailBytes'
              where   [
                      status:       outside        ['CLOSED'],
                      severity:     equals         'WARNING',
                      hostname:            equals         _hostname
                      ]),
       set_event_severity(_first_instance_event,'CRITICAL')
 )
).
change_rule:
'3rd_NT_Low_Virtual_Memory_Events_and_at_least_1_nt_MemAvailBytes_is_enough':
(
   event: _event of_class 'NT_Low_Virtual_Memory'
              where [
                      repeat_count: _repeat_count greater_or_equals2,
                      status:              outside        ['CLOSED','ACK'],
                      hostname: _hostname,
                      sub_source: _sub_source,
                      date: _date,
                      msg: _msg
                      ],
              slot: last_occur set_to _new_last_occur,
action:
  'Create_Trouble_Ticket_with_nt_MemAvailBytes':
 ( first_instance(event: _first_instance_of_event of_class 'nt_MemAvailBytes'
              where   [
                      status: _status2            outside        ['CLOSED'],
                      severity:  _severity2              equals         'CRITICAL',
                      hostname:       _hostname2    equals          _hostname,
                      event_handle: _event_handle2,
                      server_handle: _server_handle2,
                      date_reception: _date_reception2,
                      managed_object: _managed_object2,
                      msg: _msg2,
                      source: _source2,
                      sub_source: _sub_source2,
                      origin: _origin2
                                ]),
```

Figure 405. NT_Low_Virtual_Memory.rls (Part 3 of 4)

```
exec_program(_first_instance_of_event,'scripts/TroubleTicket.sh','"%ld" "%ld%ld%ld:" "%s:" "%s:"
"%s:" "%s:" "%s:" "%s:" "%s:" "%s"',
[_managed_object2,_event_handle2,_server_handle2,_date_reception2,'nt_MemAvailBytes',_source2,_su
b_source2,_origin2,_severity2,
_hostname2,_status2,_msg2],'YES')
 )
).

change_rule:
'5th_NT_Low_Virtual_Memory_Event_creates_Trouble_Ticket_if_not_acknowledged_before':
(
   event: _event of_class 'NT_Low_Virtual_Memory'
              where [
                     repeat_count: _repeat_count   greater_than   4,
                     status: _status2              outside        ['CLOSED','ACK'],
                     severity: _severity2          outside        ['HARMLESS'],
                     hostname: _hostname2          equals         _hostname,
                     date: _date,
                     event_handle: _event_handle2,
                     server_handle: _server_handle2,
                     date_reception: _date_reception2,
                     managed_object: _managed_object2,
                     msg: _msg2,
                     source: _source2,
                     sub_source: _sub_source2,
                     origin: _origin2
                     ],

              slot: last_occur set_to _new_last_occur,

action:
  'Create_Trouble_Ticket_on_NT_Low_Virtual_Memory':

 (
set_event_severity(_event,'CRITICAL'),
exec_program(_event,'scripts/TroubleTicket.sh','"%ld" "%ld%ld%ld:" "%s:" "%s:" "%s:" "%s:" "%s:"
"%s:" "%s:" "%s"',
[_managed_object2,_event_handle2,_server_handle2,_date_reception2,'NT_Low_Virtual_Memory',_source
2,_sub_source2,_origin2,
_severity2,_hostname2,_status2,_msg2],'YES')
```

Figure 406. NT_Low_Virtual_Memory.rls (Part 4 of 4)

A.2.3 Ruleset nt_MemAvailBytes_clear.rls

```
/* Ruleset nt_MemAvailBytes_clear.rls

Clear nt_MemAvailBytes and linked Events

*/

rule:
'nt_MemAvailBytes_And_Linked_Clear':
(
   event: _event of_class 'nt_MemAvailBytes'
            where [
                   severity:                    equals 'HARMLESS',
                   hostname: _hostname,
                   date: _date,
                   msg: _msg1
                   ],

action :
'nt_MemAvailBytes_Close':
 (
    all_instances(event: _to_be_cleared_cause_event of_class _class within
['nt_MemAvailBytes','NT_Low_Virtual_Memory']
       where [
            status:                              outside ['CLOSED'],
            severity:                            outside ['HARMLESS'] ,
            hostname:                                   equals _hostname,
            date_reception: _date_hd ,
            event_handle: _handle_hd ,
            msg: _msg2
                                   ] ),

       atomconcat(['Problem resolved: ',_msg1,'[ Original Message was: (',_msg2,')]'],_msg),
       bo_set_slotval(_to_be_cleared_cause_event,msg,_msg),
       change_event_status(_to_be_cleared_cause_event, 'CLOSED'),
       set_event_administrator(_to_be_cleared_cause_event,'SMEE again...TEC'),
       drop_received_event

  )
).
```

Figure 407. Ruleset nt_MemAvailBytes_clear.rls

A.2.4 Utility Script

```
#!/bin/sh
# This script is just for rule-development purposes...nothing fancy, but useful...
# The only parameter expected is the correct name of the Rulebase

if [ $# -ne 1 ] ; then
   #Wrong number of arguments
   echo "Wrong number of arguments!"
   echo "The only parameter expected is the correct name of the Rulebase."
   exit 1
fi

echo "Compiling Rulebase "$1" with tracing option..."

# Clear the error & the trace file first...
cat /dev/null > /tmp/compile.err
cat /dev/null > /tmp/rules.trace

wcomprules -t $1 2>/tmp/compile.err

if [ $? -ne 0 ] ; then
   dtterm -e vi /tmp/compile.err 1>/dev/null 2>/dev/null
   echo "Compile was unsuccessful..."
   exit
else
 echo "Compiled successfully..."
fi

echo "Load Rulebase MURKEL immediately"
wloadrb -u MURKEL

if [ $? -ne 0 ] ; then
   echo "Load was unsuccessful..."
   exit
else
 echo "Loaded successfully..."
fi

while true; do
echo "Do youwant to restart the Event Server? (y/n)"
   read answer
   echo
  if [ -z "$answer" ]; then
       echo "Error: a response is required."
       echo
       continue
    fi
  if [ $answer = "y" -o $answer = "Y" ]; then
       echo "Stop EventServer"
       wstopesvr
        if [ $? -ne 0 ] ; then
               echo "Event Server NOT running..."
               exit
        else
               echo "Event Server stopped..."
        fi
```

```
   echo "Start Event Server"
        sleep 2
        wstartesvr

        if [ $? -ne 0 ] ; then
                dtterm -e vi /tmp/tec_master 1>/dev/null 2>/dev/null
                exit
        else
                echo
                echo "###"
                echo "Rulebase" $1 "compiled, loaded and ready for testing..."
                echo "###"
                echo
                exit
        fi

     elif [ $answer = "n" -o $answer = "N" ]; then break

     else
        echo "Please answer with y or n!"
         echo
        continue
     fi

 done

        echo "If you made changes to any baroc file, you will have to start the Event Server before
they become activ!"
```

Figure 408. Utility Script for Rule Development

Appendix B. Special Notices

This publication is intended to help services professionals to implement the Tivoli Service Desk suite of applications. The information in this publication is not intended as the specification of any programming interfaces that are provided by the Tivoli applications. See the PUBLICATIONS section of the IBM Programming Announcement for the Tivoli Service Desk suite for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter 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 the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, NY 10594 USA.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

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

AIX	IBM
NetView	

The following terms are trademarks of other companies:

Tivoli Systems Incorporated, TME 10, Tivoli Management Environment, and Tivoli Enterprise Console are trademarks of Tivoli Systems, an IBM Company

Sybase is a trademark of Sybase Inc.

Oracle is a trademark of Oracle Inc.

Tivoli Service Desk is a trademark of Software Artistry, a Division of Tivoli.

Microsoft, Windows, Windows 95 and Windows NT are trademarks of Microsoft Corp.

Crystal Reports is a trademark of Seagate Software, Inc.

Knowledge-Pak is a trademark of ServiceWare Inc.

C-bus is a trademark of Corollary, Inc.

Java and HotJava are trademarks of Sun Microsystems, Incorporated.

Microsoft, Windows, Windows NT, and the Windows 95 logo are trademarks or registered trademarks of Microsoft Corporation.

PC Direct is a trademark of Ziff Communications Company and is used by IBM Corporation under license.

Pentium, MMX, ProShare, LANDesk, and ActionMedia are trademarks or registered trademarks of Intel Corporation in the U.S. and other countries.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

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

Appendix C. Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

C.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see “How to Get ITSO Redbooks” on page 371.

- *TEC Implementation Examples*, SG24-5216
- *Management Examples Using NetView Version 5.1*, SG24-5285(available soon)
- *TME 10 Inventory 3.2: New Features and Database Support*, SG24-2135
- *TME 10 Deployment Cookbook: Inventory and Company*, SG24-2120
- *Creating Custom Monitors for Tivoli Distributed Monitoring*, SG24-5211
- *Migrating from Systems Monitor to TME 10 Distributed Monitoring*, SG24-4936

C.2 Redbooks on CD-ROMs

Redbooks are also available on CD-ROMs. **Order a subscription** and receive updates 2-4 times a year at significant savings.

CD-ROM Title	Subscription Number	Collection Kit Number
System/390 Redbooks Collection	SBOF-7201	SK2T-2177
Networking and Systems Management Redbooks Collection	SBOF-7370	SK2T-6022
Transaction Processing and Data Management Redbook	SBOF-7240	SK2T-8038
Lotus Redbooks Collection	SBOF-6899	SK2T-8039
Tivoli Redbooks Collection	SBOF-6898	SK2T-8044
AS/400 Redbooks Collection	SBOF-7270	SK2T-2849
RS/6000 Redbooks Collection (HTML, BkMgr)	SBOF-7230	SK2T-8040
RS/6000 Redbooks Collection (PostScript)	SBOF-7205	SK2T-8041
RS/6000 Redbooks Collection (PDF Format)	SBOF-8700	SK2T-8043
Application Development Redbooks Collection	SBOF-7290	SK2T-8037

C.3 Other Publications

These publications are also relevant as further information sources:

Tivoli Service Desk Manuals:

- *ASE Tools and Utilities Guide,* GC31-5165
- *ESM Installation Guide,* GC31-5167
- *Expert Advisor Fundamentals Guide,* GC31-5172
- *Expert Advisor System Administrator’s Guide,* GC31-5173
- *Expert Advisor Technical Reference Guide,* GC31-5174
- *Expert Advisor User’s Guide*, GC31-5175
- *ExpertView Architecture Overview*, GC31-5176

- *ExpertView Extensions to Expert Advisor,* GC31-5177
- *ExpertView Gateway Administrator's Guide,* GC31-5178
- *ExpertView Network Specialist's Guide*, GC31-5179
- *ExpertView Network/System Administrator's Guide,* GC31-5180
- *ExpertView Technical Reference Guide*, GC31-5181
- *Expert Evolution System Administrator's Guide,* GC31-5188
- *Expert Evolution Technical Reference Guide,* GC31-5189
- *Expert Evolution User's Guide,* GC31-5190
- *Expert Foundation Manager Data Utilities,* GC31-5191
- *Expert Foundation Manager Technical Reference,* GC31-5193
- *Expert Foundation Manager User's Guide,* GC31-5194
- *Expert Foundation Manager System Guide,* GC31-5195
- *Expert Administrator System Administrator's Guide,* GC31-5205
- *Tivoli Inventory Integration User's Guide,* GC32-0288
- *Tivoli Inventory Integration System Administrator's Guide,* GC31-5204
- *SA-Expertise/Plus for Tivoli User's Guide,* GC31-5203

How to Get ITSO Redbooks

This section explains how both customers and IBM employees can find out about ITSO redbooks, CD-ROMs, workshops, and residencies. A form for ordering books and CD-ROMs is also provided.

This information was current at the time of publication, but is continually subject to change. The latest information may be found at `http://www.redbooks.ibm.com/`.

How IBM Employees Can Get ITSO Redbooks

Employees may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **Redbooks Web Site on the World Wide Web**

  `http://w3.itso.ibm.com/`

- **PUBORDER** – to order hardcopies in the United States
- **Tools Disks**

  To get LIST3820s of redbooks, type one of the following commands:

  ```
  TOOLCAT REDPRINT
  TOOLS SENDTO EHONE4 TOOLS2 REDPRINT GET SG24xxxx PACKAGE
  TOOLS SENDTO CANVM2 TOOLS REDPRINT GET SG24xxxx PACKAGE (Canadian users only)
  ```

  To get BookManager BOOKs of redbooks, type the following command:

  ```
  TOOLCAT REDBOOKS
  ```

  To get lists of redbooks, type the following command:

  ```
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET ITSOCAT TXT
  ```

  To register for information on workshops, residencies, and redbooks, type the following command:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ITSOREGI 1998
  ```

- **REDBOOKS Category on INEWS**
- **Online** – send orders to: USIB6FPL at IBMMAIL or DKIBMBSH at IBMMAIL

Redpieces

For information so current it is still in the process of being written, look at "Redpieces" on the Redbooks Web Site (`http://www.redbooks.ibm.com/redpieces.html`). Redpieces are redbooks in progress; not all redbooks become redpieces, and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

How Customers Can Get ITSO Redbooks

Customers may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **Online Orders** – send orders to:

	IBMMAIL	**Internet**
In United States	usib6fpl at ibmmail	usib6fpl@ibmmail.com
In Canada	caibmbkz at ibmmail	lmannix@vnet.ibm.com
Outside North America	dkibmbsh at ibmmail	bookshop@dk.ibm.com

- **Telephone Orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	(long distance charges apply)
(+45) 4810-1320 - Danish	(+45) 4810-1020 - German
(+45) 4810-1420 - Dutch	(+45) 4810-1620 - Italian
(+45) 4810-1540 - English	(+45) 4810-1270 - Norwegian
(+45) 4810-1670 - Finnish	(+45) 4810-1120 - Spanish
(+45) 4810-1220 - French	(+45) 4810-1170 - Swedish

- **Mail Orders** – send orders to:

IBM Publications	IBM Publications	IBM Direct Services
Publications Customer Support	144-4th Avenue, S.W.	Sortemosevej 21
P.O. Box 29570	Calgary, Alberta T2P 3N5	DK-3450 Allerød
Raleigh, NC 27626-0570	Canada	Denmark
USA		

- **Fax** – send orders to:

United States (toll free)	1-800-445-9269
Canada	1-800-267-4455
Outside North America	(+45) 48 14 2207 (long distance charge)

- **1-800-IBM-4FAX (United States)** or **(+1) 408 256 5422 (Outside USA)** – ask for:

  Index # 4421 Abstracts of new redbooks
  Index # 4422 IBM redbooks
  Index # 4420 Redbooks for last six months

- **On the World Wide Web**

Redbooks Web Site	http://www.redbooks.ibm.com
IBM Direct Publications Catalog	http://www.elink.ibmlink.ibm.com/pbl/pbl

Redpieces

For information so current it is still in the process of being written, look at "Redpieces" on the Redbooks Web Site (`http://www.redbooks.ibm.com/redpieces.html`). Redpieces are redbooks in progress; not all redbooks become redpieces, and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

IBM Redbook Order Form

Please send me the following:

Title	**Order Number**	**Quantity**

First name Last name

Company

Address

City Postal code Country

Telephone number Telefax number VAT number

☐ Invoice to customer number

☐ Credit card number

Credit card expiration date Card issued to Signature

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.

List of Abbreviations

ASE	Application Software Expert
DNS	Domain Name Server
EV	ExpertView
IBM	International Business Machines Corporation
ITSO	International Technical Support Organization
LAN	Local Area Network
NSM	Network System Management
RDBMS	Relational Dababase Management System
SQL	Structured Query Language
TCP/IP	Transmission Control Protocol/Internet Protocol

Index

T

W

ITSO Redbook Evaluation

Problem Management Using Tivoli Service Desk and the TEC
SG24-5301-00

Your feedback is very important to help us maintain the quality of ITSO redbooks. **Please complete this questionnaire and return it using one of the following methods:**

- Use the online evaluation form found at http://www.redbooks.ibm.com
- Fax this form to: USA International Access Code + 1 914 432 8264
- Send your comments in an Internet note to redbook@us.ibm.com

Which of the following best describes you?
_ Customer _ Business Partner _ Solution Developer _ IBM employee
_ None of the above

Please rate your overall satisfaction with this book using the scale:
(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)

Overall Satisfaction __________

Please answer the following questions:

Was this redbook published in time for your needs? Yes___ No___

If no, please explain:

What other redbooks would you like to see published?

Comments/Suggestions: (THANK YOU FOR YOUR FEEDBACK!)

SG24-5301-00

Printed in the U.S.A.

Problem Management Using Tivoli Service Desk and the TEC

SG24-5301-00