

ServiceCenter™
SCAuto for HP OpenView
VantagePoint Operations
Version 1.2

June 2001

SCA-1.1-ENG-01001-00083-06/08/01

Peregrine Systems, Inc.
3611 Valley Centre Drive
San Diego, CA 92130



© June 2001 Peregrine Systems, Inc. or its subsidiaries.

All Rights Reserved.

Information contained in this document is proprietary to Peregrine Systems, Incorporated, and may be used or disclosed only with written permission from Peregrine Systems, Inc. This book, or any part thereof, may not be reproduced without the prior written permission of Peregrine Systems, Inc. This document refers to numerous products by their trade names. In most, if not all, cases these designations are claimed as Trademarks or Registered Trademarks by their respective companies.

Peregrine Systems is a registered trademark and **ServiceCenter** is a trademark of
Peregrine Systems, Inc. or its subsidiaries.

HP OpenView VantagePoint Operations and **HP OpenView IT/Operations** are registered trademarks
of Hewlett-Packard Company.

This document and the related software described in this manual is supplied under license or nondisclosure agreement and may be used or copied only in accordance with the terms of the agreement. The information in this document is subject to change without notice and does not represent a commitment on the part of Peregrine Systems, Inc.

The names of companies and individuals used in the sample database and in examples in the manuals are fictitious and are intended to illustrate the use of the software. Any resemblance to actual companies or individuals, whether past or present, is purely coincidental.

This edition applies to version 2.0 and later of the licensed program of **ServiceCenter**[™]
and version 1.2 of **SCAuto for HP OpenView VantagePoint Operations**

Contents



Chapter 1 Introduction

Overview	1-1
Knowledge Requirements	1-2
Determining Current Product Version	1-2
SCAuto for VP Operations	1-3
ServiceCenter	1-4
Core Applications	1-4
Additional Applications	1-4
VP Operations	1-5
Functional Areas	1-5
Network Node Manager	1-5
About SCAuto for VP Operations	1-6
Operational Concepts	1-8
Bi-Directional Integration	1-8
Planning your ServiceCenter and VP Operations Integration	1-11
Mode 1: Uni-Directional Automatic Notification from VP Operations	1-11
Process and Data Flow	1-12
Mode 2: Bi-Directional Exchange (Default Mode of Operation)	1-13
Process and Data Flow	1-14
Mode 3: Automatic Notification from VP Operations via Trouble Ticket Interface (TTI).....	1-16
Mode 4: New SC Tickets Generate IT/O Events.....	1-17
Mode 5: Application Monitoring — VP Operations Monitors ServiceCenter	1-18
Mode 6: Combined User Interface — Launch ServiceCenter from IT/O Windows	1-18

VP Operations Business Logic Topics.....	1-20
ServiceCenter Business Logic Topics	1-21
SCAuto for VP Operations Business Logic Topics.....	1-23

Chapter 2 Installation

Overview.....	2-1
System Prerequisites.....	2-1
Required Kernel Parameters	2-2
Installation Prerequisites.....	2-2
ServiceCenter Client Installation.....	2-3
Installing SCAuto for VP Operations	2-4
Install Procedure	2-4

Chapter 3 Basic Operations

Starting and Stopping SCAuto for VP Operations Processes	3-1
Starting SCAuto for VP Operations	3-1
Stopping SCAuto for VP Operations.....	3-4
Basic Maintenance	3-5
scito.ini parameters.....	3-5
Basic Configuration	3-6
Troubleshooting	3-6

Chapter 4 Product Architecture

Application Integration	4-1
ServiceCenter Menu Options.....	4-2
Problem List	4-4
Open A Problem	4-5
Update A Problem.....	4-7
Close A Problem	4-9
Probable Cause	4-10

Service Information.....	4-11
Down Time	4-12
Assigned Problems.....	4-13
Other Services	4-14
Message Integration	4-14
Location Information	4-15
Vendor Information	4-16
User Directory.....	4-17
Filtering.....	4-17
Help Desk.....	4-18
Main Menu.....	4-18
Event Integration.....	4-19
Integration Components	4-20
scevmon.....	4-21
sctoito.....	4-22
scfromitoTTI	4-22
scfromitoMSI	4-22
scfromitoMEI	4-22

Chapter 5 Configuration

Introduction	5-1
SCAuto for VP Operations Business Logic Configuration	5-2
Configuration Overview.....	5-2
IT/O Variables	5-3
ServiceCenter TCL Event Object	5-7
Summary of ServiceCenter TCL commands	5-7
create_sc_event	5-7
set_evtype	5-7
set_evfield	5-8
print.....	5-8
send.....	5-8
Static Map File	5-9
IT/O Message Filtering the event.ini File.....	5-10
Sections	5-10
Using TTI - Trouble Ticketing Interface	5-14
Default Behavior.....	5-15
Example eventmapMSI.tcl Script.....	5-15

TCL Event Mapping from ServiceCenter to VP Operations.....	5-16
Configuration Overview.....	5-16
ServiceCenter TCL Variables	5-17
IT/O Programming APIs as TCL Commands.....	5-18
Summary of IT/O TCL Commands	5-18
opcif_write.....	5-19
opcmsg_annotation_add.....	5-20
opcmsg_ack.....	5-20
opcmsg_unack.....	5-20
opcmsg_own.....	5-20
opcmsg_disown	5-21
opcmsg_escalate.....	5-21
opcmsg_op_action_start.....	5-21
Event Configuration File	5-21
Sections	5-21
Default Behavior	5-22
Default pmo.tcl script	5-23
VP Operations Business Logic Configuration.....	5-25
General Process	5-25
Requirements Analysis	5-25
Design Phase.....	5-25
Implementation Phase	5-26
Design Considerations.....	5-26
Implementation Steps.....	5-27
Step 1 - Review Templates Assigned to a Node	5-27
Step 2 - Begin Modifying Templates (to Assign to a Node)	5-28
Step 3 - Modify Templates (to Assign to a Node)	5-31
Step 4 - Modify Template Conditions to Forward to Trouble Ticket.....	5-32
Step 5 - Modify Template Conditions to Use Message Stream Interface	5-34
Step 6 - Add Modified Template to Configuration	5-35
Step 7 - Install Templates to Selected Nodes.....	5-38
Step 8 - Final Configuration for use of Trouble Ticket Interface.....	5-40
Step 9 - Final Configuration for Use of Message Stream Interface.....	5-40
ServiceCenter Business Logic Configuration	5-42
ServiceCenter Event Services	5-42
Event Registration.....	5-42
Event Maps	5-44
ServiceCenter Problem Management.....	5-45

Chapter 6 Scenarios

Uni-directional Automatic Problem Ticket Creation
(Mode 1)6-1

(Out of Box) Bi-directional Problem Ticket/ITO Message
Creation/Update/Close (Mode 2).....6-3

Creating Problem Tickets with Trouble Ticket Interface
(Mode 3)6-4

(Out of Box) Node Based Problem Tickets.....6-5

Cause-Based Problem Tickets Per Node.....6-6

IT/O Message Group into ServiceCenter Category.....6-7

Converting Messages from Windows NT6-8

Appendix A Contacting Peregrine Systems

North America, South America, Asia/Pacific A-1

Europe, Africa A-1

Documentation Web Site A-2

Index

Chapter 1 Introduction



Overview

Welcome to *SCAuto for HP OpenView VantagePoint Operations*. This guide provides instructions on how to implement the interface for integration with Peregrine Systems' ServiceCenter product.

SCAuto for HP OpenView VantagePoint Operations allows you to automate the process of creating, updating, and closing trouble tickets in ServiceCenter, based on IT/O Message Stream Interface (MSI) and Message Event Interface (MEI) events. It has the capability of annotating, owning, and acknowledging IT/O messages from modifications done on ServiceCenter trouble tickets.

This product is part of the suite of SCAutomate (SCAuto) interface products that integrate ServiceCenter with premier Network and Systems Management tools. The interface is based on event messages sent over TCP connection to the ServiceCenter server. Additional information about SCAutomate can be found in the *SCAutomate Applications for Windows NT and UNIX Guide*.

Note: HP Openview IT/Operations (IT/O) has been renamed to HP OpenView VantagePoint Operations for UNIX (VP Operations). Note that the name change is not yet fully implemented across the VantagePoint Operations software and you will encounter the former name (IT/O) in this guide. The names VP Operations and IT/O are synonymous throughout this guide.

Note: SCAuto for HP OpenView VantagePoint Operations is referred to as *SCAuto for VP Operations* in the remainder of this guide.

Knowledge Requirements

This guide assumes the reader has:

- Working knowledge of ServiceCenter applications, ServiceCenter Client/Server, and the HP OpenView VP Operations graphical user interface. While some procedures for these applications are explained, others are referenced. Refer to the appropriate ServiceCenter documentation for a more detailed explanation.
- Familiarity with HP OpenView VP Operations and its components including the Network Node Manager (NNM).
- Working knowledge of the operating system environment in which they will be working (such as a GUI or text-based environment).
- (As an Administrator) a thorough knowledge of the operating system where ServiceCenter, SCAutomate, and the SCAuto for VP Operations product will be installed and implemented, as well as a basic understanding of ServiceCenter applications and Event Services.

Determining Current Product Version

Knowing the current version of your SCAuto for VP Operations product is valuable when contacting Peregrine Customer Support and deciding when to upgrade. From a ServiceCenter client, refer to the About menu to determine the current version of ServiceCenter.

Each of the executables *scfromitoMEI*, *scfromitoMSI*, *scfromitoTTI* and *sctoito* can be given a command line argument of “-v” to display their current versions and copyright information.

SCAuto for VP Operations

SCAuto for VP Operations offers integration between ServiceCenter applications and HP OpenView VantagePoint Operations. The interface of these two applications allows systems management functions of VP Operations to be extended and enhanced by the service desk functions of ServiceCenter.

This integration benefits customers of ServiceCenter and VP Operations by providing these features:

- A more robust environment for production IT operations
- Automated ticketing functions supported by the service desk processes
- The ability to structure and organize the real-time responses supported by the enterprise systems management functions.

The sections that follow give a high-level view of the operational concepts in this integration. These sections are:

- ServiceCenter
- VP Operations
- SCAuto for VP Operations
- Operational Concepts
- Planning Your ServiceCenter and VP Operations Integration
- VP Operations Business Logic Topics
- ServiceCenter Business Logic Topics
- SCAuto for VP Operations Business Logic Topics

ServiceCenter

Core Applications

ServiceCenter contains three core applications:

- **Problem Management**, which is a problem-tracking tool integrated with knowledge tools to speed problem resolution.
- **Inventory Configuration Management (ICM)**, which maintains an operational database of assets used in the enterprise.
- **Change Management**, which allows you to manage the evolution of the enterprise to meet changing needs and requirements as they arise.

Inventory Configuration Management tracks what the enterprise was expected to be, Problem Management works with the current state, and Change Management offers a way to manage towards the desired final product.

Additional Applications

ServiceCenter provides additional applications that support and enhance the core applications. These applications generally support the core applications with different angles of delivering services from a central call center.

The additional ServiceCenter applications are described next:

- **Service Management** provides a call-or incident-based front end (primarily) to Problem Management and other applications.
- **SLA Management** offers significant value with service desk automation based on service level agreements (SLAs).
- **Request Management** maintains catalogs of services and items and organizes the delivery of these.
- **Work Management** adds dynamic resource planning tools to help manage service desk actions.

VP Operations

VantagePoint Operations (VP Operations) is the foundation of HP OpenView's operational control of IT resources. It is a part of HP's OpenView suite of management applications.

VP Operations features data, event, and process level integration with other OpenView applications. It provides a central management console for enterprise systems management actions. It is built around a robust framework architecture, with an event console at the core.

Functional Areas

VP Operations has several functional areas that are built from the event messages handled by the event console. These functional areas are described next:

- Message groups and message templates directly impact the events.
- User groups collect IT/O operators and allocate authorization roles.
- Node groups perform a similar function for the managed resources.
- Application groups structure the IT/O monitored applications into organized, manageable components.

Network Node Manager

VP Operations incorporates the OpenView Network Node Manager product. It performs SNMP-based network monitoring. It acts as a perfect companion for the event console of IT/O because it generates event messages from incidents and faults on the network.

About SCAuto for VP Operations

SCAuto for VP Operations is designed in a modular way. Specific modules perform specific data processing functions. There are three separate modules that connect to three specific IT/O APIs:

- The Message Stream Interface (MSI) is a registration API that delivers notice whenever new events arrive at the event console.
- The Message Event Interface (MEI) is the interface that delivers messages upon status changes to existing events.
- The Trouble Ticket Interface (TTI) facilitates the help desk integration feature of message templates. When event messages match the conditions of templates, problem tickets may be opened automatically. The use of this API minimizes the configuration of the SCAuto for VP Operations adapter and allows the user to configure VP Operations. This creates an extension to make VP Operations interact with ServiceCenter automatically.

The MSI and the TTI function in a similar way. By registering with VP Operations at these APIs, new events received by VP Operations are output to SCAuto for VP Operations. This allows ServiceCenter to open new tickets. Subsequent actions against the event message (such as annotation or acknowledgment of the event) are output to SCAuto for VP Operations through the MEI API.

Individual APIs are used for individual data message exchange, but the combination of two separate APIs is used throughout the duration of the data exchange. Of the combinations, the TTI & MEI integration is preferred, and the MSI & MEI integration is also valid.

Figure 1-1 shows the architectural block diagram of the integration and the adapter.

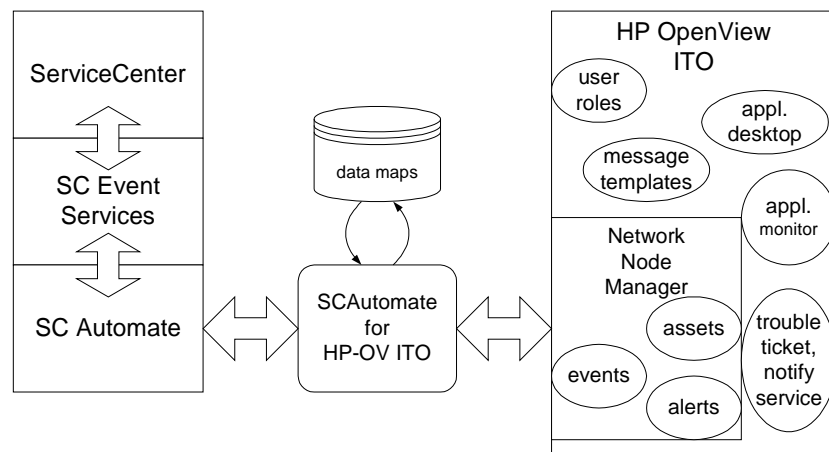


Figure 1-1. SCAuto for VP Operations Architectural Block Diagram

Figure 1-2 shows the modular components of SCAuto for VP Operations in a functional block diagram.

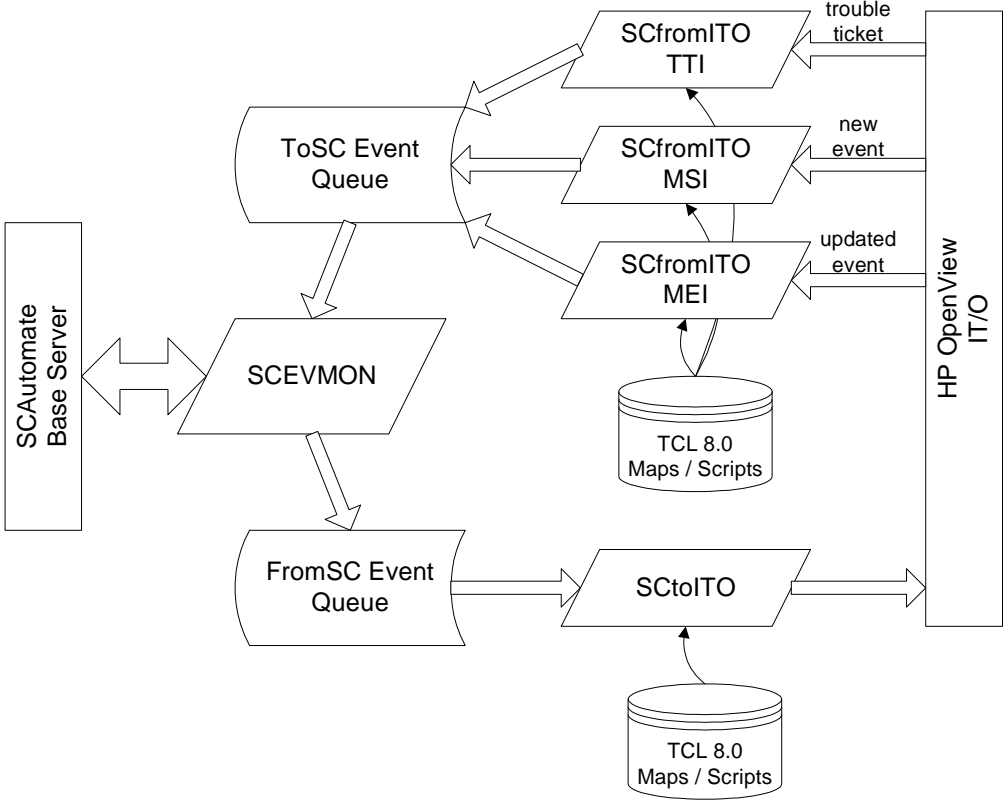


Figure 1-2. SCAuto for VP Operations Functional Block Diagram

In addition to the programmatic connections, other components within VP Operations are configured by the installation of SCAuto for VP Operations. For example, a ServiceCenter message group is configured to provide a preconfigured view of the event messages delivered by ServiceCenter to VP Operations. Upon installation, the function is available for selection and activation without any configuration or customization required.

SCAuto for VP Operations also uses components that are a part of the standard SCAutomate Software Development Kit (SDK). Key modules from the SDK are the event monitor (scevmon), the bi-directional event queues (ToSC and FromSC), and the event maps that formulate the conversion of incoming or outgoing messages.

Operational Concepts

The background of the static components was discussed previously. This section discusses the dynamic nature of the integration, which is a critical part of understanding the overall integration.

VP Operations is essentially a real-time management tool, while ServiceCenter delivers a process-oriented framework for operations. The synthesis of these domains is a dynamic environment where events and state changes drive procedure and process, and vice versa. This synthesis offers dramatic benefits for the customers of ServiceCenter and VP Operations.

For more information about integration, refer to Chapter 3, *Product Architecture*.

Bi-Directional Integration

The default mode of SC Auto for VP Operations operation is the bi-directional integration of ServiceCenter and VP Operations. In this mode, IT/O events trigger ServiceCenter processes. By default, selected events automatically open problem tickets. Subsequent actions or event messages at IT/O may update or close the tickets, or a similar exchange may occur from ServiceCenter into VP Operations.

Because most ServiceCenter applications can accept and generate event messages, this same operational flow can be applied to these ServiceCenter applications as well. This default mode of operation can be configured to meet specific end user requirements. The section *Planning Your ServiceCenter and VP Operations Integration* on page 1-11 describes the modes of operation that can be derived from the default.

All event messages that are exchanged between ServiceCenter and VP Operations must be converted from their native format into a format compatible with the destination. Event messages sent to ServiceCenter must be formatted in the event message structure defined by the SCAuto SDK. SCAuto for VP Operations performs most of this formatting, but it relies upon ASCII text map files to specify how to convert specific IT/O event message data fields into ServiceCenter event message data fields. A similar process is used for converting outbound event messages from ServiceCenter to VP Operations.

The adapter may create one-to-many relationships of IT/O events to incoming SC event messages (Figure 3-3 on page 3-9). This functionality is facilitated by the input maps that use the TCL scripting language. With this feature, for example, an SNMP Node Down event message can do any of the following:

- Open a problem ticket.
- Update an ICM record.
- Start an SLA outage metric against a logical item, such as a database that runs on the node and is reported as down.

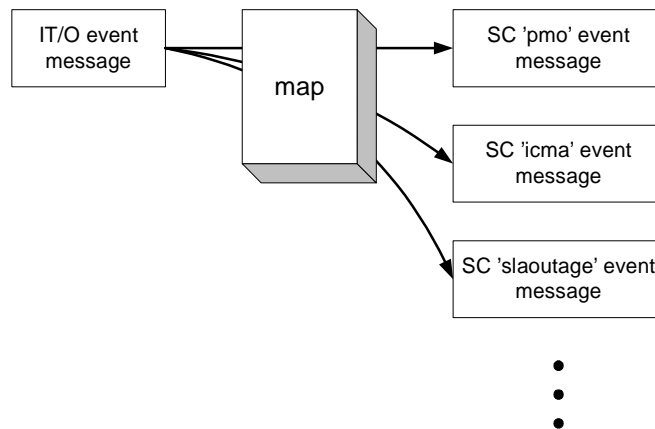


Figure 1-3. One IT/O Event Message to Many SC Event Messages

The adapter can also create one-to-many relationships of outgoing ServiceCenter events to VP Operations event messages or other actions. This output functionality is also supported by the use of TCL scripting language in the output maps. Since the maps are really script programs built to massage data fields, they can be edited easily to cause any desired actions.

The typical event message relationship will be one VP Operations event message to one ServiceCenter event message (Figure 1-4). Furthermore, many ServiceCenter event messages will be used to alter the state and the data of just one ServiceCenter problem ticket. For example, a IT/O event message will create an ServiceCenter event message that will open a problem ticket. Then another IT/O event message will create an update request type of ServiceCenter event message that adds new information to the original problem ticket. Then a final IT/O event message will create a close request that ends the active life of the ticket.

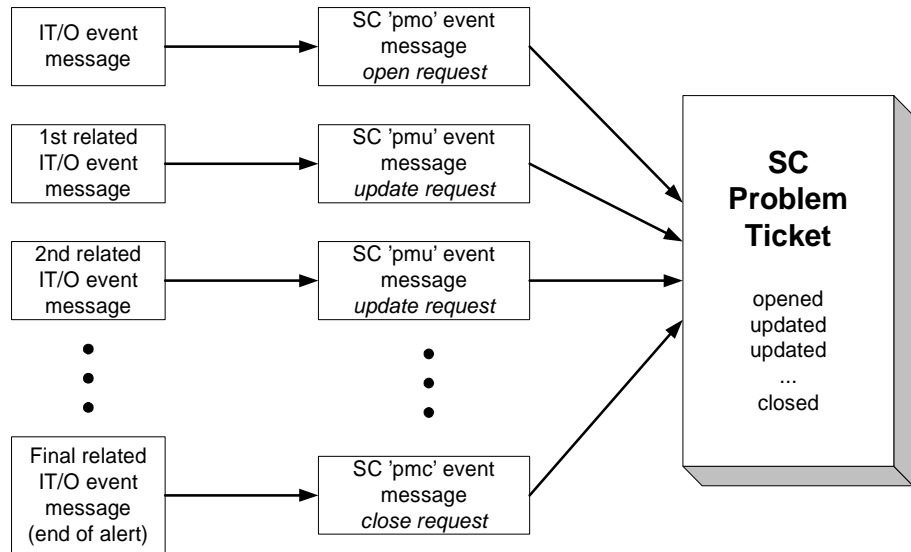


Figure 1-4. One-to-One Event Message Relationships, with Many Event Messages Linked to Just One Problem Ticket

Planning your ServiceCenter and VP Operations Integration

The following sections describe some of the ways in which SCAuto for VP Operations can be used. These scenarios help you plan and design your own implementation of VP Operations integrated with ServiceCenter via SCAuto for VP Operations. Any mode, combination of modes, or all modes can be used in a single deployment of SCAuto for VP Operations. Furthermore, an end user can extend the product to implement unique operational processes not described in any of the following mode sections.

For more scenarios, refer to Chapter 6, *Scenarios*.

Mode 1: Uni-Directional Automatic Notification from VP Operations

In this mode, IT/O events drive ServiceCenter tickets (records). To understand this mode, you should assume that:

- Business logic is applied to the IT/O event console to select which events are to be forwarded to ServiceCenter (refer to “VP Operations Business Logic Topics” on page 20).
- The only relationship between VP Operations and ServiceCenter consists of automatic tickets from IT/O events. Operationally, this implies that the monitored systems and resources are proactively creating tickets when conditions are met. However, it also means that all actions on resolving the issue, fault or problem are coordinated and administered from the ServiceCenter service desk.
- The IT/O event console becomes a “lights out” processing engine that feeds the service desk with intelligent real-time data, and then both expects and receives no further information or integration of service processes.

Process and Data Flow

This mode follows a process and data flow as shown in the following two flowchart diagrams (Figure 1-5 and Figure 1-6).

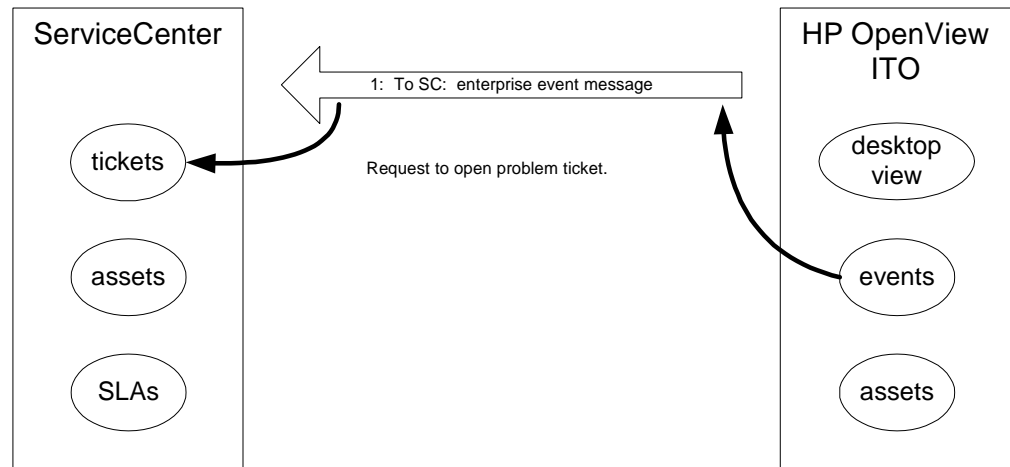


Figure 1-5. Uni-Directional Mode High Level Flowchart

This mode is configured through the registered connection from VP Operations to ServiceCenter via the MSI API. This mode will invoke the following components of SCAuto for VP Operations:

- *scfromitoMSI* and *scevmon* processes
- ToSC Queue
- Event Maps (*event.ini*, and maps referenced in *event.ini*)

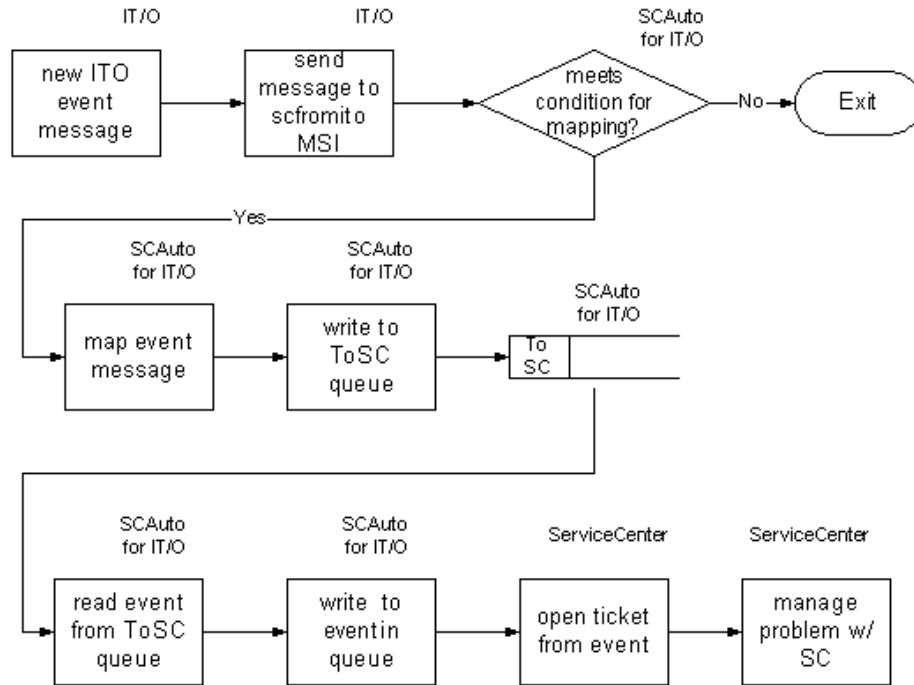


Figure 1-6. Uni-Directional Mode Flowchart

Mode 2: Bi-Directional Exchange (Default Mode of Operation)

In this mode, IT/O events drive ServiceCenter tickets (records), and the ServiceCenter tickets control IT/O events. This creates a partnership of managing the events, where each application has a significant contribution to the event and problem management process.

This mode incorporates Mode 1, but it extends it by generating ServiceCenter events that are sent to VP Operations. From this bi-directional interface, there are multiple scenarios based upon state transitions and the path of service processing. For example, there are three actions on IT/O events that are able to be taken (via specific event messages): annotate, acknowledge, or own. There are many more actions that can be taken on tickets, but the various instances always represent a generalized open, update, or close action on the ticket.

This mode of application partnership is the typical operational mode. It offers service desk functions to extend and enhance the real-time event management of VP Operations. It moves the service desk into more proactive fault management, allowing service desk analysts to respond to emerging issues, rather than reacting to fully developed faults.

This mode allows a richer protocol of integration. In effect, VP Operations may request the opening of tickets. In response, ServiceCenter acknowledges the open ticket and annotates the IT/O event. Subsequent exchanges may inform the applications of changes in state and data values. This mode also supports a complete cycle of interaction in which a close ticket closes (acknowledges) an IT/O event, and vice versa. Building this type of interaction involves business logic in both ServiceCenter and VP Operations.

Process and Data Flow

This mode follows a process and data flow as shown in the following two flowchart diagrams (Figure 1-7 and Figure 1-8).

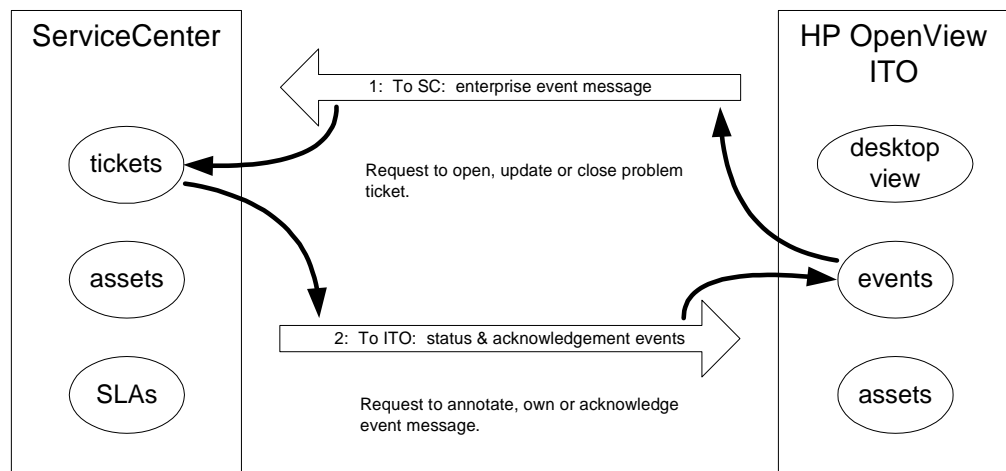


Figure 1-7. Bi-Directional Mode High Level Flowchart

This mode is constructed through the registered connection from VP Operations to SC via the MSI API, as well as the MEI API. This mode will invoke the following components of SCAuto for VP Operations:

- scfromitoMSI, scfromitoMEI, sctoito and scevmon processes
- ToSC and FromSC Queues
- Event Maps (*event.ini*, and maps referenced in *event.ini*)

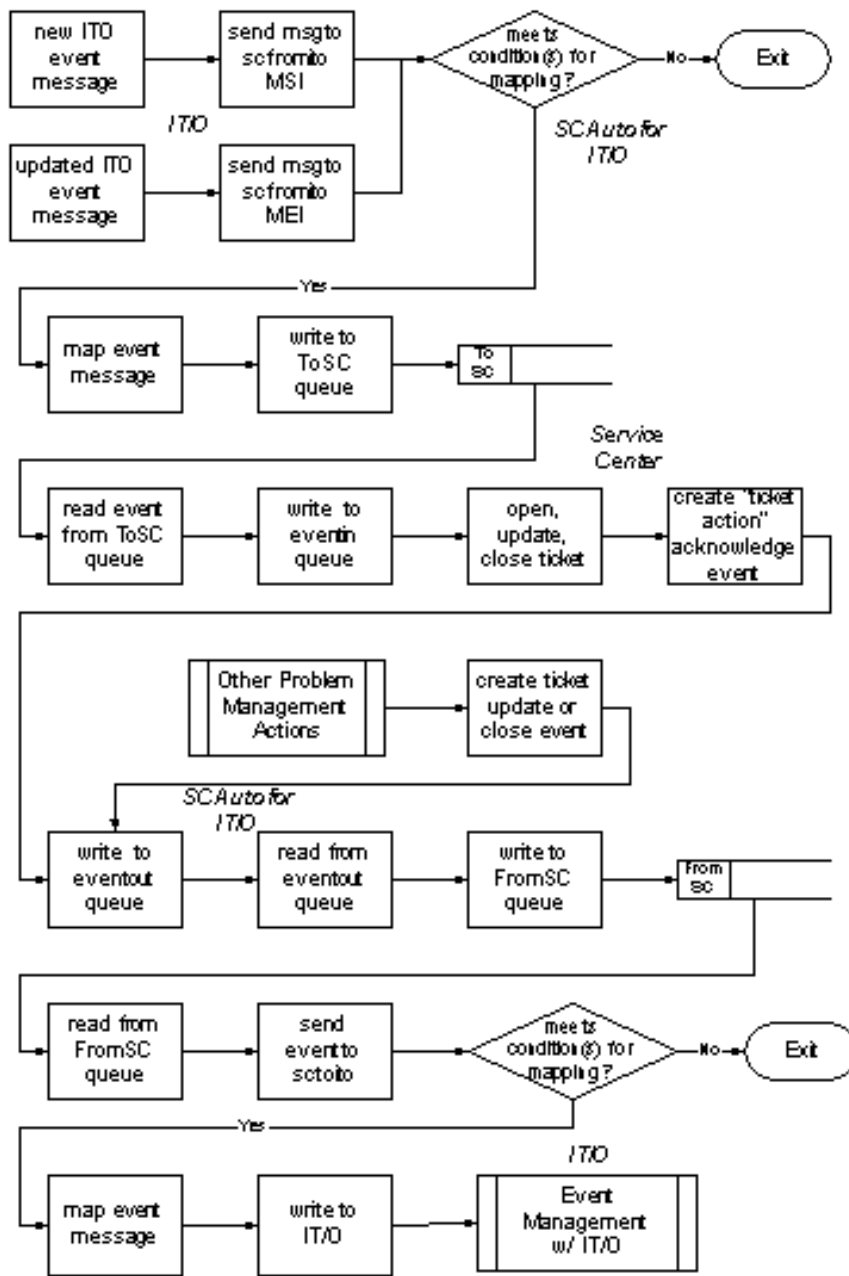


Figure 1-8. Bi-Directional Mode Flowchart

Mode 3: Automatic Notification from VP Operations via Trouble Ticket Interface (TTI)

This mode is an alternative configuration of Mode 1 (uni-directional) or Mode 2 (bi-directional). It uses the IT/O TTI API as opposed to the MSI API. This allows for a more focused configuration of IT/O Message Source Templates, freeing the MSI API to be used for other integration efforts, as well as avoiding the more complicated template formatting required with MSI configuration. With this mode, the MEI usage is identical to the previous modes.

This mode will invoke the following components of SCAuto for VP Operations:

- scfromitoTTI, scfromitoMEI, sctoito and scevmon processes
- ToSC and FromSC Queues
- Event Maps (*event.ini*, and maps referenced in *event.ini*)

This mode will use the identical flowcharts shown earlier for Modes 1 and 2. The only difference is that references to scfromitoMSI are replaced with scfromitoTTI.

Mode 4: New SC Tickets Generate IT/O Events

This mode treats ServiceCenter as an event source or event generator. Through the application of ServiceCenter business logic, new problem tickets activate logic, which generates event messages that are sent to VP Operations. IT/O Message Source Templates can be configured to treat these events like any other systems or network management events handled at the IT/O console.

This mode begins like Mode 1, but reverses the direction of the first event message. Subsequent interaction looks exactly like Mode 2. In operation, Mode 4 would allow an *IT/O-centric* model of service desk and enterprise management interaction. If desired, this would allow VP Operations to track certain ServiceCenter actions, as if they were SNMP traps, systems administration events, or any other typical VP Operations managed messages.

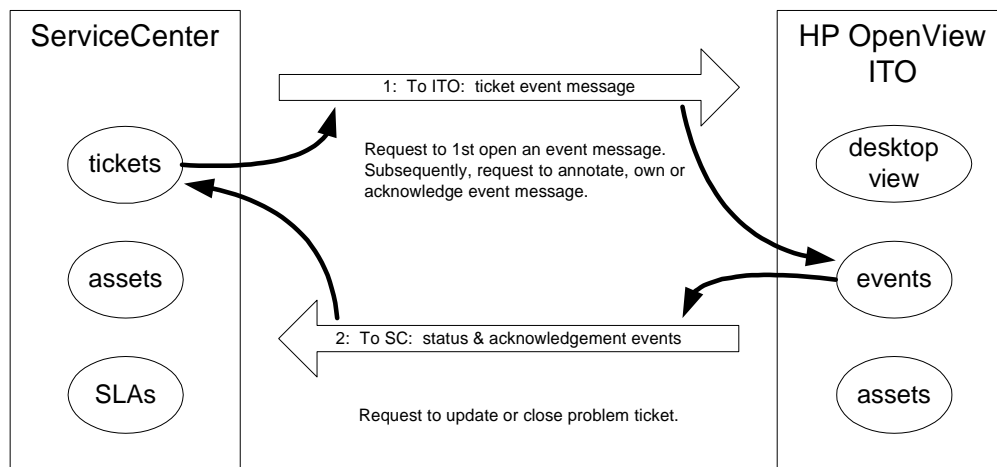


Figure 1-9. Bi-Directional Mode High Level Flowchart

Mode 5: Application Monitoring — VP Operations Monitors ServiceCenter

This mode of operation acknowledges that a primary use of VP Operations is to monitor and manage computer applications. Upon installation of SCAuto for VP Operations, additional monitoring configuration functions are installed into the IT/O Application Bank. These functions leverage the Application Response Measurement (ARM) specifications to offer consistent, standardized treatment of ServiceCenter. This mode also contains Message Source Templates, which allow VP Operations to be configured to generate events about the health of SCAuto for VP Operations and about the health of ServiceCenter itself. Further configuration of the Message Source Templates allows these events to become automatically generated problem tickets within ServiceCenter.

Mode 6: Combined User Interface — Launch ServiceCenter from IT/O Windows

To facilitate greater ease of use, SCAuto for VP Operations features several options for launching ServiceCenter clients directly from IT/O windows. From the IT/O *root* window, an icon launches a ServiceCenter client, opening directly to the main menu. This feature uses a configuration file to save the user ID and password needed to log into ServiceCenter. The function of this icon is also available from a pull-down menu on the *root* window.

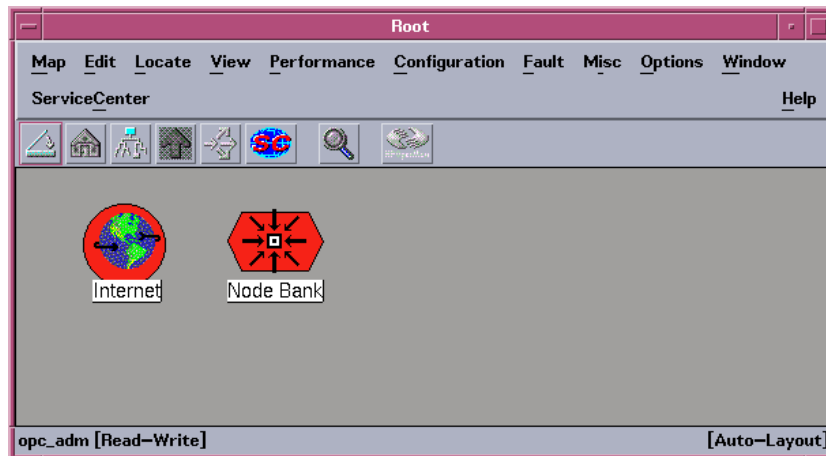


Figure 1-10. Basic IT/O root GUI with ServiceCenter Launch Methods

In addition to the *root* window, the *IT/O Node Bank* window has a similar menu and icon to launch ServiceCenter windows. In the node bank, it is possible to select a node and launch specific views of ServiceCenter data on the selected node. For example, one menu item lists open problem tickets on the node, while another shows the Inventory record on the node.

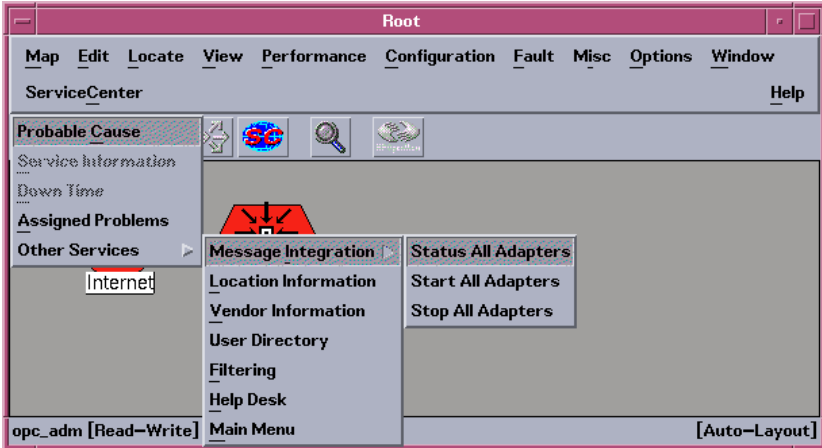


Figure 1-11. IT/O root GUI with Extended ServiceCenter Menu

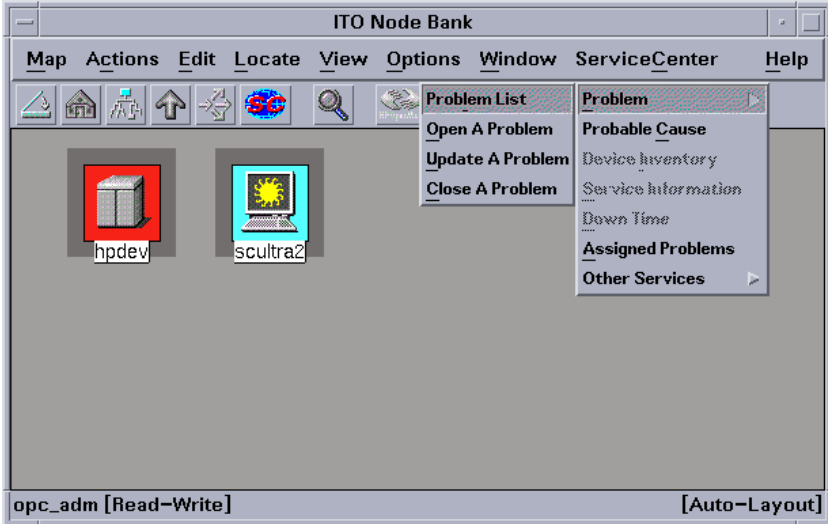


Figure 1-12. IT/O Node Bank GUI with ServiceCenter Launch Methods

VP Operations Business Logic Topics

VP Operations automates repetitive tasks; it also enhances and supports operational processes through the tools of VP Operations. These benefits can be applied broadly because VP Operations has the ability to adapt to specific needs. This adaptability comes from VP Operations' flexible approach to operations support. It results in an efficient way of collecting the logic of operations and the logic of running a business and using this business logic in repeatable, sustainable ways.

VP Operations captures business logic within its operational tools. For example, if an IT department must scrutinize user login attempts for security reasons, VP Operations can automate the process of such monitoring, and it can also automate the response to unusual conditions. The business logic is stored in Message Source Templates. These templates begin with expected enterprise events, and they configure the IT/O processing engines to consistently respond to the events.

Message Source Templates have a wide variety of configuration parameters. Some configuration parameters condition and massage the data of the event messages. Other parameters specify actions and tasks to perform in response to received event messages. These are generic examples of programming business logic into the VP Operations environment. SCAuto for VP Operations relies on specific configuration of the templates to invoke application programming interfaces.

In a basic sense, VP Operations assumes that all events from all sources are potential candidates for integration with a help desk. IT/O creates an API called the Trouble Ticket Interface (TTI) API for this purpose. Message Source Templates offer an option button to select this integration function, which automatically sends event messages to all applications that are registered and listening to the TTI API.

An advanced configuration of Message Source Templates is also available. Under these advanced options, you can specify that event messages from specific sources extend themselves to the Message Stream Interface (MSI) API. You can also select whether this is activated at the server or at the agent. When this option is selected, all event messages from the message source are copied to all applications that are registered and listening to the MSI API.

SCAuto for VP Operations leverages the above API configuration options to invoke its programs. This amounts to a business logic decision of *when events of this type occur, invoke the service desk*. This logic, in conjunction with the generic IT/O configuration options (which capture filtering, data massage, and event message routing business logic), offers the IT/O resident component capability of synthesizing the service desk with the enterprise systems management console.

ServiceCenter Business Logic Topics

ServiceCenter facilitates and automates the process of supporting business operations. Much like VP Operations, these benefits can be applied broadly because of ServiceCenter's flexibility and adaptability to address specific needs. Like VP Operations, this reflects ServiceCenter's ability to capture business logic within its applications.

ServiceCenter provides a significant amount of generic business logic within the service desk applications. The Problem Management application, for example, follows a well-defined help desk model of problem identification, tracking, and resolution. It can be used as is, or it can be customized to address specific organizational requirements. Custom business logic is most often applied to the escalation and notification aspects of the process model, although the process workflow is also frequently altered.

Business logic in ServiceCenter can be associated with several levels of ServiceCenter:

- At the bottom level are the ServiceCenter databases, where trigger mechanisms can dictate custom actions.
- Above the database level is a utilities level, which is composed of Schedules, Format Control, Macros, and Links. These tools offer the preferred location for applying custom business logic. All of the tools were designed to support this function. The ServiceCenter Event Services module essentially connects with ServiceCenter at the utilities level. This module provides equivalent functions related to processing incoming and outgoing event messages.
- The final level where business logic can be defined is at the ServiceCenter applications. This level either lends itself to easy customization of the user presentation of application data (via Forms Designer), or it involves more significant alterations that require a programming solution (via ServiceCenter's fourth generation language, RAD).

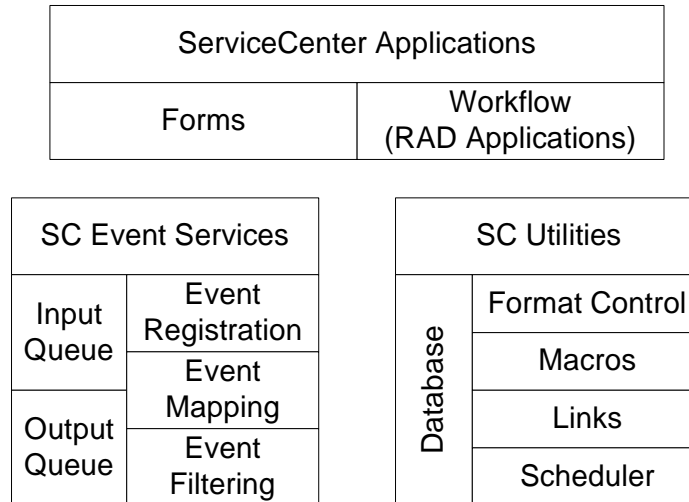


Figure 1-13. ServiceCenter Business Logic Configuration Tools

SCAuto for VP Operations utilizes the ServiceCenter Event Services module for its business logic integration with ServiceCenter. It leverages default configurations of event messages to open, update, or close a problem ticket. These event message requests are defined as event types of pmo, pmu, or pmc, respectively. The processing logic of these requests can be altered if needed, but the default configuration is satisfactory for integration with VP Operations.

Comparable to VP Operations' business logic decision of *when events of this type occur, invoke the service desk*, the SC Event Services response is *when the service desk is invoked, receive and process the data, and then launch the appropriate service desk function*. This logic is often combined with Format Control utility logic, which injects programmed queries, tests, and automated actions into the application layer (and therefore into process model of the service desk).

This configuration of utility level business logic offers the ServiceCenter resident component of synthesizing the service desk with the enterprise systems management console. When combined with the IT/O portion, the result is an extremely powerful integration that facilitates more effective and efficient IT operations and the operations of the business as a whole.

SCAuto for VP Operations Business Logic Topics

The previous two sections discussed capturing your business requirements in the configuration of VP Operations or ServiceCenter. As you define how your consolidated service desk functions, you will undoubtedly use the topics of the last two sections. However, when deploying such an enterprise solution, you may also need to put business logic into the middleware represented by SCAuto for VP Operations.

Other sections of this document discuss the features of this product that allow it to be customized and configured. Any of these features are sufficient for beginning to program business logic into the product. Realistically, any custom logic will be coded into the TCL scripts used for the data maps. Business logic captured in the map scripts will be able to affect many aspects of the overall integration: unique data items, selection of particular event messages, or creating many events from one input event. Regardless, refer to the earlier sections for more details on SCAuto for VP Operations' abilities of capturing business logic.

In today's business world, companies rely upon their IT resources. HP OpenView and Peregrine Systems ServiceCenter deliver the IT support that companies count on. The new SCAuto for VP Operations product integrates these major applications, and creates a synthesis of real-time management and process oriented service delivery. Because of the natural dynamic nature of IT, there are constantly new challenges to be addressed in IT management. This new product helps address those challenges directly, and enables enterprise solutions built from the mature concepts of the consolidated service desk.

Chapter 2 Installation



Overview

This chapter gives the prerequisites for installing SCAuto for VP Operations and provides step by step installation instructions.

System Prerequisites

To use SCAuto for VP Operations, your installation must be on a UNIX platform and must include the following:

- ServiceCenter version 1.4 or later using TCP/IP
- HP OpenView IT/Operations version 5.x on HP-UX 10.20 or HP-UX 11
- HP OpenView VantagePoint Operations (VP Operations) version 6.x on HP-UX 10.20, HP-UX 11, Solaris 7, or Solaris 8
- 35 MB temporary unpack space
- 35 MB installed footprint

The ServiceCenter server can run on Windows NT, UNIX, or MVS, but must be run using the TCP/IP protocol. SCAuto for VP Operations presently runs exclusively on the UNIX platform.

VP Operations should be implemented and operational to the extent that meaningful events arrive at the Event Message browser, and at least one designated individual is familiar with the VP Operations product as well as defining IT/O Event Message sources and IT/O Message Actions.

SCAuto for VP Operations supports ServiceCenter 1.4 or later. By default, SCAuto for VP Operations is designed to work with the default ServiceCenter configurations. If ServiceCenter is tailored, you may also have to tailor SCAuto for VP Operations.

Customers with heavily customized or older systems should give consideration to use of Peregrine Systems, Inc. Professional Services to implement SCAutomate products

Required Kernel Parameters

Your HP-UX or Solaris machine must have the following kernel parameter settings:

- The minimum value of any semaphore must be greater than or equal to 1 (SEMVMX).
- The maximum number of semaphore sets systemwide must be increased by 3 (SEMMNI).
- The maximum number of semaphores systemwide must be increased by 3 (SEMMNS).
- The minimum number of semaphores per semaphore set must be greater than or equal to 1 (SEMMSL).
- The maximum number of undo structures systemwide must be increased by 3 (SEMMNU).
- The minimum number of undo entries per undo structure must be greater than or equal to 1 (SEMUME).
- The minimum number of operations per semop must be greater than or equal to 1 (SEMOPM).

Installation Prerequisites

To install SCAuto for VP Operations, you must have administrator-level access both to the VP Operations server and the ServiceCenter server. Installation takes only a few minutes if you have determined the correct host name and port number values beforehand.

Before you begin installation, you need the following information about your VP Operations and ServiceCenter installations:

- Administrator authority at the VP Operations Server
- Administrator-level access to the ServiceCenter Server
- An authorization code for ServiceCenter, which permits the use of the SCAuto/IT/Operations.
- The host name or IP address of the ServiceCenter server and the TCP port number for use by ServiceCenter and the SCAutomate Base Server

ServiceCenter Client Installation

Install the software on the VP Operations Management server, using the SC server full-client port when prompted. For more detailed information about installation, refer to the ServiceCenter client/server installation guide for your platform.

Installing SCAuto for VP Operations

There are two major parts of the installation process:

- The Event Integration binaries and support files. This includes HP Local Registration Files (LRFs) that integrate with the OVSPMD.
- The GUI Integration components, which can be broken down into:
 - Menu/Toolbar ServiceCenter client launch and Event Integration controls
 - ServiceCenter Message Group object and Application Group Objects

During installation, these parts will be configured with the customer's options and settings.

The deliverable image will be on a CD-ROM (ISO9660 format). The installation begins by executing the *install.sh* script. Installation includes these steps:

1. Mount the CD.
 - On HP-UX (version 10.20 or 11) use the command `mount -o cdcase` or `pfs_mount`.
 - On Solaris the CD is mounted automatically.
2. Go to the appropriate directory. The directory name is:
 - **hp_10** for HP-UX 10.20.
 - **hp_11** for HP-UX 11.
 - **Solaris** for Solaris 7 or 8.
3. Execute *install.sh*.

Install Procedure

Important: The installation must be done as the *root* user.

The installation process consists of two scripts: *install.sh* and *config/install2.sh* in *scito.tar*. Executing *install.sh* unpacks *scito.tar* into a directory and then executes *config/install2.sh*. Interactive prompts enable you to stop the installation at any time. During this process, files are placed into a directory for potential manual installation. Please note that stopping the installation is *not recommended*.

To continue installation, provide the following information when prompted:

- Root directory of HP OpenView (for example, */opt/OV*).
- Target directory for product installation. Installation of default event map files is optional (for example, */opt/OV/scauto*).
- Directory where ServiceCenter client was installed (for example, */products/SC*). *
- The ServiceCenter server host name and port number.*
- The SCAutomate host name and port number. The SCAutomate host name should match the ServiceCenter host name.

* **Note:** The ServiceCenter directory is required only if you want ServiceCenter Client GUI integration.

The installation script then takes you through the following steps:

1. Prompt for location of *ov.envvars.sh* and sources file.
2. Prompt for the target directory for installations, and it copies Event Maps and binaries over.
3. Create and configure *etc/scauto/SCITO.init* file with the home directory of product.
4. Configure product ini file *scito.ini* with SCAuto Server information.
5. Configure client launch *sc.ini* file with ServiceCenter information. (Optional)
6. Configure and install HP Local Registration Files with product start/stop information and add to HP OpenView SPMD.
7. Configure and add Menu/Toolbar GUI customizations to user *opc_adm* and *opc_op*. (Optional)
8. Create ServiceCenter Message Group Object in Message Group Bank and Application Object in Application Bank. (Optional)

After installing the adapter product, you are required to create a user *opc_op* with no password in ServiceCenter if you have chosen GUI integration. The ServiceCenter client launch uses this user to access ServiceCenter.

If you encounter a problem that you are unable to resolve, please contact Peregrine Customer Support.

Chapter 3 Basic Operations



Starting and Stopping SCAuto for VP Operations Processes

The SCAuto for VP Operations applications are installed into VP Operations from LRFs (Local Registration Files). This means that you can start and stop the processes using the HP ovstart/ovstop facilities. The SCAuto for VP Operations processes consists of the event monitor (scevmon), ServiceCenter to IT/O adapter (sctoito), and the IT/O to ServiceCenter adapters (sfromitoMEI, sfromitoMSI, and sfromitoTTI).

Starting SCAuto for VP Operations

There are three ways to start the SCAuto for VP Operations adapter processes:

- If you chose the GUI integration during installation of SCAuto for VP Operations, you can start all processes by selecting the following options, in order, from the menu on the IT/O Root Window or the IT/O Node Bank Window:
 - ServiceCenter
 - Other Services
 - Message Integration
 - Start All Adapters
- You can use the IT/O command line **ovstart** to start individual processes. For example, use **ovstart scevmon** to start the scevmon process and **ovstart scevmon sfromitoMSI** to start the scevmon and sfromitoMSI processes.
- SCAuto for VP Operations can also be started from its icon (called ServiceCenter Tools) in the IT/O Application Bank window (see Figure 3-1 and Figure 3-2).



Figure 3-1. IT/O Application Bank GUI with ServiceCenter Tools Icon

You can start or stop SCAuto for VP Operations by selecting the ServiceCenter Tools icon in the ITO Application Bank window.

The ServiceCenter Tools screen is displayed.

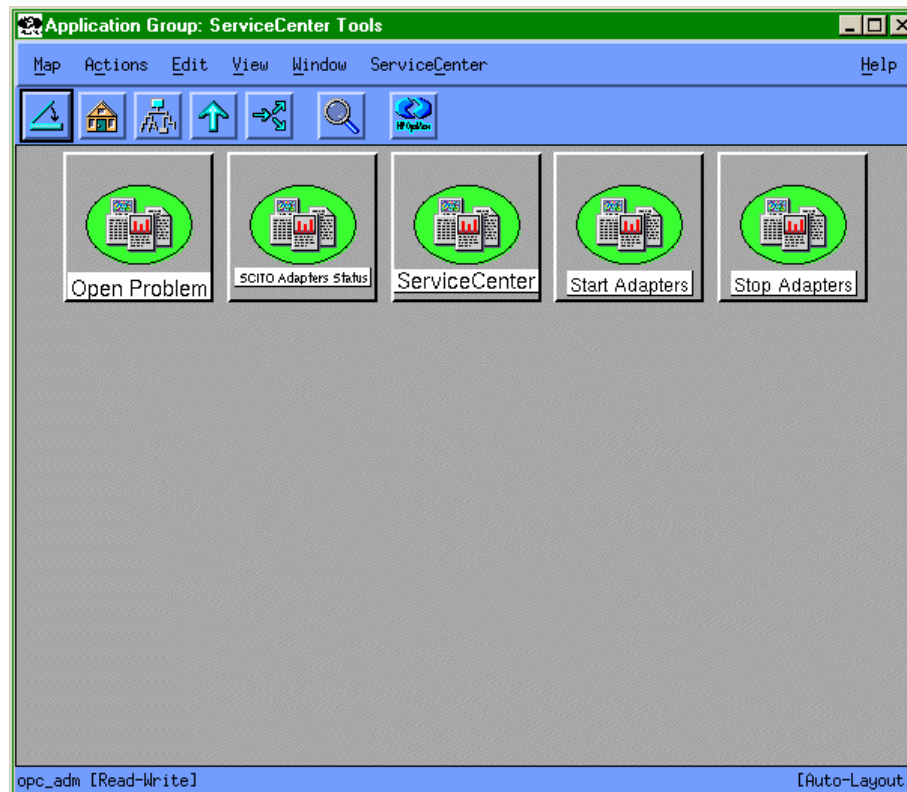


Figure 3-2. ServiceCenter Tools GUI with SCAuto for VP Operations Icons

This window contains a set of icons that controls SCAuto for VP Operations and offers some control over ServiceCenter.

- **Open Problem** - allows user to open a problem ticket on selected messages in the Message Browser.
- **SCITO Adapters Status** - shows the operational status of the adapter processes.
- **ServiceCenter** - allows user to launch the ServiceCenter client to connect to ServiceCenter.
- **Start Adapters** - starts the SCAuto for VP Operations adapter processes.
- **Stop Adapters** - stops the SCAuto for VP Operations adapter processes.

Stopping SCAuto for VP Operations

There are three ways to stop the SCAuto for VP Operations adapter processes:

- If you chose the GUI integration during installation of SCAuto for VP Operations, you can stop all processes by selecting the options listed below, in order, from the menu on the IT/O Root Window or the IT/O Node Bank Window:
 - ServiceCenter
 - Other Services
 - Message Integration
 - Stop All Adapters.
- You can use the IT/O command line **ovstop** to stop individual processes. For example, use **ovstop scevmon** to start the scevmon process and **ovstop scevmon scfromitoMSI** to stop the scevmon and scfromitoMSI processes.
- SCAuto for VP Operations can also be stopped from its icon (called ServiceCenter Tools) in the IT/O Application Bank window (Figure 3-1 and Figure 3-2).

Basic Maintenance

In the product installation directory, the SCAuto for VP Operations adapter contains a log file called *scito.log*, as well as a parameter configuration file called *scito.ini*, where all informational and error messages from the product are stored.

The product requires very little maintenance once installed and running. By default the log file is archived to *scito.log.archive* once it reaches 5M bytes. In addition, the event queues purge processed events to remove them once they reach 5M bytes each, retaining the unprocessed events.

scito.ini parameters

This is a subset of the commonly used parameters that can be set in the *scito.ini* file:

- ***sessid*** - By default, this is set to OPCENTER. This is the license string for the SCAuto for VP Operations product. Do not modify.
- ***scauto*** - This is the *hostname.port* of the SCAuto server for SCAuto for VP Operations. This parameter is configured by the installation script.
- ***event_map_dir*** - This is the root directory name (starting from the installation directory) of the event mapping scripts and files. By default, it is configured as “EventMap”.
- ***log*** - This is the file name for the log file. By default, it is configured as *scito.log*. Because the mapping scripts rely heavily on this file name for output, please be careful when changing it.
- ***debug*** - This is a general debug flag for the product to output more debugging strings. Set it to TRUE to turn it on.
- ***debugscautoevents*** - This is the SCAuto event debugging flag to output more debugging messages of the *event* type. Enable the parameter by setting it to 1.
- ***eventlogmaxlen*** - This parameter defines the maximum size, in bytes, that the event queue files can reach before being purged of processed events. The default is 5M. The minimum size is 1M.
- ***logmaxlen*** - This parameter defines the maximum size in bytes that the log file can reach before it will be wrapped down to *logpreservelen*. The default is 5M. It is not recommended to set this value to less than 1M.
- ***logpreservelen*** - This parameter defines the size of the log file to wrap down to in the event it reaches *logmaxlen*. The default is 0 which causes the *scito.log* file to be archived to *scito.log.archive* when the *logmaxlen* is reached and empties the current *scito.log* file. The maximum is 128K.
- ***usersepchar*** - Allows user to define a separator character other than “^” (^ is the default value). To specify a different character, enter a decimal value of an ASCII character between 1 and 255.

Basic Configuration

The SCAuto for VP Operations product is configured during installation. After the product is installed, it will work out of the box with the current VP Operations configuration. To further tailor the system to your business needs, refer to Chapter 5, “Configuration.”

Troubleshooting

If problem tickets in ServiceCenter are not being created by IT/O event messages, follow these steps to troubleshoot the problem:

1. Verify that the event monitor process `scevmon` is running. If this process is not running, execute `ovstart scevmon` to start it.
2. Verify that the SCAuto for VP Operations event monitor is communicating with the SCAuto Server. If the `scito.log` file contains the message `... scevmon : unable to connect ...`, do the following:
 - Check the `scito.ini` file for the parameter `scauto:`. Verify that it exists and points to the `<hostname>.<port number>` of the SCAuto Server you are trying to connect to.
 - Log in to the ServiceCenter Server host and verify that the SCAuto Server is running.
 - Check the network by pinging the ServiceCenter Server host to see if it is reachable.
3. Verify that the `scfromitoMSI` and `scfromitoMEI` processes are currently running. If not, execute `ovstart scfromito` to start them.
4. Verify that IT/O event message sources are correctly set up to communicate with SCAuto for VP Operations. (Refer to Chapter 5, “Configuration,” for more details on configuring IT/O message sources.)
 - If using TTI interface, verify that the event source has Trouble Ticketing enabled. Also verify that the Trouble Ticketing interface is correctly configured to use the `TTI.sh` script in the installed `scauto` directory.
 - If you are using the MSI interface, verify that the event source has Server MSI or Agent MSI message copy/divert enabled.
 - Verify that the `event.ini` configuration file in the `EventMap/ToSC` directory is properly configured.
5. Finally, check the `scito.log` file for any unusual error messages.

If ServiceCenter problem ticket modifications are not reaching the IT/O message browser, follow these steps to troubleshoot the problem:

1. Verify that a record is generated in the eventout file in ServiceCenter. Configure format control to output equivalent records accordingly.
2. Check to see if the event monitor process scevmon is running at the IT/O host. Execute `ovstart scevmon` to start it.
3. Check to see if the SCAuto for VP Operations event monitor is communicating with the SCAuto Server. If the scito.log file contains the message
`... scevmon : unable to connect ...`, do the following:
 - Check the scito.ini file for the parameter `scauto:`. Verify that it exists and points to the <hostname>.<port number> of the SCAuto Server you are trying to connect to.
 - Log in to the ServiceCenter Server host and verify that the SCAuto Server is running.
 - Check the network by pinging the ServiceCenter Server host to see if it is reachable.
4. Verify that the sctoito process is running using `ovstatus sctoito` on the IT/O host. If it is not running, execute `ovstart sctoito` to start it.
5. Check the configuration file EventMap/FromSC/event.ini, and verify that the user name and password are correct.
6. Finally, check the scito.log file for any unusual error messages.

Chapter 4 Product Architecture



This chapter covers the architecture of SCAuto for VP Operations, focusing on two primary topics:

- Application Integration
- Event Integration

Application Integration

SCAuto for VP Operations provides an enhanced operator interface to ServiceCenter that can run under VP Operations. From a IT/O window, you can access a number of ServiceCenter windows to gather information related to the current window or selected object.

This capability is only available if the ServiceCenter cut-throughs were enabled during the SCAuto for VP Operations installation. If the cut-throughs were not enabled, you can start a ServiceCenter client from the UNIX command line instead.

Note: Refer to the appropriate ServiceCenter documentation for more information on using ServiceCenter.

ServiceCenter clients are started through the ServiceCenter menu in a IT/O window. Depending upon the menu item selected, the client window will be opened to different ServiceCenter applications and windows.

Note: In order to run a ServiceCenter client, the IT/O window must not be started from the *root* user account.

All menu options are available if an icon for an object is selected in a IT/O window (Figure 4-1). Specific requests requiring an object selection are grayed out if an icon is not selected.

The following windows are a tutorial representation of SCAuto for VP Operations general operations. Windows and functions may change from

release to release, so reference the help files on your specific platform for the latest operational details.

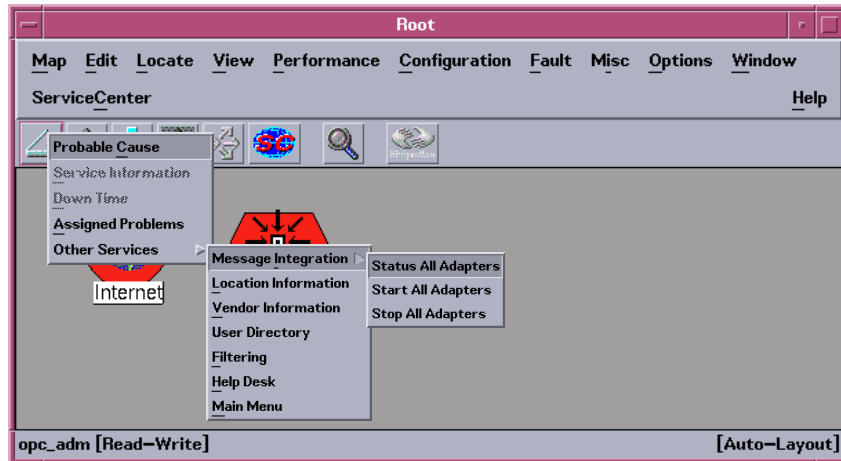


Figure 4-1 ServiceCenter menu

ServiceCenter Menu Options

The ServiceCenter menu options take you directly to the ServiceCenter applications from within VP Operations. These services save the time of logging in and navigating through ServiceCenter to get to these applications. The following sections provide a brief description of the windows.

Note: While some of the ServiceCenter application options are mentioned in this manual, you should refer to the ServiceCenter documentation for complete instructions on using the ServiceCenter applications.

To use a ServiceCenter application under SCAuto for VP Operations:

1. Select the **ServiceCenter** menu in the IT/O window and select the appropriate menu option. Some ServiceCenter menu options are not available unless an object is selected in the IT/O window.
2. Use the mouse or keyboard to navigate through a window.
3. To leave the application, select the **Back** button or press F3. This takes you to the previous window or to a logout window.
4. When the logout window is displayed (Figure 4-2), select the **EXIT** button from the popup menu or press F1 to exit the ServiceCenter session.

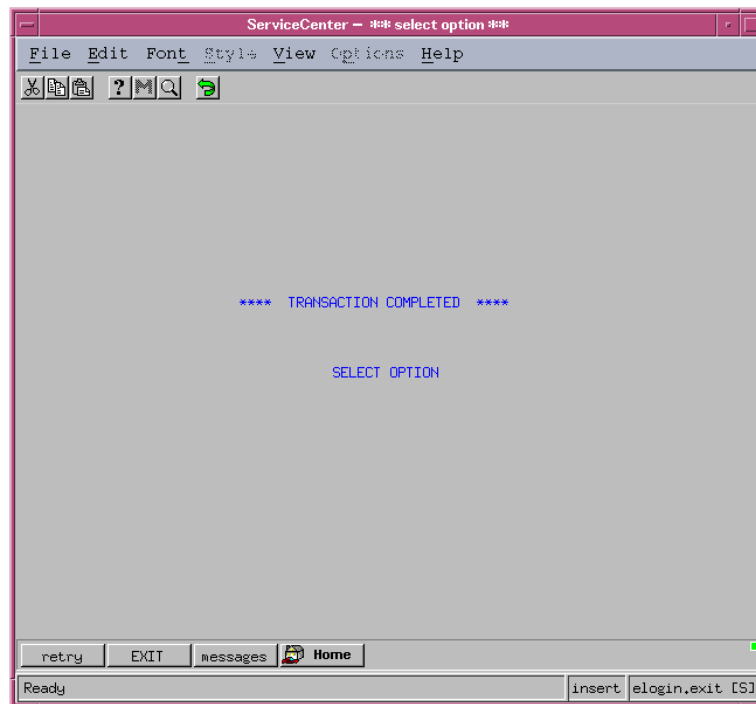


Figure 4-2 ServiceCenter Logout window

Note: The descriptions on the following pages are for ServiceCenter *Release 2*. If you are using an earlier version of ServiceCenter, the procedures will differ and the windows will not be similar to what are shown. The menu options will be the same however.

Problem List

The Problem List menu option provides a list of problems that are currently open in ServiceCenter for the selected object. When this option is selected, a problem list is displayed (Figure 4-3). Use the **Options** menu to display a list of operations to perform on the problems. Double-click an item in the problem list to display the Problem Detail window (Figure 4-4).

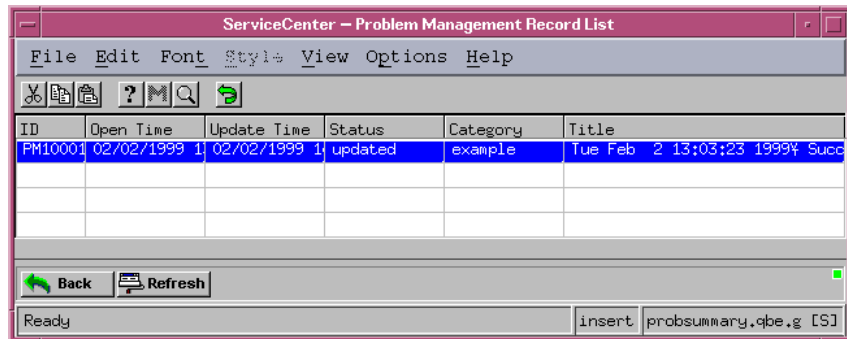


Figure 4-3 Problem Management Record List

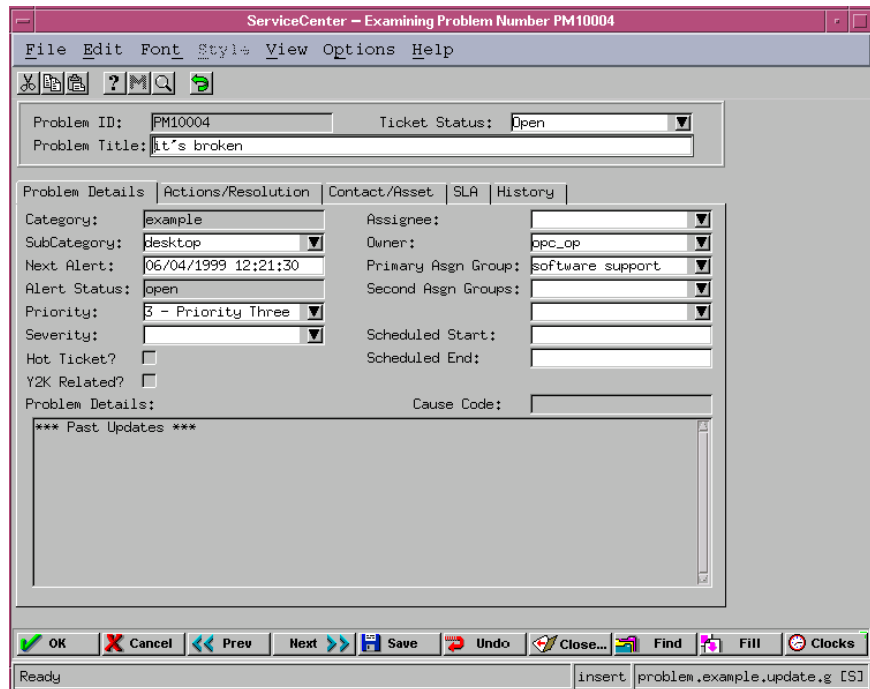


Figure 4-4 Problem Detail

Open A Problem

The Open A Problem menu option allows you to open a problem ticket in ServiceCenter for the selected IT/O node. When this menu option is selected, the Categories list is displayed (Figure 4-5). You can double-click any category in this list to display the Related Records list (Figure 4-6).

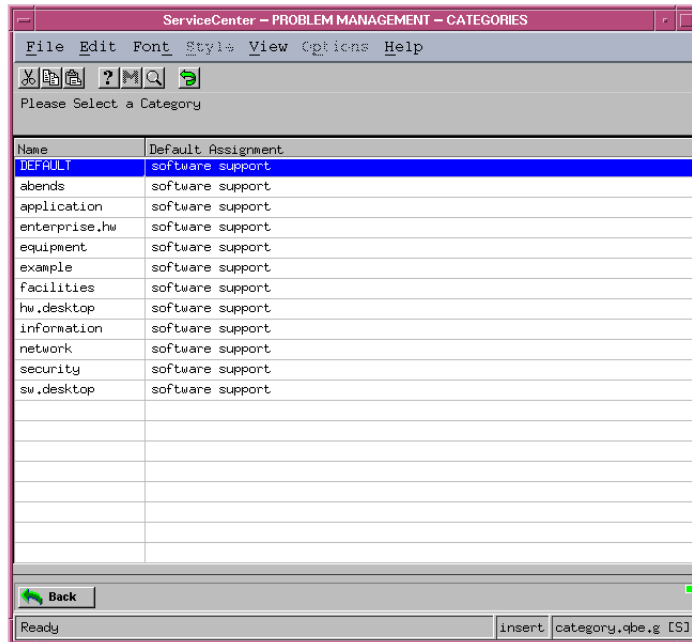


Figure 4-5 Categories list

ServiceCenter – Select related record

File Edit Font Style View Options Help

Back Forward ? M Q

Contact Name	Last Name	First Name	Phone	Extensi	Department
ABC10	sales		(818) 422-5505	323	sales
ABC12	operations		(818) 422-5505	324	sales
ABC35	marketing		(818) 422-5505	325	sales
ABC54	customer servic		(818) 422-5505	326	sales
ABC56	engineering		(818) 422-5505	327	sales
BROWN	Brown	Nicholas	404-954-4588	243	customer service
BUTLER	Butler	Richard	(818) 422-5505	328	customer service
FALCON	Falcon	Jennifer	617-455-7654	201	engineering
HAWTHORNE	Hawthorne	Greg	617-455-7654	202	operations
HELPDESK	Helpdesk	Bob	617-455-7654	203	customer service
HENNESEY	Hennesey	David	617-455-7654	204	customer service
IRWIN	Irwin	Jonathon	617-455-7654	205	engineering
JENKINS	Jenkins	Carol	617-455-7654	206	customer service
JOHNSON	Johnson	Jennifer	617-455-7654	207	engineering
JORDAN	Jordan	Susan	617-455-7654	208	marketing
KENTNER	Kentner	James	617-455-7654	209	marketing
MANAGER	Manager	Max	617-455-7654	210	customer service
MILLER	Miller	Adam	617-455-7654	211	sales
OCONNELL	O'Connell	Stacy	617-455-7654	212	operations
PETERS, J.	Peters	Jeff	617-455-7654	213	operations
PETERS, R.	Peters	Ruth	617-455-7654	214	customer service
SIMMONS	Simmons	Jeremy	617-455-7654	215	sales

Back Skip

* More than one record in contacts file matches field "contact.name". contacts.qbe.g [9]

Figure 4-6 Related Records list

In the Related Records list (Figure 4-6), you can double-click a related record to display the Create a New Problem Record window (Figure 4-7). After entering information for the new problem ticket in this window, click **Save** to save the problem ticket.

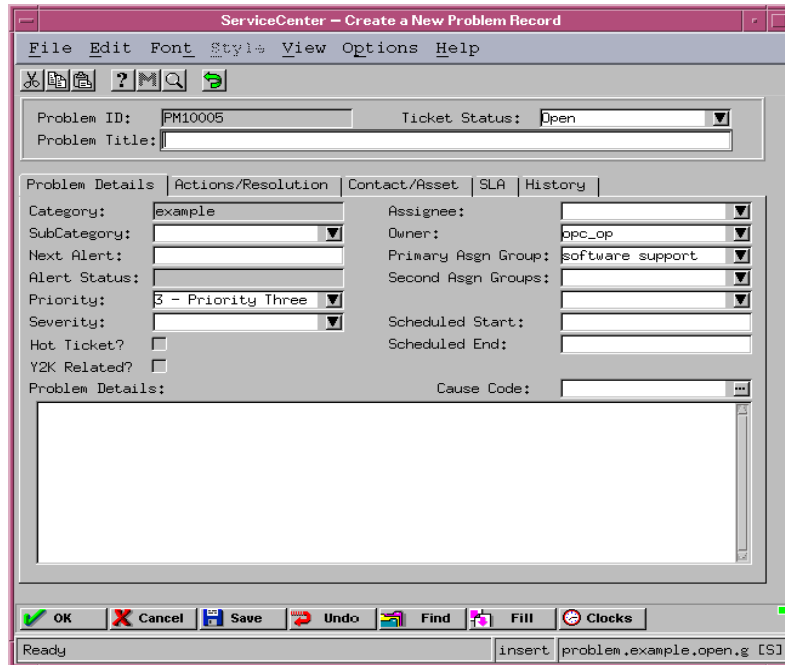


Figure 4-7 Create a New Problem Record window

Update A Problem

The Update A Problem menu option allows you to update an open problem in ServiceCenter for the selected object. When the menu option is selected, a list of ServiceCenter problems for the device is displayed (Figure 4-8). ServiceCenter displays an appropriate message if there are no open problems for the selected device.

1. Double-click the desired problem in the list. The Problem Detail window is displayed (Figure 4-4).
2. Click the **Save** or **OK** button to save your changes and update the problem ticket.

The screenshot shows a window titled "ServiceCenter - Problem Management Record List". It features a menu bar with "File", "Edit", "Font", "Style", "View", "Options", and "Help". Below the menu is a toolbar with icons for cut, copy, paste, help, search, and refresh. The main area is a table with the following columns: ID, Open Time, Update Time, Status, Category, and Title. The first row contains the data: PM10004, 04/05/1999 11, 04/05/1999 11, open, example, and it's broken. Below the table is a toolbar with "Back", "New", "Refresh", and "Count" buttons. The status bar at the bottom shows "Ready" and "insert: probsummary.qbe.g [S]".

ID	Open Time	Update Time	Status	Category	Title
PM10004	04/05/1999 11	04/05/1999 11	open	example	it's broken

Figure 4-8 Problem Management Record list

Close A Problem

The Close A Problem menu option allows you to close an open problem in ServiceCenter for the selected object. When the menu option is selected, a list of ServiceCenter problems for the device is displayed (Figure 4-9).

ServiceCenter displays an appropriate message if there are no open problems for the selected device.

1. Double-click the desired problem in the list. The Problem Detail window is displayed (Figure 4-4).
2. Click the **Close** button.
3. Click the **Save** or **OK** button to save your changes and close the problem ticket.

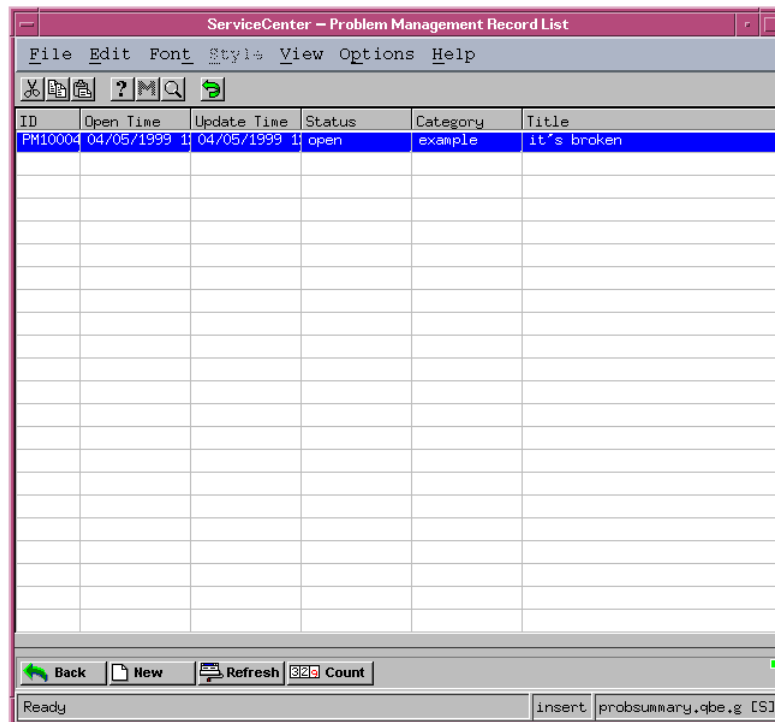


Figure 4-9 Problem Management Record list

Probable Cause

The Probable Cause menu option allows you to query ServiceCenter for the probable cause of a problem. When first accessed, a blank probable cause window appears. If you press **Enter**, a list of cause codes appears. You can select one of the listed probable causes by double-clicking on it, which will display the probable cause window (Figure 4-10). The **Resolution** field lists any solution that has been determined for the problem.

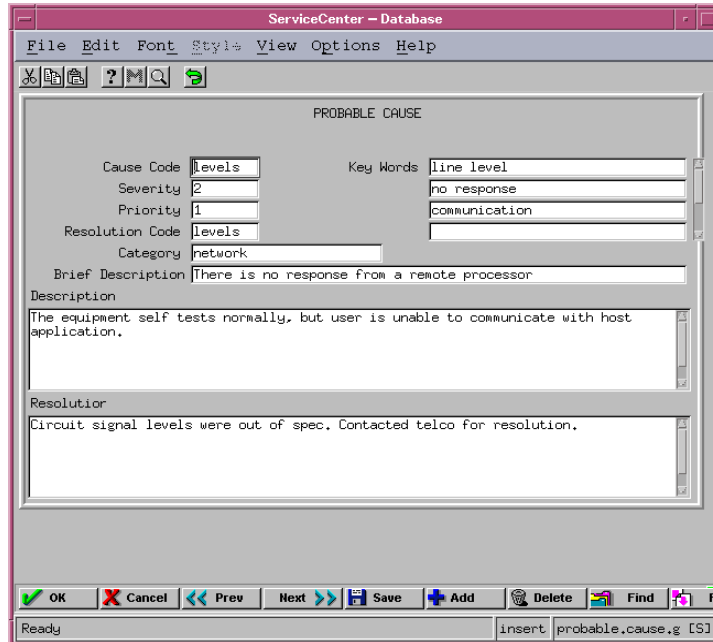


Figure 4-10 Probable Cause

Service Information

The Service Information menu option takes you to the Inventory and Configuration Management window (Figure 4-11).

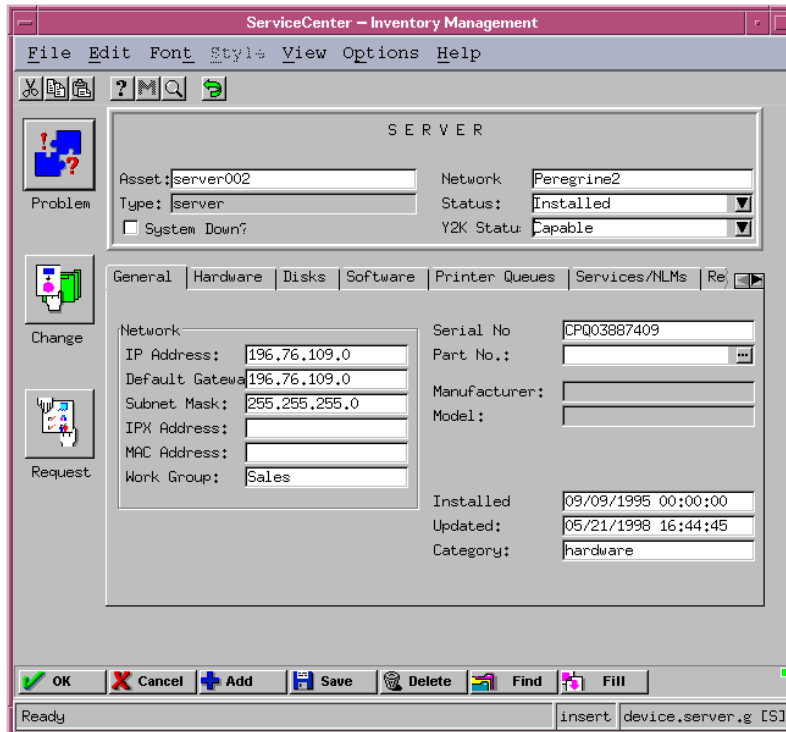


Figure 4-11 Inventory and Configuration Management

Down Time

The Down Time menu option displays the Down Time window for the selected object (Figure 4-12).

The screenshot shows a window titled "ServiceCenter - Database" with a menu bar (File, Edit, Font, Style, View, Options, Help) and a toolbar with icons for Back, Save, Delete, Add, Find, and Fill. The main content area is titled "DOWNTIME" and contains the following fields:

Logical Name	Location	Contact Name	Type	Table Name
7500e0.peregrine.com	Del Mar		node	DEFAULT

Below these fields is the "Outage Totals" section:

Last Reset	Explicit	Implicit	Perceived	Count
	01:52:47	01:52:47	01:52:47	1

The "Details" section contains a table with the following data:

Start Time	End Time	Type	Explicit	Implicit	Perceived	Problem No.
10/09/97 16:44	10/09/97 18:37	X	01:52:47	01:52:47	01:52:47	PM1018

At the bottom of the window, there is a status bar that says "You have mail waiting." and a button labeled "insert" next to the file name "downtime.graph.g".

Figure 4-12 Down Time window

Assigned Problems

The Assigned Problems menu option provides a list of open problems assigned to the operator using the current IT/O session. This summary is displayed in the Open Problems window (Figure 4-13).

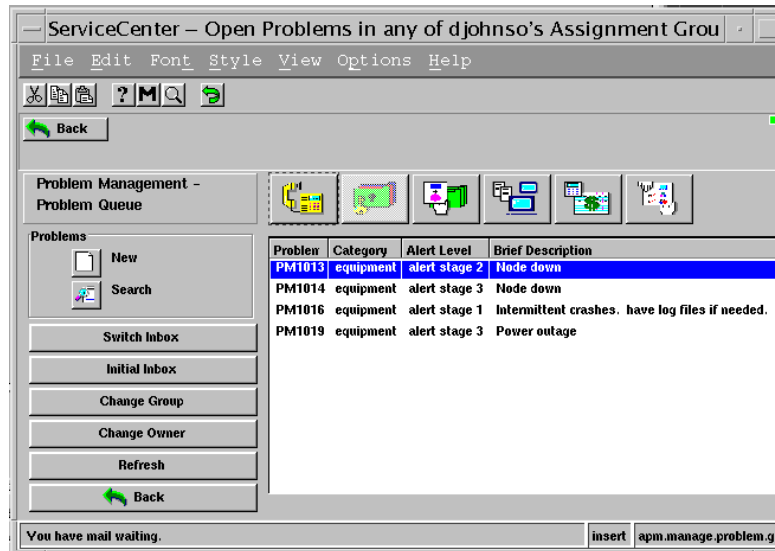


Figure 4-13 Open Problems

Other Services

The Other Services menu option provides a submenu with additional options.

Message Integration

The Message Integration menu option (Figure 4-14) provides a graphical way to start, stop, and status the SCAuto for VP Operations processes.

- To check the status of the SCAuto for VP Operations processes, select the **Status** menu option.
- To start the SCAuto for VP Operations processes, select the **Start** menu option.
- To stop the SCAuto for VP Operations processes, select the **Stop** menu option.

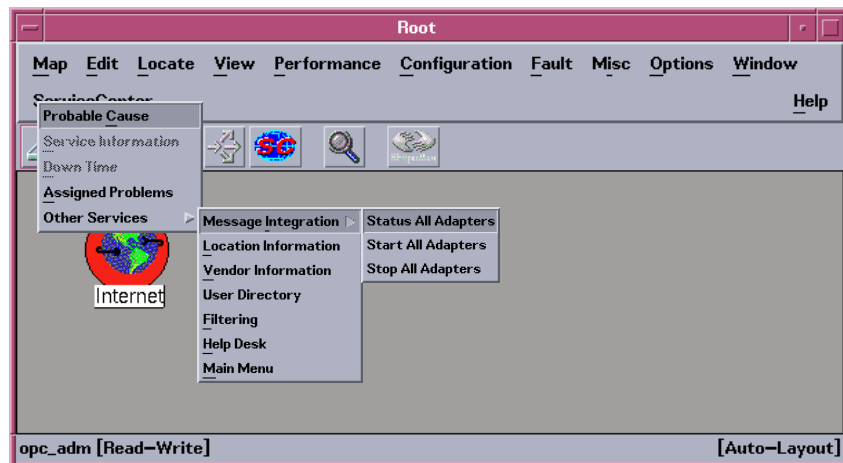


Figure 4-14 Message Integration menu options

Location Information

The Location Information menu option provides location records, much like an address book. When the Location window (Figure 4-15) is first accessed, the window is blank.

- To find location data, enter the **location name** and click the **Search** button.
- To view a list of locations, press **Enter** while in the blank window. A summary list is displayed.
- Double-click the desired location to see the data for that location.

The location window will also allow you to search, edit, add, and update location information.

The screenshot shows a window titled "ServiceCenter - Database" with a menu bar (File, Edit, Font, Style, View, Options, Help) and a toolbar. The main area is divided into two sections. The top section has three input fields: "Location Name:" with "San Mateo site", "Location:" with "San Mateo site", and "Location Code:" with "ar1". Below this is a tabbed interface with "General" and "Floor Plan" tabs. The "General" tab contains several fields: "Address:" with "5569 Turner Dr. fds", "Primary Contact:" with "JORDAN", "Department:" with "marketing", "Phone:" with "415-648-9798", "FAX:" with "415-648-9743", "Email:" with "sjordan@peregrine.com", "City/State/Zip" with "Santa Clara AR 95050", "Country" with "United States", and "Hours:" with "09:00 to 05:00". There is also a "Comments:" text area. At the bottom is a toolbar with buttons for OK, Cancel, Prev, Next, Save, Add, Delete, Find, and Print. The status bar shows "Ready" and "insert location.g [S]".

Figure 4-15 Location window

Vendor Information

The Vendor Information menu option takes you to the Vendors window (Figure 4-16). When Vendor Information is first accessed, a blank Vendors window is displayed.

- Press **Enter** to display a vendor summary list.
- Double-click on the desired vendor to access the data for that vendor. The Vendors window is displayed with the ServiceCenter information for that vendor. These fields can be edited and updated.

The screenshot shows a window titled "ServiceCenter - Database" with a menu bar (File, Edit, Font, Style, View, Options, Help) and a toolbar. The main area is titled "VENDOR FILE" and contains the following fields:

Vendor ID:	rp2	HOT LINE:	415-555-6400 x911
Vendor:	hewlett packard	Contract No.:	43875884
Address:	345 market st	Contract Person:	Ben Elton
	san francisco ca 90345	Phone:	415-555-6400 x101
Country:	USA	Sales Office:	
Phone:	415-555-6400	Sales Manager:	steve joyce
FAX:		Phone:	415 - 555 - 6400
Email:		Sales Rep.:	Frank laird
Order Contact:		Phone:	415-555-6400 x282
Phone:		Sales Hours:	09:00:00 to 17:00:00
Services:		Afterhours Contact:	
Technician:	elaine guess	Phone:	
Phone:	415-555-6400 x200	Manager:	greg robertson
Beeper:	415-555-4563	Phone:	415-555-6400
Hours:	08:00:00 to 17:00:00		
Escalation Procedures:			

The bottom of the window features a toolbar with buttons for OK, Cancel, Prev, Next, Save, Add, Delete, and Find. The status bar at the bottom shows "Ready" and "insert vendor.g [S]".

Figure 4-16 Vendors window

User Directory

The User Directory menu option allows you to add or update ServiceCenter contacts. When User Directory is first accessed, a blank ServiceCenter contact window appears. To query for user information, enter known data in the appropriate field and click the **Search** button.

- To get a user list, press **Enter** after the blank window appears. A contact list is displayed.
- Double-click on the desired user to get the User Directory information for that user.

Filtering

The Filtering menu option takes you to the Event Services Filters window (Figure 4-17). In this window, you can set ServiceCenter and SCAuto event filters, or query for an existing filter. All fields in the setup window are optional, therefore you can either set one field, all fields, or a combination of fields. Multiple filters can be set to seek problems under different conditions.

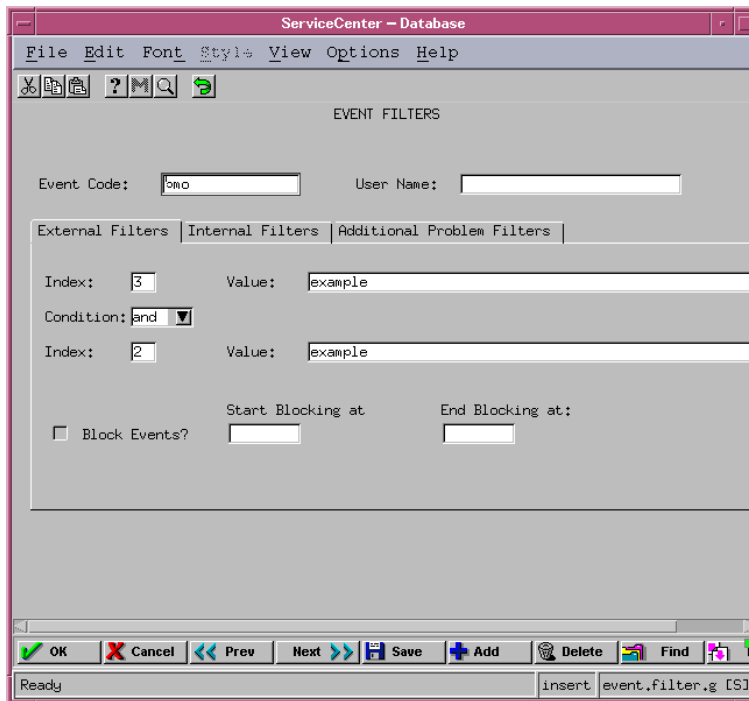


Figure 4-17 Filtering window

Help Desk

The *Help Desk* menu option takes you to a problem summary window for the current IT/O operator (Figure 4-18). For ServiceCenter 1.4, this takes you to the main Help Desk window.

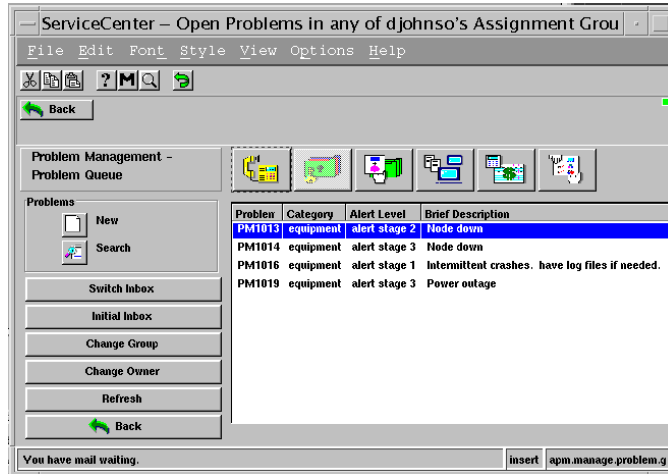


Figure 4-18 Problem Queue

Main Menu

The Main Menu option takes you to the ServiceCenter home menu (Figure 4-19). From here, you can access all ServiceCenter functions.

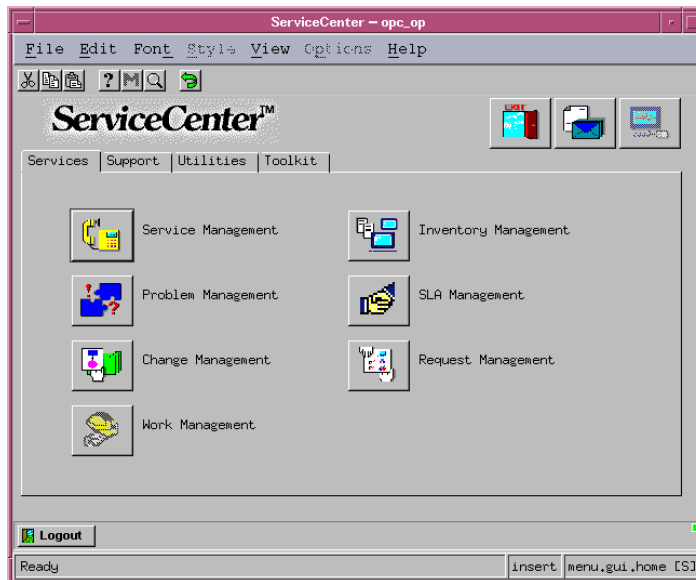


Figure 4-19 ServiceCenter Home Menu

Event Integration

SCAuto for VP Operations Event Integration is a collection of tools or components that is highly customizable for integrating event messages from IT/O to cause a ServiceCenter problem ticket to be created, updated, or closed.

Event Message integration provides these capabilities:

- Ability to tap into the Message Stream Interface (MSI) for all new event messages based on event registration conditions and to create new ServiceCenter problem tickets.
- Ability to tap into the Message Event Interface (MEI) for modifications done to existing event messages based on event registration conditions, and to update or close existing ServiceCenter problem tickets.
- Provide a trouble ticketing interface from which configured IT/O events may create ServiceCenter problem tickets.
- Ability to create new IT/O event messages from ServiceCenter *eventout* output events.
- Ability to modify existing IT/O event messages based on modification being done to related problem tickets in ServiceCenter using the *eventout* output events.
- Ability to correlate ServiceCenter problem tickets to one or more IT/O event messages.
- Ability to correlate IT/O event messages to a ServiceCenter problem ticket.

Integration Components

File Name	File Type	File Description
scevmon	executable for manipulating queue files to and from ServiceCenter.	This is the core technology of the SC Automate product that facilitates bi-directional queueing of messages to and from ServiceCenter.
sctoito	executable for reading the <i>from</i> queue file and forwarding to IT/O	This executable functions to take an event forwarded from ServiceCenter eventout and generate/modify IT/O event messages using a series of customizable TCL8.0 scripts. 80% of the IT/O API library has been converted into TCL callable functions accessible by these scripts. This executable runs in the background waiting for events. The customer can do more than is provided by the IT/O APIs, such as remote controlled processes.
scfromitoTTI	executable to interface with IT/O Trouble Ticket Interface	Basically, IT/O needs to customize interested events to forward to Trouble Ticketing first. Trouble Ticketing Interface will then call a script and pass as arguments static data describing the event. We are only interested in the first argument, which is the message ID. This executable is called once via a shell script indicated in the Trouble Ticketing custom dialog for every intended event.
scfromitoMSI	executable for writing the <i>to</i> queue from IT/O Message Stream Interface	This executable will register events for the Message Stream Interface and run in the background waiting for event notification. Upon notification, it will use event maps and generate the appropriate SC events to the <i>to</i> queue. The Message Stream Interface will notify only new events in IT/O.
scfromitoMEI	executable for writing the <i>to</i> queue from IT/O Message Event Interface	This executable will register events for the Message Event Interface and run in the background waiting for event notification. Upon notification, it will use event maps and generate the appropriate SC events to the <i>to</i> queue. The Message Event Interface will only notify about changes done to existing events in IT/O.
various map files/TCL scripts	configuration text files	TCL scripts that are customizable for creating/modifying IT/O events from ServiceCenter events as well as from IT/O to ServiceCenter. Also, there are <i>ini</i> config files that need to be set up to facilitate scevmon and the locations of these files.

scevmon

This is the core of the SCAuto SDK V3. It is best described as an event message processor using *to* (ServiceCenter) and *from* (ServiceCenter) queue files to cache events generated by the other components. The basis of this design is to facilitate a fault recovery system that enables IT/O event messages to be queued up even while ServiceCenter is currently not available, as well as queueing up ServiceCenter events while waiting for IT/O to be available. The contents of the queue files include the actual SC events in “^” delimited format preceded by a header. These files should not be modified by hand.

scevmon typically runs in the background, makes a TCP connection to the SCAuto Server, and waits for events to appear in the queue files. Once any of the other integration components writes an event string to the *to* queue file, scevmon picks it up, marks it as processed and forwards it to ServiceCenter's *eventin* file. On the other hand, if a problem ticket is modified and an entry gets created in the *eventout* file in ServiceCenter, scevmon picks it up, updates the sync file to point to the last entry in the eventout file, and writes the event string into the *from* queue file. At this point, the sctoito integration component picks it up, marks it as processed, and forwards it through the TCL mapping scripts to IT/O.

The config files used by scevmon are:

- `/etc/scauto/SCITO.init` - a file that contains a variable pointing to the installed directory of the product.
- `<installation directory>/scito.ini`

scevmon reads these files:

- `$SCITOHOME/scevents.to.<host>.<port>` - contains SC eventin events created by scfromito[MSI | MEI | TTI] programs. This file is updated to show processed events.
- `$SCITOHOME/syncfile.<host>.<port>` - syncfile for the *from* queue with ServiceCenter eventout file.

scevmon writes these files:

- `$SCITOHOME/scito.log` - log file for informational and error messages.
- `$SCITOHOME/scevents.from.<host>.<port>` - contains SC eventout events to be passed on to IT/O. This file is read in by the sctoito program.
- `$SCITOHOME/syncfile.<host>.<port>` - syncfile for the *from* queue with ServiceCenter eventout file.

sctoito

This is the background process that waits on the *from* queue, and reads in any new incoming events from ServiceCenter eventout file. It processes the event by passing it to a configurable TCL script and marks the event as *processed*.

This program is customized by <installation directory>/EventMap/FromSC/event.ini and will execute the TCL scripts pointed to by the event type names in the [EVENT] section. The [CONFIG] section is a place to specify the IT/O username and passwords to connect with. Please refer to Chapter 5, “Configuration,” for more information about customizing the component.

The following are the components in SCAuto for VP Operations that reads IT/O messages using the Message Stream Interface or through direct IT/O database APIs. They are customized by event.ini for parameters to register with the Message Stream Interface, as well as TCL scripts to execute for mapping the events to ServiceCenter event strings.

scfromitoTTI

This is the Trouble Ticket Interface adapter that can be executed using a shell script. It is the program defined for the IT/O Trouble Ticketing Interface. The event source templates are customized to forward events to this application when it is running.

scfromitoMSI

This is the background process that writes to the *from* queue when registered IT/O events appear in the Message Stream Interface.

scfromitoMEI

This is the background process that writes to the *from* queue when registered IT/O events appear in the Message Event Interface.

These programs are customized by \$SCITOHOM/EventMap/ToSC/event.ini.

Chapter 5 Configuration



Introduction

This chapter describes how to configure SCAuto for VP Operations. It covers three main topics:

- SCAuto for VP Operations Business Logic Configuration
- VP Operations Business Logic Configuration
- ServiceCenter Business Logic Configuration

SCAuto for VP Operations Business Logic Configuration

This section discusses TCL language event mapping from VP Operations to ServiceCenter. This scripted approach to configuring the SCAuto for VP Operations product results in a flexible method of configuring or customizing it to meet specific business goals and requirements.

The mapping mechanism to map IT/O message events into ServiceCenter problem tickets is made up of an *active* facility, which is a TCL 8.0 script, acting on variables populated with IT/O message elements to create a ServiceCenter TCL event object, using a *passive* facility, a static map file positionally defining each kind of interested ServiceCenter event. The **send** command of this ServiceCenter TCL object is then invoked to queue the event to be sent to ServiceCenter. In addition, the messages being handed to these facilities are themselves configurable to be filtered using OPCREG parameters at the interface program level using an *event.ini* configuration file.

Configuration Overview

By default, the mapping files reside in the *EventMap* directory in the product installation directory. The hierarchy and explanation of these files follows:

- EventMap
- **EventMap/FromSC/** -- files to configure events from ServiceCenter to VP Operations.
- **EventMap/ToSC/** -- files to configure events from VP Operations to ServiceCenter

We will be dealing with EventMap/ToSC directory in this subheading. By default, there are seven files in this directory:

event.ini Configuration file to specify MSI and MEI registration parameters to filter messages as well as which TCL map script to invoke by which application subcomponent. A detailed description will appear under the heading IT/O Message Filtering The *event.ini* file.

eventmapMSI.tcl

The event map script for the MSI interface (**scfromitoMSI**)

eventmapMEI.tcl

The event map script for the MEI interface (**scfromitoMEI**)

eventmapTTI.tcl

The event map script for the TTI interface (**scfromitoTTI**)

pmo.map, pmu.map, pmc.map

Static map file templates used in the scripts to positionally define ServiceCenter event elements in the ServiceCenter TCL event object. See “Static Map File” on page 9 for more details.

Basically, the *event.ini* file configures what the MSI and MEI interfaces will get from IT/O and which TCL scripts to invoke when there is an IT/O message. Then, the TCL scripts will use the static map files as a template to create ServiceCenter TCL event objects, populate its members, and finally send it off to the queue file to be sent to ServiceCenter *eventin* file.

IT/O Variables

The available variables are the same as the attribute names of the OPCDATA_MESSAGE. This definition can be found in Chapter 4 of the *HP OpenView Developer's Toolkit Application Integration* guide. These variables are made available as TCL_GLOBAL and will be accessible via the \$VARIABLE syntax globally from within the called TCL script.

Table 5-1. IT/O Variables

Variable Name	Short Definition
OPCDATA_MSGID	The unique IT/O message ID.
OPCDATA_NODENAME	Name of the node producing the message. The message is only handled by the IT/O manager if this system is part of the IT/O Node Bank.
OPCDATA_CREATION_TIME	(Local) Time the message was created. The time is in UNIX format (seconds since epoch).
OPCDATA_RECEIVE_TIME	Time the message was received by the management server.
OPCDATA_MSGTYPE	Message type. This attribute is used to group messages into subgroups, e.g., to denote the occurrence of a specific problem. This information may be used by event correlation engines.
OPCDATA_GROUP	Message group.
OPCDATA_OBJECT	Object name to use for the IT/O message.
OPCDATA_APPLICATION	Application which produced the message.

Table 5-1. IT/O Variables

OPCDATA_SEVERITY	Severity of the message. Possible values are: 0 - OPC_SEV_UNCHANGED 4 - OPC_SEV_UNKNOWN 8 - OPC_SEV_NORMAL 16 - OPC_SEV_WARNING 32 - OPC_SEV_CRITICAL 64 - OPC_SEV_MINOR 128 - OPC_SEV_MAJOR
OPCDATA_AACTION_NODE	Defines the node on which the automatic action should run.
OPCDATA_AACTION_CALL	Command to use as automatic action for the IT/O message.
OPCDATA_AACTION_ANNOTATE	Defines whether IT/O creates start and end annotations for the automatic action. 0 - do not create annotations 1 - create annotations
OPCDATA_AACTION_ACK	Auto Acknowledge after successful execution of the Automatic Action. 0 - do not auto-acknowledge 1 - auto-acknowledge
OPCDATA_OPACTION_NODE	Defines the node on which the operator initiated action should run.
OPCDATA_OPACTION_CALL	Command to use as operator-initiated action for the IT/O message.
OPCDATA_OPACTION_ANNOTATE	Define whether IT/O creates start and end annotations for the operator initiated action. 0 - do not create annotations 1 - create annotations
OPCDATA_OPACTION_ACK	Auto Acknowledge after successful execution of the operator initiated action. 0 - do not auto-acknowledge 1 - auto-acknowledge
OPCDATA_MSG_LOG_ONLY	Message is Server Log Only.

Table 5-1. IT/O Variables

OPCDATA_UNMATCHED	<p>Defines whether or not the message matches a condition.</p> <p>0 - the message was sent to the server because it matched a match condition</p> <p>1 - the message did not match a match condition of the assigned templates, but was forwarded nevertheless</p>
OPCDATA_TROUBLETICKET	Forward message to Trouble Ticket System.
OPCDATA_TROUBLETICKET_ACK	Acknowledge message after forwarding it to the Trouble Ticket System
OPCDATA_NOTIFICATION	Notification
OPCDATA_INSTR_IF_TYPE	<p>Type of the instruction interface</p> <p>0 - OPC_INSTR_NOT_SET</p> <p>1 - OPC_FROM_OPC</p> <p>2 - OPC_FROM_OTHER</p> <p>3 - OPC_FROM_INTERNAL</p>
OPCDATA_INSTR_IF	Name of the external instruction text interface. The external instruction text interface must be configured in IT/O.
OPCDATA_INSTR_PAR	Parameters for a call to the external instruction text interface.
OPCDATA_MSGSRC	Message Source. For example, the name of the encapsulated log file if the message originated from log file encapsulation or the interface name if the message was sent via an instance of the Message Stream Interface.
OPCDATA_MSGTEXT	Message Text.
OPCDATA_ORIGMSGTEXT	Original Message Text. Allows you to set additional source information for a message. It is only useful if the message text was reformatted but the IT/O operator needs to have access to the original text as it appeared before formatting.
OPCDATA_ANNOTATIONS	This is not a IT/O data element. It is created by the adapter product to contain a text string of all the annotations of the message.
OPCDATA_LAST_ANNOTATION	This is not a IT/O data element. It is created by the adapter product to contain a text string of the last annotation of the message.

There are also three non-OPC variables added for programmability:

Table 5-2. Non-OPC Variables

Variable Name	Short Definition
INSTALLDIR	The home directory where the product was installed.
SC_MSGTYPE	<p>A converted message type string denoting which type of message it is. Valid values are:</p> <ul style="list-style-type: none"> • New message • Message Owned by a user • Message Disowned by a user • Message Acknowledged • Message has new annotation(s) • All annotations of Message deleted • Message was escalated • Message was escalated from another server • Automatic action of message started • Automatic action of message finished • Operator initiated action of message started • Operator initiated action of message finished
SC_PROBLEM_NUMBER	The ServiceCenter Problem Number embedded in the IT/O Message annotation text when its a new message, or the Object of message Group ServiceCenter when its a ServiceCenter created message.

To check for the IT/O Group of OS only to create a new ticket in *eventmapsMSI.tcl*, do the following:

```

.....
if { $OPCDATA_GROUP != "OS" } { return }
.....

```

This statement in the beginning of the script returns from the script without creating and queueing a ServiceCenter event to be sent.

ServiceCenter TCL Event Object

LEGEND:

- <> required parameter
- [] optional parameter

Summary of ServiceCenter TCL commands

- **create_sc_event** <object name>
- <object name> **set_evtype** <event type> [**use_template** <template name>]
- <object name> **set_evfield** <0...80>|<field name> <field value>
- <object name> **print**
- <object name> **send**

create_sc_event

```
create_sc_event <ObjectName>
```

The resultant object from this command will have methods to populate its data members and a **send** method to queue the event to ServiceCenter. The methods are invoked by using the newly created object name itself and therefore will have its own context. There is a limit of 10 objects that can be created in each script.

```
create_sc_event eventObject
```

The object named **eventObject** is created and can be used to set values and finally *send* it to ServiceCenter.

set_evtype

```
<ObjectName> set_evtype <evtype name> [use_template <template name>]
```

This command sets the **evtype** field of the event to <evtype name> (e.g., pmo, pmu pmc etc.). There are two optional parameters that can be specified for the event to use a Static Map File for setting the event values:

- **use_template**
- <template name>

If these optional parameters are not specified, setting of object fields can still be done using integer indexes., as shown in the following examples:

```
create_sc_event eventObject
# create the event object
eventObject set_evtype pmo use_template "EventMap/ToSC/pmo.map"
# make it of type pmo, and use the template for
# named indexes.
```

The event object **eventObject** is created. It is then set to type *pmo*, essentially, the **evtype** field of the event is set to *pmo*. The static map file

EventMap/ToSC/pmo.map is used to define the named indexes that will be used later to set the other fields.

```
create_sc_event eventObject
# create the event object
eventObject set_evtype pmo
# set the evtype to pmo. do not use templates for indexing
```

The event object **eventObject** is created. It is then set to *pmo* for **evtype**. Because a template was not used here, subsequent setting of **evfields** will have to use integer indexes.

set_evfield

```
<ObjectName> set_evfield <0 ... 80>|<evfield name> <value>
```

This command sets the event fields with values. If the **use_template** option was used during the **set_evtype** command, then you may use field names defined in the static map file templates to set the values, otherwise, you have to use positional integer indexes to set the **evfield** values. See the following examples:

```
create_sc_event eObject
# create the event object
eObject set_evtype pmo
# set the evtype to "pmo". do not use templates for indexing
eObject set_evfield 1 $OPCDATA_NODENAME
# set evfield position 1 to ITO nodename variable
```

The event object is created, set to *evtype* “*pmo*” without using a static map file for template. The first field of the event object is set to the IT/O variable for **nodename**.

```
create_sc_event eObject
# create the event object
eObject set_evtype pmo use_template "EventMap/ToSC/pmo.map"
# set the evtype
# use a templates for indexing
eObject set_evfield logical.name $OPCDATA_NODENAME
# set logical.name evfield to ITO nodename variable
```

The event object is created, set to *evtype* “*pmo*” using a static map file for template. The **logical.name** field of the event object is set to the IT/O variable for **nodename**.

print

```
<objectName> print
```

This command outputs to **stderr** all the contents of the object including the template field names if used.

send

```
<objectName> send
```

This command queues the event object to be sent to ServiceCenter's *eventin* file.

For more examples on how to use the commands, check out the example script file.

Static Map File

The static map file that is usable in a **set_evtype** command as the template is an ascii text file. Each row denotes a positional field name of the intended ServiceCenter event map starting at index 1, for example, row 1 = field 1 in event map.

By default, three maps for event types *pmo*, *pmu* and *pmc* are delivered. These maps match the ServiceCenter event maps for version 2.1SP3.

Example *pmo.map* used in **eventmapMSI.tcl**.

```
----- begin file -----
logical.name
network.name
reference.no
cause.code
action
action2
action3
network.address
type
category
domain
objid
version
model
serial.no
vendor
location
contact.name
contact.phone
resolution
assignee.name
priority.code
failing.component
system
----- end file -----
```

The map template file is not a required piece of the configuration; it is used to make it convenient and manageable to name the event fields in the ServiceCenter TCL event object. Without this file, however, you can still set the values using integer indexes.

IT/O Message Filtering the *event.ini* File

The *event.ini* file in the `<install dir>/EventMap/ToSC` directory is a means of configuring the **scfromitoMSI**, **scfromitoMEI** and **scfromitoTTI** components of the interface.

In the *event.ini* file, you can specify:

- How you want to filter MSI event messages
- How you want to filter MEI event messages
- Which TCL map files to execute when a message of type MSI, MEI and/or TTI is available

There are two main structures in the *ini* file, and this file closely follows the format of a Microsoft Windows ini file format.

Sections

Section header names are identified by enclosing them in square brackets [...]. The brackets group sets of configurable parameters. There are three required section header names:

- **MSI_CONFIG** Section for configuring the MSI interface
- **MEI_CONFIG** Section for configuring the MEI interface
- **TTI_CONFIG** Section for configuring the TTI interface

Each section header name contains its own set of required parameters.

Table 5-3. IT/O Sections and Parameters

Section Name	Required Parameters
MSI_CONFIG	<ol style="list-style-type: none"> 1. OPC_USER - This is the IT/O user the MSI interface will connect as to retrieve message details. 2. OPC_PASSWORD - This is the password for the OPC_USER. Optionally, you may specify the string literal USE_ETC_PASSWD in this field. This tells the API to access the equivalent UNIX user account for the password. This brings in the requirement to have a UNIX user of the same username and password as the ITO user you want to access the ITO database, but hides the super user password using standard UNIX user accounting. 3. OPC_USER.MSI_OPCREG_MSGTYPE - This is an MSI registration attribute to filter on which IT/O message type to notify. This is a String value of maximum 36 characters, no spaces. Consult your IT/O Message Source Templates for available message types. If you specify the keyword "ALL" in this field, all messages copying/diverting to the Message Stream Interface will be notified. 4. MSI_OPCREG_GROUP - This is an MSI registration attribute to filter on which IT/O message group to notify. This is a String value of maximum 32 characters, no spaces. Consult your IT/O Message Source Templates for available message groups. 5. MSI_OPCREG_NODENAME - This is an MSI registration attribute to filter on which IT/O node name to notify. This is a String value of maximum 254 characters, no spaces. 6. MSI_OPCREG_OBJECT - This is an MSI registration attribute to filter on which IT/O message object to notify. This is a String value of maximum 32 characters, no spaces. Consult your IT/O Message Source Templates for available message objects. 7. MSI_OPCREG_SEVERITY - This is an MSI registration attribute to filter on which IT/O message severity to notify. This is a String value of 1 or more characters to denote ORing of values. Valid characters are: <ol style="list-style-type: none"> a. "c" - critical b. "w" - warning c. "n" - normal d. "u" - unknown e. "j" - major f. "m" - minor g. "cwj" - critical or warning or major. This is the only attribute you don't need the ' ' to denote ORing.

Table 5-3. IT/O Sections and Parameters

MSI_CONFIG	<p>8. MSI_OPCREG_APPLICATION - This is an MSI registration attribute to filter on which IT/O message application to notify. This is a String value of maximum 32 characters, no spaces. Consult your IT/O Message Source Templates for available message applications.</p> <p>9. POLL_INTERVAL</p> <ul style="list-style-type: none">- 0 means blocking read for opcif_read call.- # means number of seconds to sleep before calling non-blocking opcif_read. <p>10. MSI_EVENTMAPS - This parameter specifies which section header to fall into, to look for the attribute "mapname" for specifying which TCL script to execute for this interface. Currently, only the first "mapname" attribute is taken.</p> <p>Note: Values with more than one attribute implicitly assume a logical ANDing of all non-null attributes.</p> <p>Note: Within each individual attribute, use the ' ' character to logically OR more than one value.</p>
------------	--

Table 5-3. IT/O Sections and Parameters

<p>MEI_CONFIG</p>	<ol style="list-style-type: none"> 1. OPC_USER - This is the IT/O user the MEI interface will connect as to retrieve message details. 2. OPC_PASSWORD - This is the password for the OPC_USER. Optionally, you may specify the string literal USE_ETC_PASSWD in this field. This tells the API to access the equivalent UNIX user account for the password. This brings in the requirement to have a UNIX user of the same username and password as the ITO user you want to access the ITO database, but hides the super user password using standard UNIX user accounting. 3. MEI_OPCREG_MSG_EVENT_MASK - This is an IT/O Message Event Interface event mask to specify which type of change-in-message should be notified. Valid values are: <ol style="list-style-type: none"> a. OPC_MSG_EVENT_OWN - when a message is owned b. OPC_MSG_EVENT_DISOWN - when a message is disowned c. OPC_MSG_EVENT_UNACK - when a message is unacknowledged d. OPC_MSG_EVENT_ACK - when a message is acknowledged e. OPC_MSG_EVENT_ANN - when a message is annotated f. OPC_MSG_EVENT_NO_ANN - when there are no annotations 4. POLL_INTERVAL: <ul style="list-style-type: none"> - 0 means blocking read for opcif_read call - # means seconds to sleep before calling non-blocking opcif_read 5. MEI_EVENTMAPS - This parameter specifies which section header to fall into, to look for the attribute "mapname" for specifying which TCL script to execute for this interface. Currently, only the first "mapname" attribute is taken. <p>Note: For MEI_OPCREG_MSG_EVENT_MASK, use the ' ' character to denote logical OR'ing of event types.</p>
<p>TTI_CONFIG</p>	<ol style="list-style-type: none"> 1. OPC_USER - This is the IT/O user the MEI interface will connect as to retrieve message details. 2. OPC_PASSWORD - This is the password for the OPC_USER. Optionally, you may specify the string literal USE_ETC_PASSWD in this field. This tells the API to access the equivalent UNIX user account for the password. This brings in the requirement to have a UNIX user of the same username and password as the ITO user you want to access the ITO database, but hides the super user password using standard UNIX user accounting. 3. TTI_EVENTMAPS - This parameter specifies which section header to fall into, to look for the attribute "mapname" for specifying which TCL script to execute for this interface. Currently, only the first "mapname" attribute is taken.

Example event.ini file:

```
----- begin file -----
[ MSI_CONFIG ]
OPC_USER=opc_adm
OPC_PASSWORD=OpC_adm
MSI_OPCCREG_MSGTYPE=ALL
MSI_OPCCREG_GROUP=
MSI_OPCCREG_NODENAME=
MSI_OPCCREG_OBJECT=
MSI_OPCCREG_SEVERITY=
MSI_OPCCREG_APPLICATION=
POLL_INTERVAL=1
MSI_EVENTMAPS=MSI_EVENTMAPS

[ MEI_CONFIG ]
OPC_USER=opc_adm
OPC_PASSWORD=OpC_adm
MEI_OPCCREG_MSG_EVENT_MASK=OPC_MSG_EVENT_OWN|OPC_MSG_EVENT_DISOWN|OPC_MSG_EV
ENT_UNACK|OPC_MSG_EVENT_ACK|OPC_MSG_EVENT_ANNO|OPC_MSG_EVENT_NO_ANNO
POLL_INTERVAL=1
MEI_EVENTMAPS=MEI_EVENTMAPS

[ TTI_CONFIG ]
OPC_USER=opc_adm
OPC_PASSWORD=OpC_adm
TTI_EVENTMAPS=TTI_EVENTMAPS

[ MSI_EVENTMAPS ]
mapname=EventMap/ToSC/eventmapMSI.tcl

[ MEI_EVENTMAPS ]
mapname=EventMap/ToSC/eventmapMEI.tcl

[ TTI_EVENTMAPS ]
mapname=EventMap/ToSC/eventmapTTI.tcl
----- end file -----
```

Using TTI - Trouble Ticketing Interface

To use the TTI:

1. Identify which message source you want to forward to the TTI in the Message Source Template and enable it.
2. In the Node Bank submap *ITO*, select the following menu commands, in order: **Actions->Utilities->Trouble Ticket**.
3. Enable the interface by clicking on the check box.
4. Enter in the **Call of trouble ticket** field <install dir>/TTI.sh.
<install dir> is the SCAuto product install directory.

Default Behavior

The **scfromitoMSI** process picks up all events that copy or divert to the Message Stream Interface. If the host name of the occurring event is different, ServiceCenter creates a new problem ticket. If the host name of the occurring event is the same, the same ticket is updated by default, and a new ServiceCenter IT/O event is created with a ServiceCenter Group attribute. You can use the ServiceCenter Message Group in the IT/O Message Group Bank to filter the messages.

scfromitoMEI only picks up events that were previously opened as ServiceCenter problem tickets or ServiceCenter event group events *if the message for the problem or event has been modified by the IT/O message browser*. This modification can consist of an operator owning, acknowledging, or annotating a ServiceCenter event type or a IT/O event that caused a problem ticket. When this type of modification occurs, a *pmu* is generated to modify the same problem ticket in ServiceCenter.

Example eventmapMSI.tcl Script

```
----- begin file -----
[comments removed]

# setup log file
set debugFile [open $INSTALLDIR./scito.log a]

# create only if its a new message from OPC
if { $SC_MSGTYPE != "New message" } {
    puts $debugFile "eventmapMSI.tcl: skipping event - $SC_MSGTYPE"
    close $debugFile
    return
}

# skip ServiceCenter events
if { $OPCDATA_GROUP == "ServiceCenter" } {
    puts $debugFile "eventmapMSI.tcl: skipping ServiceCenter event -
$SC_MSGTYPE"
    close $debugFile
    return
}

# create the event object
create_sc_event eventObject

# set the event type using a template to define field names
# * if you don't use a template, you can use integer
# * indexes into the evfield array.
eventObject set_evtype pmo use_template "EventMap/ToSC/pmo.map"

# start mapping field names to ITO values
eventObject set_evfield logical.name $OPCDATA_NODENAME
```

```

eventObject set_evfield network.name $OPCDATA_NODENAME
eventObject set_evfield reference.no $OPCDATA_MSGID
eventObject set_evfield cause.code $OPCDATA_MSGSRC
eventObject set_evfield action [ format "%s %s (Original message: %s) %s"
$OPCDATA_CREATION_TIME          $OPCDATA_MSGTEXT          $OPCDATA_ORIGMSGTEXT
$OPCDATA_LAST_ANNOTATION ]
eventObject set_evfield network.address $OPCDATA_NODENAME
eventObject set_evfield type ITOEvent
eventObject set_evfield category example
eventObject set_evfield priority.code $OPCDATA_SEVERITY
eventObject set_evfield failing.component [ concat $OPCDATA_GROUP ","
$OPCDATA_OBJECT "," $OPCDATA_APPLICATION ]
eventObject set_evfield system $OPCDATA_MSGSRC

# print out a debug of event created
#eventObject print

# send the event to queue
puts $debugFile "eventmapMSI.tcl: queueing new ticket to be opened."
eventObject send

close $debugFile
----- end file -----

```

TCL Event Mapping from ServiceCenter to VP Operations

The mapping mechanism to map ServiceCenter events into IT/O messages is done by invoking a TCL 8.0 script. ServiceCenter event fields are made into **TCL GLOBAL** variables that can be accessed in the targeted script with the \$VARIABLE syntax. Various IT/O programming APIs are also made into **TCL Command** commands callable from within the same script, thus effectively bridging the two domains.

Configuration Overview

By default, the mapping files reside in an EventMap directory within the product installation directory. The hierarchy and function of the EventMap directory structure follows:

- EventMap
- EventMap/FromSC/ - files to configure events from ServiceCenter to VP Operations.
- EventMap/ToSC/ - files to configure events from VP Operations to ServiceCenter

The following files are located in the EventMap/FromSC/ directory:

- event.ini** The configuration file that specifies IT/O user and password, and TCL scripts to invoke based on event types.
- pmo.tcl** The event map script to invoke when there is a pmo (Problem Opened).
- pmu.tc** The event map script to invoke when there is a pmu (Problem Updated).
- pmc.tcl** The event map script to invoke when there is a pmc (Problem Closed).
- util.tcl** Utility TCL script that contains command to convert ServiceCenter priority codes to IT/O Severity values.
- default.tcl** A default TCL script to invoke when an event type is received and there is no equivalent TCL script.
- showEvent.tcl** A debug script that will display all event field values from a ServiceCenter event.

The *event.ini* file configures the IT/O user name and password, along with the TCL scripts that are invoked based on the **evtype** field of the incoming ServiceCenter event.

ServiceCenter TCL Variables

The available variables depend on the **evtype type** field and are in accordance with the event maps from Events Services in ServiceCenter for that type. These variables are made available as **TCL_GLOBAL** and will be accessible via the \$VARIABLE syntax globally from within the called TCL script. See the **Event Services User's Guide** for more detail about event maps.

Table 5-4. SC TCL Variables

Variable Name	Short Description
INSTALLDIR	The directory path where the product was installed. This is not part of an event field and is added for convenience in locating helper scripts.
SCEVTYPE	ServiceCenter registration name for the event, for example, <i>pmo</i> .
SCEVTIME	Date and time event occurred.
SCEVSYSSEQ	ServiceCenter eventout queue sequence number. Used as checkpoint in sync file.
SCEVUSRSEQ	A user-assigned sequence number used to trace an event through ServiceCenter.
SCEVSYSOPT	A code to identify system options.

Table 5-4. SC TCL Variables

SCSCEVUSER	The event user name; if passed, it is used as the operator name.
SCEVPSWD	The event user's password.
SCEVSEPCHAR	The character used to separate fields in the \$SCEVFIELDS variable. Default is "^".
SCEVFIELDS	The data describing the event, with fields separated by the \$SCEVSEPCHAR character.
SCEVFIELDS_1..2..3 ...*	The event fields from the \$SCEVFIELDS string parsed out as individual field values using the \$SCEVSEPCHAR separator value.

Note: The specific field definition depends on the evtype coming back and is documented in the <evtype>.tcl scripts themselves. It is up to the user to define types that do not have a corresponding TCL script.

IT/O Programming APIs as TCL Commands

LEGEND:

< > required parameter

[] optional parameter

These are IT/O Programming APIs that have been wrapped in TCL commands and made available during the script invocation. Specifically, the available APIs consists of *opcif_** types for writing to the Message Stream Interface, and *opcmg_** types for retrieving message details from the database.

Summary of IT/O TCL Commands

- **opcif_write** <Message Text> <Application> <Message Group> <Message Type> <Node Name> <Object> <Severity>
- **opcmg_annotation_add** <Message Id> <Annotation Text>
- **opcmg_ack** <Message Id>
- **opcmg_unack** <Message Id>
- **opcmg_own** <Message Id>
- **opcmg_disown** <Message Id>
- **opcmg_escalate** <Message Id>
- **opcmg_op_action_start** <Message Id>

opcif_write

opcif_write <Message Text> <Application> <Message Group> <Message Type> <Node Name> <Object> <Severity>

The **opcif_write** command creates a new IT/O message. To pass a null value, use "" for an empty string instead. There are seven required arguments to this command:

- Message Text** The message text of the newly created IT/O message.
- Application** The application that this message belongs to. Check your Message Source Templates for valid application names.
- Message Group** The group that this message belongs to. Check your Message Source Templates for valid application names.
- Message Type** The type that this message belongs to. Check your Message Source Templates for valid application names.
- Node Name** The IT/O node name that this message is created for.
- Object** The object that this message belongs to. Check your Message Source Templates for valid application names.
- Severity** The IT/O severity value for this message. Check the utility script EventMap/FromSC/util.tcl for the conversion utility to convert from ServiceCenter priority codes. Valid values are:
- a. *c* - Critical
 - b. *j* - Major
 - c. *m* - Minor
 - d. *w* - Warning
 - e. *n* - Normal

Example:

```
# source in the util.tcl script to get the ConvertSeverity command
source "$INSTALLDIR./EventMap/FromSC/util.tcl"

# construct a formatted message text to be passed to opcif_write
set OPC_Message_Text "Problem ticket $SCEVFIELDS_2 opened on $SCEVFIELDS_4
by $SCEVFIELDS_5 for Message ID $SCEVFIELDS_14\n Category: $SCEVFIELDS_3\n
Assigned to: $SCEVFIELDS_9\n Severity: $SCEVFIELDS_7\n CauseCode:
$SCEVFIELDS_17\n Hostname: $SCEVFIELDS_18\n Group: $SCEVFIELDS_19\n
Location: $SCEVFIELDS_21\n Action: $SCEVFIELDS_26\n Contact:
$SCEVFIELDS_32 $SCEVFIELDS_34\n NetworkName: $SCEVFIELDS_35\n Resolution:
$SCEVFIELDS_37\n UpdateAction: $SCEVFIELDS_38"

# convert ServiceCenter priority code to ITO Severity indicator
set SCSEVERITY [ ConvertSeverity "$SCEVFIELDS_7" ]
```

```
# create the new ITO message
opcif_write "$OPC_Message_Text" "ServiceCenter_pmo" "ServiceCenter"
"TroubleTicket" "$SCEVFIELDS_18" "pmo" "$SCSEVERITY"
```

In this example, a new IT/O message is created with a formatted message text from ServiceCenter event fields of Application="ServiceCenter_pmo", Group="ServiceCenter", Type="TroubleTicket", node name from ServiceCenter event logical.name field, Object="pmo" and a converted Severity that is equivalent to the ServiceCenter priority code.

opcmsg_annotation_add

```
opcmsg_annotation_add <Message Id> <Annotation Text>
```

This command adds an annotation to an existing IT/O message. This command requires two arguments, consisting of a IT/O message ID and free-form annotation text.

Example:

```
# pre-format an annotation text to use from ServiceCenter field values
set OPC_Annotate_Text "Problem ticket $SCEVFIELDS_2 opened on $SCEVFIELDS_4
by $SCEVFIELDS_5\n Category: $SCEVFIELDS_3\n Assigned to: $SCEVFIELDS_9\n
Severity: $SCEVFIELDS_7"
```

```
# annotate and existing ITO message
opcmsg_annotation_add "$SCEVFIELDS_14" "$OPC_Annotate_Text"
```

The previous example pre-formats annotation text to be used to annotate an existing IT/O message identified by the message ID stored in event field index 17 of the evfields string.

opcmsg_ack

```
opcmsg_ack <Message Id>
```

Given the message ID of an existing IT/O message, this command will acknowledge it as the user defined in the event.ini file.

opcmsg_unack

```
opcmsg_unack <Message Id>
```

Given the message ID of an existing IT/O message, this command will unacknowledge it as the user defined in the event.ini file.

opcmsg_own

```
opcmsg_own <Message Id>
```

Given the message ID of an existing IT/O message, this command will own it as the user defined in the event.ini file.

opcmsg_disown

`opcmsg_disown <Message Id>`

Given the message ID of an existing IT/O message, this command will disown it as the user defined in the `event.ini` file.

opcmsg_escalate

`opcmsg_escalate <Message Id>`

Given the message ID of an existing IT/O message, this command will escalate it as the user defined in the `event.ini` file.

opcmsg_op_action_start

`opcmsg_op_action_start <Message Id>`

Given the message ID of an existing IT/O message, this command will start the pre-defined operator action that the user defined in the `event.ini` file.

Event Configuration File

The `event.ini` file in the `<install dir>/EventMap/FromSC` directory is a means of configuring the `sctoito` component of the interface. In here, you may specify:

- The user name/password to connect to IT/O to retrieve message details from its database.
- The TCL map script to invoke when a certain event type is read from ServiceCenter.

Sections

Section header names are identified by square brackets "[...]". The brackets group sets of configurable parameters. There are two required section header names:

CONFIG	Section for specifying IT/O username/password to connect as.
EVENT	Section for configuring event types to the TCL map script to execute.

Each section header name has its own set of required parameters.

Table 5-5. Section Parameters

Section Name	Required Parameter
CONFIG	<ul style="list-style-type: none"> • OPC_USER - The IT/O user name. • OPC_PASSWORD - The password.
EVENT	<p><event type>=<TCL map script to execute></p> <p>For example: pmo=EventMap/FromSC/pmo.tcl</p> <p>This will tell the interface to execute the script <install dir>/EventMap/FromSC/pmo.tcl when a "pmo" event is received from ServiceCenter.</p>

Default *event.ini* file

```

----- begin file -----
[ CONFIG ]
OPC_USER=opc_adm
OPC_PASSWORD=OpC_adm

[ EVENT ]
pmo=EventMap/FromSC/pmo.tcl
pmu=EventMap/FromSC/pmu.tcl
pmc=EventMap/FromSC/pmc.tcl
----- end file -----

```

Default Behavior

When there is a *pmo* in the eventout file (such as when a ticket is newly created), **scito** treats it as an acknowledgment of the new problem ticket and annotates the originating event. **scito** also creates a new ServiceCenter *opened* event with the PM number being the Object.

When there is a *pmu* in eventout (such as when its created by a pmu in eventin, or when a ServiceCenter operator modifies the ticket), if the actor or user field is not SCITO, **scito** annotates the originating event and creates a new ServiceCenter *update* event with the PM number being the Object. This prevents automatic annotations from IT/O from going into an infinite loop, updating ServiceCenter and vice-versa.

Similarly, when eventout has a *pmc* that is not from user SCITO, the originating event will be annotated and a new ServiceCenter *closed* IT/O event will be generated.

Default pmo.tcl script

```
[comments deleted]

#
*****
**
# BEGIN data block
# Define data value mapping between from ServiceCenter event variables to
# ITO variables to be used by OPC commands in action block.
#
*****
**

set OPC_Annotate_Text "Problem ticket $SCEVFIELDS_2 opened on $SCEVFIELDS_4
by $SCEVFIELDS_5\n Category: $SCEVFIELDS_3\n Assigned to: $SCEVFIELDS_9\n
Severity: $SCEVFIELDS_7"

set OPC_Message_Text "Problem ticket $SCEVFIELDS_2 opened on $SCEVFIELDS_4
by $SCEVFIELDS_5 for Message ID $SCEVFIELDS_14\n Category: $SCEVFIELDS_3\n
Assigned to: $SCEVFIELDS_9\n Severity: $SCEVFIELDS_7\n CauseCode:
$SCEVFIELDS_17\n Hostname: $SCEVFIELDS_18\n Group: $SCEVFIELDS_19\n
Location: $SCEVFIELDS_21\n Action: $SCEVFIELDS_26\n Contact:
$SCEVFIELDS_32 $SCEVFIELDS_34\n NetworkName: $SCEVFIELDS_35\n Resolution:
$SCEVFIELDS_37\n UpdateAction: $SCEVFIELDS_38"

set OPC_Application "ServiceCenter_pmo"

set OPC_Message_Group "ServiceCenter"

set OPC_Message_Type "TroubleTicket"

set OPC_Node_Name "$SCEVFIELDS_18"

set OPC_Object "$SCEVFIELDS_2"

# convert ServiceCenter priority code to ITO severity
# proc is defined in util.tcl
set rc [catch {set SCSEVERITY [ ConvertSeverity "$SCEVFIELDS_7" ]}]

set OPC_Severity "$SCSEVERITY"

#
*****
**
# BEGIN Action block
# Define ITO message updating/creating logic using variables mapped in data
# block.
#
# See GUIDE.txt for available OPC Tcl commands and their parameters.
```

```

#
*****

#
# first try to annotate original message, if message doesnt exist, create a
# new one.
#
set rc [catch [opcmsg_annotation_add "$SCEVFIELDS_14" "$OPC_Annotate_Text" ]
]

if {[expr $rc == 0]} {
    puts $debugFile "ITO message annotated for pmo. $SCEVFIELDS_2"
}

#
# lets make a new ServiceCenter message.
#
set rc [catch [opcif_write "$OPC_Message_Text" "$OPC_Application"
"$OPC_Message_Group" "$OPC_Message_Type" "$OPC_Node_Name" "$OPC_Object"
"$OPC_Severity" ] ]

if {[expr $rc == 0]} {
    puts $debugFile "ITO message created for pmo. $SCEVFIELDS_2"
} else {
    puts $debugFile "After running opcif_write pmo, $SCEVFIELDS_2, ret = ($rc)
$errorCode\n$errorInfo."
}

#
# now lets try to own the message.
#
#set rc [catch [opcmsg_own "$SCEVFIELDS_14" ] ]
#
#if {[expr $rc == 0]} {
#puts $debugFile "ITO message owned. $SCEVFIELDS_2"
#} else {
#puts $debugFile "ITO message unable to own. $SCEVFIELDS_2"
#}

```

VP Operations Business Logic Configuration

This section contains a step-by-step process for configuring VP Operations to enable automatic event message integration with ServiceCenter. This process is part of the normal configuration process upon installation of SCAuto for VP Operations.

General Process

Requirements Analysis

To take full advantage of VP Operations and ServiceCenter integration, you need a high-level plan for application integration. While the integration can automatically create problem tickets from any events managed by VP Operations, a defined set of application requirements enables you to focus this general capability into specific measurable results.

For example, you can choose to have all kernel alarms open problem tickets automatically, but this process will create hundreds of tickets. If the design is reviewed from a requirements standpoint, where the service desk is pursuing operations in accordance with a service level agreement (SLA), perhaps it can be refined to open tickets only on *critical* events from a small set of *critical hosts*. In this case, defining the requirements helps meet the specific goals of the SLA by notifying the service desk of potentially huge impacts on meeting the contract.

The process of defining requirements precedes the configuration of any software. It consists of analyzing your business and application and service desk requirements, and organizing these requirements to suit your needs. It is the first step in the Analyze, Design, and Implement process described in this section.

Design Phase

After you have defined the requirements of your consolidated service desk environment, you can determine the specific monitored sources help needed to meet these requirements. This is the Design step of the process. In VP Operations, this amounts to identifying what Message Sources are relevant. This may be done for individual Message Sources or for groups. Beyond noting which Message Sources are relevant, the next step is to define the parameterization and configuration of each relevant Message Source.

Implementation Phase

With a coherent design that meets the requirements, it is then possible to modify the IT/O Message Source Templates to meet this design specification. This is the Implement step of the process.

After the modified templates are saved, they need to be *pushed* to the managed nodes to which they apply. This step, which is the final configuration action, acknowledges that the modified templates are controlled centrally from IT/O but are distributed broadly to the nodes or groups that are specified as the appropriate recipients of the policy.

Design Considerations

When designing your system, consider these guidelines:

- If you expect to receive many of the same types of messages that will cause VP Operations to ServiceCenter interaction, use the Message Stream Interface (MSI). SCAuto for VP Operations support of MSI involves a daemon process that is constantly waiting for new messages in the message stream. This is not processor intensive, and it is implemented to be a lightweight background process for performing regular, repeated event message processing.
- If you expect to receive fewer and less frequent, and consequently more important messages, use the Trouble Ticket Interface (TTI). SCAuto for VP Operations support of TTI involves a single thread of execution that is started upon receipt of the event message. This amounts to a processing intensive approach that offers a direct message delivery.

Implementation Steps

The following steps describe how to modify a Message Source Template to activate automated event messages from VP Operations to ServiceCenter. These steps are modeled from a process that implemented a high-level goal, stated as:

Integrate Security alerts into Problem Tickets where any failed switch user (su) command causes IT/O alerts.

Step 1 - Review Templates Assigned to a Node

From the IT/O Node Bank window, go to the **Actions** menu, click **Agents**, then click **Assign Templates**.

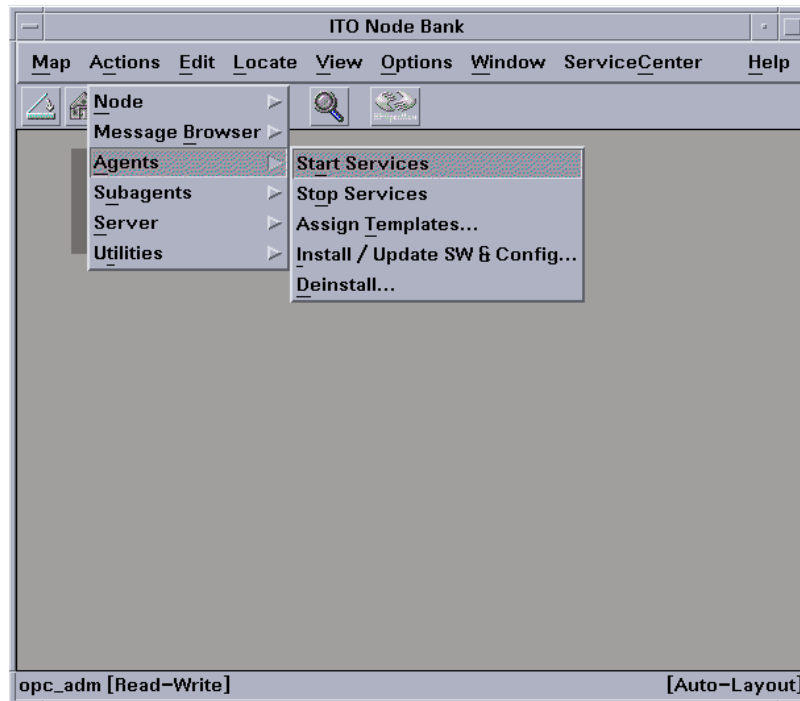


Figure 5-1 IT/O Node Desk window

The Define Configuration window is displayed.

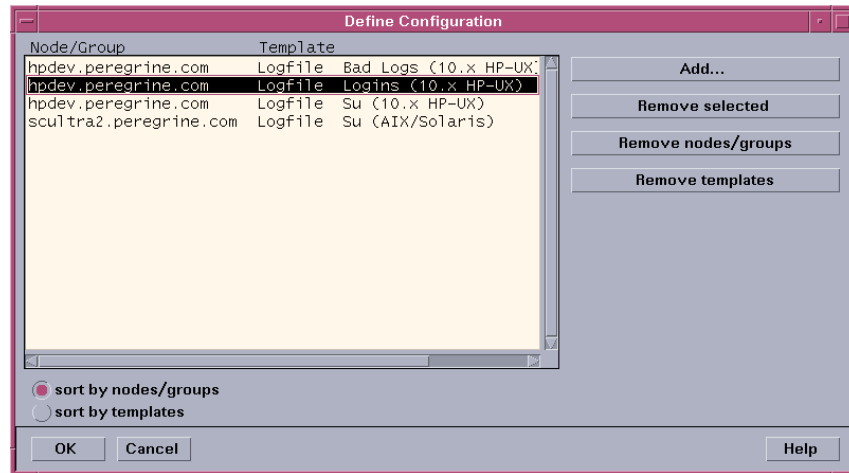


Figure 5-2 Define Configuration window

In this window, identify which templates have been assigned to specific nodes or groups. If the template that you want is already assigned, you need to push it out again to its subscribers once you complete your modifications. Plan to do this in addition to, or along with, the template that you push out in Step 7.

Step 2 - Begin Modifying Templates (to Assign to a Node)

In the IT/O Node Bank window, go to the **Window** menu and click **Message Source Templates** (Figure 5-3). The Message Source Templates window is displayed (Figure 5-4). This window contains two panes:

- The left pane contains the message source groups available for you to select (Figure 5-5).
- The right pane displays the unique Message Source Templates in a selected group (Figure 5-6). Navigate the groups by clicking on items to show values in the right pane or double-clicking to expand items in the left pane.

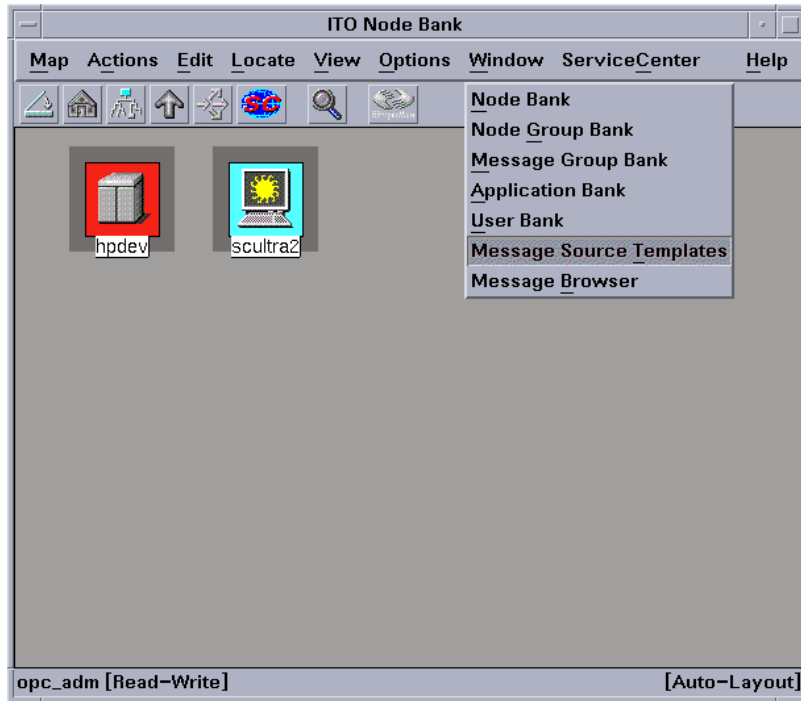


Figure 5-3 IT/O Node Bank window: Window menu

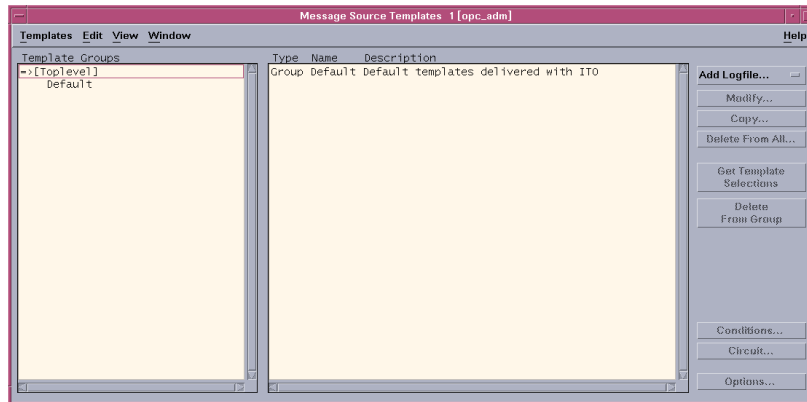


Figure 5-4 Message Source Templates window

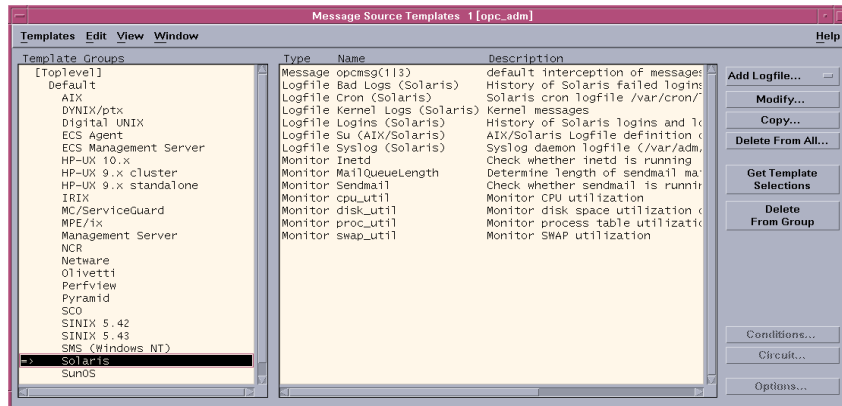


Figure 5-5. Message Source Templates window: Left pane navigation

When you find the group you are interested in, select it in the left pane. The **Solaris** group is used in this example.

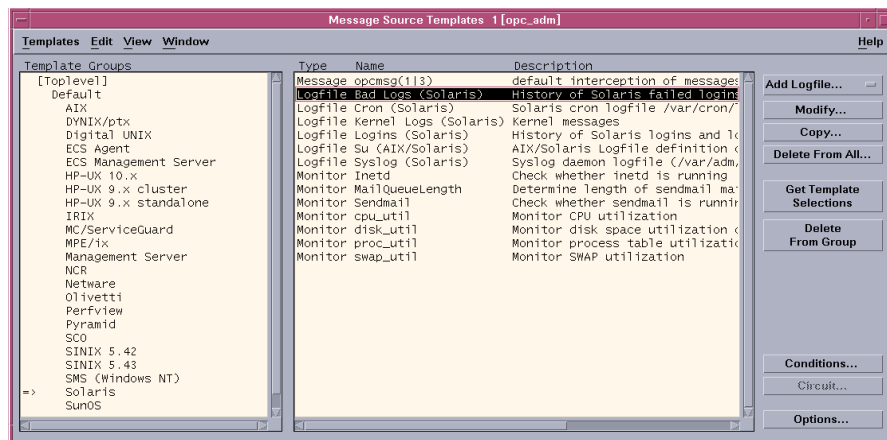


Figure 5-6. Message Source Templates window: Right pane navigation

The right pane contains the members of the selected group (Figure 5-6). For the **Solaris** group, there is a **Logfile Bad Logs (Solaris)** item. When this item is selected, it activates the **Conditions** button. Press this button to edit the configuration of the Logfile Bad Logs (Solaris) template. This displays the Message & Suppress Conditions window (Figure 5-7).

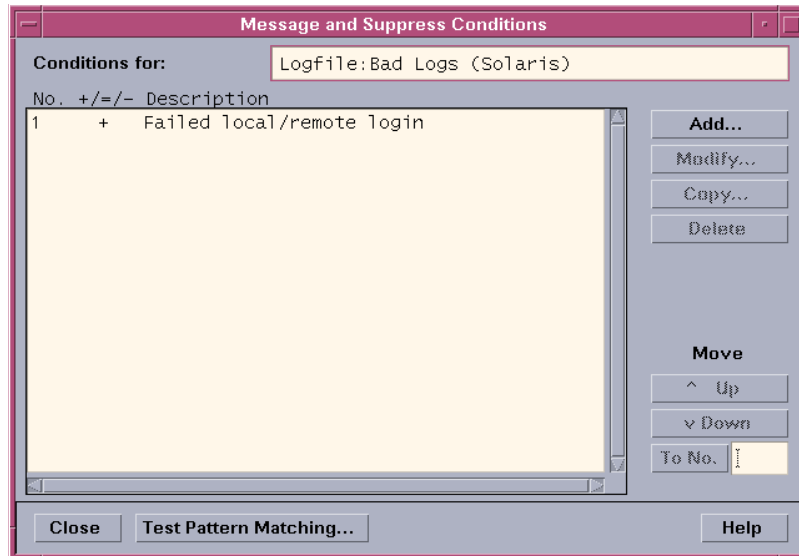


Figure 5-7. Message & Suppress Conditions window

Step 3 - Modify Templates (to Assign to a Node)

From the Message & Suppress Conditions window, ensure that the conditions are set as you require. For example, to suppress a condition, be sure that it has a minus sign (-) in the window. To activate a condition, it needs a plus sign (+). To change the setting, select the condition and press the **Modify** button. This displays the Condition No. <N> window (where *N* is the number of the condition in the list, according to the condition you select). See Figure 5-8 for an example.

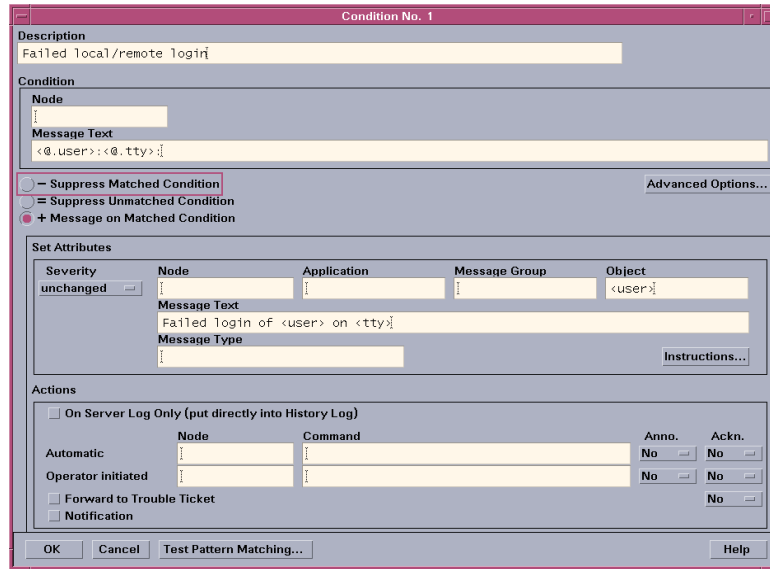


Figure 5-8. Condition No. 1 window

Step 4 - Modify Template Conditions to Forward to Trouble Ticket

To invoke the TTI interface, select the **Forward to Trouble Ticket** check box on the Condition No. <N> window. This action completes the Message Source Template configuration. Save your edits by clicking the **OK** button (closes the Condition No. <N> window), and proceed to Step 7 - Install Templates to Selected Nodes.

Note: If you wish to use the MSI interface, proceed to the next step without configuring the Trouble Ticket Interface.

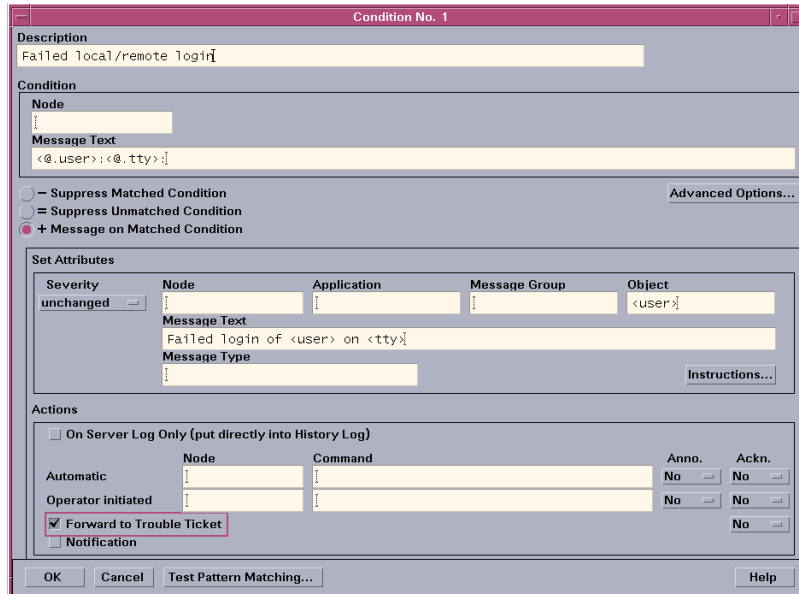


Figure 5-9. Condition No. 1 window

This form is the same place where you would set up automatic actions, such as executing a command on the node that generated this event message. Certain **attributes** can also be set here. These attributes can then be used for processing actions in SCAuto for VP Operations maps (scripts) and can later be used within ServiceCenter. The attribute settings will apply regardless of the choice of MSI or TTI setting.

For example, the **Message Group** attribute in the center of this form can be set to variable values or literal values (constants). This attribute value is passed to SCAuto for VP Operations in an environment variable named \$OPCDATA_GROUP. For example, the Message Group attribute can be given a value of TEST in the form. Then, the SCAuto for VP Operations TCL script can use the \$OPCDATA_GROUP as the name of the *category* variable used in the ServiceCenter event message. This implies that a new ticket will be created in the TEST problem ticket category, if it exists.

Furthermore, if you create IT/O Message Groups that match ServiceCenter problem ticket categories, there will be both a high-level logical relationship between your VP Operations implementation and your ServiceCenter implementation, and you will also have a detailed connection through this use of message source template attributes.

Step 5 - Modify Template Conditions to Use Message Stream Interface

To use the MSI interface, click the **Advanced Options** button in the Condition No. <N> window. The Message Conditions Advanced Options window is displayed (Figure 5-10).

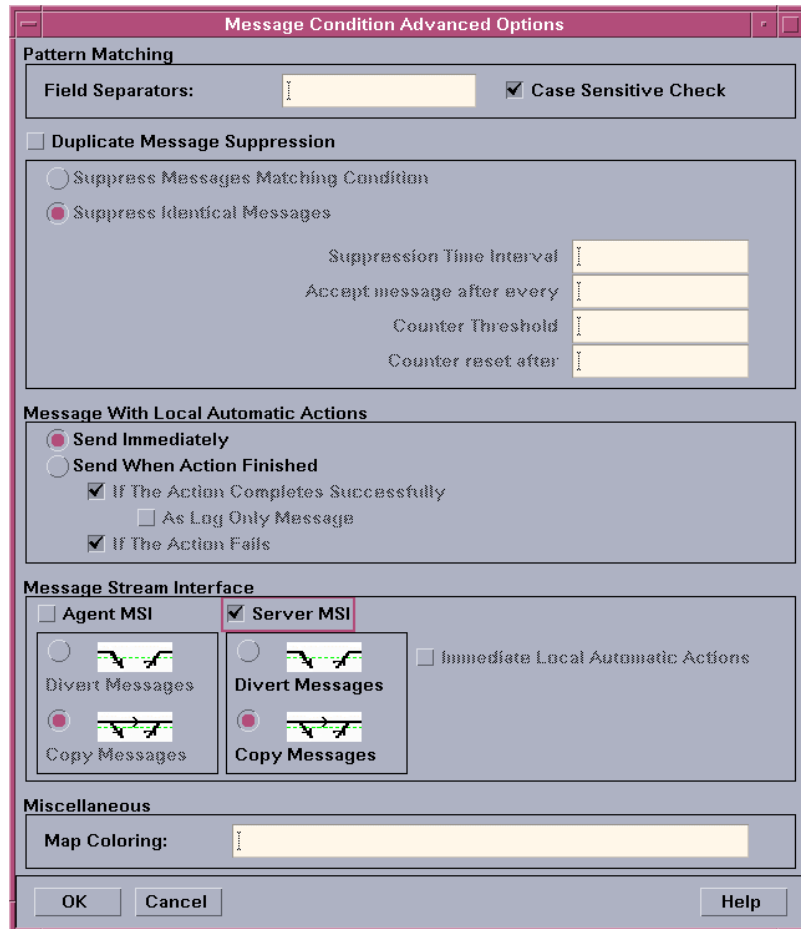


Figure 5-10. Message Conditions Advanced Options window

In the Message Conditions Advanced Options window, there is a section labeled **Message Stream Interface**. Within this section, there are two areas: Agent MSI and Server MSI. Activate just the **Server MSI** interface by checking the box. Then, select the **Copy Message** button once the Server MSI interface is activated.

Save your edits by clicking the **OK** button. This closes the Message Conditions Advanced Options window. Continue to click the **OK** or **Close** buttons in the remaining open windows until you return to the Message Source Templates window (Figure 5-4).

Using this form, you can also set up various Message Stream parameters, such as suppressing duplicate event messages.

Note: If you leave the Message Source Template window open, with the selected (now modified) template, this streamlines step 6.

Step 6 - Add Modified Template to Configuration

From the IT/O Node Bank window (Figure 5-11), go to the **Actions** menu and click **Agents**. Then, click **Assign Templates** to display the Define Configuration window (Figure 5-12). This process repeats the commands and actions of step 1.

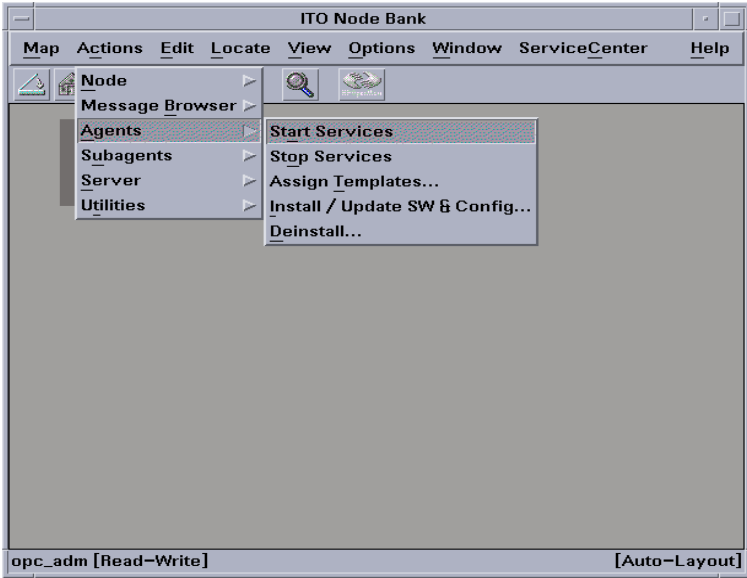


Figure 5-11. IT/O Node Bank window

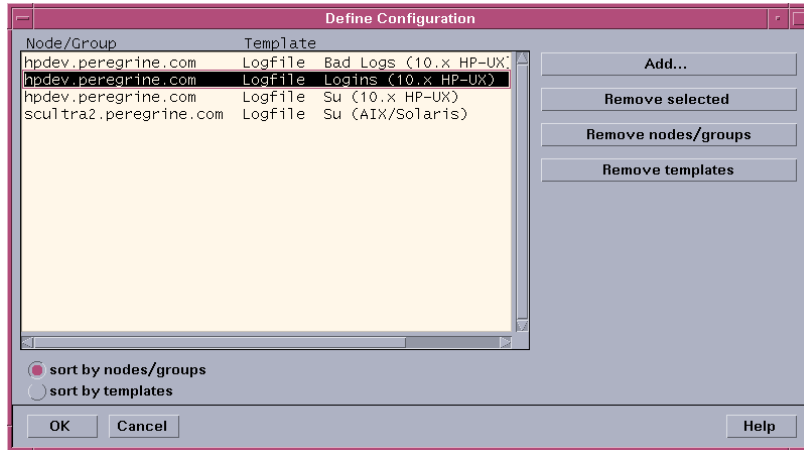


Figure 5-12. Define Configurations window

In the Define Configurations window (Figure 5-12), click the **Add** button to launch an Add Configurations window (Figure 5-13).

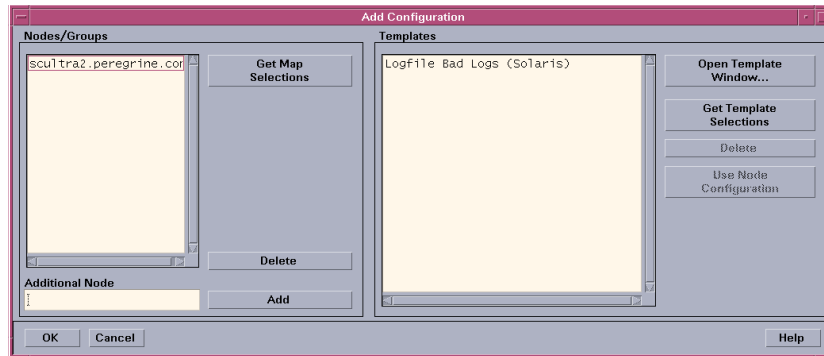


Figure 5-13. Add Configuration window

This window contains a **Get Template Selections** button. This is used, for example, when you have an unusual workflow. In this case, you would open a new window, select your modified template in the new window, and then return to the first window and click the **Get Template Selections** button to import your selection.

If your Message Source Template window is still open from the last step, and contains a selected template, you can click the **Get Template Selections** button to complete this step. Otherwise, continue with the instructions here.

To open the window where you can select a template, click the **Open Template Window** button in the Add Configurations window. This displays the Message Source Templates window (Figure 5-14).

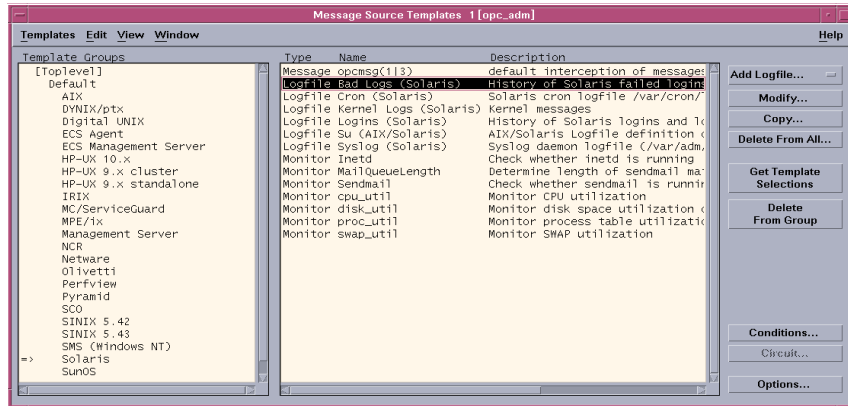


Figure 5-14. Message Source Templates window

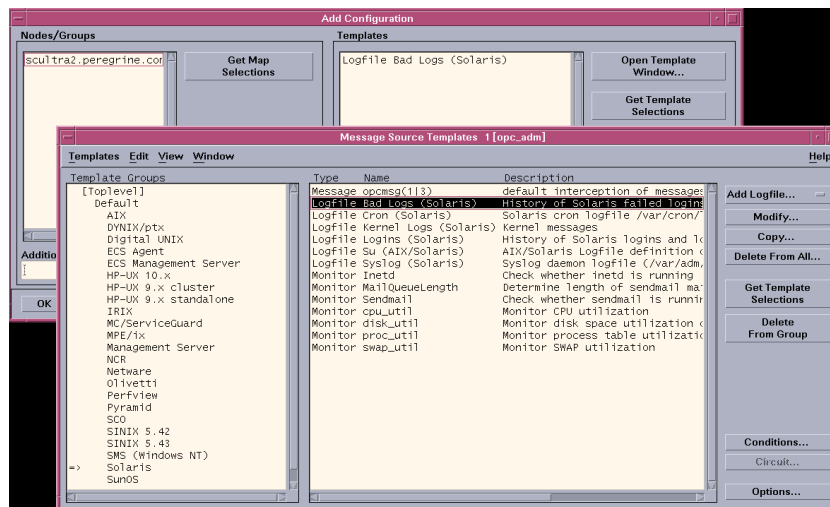


Figure 5-15. Message Source Templates and Add Configuration windows

When you finish assigning your modified templates, click the **OK** button to save your edits. This returns you to the Define Configuration window (Figure 5-12). Verify that your newly assigned template is displayed in this window. Press the **OK** button until you return to the IT/O Node Bank window. This completes the steps to define and add the desired templates.

Note: The Message Source Templates window (Figure 5-15) may still be open from step 5. You can close it now as well.

Step 7 - Install Templates to Selected Nodes

This step involves “pushing” the configuration to the endpoints. Once the message source templates are installed at the endpoints, any activity that triggers the template at the nodes will cause IT/O event messages and subsequent activation of SCAuto for VP Operations.

From the IT/O Node Bank window (Figure 5-16), highlight one or more nodes to receive the configuration. On the **Actions** menu, click **Agents**, then click **Install / Upgrade S/W Configuration**. This displays the Install/Upgrade IT/O Software and Configuration window (Figure 5-17).

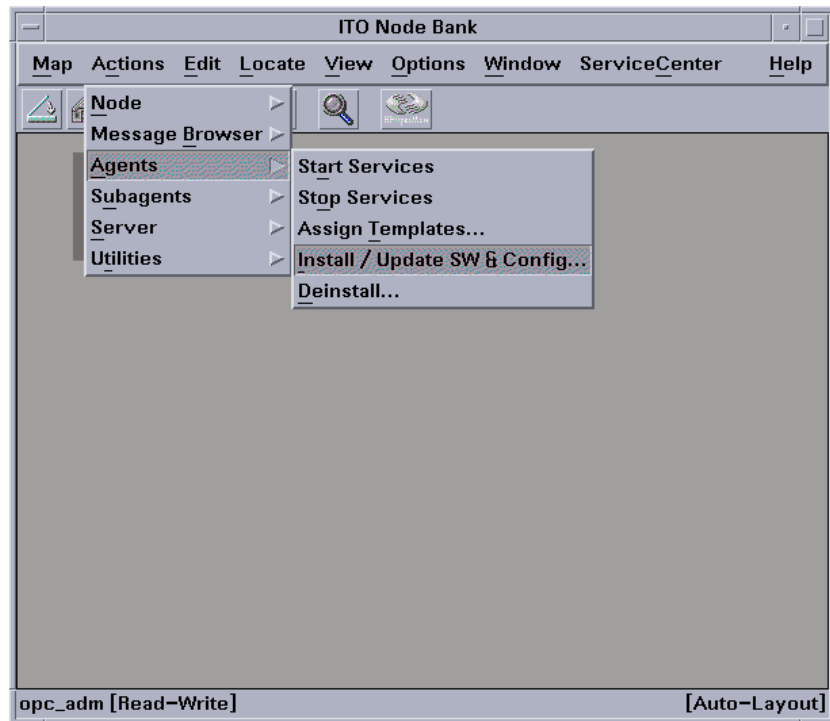


Figure 5-16. ITC Node Bank window: Actions menu

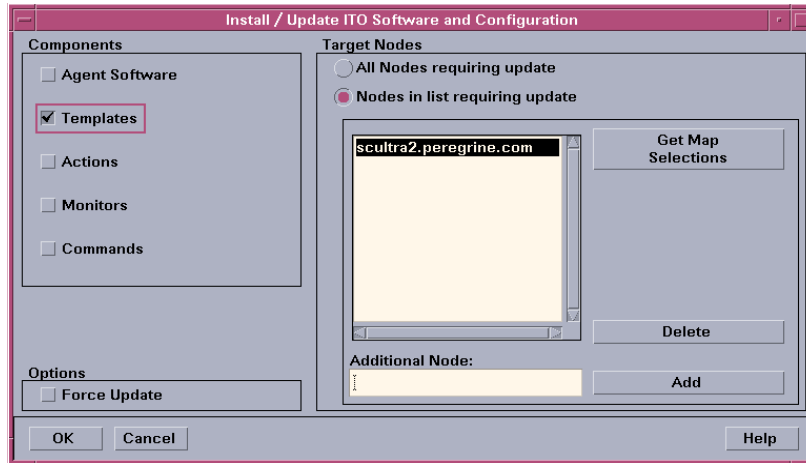


Figure 5-17. Install / Upgrade IT/O Software and Configuration

Verify that your node or group is in the list of target nodes. You must also verify that the **Templates** checkbox is selected in the **Components** area in the left part of the window. Your message source may also require Actions, Monitors, and Commands.

When you click **OK**, the template configuration is sent to the node and it replaces any existing configuration at the node. If you have an IT/O message browser open (Figure 5-18), you will see an *opcdista* type of event message that confirms that the configuration has been received by the endpoint).

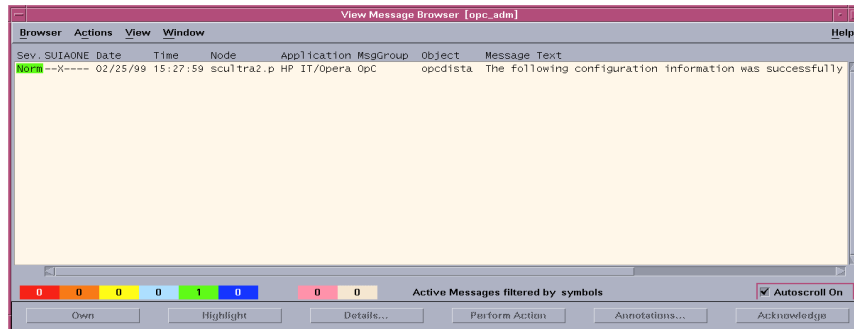


Figure 5-18. View Message Browser

Step 8 - Final Configuration for use of Trouble Ticket Interface

If you chose to use the TTI interface, there is one additional configuration step. You need to specify which program IT/O should run whenever the TTI function is invoked.

From the IT/O Node Bank window, go to the **Actions** menu and click **Utilities**. Then, click **Trouble Ticket** to display the Trouble Ticket dialog box (Figure 5-19).

In the Trouble Ticket dialog box, select the **Use Trouble Ticket System** checkbox. In the **Call of Trouble Ticket** field, enter a full path name and script name to invoke the SCAuto for IT/O TTI program. (The program's name is **TTI.sh**. The default location for this is `/opt/OV/scauto/TTI.sh`; however, your installation may have placed this program in a different location.)

When you click **OK**, IT/O will look for and validate that the program is available and executable.

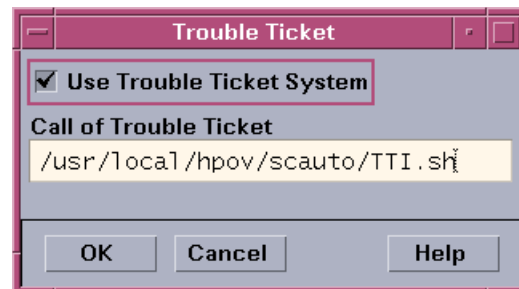


Figure 5-19. Trouble Ticket dialog box

Step 9 - Final Configuration for Use of Message Stream Interface

There is one more parameter that must be verified before the configuration is complete. To allow specific nodes to use the MSI or not, IT/O has a parameter related to enabling outputs from nodes to the MSI.

From the IT/O Node Bank window, select one or more nodes. Right-click the mouse, and click the **Modify** command. This opens the Modify Node window. Click on the **Advanced Options** button, and the Node Advanced Options window appears (Figure 5-20). In the section of this window labeled Message Stream Interface, click on the **Enable Output** checkbox. This is also the place to enable automatic actions and operator initiated actions.

To save your edits, click **OK** until you get back to the IT/O Node Bank window. You are now fully configured for automated event integration between VP Operations and ServiceCenter.

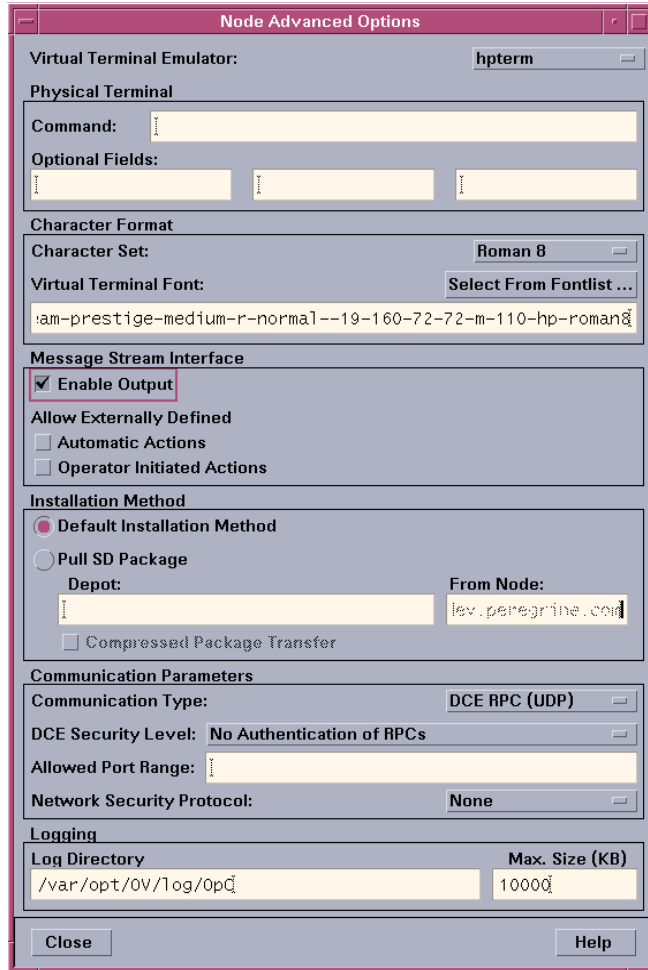


Figure 5-20. Node Advanced Options window

This completes the configuration of the VP Operations business logic inherent in the Message Source Templates.

ServiceCenter Business Logic Configuration

ServiceCenter Event Services

To access Event Services, click the **Event Services** button on the Main ServiceCenter menu. Select the **Administration** tab (Figure 5-21) to reach Event Registration, Filters, and Maps.

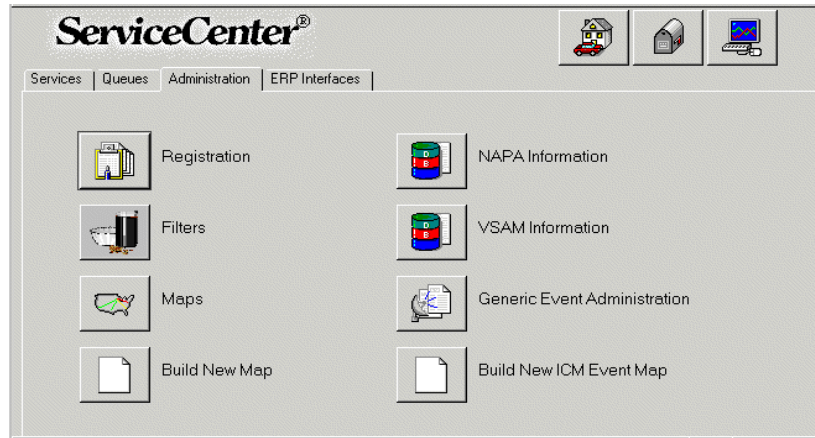


Figure 5-21. ServiceCenter Administration tab

Event Registration

On the Administration tab, click the **Registration** button to display the Event Registration form (Figure 5-22), which contains the definitions for events processed by the system. The Event Registration form contains three tabs: Basic, Expressions, and Application.

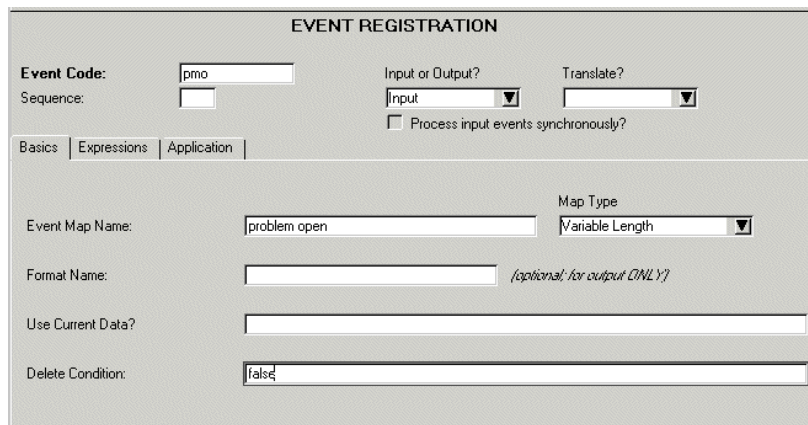
The screenshot shows the 'EVENT REGISTRATION' form, 'Basics' tab. It includes fields for 'Event Code' (pmo), 'Sequence' (empty), 'Input or Output?' (Input), 'Translate?' (empty), and a checkbox for 'Process input events synchronously?'. Below these are tabs for 'Basics', 'Expressions', and 'Application'. The 'Basics' tab contains fields for 'Event Map Name' (problem open), 'Map Type' (Variable Length), 'Format Name' (empty, with a note '(optional; for output ONLY)'), 'Use Current Data?' (empty), and 'Delete Condition' (false).

Figure 5-22. Event Registration form: Basics tab

The **Expressions** tab (Figure 5-23) displays the processing logic associated with the event type (for example, *pmo*).

Figure 5-23. Event Registration form: Expressions tab

The **Applications** tab (Figure 5-24) displays the RAD application that is associated with the event, as well as the parameters that are used when running the application upon processing of the event.

Description	Parameter Names	Parameter Values
	record	\$axces
	prompt	evmap in \$axces.register
	string1	problem
	text	open
	query	\$ax.query.passed
	boolean1	evstatus in \$axces~##"error"
	cond.input	\$ax.open.flag

Figure 5-24. Event Registration form: Applications tab

Event Maps

Event Maps are the guides to recording and processing events, including data type and the name of the file where event data is stored.

The screenshot shows the 'EVENT MAP' configuration window with the 'Basics' tab selected. The 'Map Name' is 'problem update', 'Type' is 'Input', and 'Fixed or Variable' is set to a dropdown. 'Sequence' is 1, 'Position' is 1, and 'Length' is an empty field. Below the tabs, there are fields for 'File Name' (containing 'problem'), 'Query' (empty), 'Field Name' (containing 'logical.name'), 'Data Type' (set to 'Character'), 'Nullsub' (containing '\$xces.field'), and 'Translate' (empty). The 'Array Information' section includes 'Element Type' (empty), 'Element Separator' (checkbox), and 'Element Length' (checkbox).

Figure 5-25. Event Map: Basics tab

The **Expressions** tab (Figure 5-26) contains additional instructions for event processing.

The screenshot shows the 'EVENT MAP' configuration window with the 'Expressions' tab selected. The 'Map Name' is 'problem update', 'Type' is 'Input', and 'Fixed or Variable' is set to a dropdown. 'Sequence' is 1, 'Position' is 1, and 'Length' is an empty field. Below the tabs, there is an 'Initialization' section with three empty text boxes. The 'Condition for Mapping' section has one empty text box. The 'Post-Map Instructions' section contains the following text: 'if null(logical.name in \$xces.target) then (logical.name in \$xces.target=network.name in \$xces.target) factor in \$xces.target=nullsub(operator(), "external")'. There are four additional empty text boxes below the instructions.

Figure 5-26. Event Map: Expressions tab

All Event Services features are addressed in greater detail in the *Event Services User's Guide*. See this guide if you plan to make modifications discussed in this section.

ServiceCenter Problem Management

You may want to customize and configure the definition of problem ticket formats and management in ServiceCenter. Based on your needs, you may need to create additional problem categories to reflect the automatic problem ticket processing supported by the SCAuto for VP Operations product. You can use the problem category named *example* as a guide or template to build new automated ticket categories and their associated forms and database dictionary definitions.

Figure 5-27 shows a portion of the database dictionary structure for the problem file. This can be used as reference for custom categories based on the problem file. For more detailed information, refer to the “Database Dictionary” section in the *Base Utilities Guide*.

Field Name	Type	Index	Level
descriptor	Structure	1	0
header	Structure	1	1
number	Character	1	2
number.vj	Character	1	2
page	Number	2	2
total.pages	Number	3	2
open.time	Date/Time	4	2
category	Character	5	2
alert.time	Date/Time	6	2
assignment	Character	7	2
update.time	Date/Time	8	2
asgnchg	Number	9	2
status	Character	10	2
close.time	Date/Time	11	2
reopen.time	Date/Time	12	2

Keys	File Number/Pools
No Nulls	header.number header.last
Unique	header.number header.page

Figure 5-27 . Portion of the Database Dictionary structure for the Problem File

Event data is channeled and presented in ServiceCenter according to controls defined in the **Format Control** records associated with specific problem category formats (display forms). Refer to the *Format Control Guide* for complete information on Format Control in ServiceCenter.

Additionally, the **Forms Designer** tool is used to create custom forms for added problem categories and customizations to accommodate the automated problem ticket generation from SCAuto for VP Operations events. Refer to the *Forms Designer Guide* for more details on form (format) development and customization.

Chapter 6 Scenarios



This chapter provides scenarios of different high-level configurations that can be accomplished using SCAuto for VP Operations. Depending on your business needs, the following examples may be implemented individually or in combination to achieve your business goals.

Uni-directional Automatic Problem Ticket Creation (Mode 1)

This implementation is described in Chapter 1, “Introduction,” as Mode 1 of the Operational Concepts.

In this mode, IT/O events drive SC tickets. This occurs through the registered connection from VP Operations to ServiceCenter via the MSI API. This mode requires the following components of SCAuto for VP Operations:

- scfromitoMSI and scevmon processes
- ToSC Queue file
- Event Maps (event.ini, TCL scripts, and maps referenced in event.ini)

You may start the adapter processes using the HP OpenView “ovstart <process name>” facility. Since by default, all the other SCAuto for VP Operations components are installed and configured to execute, using the integrated GUI menu options in the Root/Node Bank Windows will start all the adapters.

You may be able to customize and remove the unwanted components from the menu by doing the following:

- Modify these files:
 - `$OV_CONF/OpC/mgmt_sv/ui/registration/C/opc_adm/scauto`
 - `$OV_CONF/OpC/mgmt_sv/ui/registration/C/opc_adm/scauto`
- Under the Action “startALL”, “stopAll”, and “statusAll” sections, remove the components that you do not want to start/stop.

Since only the `scfromitoMSI` and `scevmon` processes are required in this implementation, you can use the LRF files supplied in the `<installed directory>/lrf_files` directory and execute `ovdelobj scfromitoMEI.lrf`, and `ovdelobj sctoito` to remove the registration of these components.

In addition, you can modify the LRF for `scfromitoMSI` and `scevmon` and change the first parameter to `OVs_YES_START`, and then execute `ovaddobj <filename>`. This causes the HP OpenView ovstart facility to start the process by default.

For more information, refer to the *HP OpenView Administrator's Guide*, or the man pages for “`ovaddobj`” and “`ovdelobj`” commands as well as the man pages for “`lrf`”.

(Out of Box) Bi-directional Problem Ticket/ITO Message Creation/Update/Close (Mode 2)

This implementation is described in Chapter 1, “Introduction,” as Mode 2 of the Operational Concepts. This is the out-of-box default behavior.

In this mode, IT/O events drive ServiceCenter tickets, and the ServiceCenter tickets control IT/O events. This amounts to a partnership of managing the events, where each application provides a significant contribution to the event and problem management process. This mode is constructed through the registered connection from IT/O to ServiceCenter via the MSI API, as well as the MEI API. This mode requires the following components of SCAuto for VP Operations:

- scfromitoMSI, scfromitoMEI, sctoito, and scevmon processes
- ToSC and FromSC Queues
- Event Maps (event.ini, TCL scripts, and maps referenced in event.ini)

You can start or stop the adapter processes using the HP OpenView ovstart <process name> facility or the integrated GUI menu option from the Root or Node Bank windows.

In addition, you can modify the LRF (in the <installed directory>/lrf_files directory) for these components and change the first parameter to *OVs_YES_START*, and execute **ovaddobj <filename>**. This causes the HP OpenView ovstart facility to start the process by default.

Refer to the *HP OpenView Administrator's Guide*, or the man pages for **ovaddobj** and **ovdelobj** commands as well as the man pages for “lrf”.

Creating Problem Tickets with Trouble Ticket Interface (Mode 3)

This implementation is described in Chapter 1, “Introduction”, as Mode 3 of the Operational Concepts.

This mode is an alternative configuration of Mode 1 (Uni-directional) or Mode 2 (Bi-directional). It uses the IT/O TTI API instead of the MSI API. This allows for a more focused configuration of IT/O Message Source Templates, freeing the MSI API to be used for other integration efforts, if desired. MEI usage with this mode is identical to the previous modes.

This mode requires the following components of SCAuto for VP Operations:

- `scfromittoTTI` and `SCEVMON` processes
- ToSC and FromSC Queues
- Event Maps (`event.ini`, TCL scripts, and maps referenced in `event.ini`)
- IT/O configurations for TTI to execute `<installed directory>/TTI.sh`, as well as source templates to *Forward* message to Trouble Ticket Interface.

Because this implementation does not require the execution of any of the other IT/O interfaces, you can remove them from the GUI integration as well as the SPMD (`ovstart`) profile, if desired.

You can start or stop the `scevmon` process using the HP OpenView `ovstart scevmon` facility. Since by default, all the other SCAuto for VP Operations components are installed and configured to execute, using the integrated GUI menu options in the Root/Node Bank Windows will start all the adapters.

You may be able to customize and remove the unwanted components from the menu by doing the following:

- Modifying these files:
 - `$OV_CONF/OpC/mgmt_sv/ui/registration/C/opc_adm/scauto`
 - `$OV_CONF/OpC/mgmt_sv/ui/registration/C/opc_adm/scauto`
- Under the Action *startALL*, *stopAll*, and *statusAll* sections, remove the components that you do not want to start or stop.

Because only the `scevmon` process is required in this implementation, you can use the LRF files supplied in the `<installed directory>/lrf_files` directory and execute `ovdelobj <component file name>` to remove the registration of these components.

In addition, you can modify the LRF `scevmon` and change the first parameter to `OVs_YES_START`, and then execute `ovaddobj <scevmon LRF filename>`. This causes the HP OpenView `ovstart` facility to start the process by default.

Refer to the *HP OpenView Administrator's Guide*, or the man pages for `ovaddobj` and `ovdelobj` commands as well as the man pages for `lrf`.

(Out of Box) Node Based Problem Tickets

By default, after installing the product, the *To ServiceCenter* TCL scripts are configured to generate pmo events with the category of *example*. Also, the default behavior for the *example* category in ServiceCenter is to match problem tickets based on the **logical.name** field in the pmo event. If this is the desired effect, you do not have to customize the product.

If you want the pmo to go into a specific ServiceCenter category, you must take the following steps:

- Make sure that the targeted category contains format control logic to handle a pmo the same way as the example category.
- Customize the TCL scripts to generate the desired category in the resulting pmo event.

Refer to Chapter 5, “Configuration,” for more information about these steps.

Cause-Based Problem Tickets Per Node

This implementation is primarily designed for a typical IT/O configuration. It essentially takes the node-based concept of OpenView NNM to another level, to an Application-based or Cause-based event per node in the managed environment.

For example, in a node-based system, every event that is configured to be captured by IT/O on the same node will funnel into the same problem ticket, regardless of the nature or cause of the message. In contrast, a Cause-based or Application-based system makes a distinction, for example, between a Security event and an OS event by opening different problem tickets for these events. By default, ServiceCenter and SCAuto for VP Operations are node-based.

To configure for a Cause-based or Application-based system, the default ServiceCenter Event Registration expression must be modified in ServiceCenter to match pmo events with existing problem tickets based on the **logical.name**, **category**, and **cause.code** fields in the pmo event.

IT/O Message Group into ServiceCenter Category

This implementation facilitates the management of large numbers of IT/O event messages by funneling them into different ServiceCenter categories based on the IT/O Message Group name. This enables the IT/O Administrator to control which IT/O message opens a problem ticket in which category. This grouping of problem tickets by IT/O Message Groups is a logical approach to eventually assigning problem tickets to the same group of technical support staff based on their expertise.

To configure for problem tickets to be opened in categories specified by the Message Group of the IT/O Message:

- In ServiceCenter, create categories matching the IT/O Message Group names. This will later enable the SCAuto for VP Operations adapter to assign events to these specific categories based on the IT/O Message Group of the interested event. Also, make sure that the correct event registration and format control logic are built into these new categories.
- In SCAuto for VP Operations, modify the default “ToSC” TCL scripts to generate specific category and cause.code in the resulting pmo. By default, the **category** field is hard-wired to the literal example. If desired, you can use the IT/O message group value to populate this field by changing the line:

```
eventObject set_evfield category example
```

to

```
eventObject set_evfield category $OPCDATA_GROUP
```

Converting Messages from Windows NT

Message text from Windows NT-managed nodes contains an extra linefeed character at the end of lines and will appear as a “^M” character on UNIX systems. This extra control character causes errors in the event registration mapping of ServiceCenter. To filter the text and strip out the extra “^M”, use the following TCL syntax:

```
[join [split $TEXT_VAR ^M]]
```

Make sure that the “^M” character is entered as one character in the “vi” editor. You can use **CTRL+V** and **CTRL+M** to enter the “^M” character.

Appendix A Contacting Peregrine Systems



For further information and assistance with SCAuto for VP Operations, contact Peregrine Systems' Customer Support. Current details of local support offices are available through these main contacts.

North America, South America, Asia/Pacific

Telephone: (1) (800) 900-8152 (within US only, toll free)

+ (1) (858) 794-7402

Fax: + (1) (858) 480-3986

Email: support@peregrine.com

Headquarters: Peregrine Systems, Inc.

Attn: Customer Support
3611 Valley Centre Drive
San Diego, CA 92130

Europe, Africa

Telephone: (0) (800) 834 770 (within United Kingdom only, toll free)

+ (44) (0) (02) 8334-5844

Fax + (44) (0) (02) 8334-5890

Email: uksupport@peregrine.com

Documentation Web Site

For a complete listing of the current SCAutomate documentation, see the Documentation pages on the Peregrine Systems, Inc. Customer Support Web site at:

<http://support.peregrine.com>

You will need the current login and password to access this Web page.

For copies of the manuals, you can download .pdf files of the documentation using the Adobe Acrobat Reader (also available on the Web site). Additionally, you can order printed copies of the documentation through your Peregrine Systems sales representative.

Index



A

- adding the modified template to the configuration, 5-35
- application integration, 4-1
- Assigned Problems, 4-13
- assigned problems, 4-13

B

- bi-directional integration, 1-8
- bi-directional problem ticket/ITO message creation/update/close, 6-3

C

- cause based problem tickets per node, 6-6
- Change Management, 1-4
- closing a problem, 4-9
- configuring SCITO, 3-6
- contacting Peregrine Systems, A-1
- converting messages from Windows NT, 6-8
- core applications, 1-4
- create_sc_event, 5-7
- creating a new problem record, 4-18
- creating problem tickets with Trouble Ticket Interface, 6-4

D

- default behavior, 5-15, 5-22
 - example eventmapMSI.tcl script, 5-15
- default pmo.tcl script, 5-23
- design considerations, 5-26
- design phase, 5-25
- determining current product version, 1-2
- directory, 4-17
- Down Time, 4-12

E

- event configuration file, 5-21
- event filtering, 4-17
- event integration, 4-19
- event mapping
 - configuration overview, 5-2
 - ITO variables, 5-3
- event maps, 5-44
- event registration, 5-42
- Event Services, 5-42

- event.ini, 5-2
- eventmapMEI.tcl, 5-2
- eventmapMSI.tcl, 5-2
- eventmapTTI.tcl, 5-3
- Events
 - problem open, 5-43

F

- Filtering, 4-17
- filtering, 4-17
- final configuration for use of Message Stream Interface, 5-40
- final configuration for use of Trouble Ticket Interface, 5-40

H

- Help Desk, 4-18

I

- ICM, 1-4
- implementation phase, 5-26
- implementation steps, 5-27
 - add modified template to configuration, 5-35
 - begin to modify templates to assign to a node, 5-28
 - final configuration for use of Message Stream Interface, 5-40
 - final configuration for use of Trouble Ticket Interface, 5-40
 - install templates to selected nodes, 5-38
 - modify template conditions to forward to trouble ticket, 5-32
 - modify template conditions to use Message Stream Interface, 5-34
 - modify templates (to assign to a node), 5-31
 - review templates assigned to a node, 5-27
- installation
 - client, 2-3
- installation prerequisites, 2-2
- installing SCAuto for VP Operations, 2-4
 - install procedure, 2-4
- installing templates to the selected nodes, 5-38
- integration components, 4-20
- Inventory
 - down time, 4-12

- service information, 4-11
- Inventory Configuration Management, 1-4
- IT/O programming APIs as TCL commands, 5-18
 - opcif_write, 5-19
 - opcmsg_ack, 5-20
 - opcmsg_annotation_add, 5-20
 - opcmsg_disown, 5-21
 - opcmsg_escalate, 5-21
 - opcmsg_op_action_start, 5-21
 - opcmsg_own, 5-20
 - opcmsg_unack, 5-20
 - summary of IT/O TCL commands, 5-18
- IT/O message filtering, 5-10
- IT/O message group into ServiceCenter category, 6-7
- IT/O sections, 5-10, 5-21
- IT/O sections and parameters table, 5-11
- IT/O variables, 5-3

K

- knowledge requirements, 1-2

L

- Location Information, 4-15
- location information, 4-15

M

- Main Menu, 4-18
- Main Menu option, 4-18
- maintaining SCITO, 3-5
- maps, 5-44
- menu options, 4-2
- Menus
 - Service Information, 4-11
- Menus
 - Assigned Problems, 4-13
 - Down Time, 4-12
 - Filtering, 4-17
 - Hlep Desk, 4-18
 - Location Information, 4-15
 - Main, 4-18
 - Other Services, 4-14
 - User Directory, 4-17
 - Vendor Information, 4-16
- Message Event Interface (MEI), 1-6
- message integration, 4-14
- Message Stream Interface (MSI), 1-6
- modifying template conditions to forward to trouble ticket, 5-32
- modifying template conditions to use Message Stream Interface, 5-34
- modifying templates (to assign to a node), 5-31
- modifying templates to assign to a node, 5-28

N

- node based problem tickets, 6-5
- non-OPC variables, 5-6

O

- opcif_write, 5-19
- opcmsg_ack, 5-20
- opcmsg_annotation_add, 5-20
- opcmsg_disown, 5-21
- opcmsg_escalate, 5-21
- opcmsg_op_action_start, 5-21
- opcmsg_own, 5-20
- opcmsg_unack, 5-20
- opening a problem, 4-5
- Other Services, 4-14

P

- Peregrine Systems, contacting, A-1
- pmc.map, 5-3
- pmo, 5-43
- pmo.map, 5-3
- pmo.tcl script, 5-23
- pmu.map, 5-3
- print, 5-8
- probable cause, 4-10
- Problem
 - assigned problems, 4-13
 - problem list, 4-4
 - Problem List menu option, 4-4
 - Problem Management, 1-4, 5-45
 - problem open, 5-43
 - process, 5-25
 - design phase, 5-25
 - implementation phase, 5-26
 - requirements analysis, 5-25

Q

- querying ServiceCenter for probably cause, 4-10

R

- Request Management, 1-4
- requirements
 - installation, 2-2
 - system, 2-1
- requirements analysis, 5-25
- reviewing the templates assigned to a node, 5-27

S

- SCAuto for VP Operations business logic topics, 1-23
- SCAutomate
 - closing problems, 4-9
 - Events
 - problem open, 5-43
 - problem list, 4-4
 - problem open, 5-43
- scenarios
 - bi-directional problem ticket/IT/O message creation/update/close, 6-3
 - cause based problem tickets per node, 6-6

- creating problem tickets with Trouble Ticket Interface, 6-4
- ITO message group into ServiceCenter category, 6-7
- node based problem tickets, 6-5
- uni-directional automatic problem ticket creation, 6-1
- scevmon, 4-21
- scfromitoMEI, 4-22
- scfromitoMSI, 4-22
- scfromitoTTI, 4-22
- scito.ini parameters, 3-5
- sctoito, 4-22
- section header names, 5-10, 5-21
- send, 5-8
- Service Information, 4-11
- service information, 4-11
- Service Management, 1-4
- ServiceCenter
 - closing problems, 4-9
 - core applications, 1-4
 - down time, 4-12
 - Events
 - problem open, 5-43
 - Filtering, 4-17
 - Help Desk, 4-18
 - Location Information, 4-15
 - Main Menu, 4-18
 - Main menu, 4-18
 - other services, 4-14
 - Problem
 - assigned problems, 4-13
 - problem listing, 4-4
 - query for probable cause, 4-10
 - service information, 4-11
 - TCL event object
 - See TCL event object, 5-7
 - User Directory, 4-17
 - Vendor Information, 4-16
- ServiceCenter business logic configuration, 5-42
- ServiceCenter Business Logic Topics, 1-21
- ServiceCenter client installation, 2-3
- ServiceCenter Event Services, 5-42
- ServiceCenter menu options, 4-2
- ServiceCenter Problem Management, 5-45
- Services
 - Filtering, 4-17
 - Help Desk, 4-18
 - Location Information, 4-15
 - Main Menu, 4-18
 - other services, 4-14
 - User Directory, 4-17
 - Vendor Information, 4-16
- set_evfield, 5-8
- set_evtype, 5-7
- SLA Management, 1-4
- start process, 4-14
- starting SCITO, 3-1
- starting SCITO processes, 3-1
- static map file, 5-9
- status, 4-14
- stop process, 4-14
- stopping SCITO, 3-4
- stopping SCITO processes, 3-1
- support, contacting, A-1
- system prerequisites, 2-1

T

- TCL event mapping (ServiceCenter to VP Operations), 5-16
 - configuration overview, 5-16
 - TCL variables, 5-17
- TCL event object, 5-7
 - create_sc_event, 5-7
 - print, 5-8
 - send, 5-8
 - set_evfield, 5-8
 - set_evtype, 5-7
 - summary of ServiceCenter TCL commands, 5-7

TEC

- Events
 - problem open, 5-43
- technical support, contacting, A-1

Tivoli

- Events
 - problem open, 5-43

Trouble Ticket Interface (TTI), 1-6

troubleshooting

- problem ticket modifications not reaching ITO message browser, 3-7
- problem tickets not being created by ITO event messages, 3-6

troubleshooting SCITO, 3-6

U

- uni-directional automatic problem ticket creation, 6-1
- updating a problem, 4-7
- User Directory, 4-17
- user directory, 4-17

V

- Vendor Information, 4-16
- vendor information, 4-16
- VP Operations business logic configuration, 5-25
- VP Operations business logic topics, 1-20

W

- Web site, A-2
- Work Management, 1-4

