HP Business Availability Center

for the Windows and Solaris operating systems

Software Version: 7.50

Discovery and Dependency Mapping

Document Number: BACDISC7.50/01
Document Release Date: May 2008
Software Release Date: May 2008

Legal Notices

Warranty

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Third-Party Web Sites

HP provides links to external third-party Web sites to help you find supplemental information. Site content and availability may change without notice. HP makes no representations or warranties whatsoever as to site content or availability.

Copyright Notices

Trademark Notices

Adobe® and Acrobat® are trademarks of Adobe Systems Incorporated.

Intel®, Pentium®, and Intel® Xeon™ are trademarks of Intel Corporation in the U.S. and other countries.

Java™ is a US trademark of Sun Microsystems, Inc.

Microsoft®, Windows®, Windows NT®, and Windows® XP are U.S registered trademarks of Microsoft Corporation.

Oracle® is a registered US trademark of Oracle Corporation, Redwood City, California.

Unix® is a registered trademark of The Open Group.

Documentation Updates

This manual's title page contains the following identifying information:

- Software version number, which indicates the software version
- Document release date, which changes each time the document is updated
- Software release date, which indicates the release date of this version of the software

To check for recent updates, or to verify that you are using the most recent edition of a document, go to:

http://h20230.www2.hp.com/selfsolve/manuals

Support

You can visit the HP Software Support Web site at: **www.hp.com/go/hpsoftwaresupport**

HP Software online support provides an efficient way to access interactive technical support tools. As a valued support customer, you can benefit by using the support site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HP support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and sign in. Many also require a support contract. To find more information about access levels, go to:
http://h20230.www2.hp.com/new_access_levels.jsp

To register for an HP Passport ID, go to:
http://h20229.www2.hp.com/passport-registration.html

Table of Contents

PART II: ADMINISTRATION

PART III: ADVANCED DISCOVERY

Welcome to This Guide

This guide describes how to install the Discovery and Dependency Mapping (DDM) Probe, how to manage the DDM process and to automatically discover and map IT infrastructure resources and their interdependencies. DDM can discover such resources as applications, databases, network devices, different types of servers, and so on. For users with an advanced knowledge of discovery, there is a section on pattern-writing.

This chapter includes:

How This Guide Is Organized

The guide contains the following chapters:

Part I Probe Installation

Explains how to install the DDM Probe and explains license types.

Part II Administration

Describes the main concepts, tasks, and reference for the Run Discovery, Set Up Discovery Probes, Manage Discovery Resources, and Show Status Snapshot applications.

Part III Advanced Discovery

This section is intended for users with an advanced knowledge of Discovery and Dependency Mapping. It explains how to write custom patterns and how to run discovery for many system components. This section also explains how to use the HP Discovery and Dependency Mapping Web Service API to manage discovery and dependency.

Who Should Read This Guide

This guide is intended for the following users of HP Business Availability Center:

- HP Business Availability Center administrators
- HP Business Availability Center platform administrators
- HP Business Availability Center application administrators
- HP Business Availability Center data collector administrators

Readers of this guide should be knowledgeable about enterprise system administration, have familiarity with ITIL concepts, and be knowledgeable about HP Business Availability Center in general and HP Universal CMDB technology specifically.

Getting More Information

For a complete list of all online documentation included with HP Business Availability Center, additional online resources, information on acquiring documentation updates, and typographical conventions used in this guide, see the *HP Business Availability Center Deployment Guide* PDF.

Part I

Probe Installation

1

Installing the Probe

This chapter describes the procedures that are needed for the installation of the Discovery and Dependency Mapping (DDM) Probe on a Windows platform.

You cannot install the Probe on a Solaris machine. If the HP Universal CMDB server is installed on a Solaris machine, install the DDM Probes from the Windows CD-ROM.

This chapter includes:

Tasks

Reference

Install the Probe

This task describes how to install the DDM Probe.

The Probe can be installed before or after you install the HP Business Availability Center server. However, during Probe installation you must provide the server name, so it is preferable to install the server before installing the Probe.

It is recommended to install the Probe on a separate server from the Business Availability Center server, to distribute the overall system load.

To install the DDM Probe:

1 Select **Admin** > **Platform** > **Setup and Maintenance** > **Downloads**.

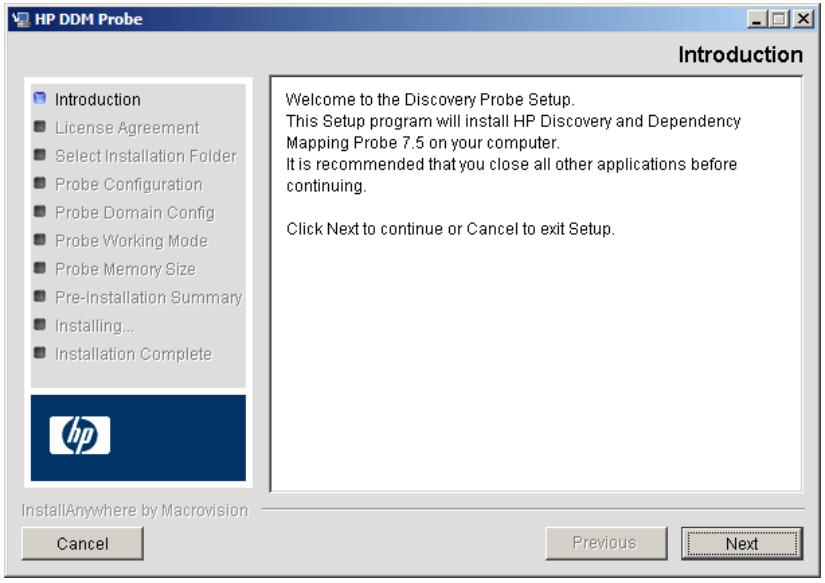
Note: The Probe link in the Downloads page is displayed only if you have purchased a standard or advanced license for Discovery and Dependency Mapping, and if the administrator has added the Probe link to the Downloads page. For details, see "Licensing Models" on page 25 and "Installing Component Setup Files" in the *HP Business Availability Center Deployment Guide* PDF.

---

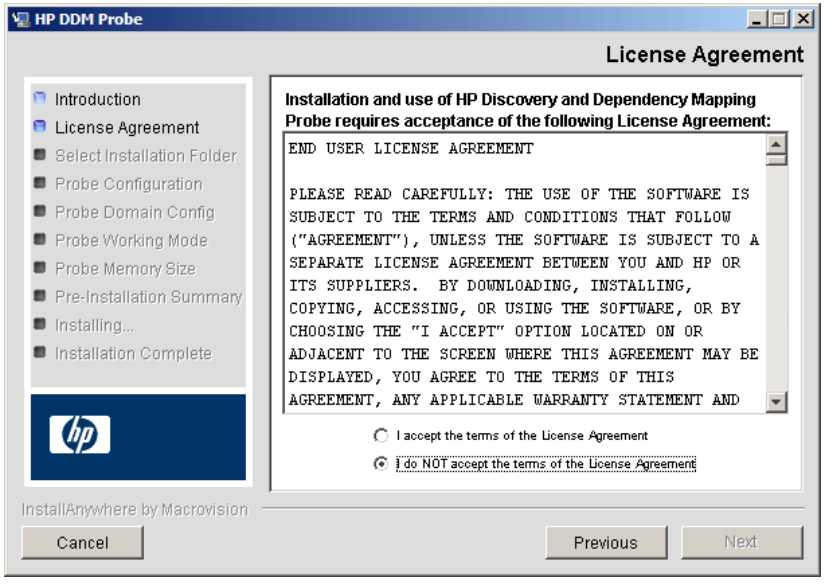
2 Click **Discovery Probe for Windows 2000/2003/XP**. You can open the Setup file or save it to your computer:

- If you choose to open the file, it is not saved to your computer, and the setup program starts immediately. In this case, depending on your browser security settings, a security warning dialog box may open. Confirm that you want to proceed.
- If you choose to save the file to your computer, double-click the downloaded file to begin installation.

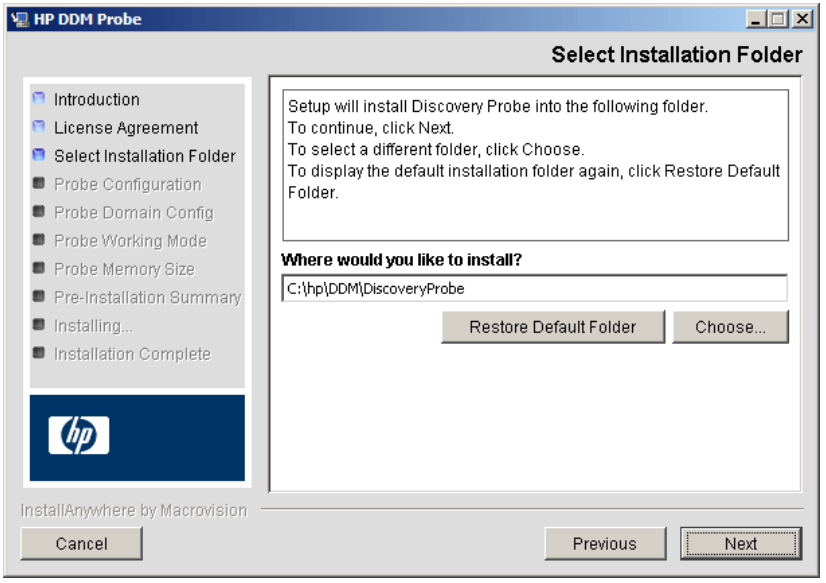
A progress bar is displayed. Once the initial process is complete, the Introduction dialog box opens.



3 Click **Next** to open the License Agreement dialog box.



4 Accept the terms of the agreement and click **Next** to open the Select Installation Folder dialog box.

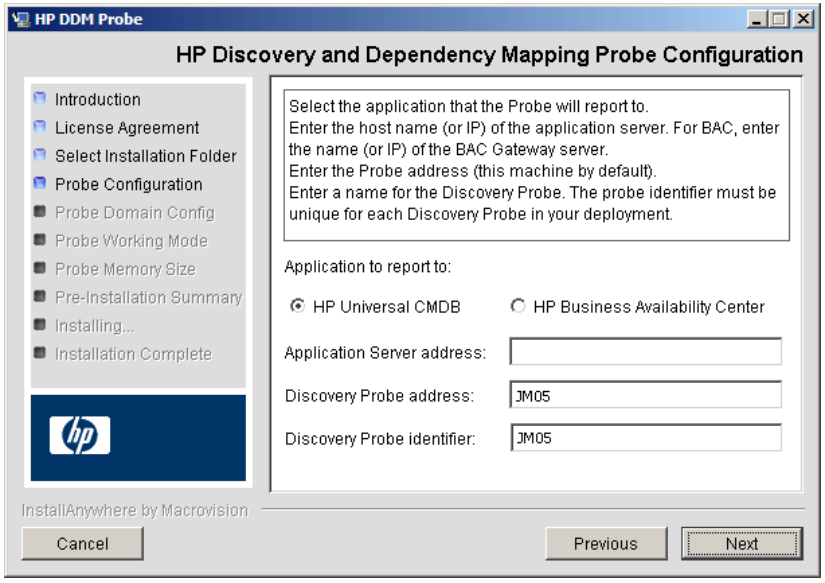


Accept the default entry or click **Choose** to display a standard Browse dialog box. To install to a different directory, browse to and select the installation folder.

Note: To restore the default installation directory, after selecting a directory in the Browse dialog box, click **Restore Default Folder**.

---

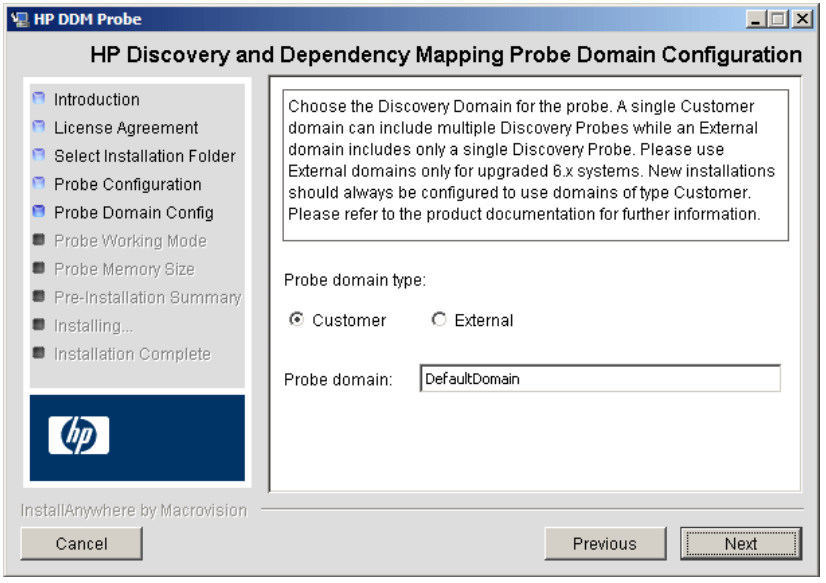
5 Click **Next** to open the HP Discovery and Dependency Mapping Probe Configuration dialog box.



- **Application to report to.** Choose the application server with which you are working. You can use the Probe with either HP Universal CMDB or HP Business Availability Center.
- If you select HP Universal CMDB, in the **Application Server address** box, enter the name or the IP address of the HP Universal CMDB server to which the Probe is to be connected.
- If you select HP Business Availability Center, in the **Application Server address** box, enter the IP or the DNS name of the Gateway Server.
- In the **Discovery Probe address** box, enter the IP address or the DNS name of the machine on which you are currently installing the Probe, or accept the default.
- In the **Discovery Probe identifier** box, enter a name for the Probe that will be used to identify it in the world.

Important: The UCMDB Probe identifier must be unique for each Probe in your deployment.

6 Click **Next** to open the HP Discovery and Dependency Mapping Probe Domain Configuration dialog box.



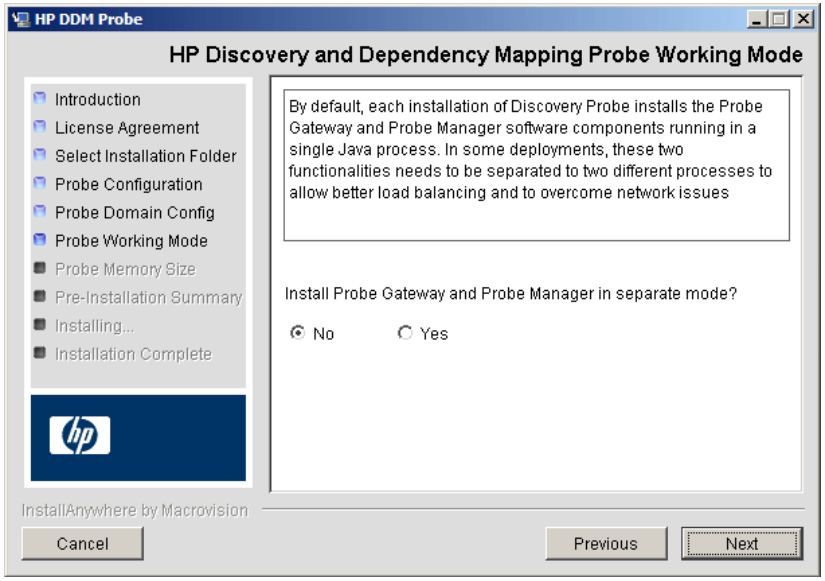
Choose between **Customer** and **External**, depending on the type of domain on which the Probe is to be running:

- **Customer.** Select if you are installing one or more Probes in your deployment.

Important: For new installations, always select **Customer**.

- **External.** Select if you are upgrading from version 6.x systems.
- **Probe domain.** Accept the default domain name or enter another domain name.

7 Click **Next** to open the HP Discovery and Dependency Mapping Probe Working Mode dialog box.

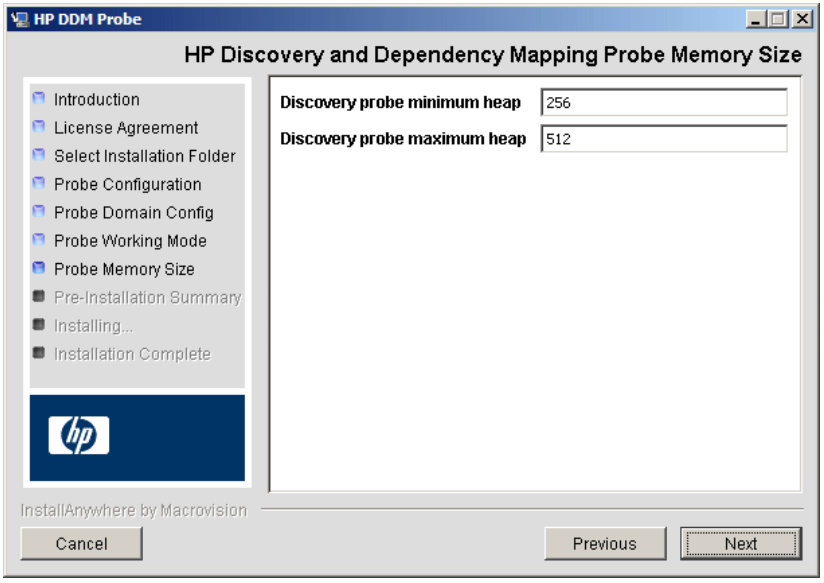


You can run the Probe Gateway and Manager as one Java process or as separate processes. You would probably run them as separate processes in deployments that need better load balancing and to overcome network issues.

For details about running Probe Gateway and Probe Manager, see “DDM Probe Tasks” on page 28.

Click **No** to run Probe Gateway and Probe Manager as one process. Click **Yes** to run Probe Gateway and Probe Manager as two processes.

8 Click **Next** to open the HP Discovery and Dependency Mapping Probe Memory Size dialog box.

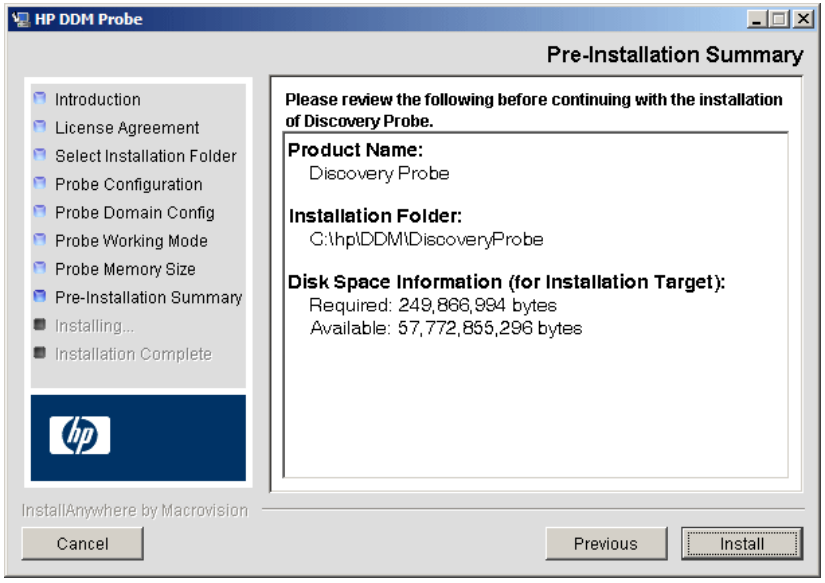


Define the minimum and maximum memory to be allocated to the Probe. The values are measured in megabytes.

Note: For large systems (more than one hundred thousand CIs), it is recommended to enlarge the minimum and maximum heaps to 1 GB. Increase your memory max heap size for the Probe Manager. Also, separate the Probe Gateway and Manager to run as individual processes (defined in the previous step).

---

9 Click **Next** to open the Pre-Installation Summary dialog box and review the selections you have made.



10 Click **Install** to complete the installation of the Probe. When the installation is complete the Install Complete page is displayed.

Note: Any errors occurring during installation are written to the following file: **<DDM Probe root directory>\DiscoveryProbe \Discovery_Probe_InstallLog.log**.

---

11 Click **Done**. The following shortcut is added to the Windows **Start** menu:

Programs > HP DDM > DDM Probe

12 Activate the Probe by selecting the shortcut.

Upgrade the Probe

This task describes how to upgrade the DDM Probe.

This task includes the following steps:

- "Uninstall the Old Probe" on page 22
- "Install the New Probe" on page 22

1 Uninstall the Old Probe

Uninstall all existing Probes. If a Probe is running, stop it before you uninstall it.

2 Install the New Probe

For details on installation, see "Install the Probe" on page 14.

Note:

- You should install the new Probe with the same configuration, that is, use the same Probe ID, domain name, and server name as for the previous Probe installation.
- You must reactivate active jobs after the upgrade so that the newly installed Probes receive the tasks they are assigned.

Probe Installation Requirements

Hardware Requirements

Computer/processor	**Windows:** Pentium IV 2.4 GHz or later processor
Memory	**Windows:** Minimum 1 GB RAM (Recommended: 2 GB RAM)
Virtual memory (for Windows deployment)	Minimum 2 GB **Note:** The virtual memory size should always be at least twice the physical memory size.
Free hard disk space	**Windows:** Minimum 4 GB (at least 4 GB for database software and data files) (Recommended: 20 GB hard disk)
Display	**Windows:** Color palette setting of at least 256 colors (32,000 colors recommended).

Software Requirements

Operating system	**Windows:** ➤ Windows 2000 Server/Advanced Server, Service Pack 4 or later ➤ Windows 2003 Standard/Enterprise editions, Service Pack 1, Service Pack 2
Java Runtime Environment	JRE 1.5.0 (installed with the product)

2

Licensing Models

This chapter provides information on the Discovery and Dependency Mapping (DDM) licensing models.

This chapter includes:

Concepts

Licensing Models Overview

There are three levels of licensing:

- **Universal CMDB Foundation (included in all Business Availability Center licenses).** Business Availability Center includes the complete class model (all CITs) and all packages. (You populate the UCMDB either manually or using integration.) The Universal CMDB Foundation license does not include Discovery and Dependency Mapping (DDM), but does include Federation (the ability to federate with other data sources and reconcile CIs), CIT Manager, and Package Manager. This is the default license.
- **DDM Standard.** This license includes DDM and enables discovery of the infrastructure and network.
- **DDM Advanced.** This license includes DDM, enables discovery of all components at your site, and supports advanced custom pattern-writing.

  For a complete list of features that are supported by each license, contact HP Software Support.

Upgrading to DDM Standard or DDM Advanced License

When you install Business Availability Center you receive the Universal CMDB Foundation license. To obtain the file needed to upgrade to the DDM Standard or DDM Advanced license, contact HP Software Support.

To upgrade Business Availability Center:

1. Obtain the appropriate file from HP Software Support: **standard_ucmdb_license.xml** or **advanced_ucmdb_license.xml**.
2. Stop the Business Availability Center server.
3. Place the file in the <**Business Availability Center root directory**>**mam_lib\server** folder on the Processing server machine.

   If Business Availability Center is installed in a distributed deployment, on the Gateway Server machine, use the JMX console to force a license change:

   a. Launch the Web browser and enter the address **http://<server_name>:8080/jmx-console**, where <**server_name**> is the name of the machine on which Business Availability Center is installed. When prompted, enter the JMX Console authentication credentials (if you do not have these credentials, contact your system administrator).

   b. Under MAM, click **service=UCMDB UI** to open the JMX MBEAN View page.

   c. Locate **java.lang.String getLicenseForCustomer()** and enter the following information:

      In the force parameter box, select **True**.

      In the ParamValue box for the parameter **customerId**, enter **1**.

      Click **Invoke**.

   Note: To verify the type of license that is installed, select **False** and enter the customer ID. Details about the license are displayed.

   ---

4. Start the Business Availability Center server.

3

The Discovery and Dependency Mapping (DDM) Probe

This chapter provides information on the DDM Probe.

This chapter includes:

Concepts

Tasks

Reference

DDM Probe Tasks

This section explains how the DDM Probe manages tasks.

The DDM Probe comprises two components: the Probe Gateway and the Probe Manager.

- The Probe Gateway provides communication (http or https) between the Probe Manager and the Business Availability Center server, for processes such as downloading tasks and returning task results.
- The Probe Manager runs the DDM process itself.

The Probe Gateway communicates with the Probe Manager using RMI.

By default, the Gateway and Manager run as a single process but they can be configured (during installation) to reside on separate processes. Moreover, several Probe Managers can be configured to connect to a single Probe Gateway. This can be useful if you want a specific Probe Manager to deal with certain DDM jobs.

This section includes the following topics:

Stage 1. Probe Gateway

The Business Availability Center server does not initiate tasks on the Probe; it is the Probe's responsibility to request relevant tasks to run.

Stage 2. Business Availability Center Server

At the same time as the Probe requests tasks from the server, it also sends to the server the last update time of its configuration and the last task ID received. The server returns to the Probe one of the following:

- Updated server data (in the case that the configuration on the Probe is not current). The server data includes: Python scripts, patterns, the Domain Scope Document dictionary file, and so on. For details, see "Probe Configuration Update" on page 32. For details on using the Domain Scope Document to harden DDM, see "Hardening Discovery and Dependency Mapping" in the *HP Universal CMDB Deployment Guide* PDF.
- The last task sent (if there is a mismatch between the Probe and the server last task ID).
- New tasks to run (if any exist):
  - If a job has been deactivated, the server sends a **delete job** message to the Probe.
  - If a job has been activated, the server sends a **run new job** message to the Probe.
- The Business Availability Center server sends the Probe a response (in XML format) with the new task data. Each task contains the job and pattern names, and the relevant Trigger CI data.

  The number of Trigger CIs for a task is limited (100 by default). For example, if an active job includes 1000 destinations, the job is sent to the Probe in 10 tasks with 100 Trigger CIs in each task.

Stage 3. Probe Gateway

- When the Probe Gateway receives tasks from the Business Availability Center server, it saves them to its local database (MySQL).
- Periodically, a thread on the Probe Gateway scans the database for tasks and sends them to the Probe Manager. This process enables load balancing in DDM when there are multiple Probe Managers for each Probe Gateway.

Stage 4. Probe Manager

- The tasks on the Probe Manager are scheduled using the Quartz third-party library. When tasks are completed, the Probe Manager sends the results (in XML format) to the Probe Gateway. (For details on Quartz, refer to the documentation at http://www.opensymphony.com/quartz/.)
- The Probe Manager receives a set of result objects. The Probe Manager first performs processing on the results (for example, filters results, runs the result redundant mechanism), and only then prepares the results for sending to the Probe Gateway.
- The results are stored in the Probe Manager database.
- A thread seeks the database for results that are ready to be sent to the Probe Gateway. These results are merged into a single result, whose size does not exceed a maximum result size (currently 20,000), as defined in the discoveryProbe.properties file:

```
appilog.agent.local.maxTaskResultSize = 20000
```

  When results reach the Gateway, it immediately responds with a success or failure reply. Based on this acknowledgement from the Gateway, the Probe Manager marks the results as **ack** in the database, so that they are not sent again during the next cycle.
- The task results that have been acknowledged by the Gateway remain in the Probe Manager database till they are deleted—once a week.
- When results reach the Probe Gateway, they are not sent directly to the server, but are stored in the Gateway database, to avoid flooding the server with data.

Stage 5. Probe Gateway

- A dedicated thread on the Probe Gateway scans the database and searches for task results that are ready to be sent to the server. These results are sent to the server by the Probe Gateway, using the **sendResultsToServer()** API.
- If the size of data that needs to be sent is too large, it is sent in chunks (max. 50,000). The information is then updated in the CMDB (using create, update, or remove).

➤ Finally, the Probe Gateway verifies that the server has finished handling the results, deletes those results from its database (so they are not sent again to the server), and continues sending results to the server, if any more results exist.

Stage 6. Probe and Server Synchronization Process

After reading a predefined number of tasks, the Probe confirms these tasks with the server. (This process prevents the need for manually reactivating patterns or jobs.)

1 The Probe sends to the server the names of all activated jobs and the number of Trigger CIs for each job.

2 The server checks that the number matches that in the CMDB:

- ➤ If a job is missing from the Probe, the server redispatches the job to the Probe.
- ➤ If the Probe has less or more than the number of CIs on the server, the server returns the names of the problematic jobs and their CIs to the Probe.

3 The Probe checks the problematic jobs' list. If the Probe includes a job that does not exist on the server, the Probe sends a **remove job** remote method call to all Probe Managers.

4 If the Probe includes a CI that does not exist on the server, the Probe sends a **remove CI** remote method call to all Probe Managers.

5 If the server includes a Trigger CI that does not exist on the Probe, the Probe requests this CI from the server. The server returns the task (in XML format) and the Probe distributes this task.

Probe Configuration Update

- To perform discovery, the Probe needs resource data, such as the Domain Scope Document dictionary file, scripts, and so on. The Probe is updated automatically with these resources. Along with each task request from the Probe Gateway to the server, the Probe Gateway sends the last (server) update time of its latest updated resources.
- The server, before returning any new tasks, validates that there are no more recently updated resources. If there are, instead of returning the regular queued tasks for the Probe, the server returns a special, crafted task for updating the Probe's resources.
- When the Probe receives this task, it sends a **GetResouces()** request to the server, which returns a list of resources that have not been updated to the Probe. In that way the Probe is always updated with the latest system configuration files.

Data Validation on the DDM Probe

From version 7.0, the CIT model also resides on the DDM Probe. This enables data validation to take place on the Probe when receiving data from services. Problems are generated for a specific Trigger CI and displayed to the user. For details, see "Discovery Status Pane" on page 126.

The following validation takes place on the Probe:

- The CIT of the CI is compared to that in the CIT model.
- The CI is checked to verify that all key attributes are present (on condition that the CmdbObjectId attribute is not defined).
- The CI's attributes are checked to verify that they are all defined in the CIT.
- The CI's attributes of type STRING are checked to verify that they do not exceed the size limit. If an attribute is longer than the limit, DDM checks whether an AUTO_TRUNCATE qualifier is defined for the attribute. If there is a qualifier, the value is truncated and a warning message is written to the Probe error.log file.

All invalid attributes raise a CollectorsProcessException exception, which reports on a specific CI. When the Probe finds invalid data that is related to the CITs, all data that the Probe has collected on that CI is dropped by the Probe and is not sent to the server.

For details on attributes, see "CI Type Attributes" in *CI Attribute Customization*.

Filtering Results

You can filter results sent by the Probe to the Business Availability Center server. You would probably want to filter irrelevant data regularly during production runs and specifically when you are testing a limited environment.

There are two levels of filtering: pattern filtering and global filtering:

- **Pattern filtering.** DDM filters the results for a specific pattern and sends to the CMDB only those filtered CIs. You define a pattern filter in the Pattern Management pane. For details, see "Pattern Management Tab" on page 189.
- **Global filtering.** DDM filters the results of all jobs running on a Probe. You define global filters in the globalFiltering.xml file. For details, see "globalFiltering.xml" on page 171.

  The order of filtering is as follows: during a run, DDM first searches for a pattern filter and applies the filter to the results of the run. If there are no pattern filters, DDM searches for a global filter and applies that filter to the results. If DDM finds no filters, all results are sent to the server.

Blocking the Domain Scope Document Credentials

The Probe's file system holds (by default) both the encryption key and the Domain Scope Document. Each time the Probe is started, the Probe retrieves the Domain Scope Document from the server and stores it on the file system. To prevent unauthorized users from obtaining these credentials, you can configure the Probe so that the Domain Scope Document is held in the Probe's memory and is not stored on the Probe file system.

To change the configuration, access **DiscoveryProbe.properties** and change:

```
appilog.collectors.storeDomainScopeDocument=true
```

to

```
appilog.collectors.storeDomainScopeDocument=false
```

The Probe Gateway and Probe Manager serverData folders no longer contain the domainScopeDocument.bin file.

For details on using the Domain Scope Document to harden DDM, see "Hardening Discovery and Dependency Mapping" in the *HP Universal CMDB Deployment Guide* PDF.

Install the DDM Probe

This section describes the DDM Probe installation procedure.

Note:

- The Probe link in the Downloads page is displayed only if you have purchased a license for the DDM application.
- The managed environment is defined by the IP ranges of the domains. However, with some patterns it is possible to override this behavior and discover CIs that are out of a Probe's range.

This task includes the following steps:

- "Install the Probe" on page 35
- "Launch the Probe" on page 35
- "Run Discovery and Dependency Mapping" on page 36

1 Install the Probe

For details, see Chapter 1, "Installing the Probe."

2 Launch the Probe

On the machine on which the Probe is installed, select **Start** > **Programs** > **HP DDM** > **DDM Probe** to start the Probe. A Command Prompt window opens. To verify that the Probe has been launched successfully, in Business Availability Center select **Admin** > **Universal CMDB** > **Discovery** > **Set Up Discovery Probes**. Select the Probe and, in the Details pane, verify that the status is **connected**.

For details on how the Probe works, see "DDM Probe Tasks" on page 28.

3 Run Discovery and Dependency Mapping

For details, see "Working in Basic Mode or Advanced Mode" on page 96.

The DiscoveryProbe.properties File

A DDM process needs several parameters to be activated. These parameters specify the method to be used (for example, ping five times before declaring a failure) and on which CI a method should be used. If parameters have not been defined by the user, the DDM process uses the default parameters defined in the **DiscoveryProbe.properties** file. To edit the parameters, open **DiscoveryProbe.properties** in a text editor.

The DiscoveryProbe.properties file is located in **<DDM Probe root directory>\DiscoveryProbe\root\lib\collectors**.

Note: If you update the **DiscoveryProbe.properties** parameters, you must restart the Probe so that it is updated with the changes.

The **DiscoveryProbe.properties** file is divided into the following sections:

- **Server Connection Definitions.** Contains parameters that are needed to set up the connection between the server and the Probe, such as the protocol to be used, machine names, default Probe and domain names, timeouts, and basic authentication.
- **DDM Probe Definitions.** Contains parameters that define the Probe, such as root folder location, ports, and Manager and Gateway addresses.
- **Probe Gateway Configurations.** Contains parameters that define time intervals for retrieving data.
- **Probe Manager Configurations.** Contains parameters that define Probe Manager functionality, such as scheduled intervals, result grouping, chunking, threading, timeouts, and filtering.
- **I18N Parameters.** Contains parameters that define language settings.

- **Internal Configurations.** (Caution: These parameters should not be changed without an advanced knowledge of Discovery and Dependency Mapping.) Contains parameters that enable DDM to function efficiently, such as thread pool size.

Part II

Administration

4

Discovery and Dependency Mapping Introduction

This chapter provides information on Discovery and Dependency Mapping.

This chapter includes:

Concepts

Tasks

Reference

Discovery and Dependency Mapping – Overview

The Discovery and Dependency Mapping (DDM) process is the mechanism that enables you to collect information about your system by discovering the IT infrastructure resources and their interdependencies. DDM automatically discovers and maps logical application assets in Layers 2 to 7 of the Open System Interconnection (OSI) Model.

DDM discovers resources such as applications, databases, network devices, servers, and so on. DDM also communicates with industry standard or application APIs. Each discovered IT resource is delivered to, and stored in, the configuration management database (CMDB) where the resource is represented as a managed CI.

DDM is an ongoing, automatic process that continuously detects changes that occur in the IT infrastructure and updates the CMDB accordingly. You do not need to install any agents on the devices to be discovered.

Following installation, the network on which the DDM Probe is located, the host on which the Probe resides, and the host's IP address are automatically discovered and a CI is created for each of these objects. These discovered CIs are placed in the CMDB. They act as triggers that activate a DDM job. Every time a job is activated, the job discovers more CIs, which in turn are used as triggers for other jobs. This process continues until the entire IT infrastructure is discovered and mapped.

Once you configure DDM and activate the required patterns, DDM runs on the system, discovers system components, and saves them as CIs in the CMDB. You can discover new objects either manually or automatically. Objects that are outside the Probe's network require additional, manual configuration.

Agentless Technology

DDM is an agentless technology that discovers IT environment components through a dedicated Probe residing on the customer's site. For example, the Netlinks discovery module discovers TCP/IP connections from received NetFlow data.

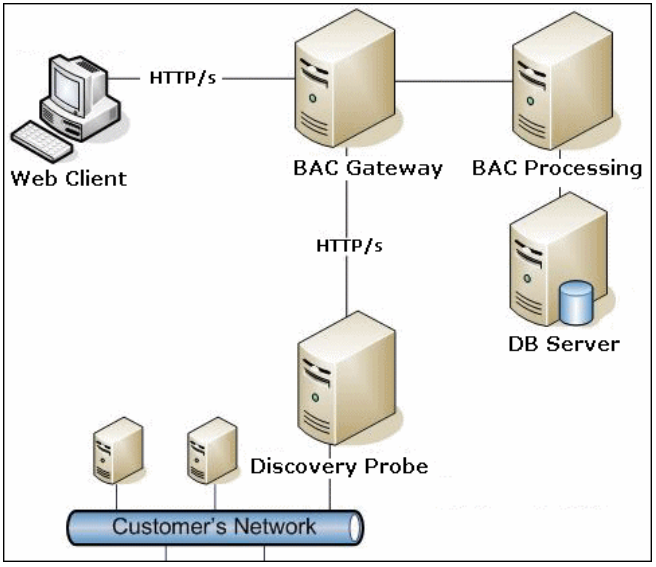
The Probe connects to HP Business Availability Center via http or https traffic to receive new tasks, send task results, and so on. For details on the Probe workflow, see "DDM Probe Tasks" on page 28.

Although DDM is agentless, that is, it does not require the installation of any agent on a customer's machine, it does depend on agents that are already installed such as:

- **SNMP Agent.** Provides information about the operating systems, device types, installed software, and other system resources information. SNMP agents can usually be extended to support new MIBs, exposing more data for managerial purposes.
- **WMI Agent.** Microsoft's remote management agent, which is usually available for access by a remote administrator. The WMI agent is also extensible by adding WMI providers to the generic agent.
- **Telnet/SSH Agent (or daemon).** Used mostly on UNIX systems to connect remotely to a machine and to launch various commands to obtain data.
- **xCmd.** A remote administration technology similar in functionality to Telnet/SSH that enables launching any console command over Windows machines. xCmd relies on Administrative Shares & Remove Service Administration APIs to function correctly.
- **Application specific.** This agent depends on the remote application to function as an agent and respond appropriately to the Probe's remote queries, for example, database discoveries, Web server discoveries, and SAP and Siebel discoveries.

Discovery and Dependency Mapping Architecture

Discovery and Dependency Mapping architecture is deployed as follows:



DDM User Interface

- The DDM Probe is the component that performs the data collection and runs on the customer's network.
- The user interface servlet (collectorsUtilities) and the Probe servlets (collectorsServlet, collectorsResultsServlet, and collectorsDownloadServlet) reside on the HP Business Availability Center Gateway Server machine.
- To process clients' requests, these components interact with the HP Business Availability Center Data Processing Server running the DDM components, the viewing system, and the CMDB.

Discovery and Dependency Mapping Components

This section includes the following topics:

DDM Probe

The Probe is the main component responsible for requesting tasks from the server, dispatching them, and sending the results back to the CMDB through the server. You define a range of network addresses for a specific, installed Probe. Each Probe is identified by its name. For details on how the Probe functions, see "DDM Probe Tasks" on page 28.

The DiscoveryProbe.properties file contains configuration parameters. The file is located in **\<DDM Probe root directory>\DiscoveryProbe \root\lib\collectors**. For details, see "The DiscoveryProbe.properties File" on page 36.

HP Business Availability Center Servers

The HP Business Availability Center Gateway Server hosts the servlets that deliver requests to the Probe. The Processing Server receives the results and stores the collected data in the CMDB.

Discovery Modules

The module is a grouping of jobs that logically belong together, can be operated and managed together, and so on. This helps to reduce clutter in the main view when many jobs need to be written, and can also offer better manageability.

When creating a job, you need to choose a module for it or create a new module. If you are creating several jobs, the best practice is to split them into logical groups and assign them to modules accordingly.

Discovery Jobs

A job enables reuse of a pattern for different DDM processes. Jobs enable scheduling the same pattern differently over different sets of triggered CIs and also supplying different parameters to each set.

Note: To activate DDM, you activate jobs—organized in modules—and not patterns.

For details on discovering specific components, see Chapter 9, "Discovery and Dependency Mapping Content."

Jobs are organized in modules as follows:

- **Network and host resources.** You can discover resources on Windows and UNIX hosts, for example, disk information, running processes or services, and so on.
- **Applications.** The Application modules discover Oracle E-Business Suite components, the SAP environment based on Computer Center Management System (CCMS), the Siebel environment (such as the Siebel topology and database), and Web services such as the UDDI registry.
- **Database modules** (Oracle, DB2, Sybase, Microsoft SQL Server). DDM first finds instances of databases, then of the database resources (for example, users, tables, tablespaces) for each database instance. Business Availability Center includes predefined default views of the DB2, Oracle, and Microsoft SQL Server databases.

- **J2EE applications.** DDM discovers JBoss, Oracle Application Server, WebLogic and WebSphere components.
- **Web servers.** DDM discovers Microsoft IIS for Windows and IBM HTTP Server.

Discovery and Dependency Mapping Wizards

You use one of the DDM wizards (to discover the network, databases, and J2EE applications) when you want to use the default values set for IP ranges, network credentials, and so on. For details on using a wizard to run DDM, see "Basic Mode Window" on page 108.

Protocols

Discovery of the IT infrastructure components uses protocols such as SNMP, WMI, JMX, Telnet, and so on. For details, see "Domain Credential References" on page 83.

Patterns

A pattern includes parameters and an input TQL that describes the potential input CIs. Patterns are one of the resources of a Discovery and Dependency Mapping job. A pattern also includes scripts and other code needed for discovery. For details on pattern-writing, see Chapter 10, "Content Development and Pattern-Writing."

Note: From version 7.0, you no longer have to edit pattern XML files. To make pattern changes, use Manage Discovery Resources. For details, see "Manage Discovery Resources Window" on page 199.

Configuration Files

Configuration files include properties and parameters that are relevant for the DDM patterns. For example, the portNumberToPortName.xml file (that maps a discovered port's number to a port name) includes a list of ports used by DDM when discovering networks. For details on user-definable files, see "Resource Files" on page 169.

External Resources

External resources include all resources external to Business Availability Center that are needed in DDM, for example, a Visual Basic file, a credentials file, and so on.

Packages

Packages contain definitions, resources, and tools that enable you to discover IT infrastructure resources such as network extensions, applications, and databases. For details, see "Package Manager" in *Model Management*.

Scripts

Business Availability Center uses Jython scripts for pattern writing. For example, the SNMP_Connection.py script is used by the SNMP_NET_Dis_Connection pattern to try and connect to machines using SNMP. Jython is a language based on Python and powered by Java. For details on pattern-writing, see Chapter 10, "Content Development and Pattern-Writing."

For details on how to work in Jython, you can refer to these Web sites:

- http://www.jython.org
- http://www.python.org

Discovery and Dependency Mapping Applications

Discovery and Dependency Mapping includes the following applications:

- "Run Discovery" on page 49
- "Set Up Discovery Probes" on page 49
- "Manage Discovery Resources" on page 49
- "Show Status Snapshot" on page 49

Run Discovery

The Run Discovery application enables you to manage the DDM modules and jobs (required for discovering a specific group of CIs). You run the process by activating jobs. You can choose to activate all or some of the jobs in a module. You can also edit jobs, and you can schedule when a job should run.

For details, see Chapter 6, “Run Discovery.”

Set Up Discovery Probes

Set Up Discovery Probes enables you to add Probes to the system and to edit existing Probes. You define the network range that each Probe must cover.

For details, see Chapter 5, “Set Up Discovery Probes.”

Manage Discovery Resources

Note: Only users with an advanced knowledge of Discovery and Dependency Mapping should make changes to the resources.

Manage Discovery Resources enables you to view the resources that are needed to perform discovery. You can edit patterns, scripts, configuration files, and you can replace or remove external resources needed in DDM. For details, see “Manage Discovery Resources Window” on page 199.

For details on the Manage Discovery Resources user interface, see Chapter 7, “Manage Discovery Resources.”

Show Status Snapshot

Show Status Snapshot enables you to view details about the scheduling of a particular job as well as job statistics. You can also view the report results in a My BAC portlet.

For details, see Chapter 8, “Show Status Snapshot.”

Trigger CITs, Trigger CIs, Input TQLs, and Trigger TQLs

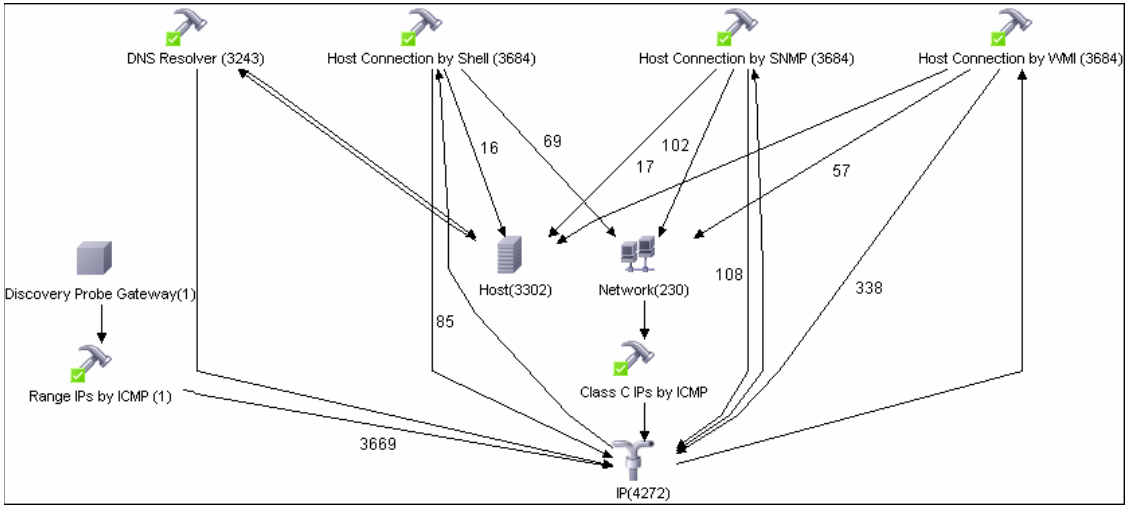
This section describes the functions of Trigger CITs, Trigger CIs, Input TQLs, and Trigger TQLs. For details on these objects, see "Define Pattern Input (Trigger CIT and Input TQL)" on page 294 and other sections in Chapter 10, "Content Development and Pattern-Writing." For details on TQLs, see "Topology Query Language" in *Model Management*.

Trigger CITs

The Trigger CIT defines which CIT is used as the input for a pattern. For example, for a pattern that is going to discover IPs, the input CIT is Network.

Example – Trigger CIT Activating Jobs

In the following example, the **Network** CIT is a trigger that activates the **Class C IPs by ICMP** job. The **Class C IPs by ICMP** job then discovers instances of IP addresses. These discovered IP addresses themselves become CIs that act as triggers to activate the **Host Connection** jobs, which in turn discover more IP addresses and Network CIs. The process ends when all the IP addresses included in the range defined for the Probe are discovered.



Trigger CIs

A Trigger CI is a CI in the CMDB that activates a job. Every time a job is activated, the job discovers more CIs, which in turn are used as triggers for other jobs. This process continues until the entire IT infrastructure is discovered and mapped.

For details on adding Trigger CIs to a job, see "Discovery Status Pane" on page 126.

Input TQLs

An Input TQL has two functions:

- **An Input TQL is associated with a pattern.** The Input TQL defines a minimal set of requirements for every Trigger CI included in a job that runs this pattern. (This is true even when no trigger TQL is associated with the job.)

  For example, an input TQL can query for IPs running SNMP, that is, only IPs with installed SNMP agents can trigger this pattern. This prevents the case where a user could manually create a Trigger CI that adds all hosts as triggers to a pattern.

- **An Input TQL defines how to retrieve data information from the CMDB.** Destination data information, even if it is not included in a Trigger CI, can be retrieved by the Input TQL. The Input TQL defines **how** to retrieve the information.

  For example, you can define a relationship between a Trigger CI (a node with the node name of **SOURCE**) and the target CI and then can refer to the target CI according to this node name, in the Triggered CI Data pane. For details, see "Triggered CI Data Pane" on page 197.

  For details on using input TQLs when writing patterns, see "Step 1: Create a Discovery and Dependency Mapping Pattern" on page 294.

Trigger TQLs

A Trigger TQL associated with a job is a subset of the Input TQL, and defines which specific CIs should be the Trigger CIs for a job. That is, if an Input TQL queries for IPs running SNMP, a Trigger TQL queries for IPs running SNMP in the range 195.0.0.0-195.0.0.10.

Note: A Trigger TQL must refer to the same objects as the Input TQL. For example, if an Input TQL of a pattern queries for IPs running SNMP, you cannot define a Trigger TQL for an associated job to query for IPs connected to a host. This is because some of the IPs may not be connected to an SNMP object, as required by the Input TQL.

Manually Activate a Job

You can activate a job by clicking the **Activate** button in the Discovery Modules pane. You can manually activate a CI by disabling the TQL and clicking the **Add TQL** button. (You disable a TQL in the **Edit Probe Limitation for TQL Output** dialog box.) The job runs using only the redispatched CIs. For details, see "Discovery Modules Pane" on page 135.

Manually Create a Network CI

A Probe starts by discovering the network on which it is running, so usually there is no need for you to create a network CI. However, if you install the Probe on a certain network, but configure the Probe to discover objects on another network, DDM is not able to discover the network. You must manually create a network CI for the network that the Probe must discover.

To verify that a network CI exists, access the View Manager (**Admin** > **Universal CMDB** > **Modeling** > **View Manager**). Locate the Network folder and verify that the folder contains a Network Topology view.

For details on manually creating a network CI, see "New CI Wizard" in *Model Management*.

Schedule Modules to Run

You can set a schedule for a job or a module so that it runs at a certain time. For details, see "Discovery Scheduler Dialog Box" on page 138.

Naming Conventions

When naming entities in DDM, you can use the following characters: a-z, A-Z, 0-9. When entering IP addresses, use only digits and asterisks (*).

Log Files

This section describes the DDM log files and explains how to perform basic troubleshooting. Log files store messages, including errors, relating to the activation of jobs. For details on problem management, see "Managing Problems With Error Reporting" on page 97. For details on setting options for communication logs, see "Execution Options Pane" on page 190.

Severity Levels

Each log is set so that the information it records corresponds to a certain severity threshold. Because the various logs are used to keep track of different information, each is pre-set to an appropriate default level. For details on changing the log level, see "Changing Log Levels" below.

Severity levels are listed here from narrowest to widest scope:

- **Fatal.** This level reports serious errors such as a problem with the infrastructure, missing DLL files, or exceptions.
- **Debug.** This level is used by HP Software Support when troubleshooting problems.
- **Error.** This level reports problems that cause DDM not to retrieve data. Look through these errors as they usually require some action to be taken (for example, to increase timeout, to change a range, to change a parameter, to add another user credential, and so on).

- **Warning.** When a run is successful but there may be non-serious problems that you should be aware of, DDM marks the severity as **Warning**. You should look at these CIs to see whether data is missing, before beginning a more detailed debugging session. **Warning** can include messages about the lack of an installed agent or remote host, or that invalid data caused an attribute not to be properly calculated.
- **Info.** The log records all activity. Most of the information is normally routine and of little use and the log file quickly fills up.
- **Success.** The Trigger CI ran successfully.

Note: The names of the different log levels may vary slightly on different servers and for different procedures. For example, **Info** may be referred to as **Always logged** or **Flow**.

Changing Log Levels

If requested by HP Software Support, you may have to change the severity threshold level in a log (that is, to set verbosity), for example, to a debug level.

To change the severity threshold level:

1. Open the log properties file in a text editor. Log file properties are defined in files in the following directory: C:\hp\DDM\DiscoveryProbe\root\logs.
2. Locate the log parameter. For example:

```
log4j.appender.LOGFILE_Performance.Threshold=DEBUG
```

3. Change the level to the required level. For example:

```
log4j.appender.LOGFILE_Performance.Threshold=INFO
```

For a description of the log levels, see "Severity Levels" on page 53.

4. Save the file.

This section includes the following topics:

Server Logs

Server log files reside on the HP Business Availability Center server. They store information about server activity, including error messages, that occurs on the server side.

The following logs are located in **<HP Business Availability Center root directory>\log**.

mamAutoDiscovery.log

Description. Contains information about tasks running on the server. The server provides services to the user interface, such as: activating jobs, processing results from the Probe, or creating tasks for the Probe. In a distributed environment, the file resides on the Data Processing server.

Error Level. All DDM process errors on the server side.

Information Level. Information about requests being processed.

Debug Level. Logs mainly for debugging purposes.

Basic Troubleshooting. Check this log when you have invalid user interface responses or errors you want to explore. This log provides information to enable you to analyze the problems.

mamWebAutoDiscovery.log

This log receives messages from:

- **The Collectors Utilities Servlet.** The user interface connects to the server through this servlet.
- **The Collectors Servlet.** The Probe requests new tasks from the server through this servlet.
- **The Collectors Results Servlet.** The Probe sends new results through this servlet.

- **The Collectors Download Servlet.** The Probe downloads new server data through this servlet.

  In a distributed environment, the file resides on the Gateway server.

  Error Level. All errors in the servlet.

  Information Level. Information about user requests and Probe task requests.

  Debug Level.

  - User requests
  - Probe requests to read DDM tasks.
  - Probe access of the servlet.

  Basic Troubleshooting.

  - User Interface–Server communication problems.
  - Probe–Server communication problems.

  Some processing problems may be written to this log instead of to **mamAutoDiscovery.log**.

mamAutoDiscoveryUpgrade.log

Contains information about the upgrade process.

mamAutoDiscoveryResultsStat.log

Contains the statistics of the results received from the Probe.

Probe Logs

Probe logs store information involving job activation that occurs in the Probe Gateway and Probe Manager.

The logs in this section are located in **<DDM Probe root directory>\DiscoveryProbe\root\logs**.

General Logs

wrapperProbe.log

Records all the Probe's console output in a single log file.

Error Level. Any error that occurs within the Probe Gateway.

Information Level. Important information messages, such as the arrival or removal of a new task.

Debug Level. Record of every Probe access of the servlet.

Basic Troubleshooting. Use this file for any Probe Gateway problems to verify what occurred with the Probe Gateway at any time as well as any important problems it encountered.

probe-error.log

Summary of the errors from the Probe.

Error Level. All errors in the Probe components.

Information Level. N/A

Debug Level. N/A

Basic Troubleshooting. Check this log to verify if errors occurred in the Probe components.

probe-infra.log

List of all infrastructure messages.

Error Level. All infrastructure errors.

Information Level. Information about infrastructure actions.

Debug Level. Messages mainly for debug purposes.

Basic Troubleshooting. Messages from the Probe's infrastructure only.

wrapperLocal.log

Error Level. Any error that occurs within the Probe Manager.

Information Level. Important information messages such as received tasks, task activation, and the transferring of results.

Debug Level. N/A

Basic Troubleshooting. Use this file for any Probe Manager problems to verify what occurred with the Probe Manager at any time as well as any important problems it encountered.

Probe Gateway Logs

probeGW-taskResults.log

This log records all the task results sent from the Probe Gateway to the server.

Error Level. N/A

Information Level. Result details: task ID, job ID, number of CIs to delete or update.

Debug Level. The **ObjectState HolderVecto**r results that are sent to the server (in an XML string).

Basic Troubleshooting.

- If there is a problem with the results that reach the server, check this log to see which results were sent to the server by the Probe Gateway.
- The results in this log are written only after they are sent to the server. Before that, the results can be viewed through the Probe JMX console (use the ProbeGW Results Sender MBean). You may have to log in to the JMX console with a user name and password.

probeGW-tasks.log

This log records all the tasks received by the Probe Gateway.

Error Level. N/A

Information Level. N/A

Debug Level. The task's XML.

Basic Troubleshooting.

- If the Probe Gateway tasks are not synchronized with the server tasks, check this log to determine which tasks the Probe Gateway received.
- You can view the current task's state through the JMX console (use the Discovery Scheduler MBean).

Probe Manager Logs

probeMgr-services.log

Java services debug messages.

Error Level. N/A

Information Level. N/A

Debug Level. N/A

Basic Troubleshooting. Check this log to view Java services debug messages.

probeMgr-performance.log

Performance statistics dump, collected every predefined period of time, which includes memory information and thread pool statuses.

Error Level. N/A

Information Level. N/A

Debug Level. N/A

Basic Troubleshooting.

- Check this log to investigate memory issues over time.
- The statistics are logged every 1 minute, by default.

probeMgr-patternsDebug.log

This log contains messages used to debug pattern issues.

Error Level. N/A

Information Level. N/A

Debug Level. N/A

Basic Troubleshooting. Use this log file for debugging patterns.

Troubleshooting and Limitations

For details on using the log files to perform basic troubleshooting, see "Log Files" on page 53.

This section includes the following topics:

The Probe Gateway and Probe Manager Activation

Problem. The Probe Gateway or Probe Manager cannot be activated.

Indication. When trying to activate the Probe Gateway or Probe Manager, the console opens and immediately closes.

Verification. To view the exception message, open the following files located in **<Business Availability Center root directory>\UCMDBServer\root\logs**:

- For the Probe Gateway: **wrapperProbe.log**
- For the Probe Manager: **wrapperLocal.log**

A message is displayed. If one of the following messages appears, the problem lies in the memory size definition:

```
Initial heap too small for new size specified.
Incompatible initial and maximum heap sizes specified.
The port number is being used.
```

To solve this problem, see the following Solution section. If another message appears and you cannot fix the problem, contact HP Software Support.

Solution. There can be several reasons for the activation problem, for example:

- **Inappropriate memory size.** Minimum and maximum memory sizes are allocated for each CMDB component. These definitions are set in the batch.cmd file, under the set memory sizes section. If memory sizes are too high for your workstation, are illegal, or incompatible with one another, you must change them, save the file, and restart the component whose values you changed.
- **Installation path is too long.** If you installed Business Availability Center to a directory with a long path, the operating system or JVM may have problems running the Business Availability Center execution commands. Reinstall Business Availability Center to a different directory that creates a shorter path.

The Probe Gateway and Probe Manager Connection

Problem. The connection between the Probe Gateway and Probe Manager cannot be established.

Indication. The DDM process is not working properly.

Verification. For the Probe Gateway: An error message is displayed in the Probe Gateway log (**wrapperProbe.log**, located in <**Business Availability Center root directory**>**UCMDBServer****root****logs**), as shown in the following example:

```
Failed to connect to probe manager at <server>. Will retry later
```

For the Probe Manager: An error message is displayed in the Probe Manager log (**probe-infra.log**, located in <**Business Availability Center root directory**>**UCMDBServer****root****logs**), as shown in the following example:

```
Connection attempt to service:jmx:rmi:///jndi/rmi://<Probe GW HOST>:1742/jmxrmi
failed, probe GW may be down
```

Solution. Check the following:

- Verify that the correct port—1742—is defined. The RMI connection port parameter is called appilog.collectors.rmi.port. It is defined in the **DiscoveryProbe.properties** file, located in <**Business Availability Center root directory**>**UCMDBServer****root****lib****collectors**.
- Verify whether the Probe Manager port is being used by another application. To verify this, in the Windows command interpreter (cmd.exe) type: netstat -na. A list of ports that are currently in use is displayed. If the port is in use, either close the other application or change the port number in the **DiscoveryProbe.properties** file.

Host Name Cannot Be Resolved to IP Address

Problem. A host name cannot be resolved to its IP address. If this happens, the host cannot be discovered, and patterns do not run.

Solution. Add the host machine name to the Windows HOSTS file on the Probe machine.

Discovery Tab Missing from Tabs

Problem. The Discovery tab is not displayed in the main page of Business Availability Center.

Solution. Install a license for the Probe. For details, see Chapter 2, "Licensing Models."

Connection Fails

Problem. The connection between the HP Universal CMDB server and the Probe fails due to an RMI or HTTP exception.

Solution. Ensure that none of the Probe ports are in use by another process.

DDM Results Do Not Appear in the Topology Map

Problem. Data that should have been discovered during the DDM process does not appear in the topology map.

Verification. The CMDB cannot retrieve the data or build the TQL results. Check the Discovery Statistics pane. If the CIs were not created, the problem is occurring during the DDM process.

Solution. Check the error messages in the **probeMgr-services.log** file located in **<Business Availability Center root directory>\UCMDBServer\root\logs**.

Networks and IPs

Problem. Not all networks or IPs have been discovered.

Indication. Not all the networks or IPs appear in the topology map results.

Verification. The IP address range in the Set Up Discovery Probes window does not encompass the scope of the networks or IPs that should have been discovered.

Solution. Change the scope of the DDM range:

1. Select **Admin** > **Discovery** > **Set Up Discovery Probes** to open the Set Up Discovery Probes window.
2. Select the Probe and the range.

3 Change the IP address range in the Ranges box as required.

TCP Ports

Problem. Not all TCP ports have been discovered.

Indication. Not all TCP ports appear in the topology map results.

Verification. Open the **portNumberToPortName.xml** file (**Admin** > **Universal CMDB** > **Manage Discovery Resources** > **Network** > **Configuration Files** > **portNumberToPortName.xml**), and search for the missing TCP ports.

Solution. Add the port numbers that need to be discovered to the **portNumberToPortName.xml** file.

Resolving DNS Names

Problem. The Probe Manager cannot resolve DNS names.

Indication. The Host's label in the topology map contains only the CIT to which it belongs.

Verification. In the topology map, right-click the host whose DNS name has not been resolved. Select **Show CI Attributes** to open the CI Attributes dialog box. Check the **host_dnsname** box. If it is empty, the DNS name has not been resolved.

Solution. Verify that:

- The protocol agent is installed on the remote machine.
- Business Availability Center can communicate with the protocol agent.
- You have the correct permissions to access the protocol agent.

Resolve the DNS name using any of the following protocols:

- SNMP
- Telnet
- SSH
- WMI

Status 'Disconnected' for Probe

Problem. Discovery and Dependency Mapping shows a disconnected status for a Probe.

Solution. Check the following on the Probe machine:

- That the Probe is running.
- That there are no network problems.

SSH/Telnet Credentials

Problem. Failure to connect to the TTY (SSH/Telnet) agent.

Solution. To troubleshoot connectivity problems with the TTY (SSH/Telnet) agent, use a utility that can verify the connectivity with the TTY (SSH/Telnet) agent. An example of such a utility is the client tool PuTTY.

SNMP Credentials

Problem. Failure to collect information from SNMP devices.

- **Solution 1**. Verify that you can actually access information from your Network Management station by using a utility that can verify the connectivity with the SNMP agent. An example of such a utility is GetIf.
- **Solution 2**. Verify that the connection data to the SNMP protocol has been defined correctly in the Add Protocol Parameters dialog box. For details, see "Protocol Parameters Dialog Box" on page 81.
- **Solution 3**. Verify that you have the necessary access rights to retrieve data from the MIB objects on the SNMP agent.

SAP Discovery Fails

Problem. The SAP discovery fails and a java.exe message is displayed

```
This application has failed to start because MSVCR71.dll was not found.
```

Solution. Two .dll files are missing. For the solution, read Note #684106 in https://websmp205.sap-ag.de/~form/sapnet?_FRAME=CONTAINER&_OBJECT=012003146900000245872003.

Host Fingerprinting in Nmap Cannot Run on Probe

Problem. When running the Host Fingerprinting feature in Nmap on a Probe that is installed on a Windows 2003 Server, you see an error message.

Solution. You cannot run Host Fingerprinting in Nmap when the Probe is installed on a Windows 2003 Server. Install the Probe on a different operating system, for example, Windows 2000.

Limitations

- When the Discovery Probe is installed on a non-English operating system, the installation wizard remains in English.
- When performing an Oracle Real Application Clusters (Oracle RAC) discovery, note that DDM cannot discover links to the remote machines (the database clients) in the following situation: The discovered database reports its clients by their host names and not by their IP addresses, and the host name cannot be resolved to an IP address. In this case, the remote client cannot be created.

5

Set Up Discovery Probes

This chapter provides information on setting up Discovery and Dependency Mapping (DDM) Probes.

This chapter includes:

Concepts

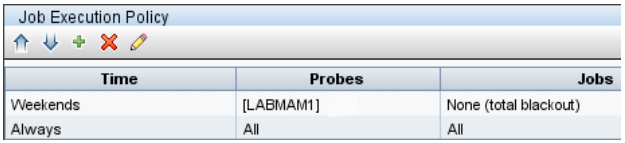
Reference

Job Execution Policies

You can define periods of time when a Probe must not run. You can choose to disable specific jobs running on any Probe or all jobs running on a specific Probe. You can also exclude jobs from a job execution policy so that they continue running as usual.

For details on defining a job execution policy, see “Add/Edit Policy Dialog Box” on page 72.

We recommend leaving the policy that runs all the time as the last policy in the list so that even if a job is not connected to a policy, it still runs:



If the job is not connected to any policy, and the policy that runs all the time is not the last policy in the list, the job does not run.

Example of Policy Ordering

There are two policies, **Total TCP Blackout** and **Always**. **Total TCP Blackout** does not allow any TCP discovery jobs to run. The policies appear in the list:

A job (Class C IPs by ICMP) starts running. It checks the policies in the policy list from top to bottom. It starts by checking **Total TCP Blackout**. The job does not appear in this policy, so it continues down the list and checks **Always**. The job does appear here (**Allow All** is selected in the Edit Policy dialog box) so the job runs.

The next job (IP Traffic by Network Data) starts running. It checks the policies in the policy list from top to bottom. It starts by checking **Total TCP Blackout**. The job appears in this policy (**Disallowed Jobs** is selected in the Edit Policy dialog box), so it does not run.

Running Jobs When a Job Execution Policy Is Running

If a policy begins to operate while a Probe is executing a job, the job pauses. When the policy finishes, the job continues to run from where it ceased. For example, say a job contains 10,000 Trigger CIs. The job finishes working on 7,000 of them and then the policy starts to operate. When the job continues (after the policy finishes), it works on the remaining 3,000 Trigger CIs—the job does not start running from the beginning.

Set Up Discovery Probes User Interface

This section describes:

Add/Edit IP Range Dialog Box

Description	Enables you to set the network range for discovery. The results are retrieved from the addresses in the range you define. You can also define IP addresses that must be excluded from a range. **To access:** Click the **Add IP range** button in the Ranges pane (**Admin** > **Universal CMDB** > **Discovery** > **Set Up Discovery Probes** > **Details pane)**.
Important Information	If you define a range that is out of the scope of the network on which the Probe is installed, HP Business Availability Center automatically defines the range of the IP on which the Probe is running. A message informs you that the Probe does not include the Probe range. Answer **Yes** to include the IP address in the range.
Included in Tasks	"Run Discovery – Advanced Mode Workflow" on page 99

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	To exclude an IP range from discovery, click the **Add IP range** button.
	To delete the excluded part of an IP range, select the excluded range and click the **Remove IP range** button.
	To edit the excluded part of an IP range, click the **Edit IP range** button. For details, see Exclude Ranges.

GUI Element (A–Z)	Description
Exclude Ranges	Click the **Add IP range** or **Edit IP range** button to exclude part of a range. In the Exclude IP Range dialog box, enter the range to exclude. **Note:** ➤ You must enter a range (in the Add/Edit IP Range dialog box) before you can enter the excluded range. ➤ The rules for entering an excluded range are the same as for entering a range. For details, see Range. ➤ Use this feature to divide a network range into several subranges. For example, say a range is 10.0.64.0 – 10.0.64.255. You define three excluded ranges: 10.0.64.45 – 10.0.64.50 10.0.64.65 – 10.0.64.70 10.0.64.89 – 10.0.64.95 Therefore, the ranges to be discovered are: 10.0.64.0 – 10.0.64.44 10.0.64.51 – 10.0.64.64 10.0.64.71 – 10.0.64.88 10.0.64.96 – 10.0.64.255

GUI Element (A–Z)	Description
Range	The rules for defining an IP address range are as follows: ➤ The IP address range must have the following format: start_ip_address – end_ip_address For example: 10.0.64.0 - 10.0.64.57 ➤ The range can include an asterisk (*), representing any number in the range of 0-255. ➤ If you use an asterisk, you do not need to enter a second IP address. For example, you can enter the range pattern 10.0.48.* to cover the range from 10.0.48.0 to 10.0.48.255. ➤ Use an asterisk in the lower bound IP address of the IP range pattern only. (If you use an asterisk in the lower bound IP address and also enter an upper bound IP address, the upper bound IP address is ignored.) ➤ You can use more than one asterisk (*) in an IP address as long as they are used consecutively. The asterisks cannot be situated between two numbers in the IP address, nor can they be substituted for the first digit in the number. For example, you can enter 10.0.*.* but not 10.*.64.*. ➤ Two Probes in the same domain cannot have the same IP address in their range.

Add/Edit Policy Dialog Box

Description	Enables you to add a job execution policy, to disable jobs from running at specific times. **To access: Admin** > **Universal CMDB** > **Discovery** > **Set Up Discovery Probes** > **Details pane** > **Job Execution Policy** section. Select an existing policy and click **Edit**, or click the **Add** button.
Useful Links	"Job Execution Policies" on page 67 "Job Execution Policy Pane" on page 77 "Domain Credential References" on page 83

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
Related jobs	➤ **Allow all.** Run the job execution policy on all jobs. ➤ **Total blackout.** The policy does not run on any jobs. ➤ **Allowed jobs.** Choose jobs to run even during the configured blackout time. ➤ **Disallowed jobs.** Choose jobs that do not run during the configured blackout time. For allowed and disallowed jobs, click the **Add job** or **Remove job** button to choose specific jobs to be included in, or excluded from, the policy. If you click the **Add job** button, the Choose Discovery Jobs dialog box opens.
Related Probes 	The Probes on which to run the policy. Click the button to open the **Edit Related Probes** dialog box to define which Probes are included in the policy.
Time 	The date and time during which the policy is active. Click the button to open the **Edit Timetable** dialog box.

Add New Domain Dialog Box

Description	Enables you to add a domain. **To access:** Click the **Add Domain or Probe** button in the Domains and Probes pane.
Important Information	In an upgraded 6.x environment, to enable data to be modelled similarly as in the previous version, you must define the Probes as belonging to the **External** domain and not to the **Customer** domain.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
Description	Enter a description to appear in the Details pane of the Set Up Discovery Probes window.
Domain Type	➤ **Customer.** A private domain used for your site. You can define several domains and each domain can include multiple Probes. Each Probe can include IP ranges but the customer domain itself has no range definition. ➤ **External.** Internet/public domain. A domain that is defined with a range. The external domain can contain only one Probe whose name equals the domain name. However, you can define several external domains in your system.
Name	Enter a unique name for the domain.

Add New Probe Dialog Box

Description	Enables you to add a Probe. **To access:** Click the **Add Domain or Probe** button in the Domains and Probes pane.
Important Information	➤ To add a Probe to an existing domain, select **Probes** in the Domains and Probes pane and click the **Add Domain or Probe** button. ➤ To add a Probe to a new domain, create a domain, then add the Probe to the domain. ➤ Two Probes in the same domain cannot have the same IP address in their range. ➤ When a Probe is activated, it is added automatically and its status changes to connected. For details, see “Launch the Probe” on page 35.

Choose Discovery Jobs Dialog Box

Description	Enables you to choose the jobs that are to be added to, or excluded from, the job execution policy. **To access:** Select **Allowed Jobs** or **Disallowed jobs** in the Edit Policy dialog box and click the button .

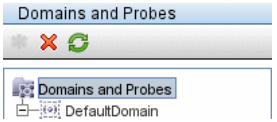
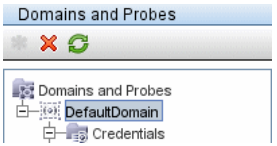
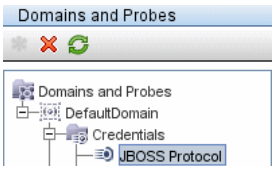
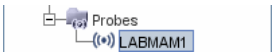
The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
<Installed packages>	Locate the job to be included in, or excluded from, the policy. To select jobs from several packages, hold down the CTRL key and make the selection.

Details Pane

Description	Enables you to view the Probes running under all domains and to add an execution policy to jobs (that is, to schedule time periods when jobs should not run). **To access:** Click an object in the Domains and Probes pane.
Important Information	Depending on what you select in the Domains and Probes pane, different information is displayed in the Details tab. For details, see "Displayed Information" on page 76 in the next section.

Displayed Information

If You Select...	The Information Displayed Is...
	Domains and Probes, you can view details on all Probes and you can define and edit job execution policies. For details, see "Discovery Probes Pane" on page 76 and "Job Execution Policy Pane" on page 77.
	A specific domain, you can add a description and view a list of Probes running in that domain. For details, see "Discovery Probes Pane" on page 76 and "Description Pane" on page 78.
	A specific protocol, you can add protocol parameters and you can view details on the protocol, including user credentials. For details, see "Domain Credential References" on page 83.
	A specific Probe, you can view details on the Probe, including range information. You can also add ranges to, or exclude ranges from, the Probe. For details, see "Ranges Pane" on page 78, "Discovery Probes Pane" on page 76, and "Details Pane" on page 78.

Discovery Probes Pane

The Discovery Probes pane includes a list of all Probes connected to the server. The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
IP	The IP range defined during Probe creation.
Last Access Time	The last time that the Probe requested tasks from the server.
Name	The Probe name as it appears in DDM.
Status	Can be Connected or Disconnected.

Job Execution Policy Pane

Description	Enables you to configure the periods of time when jobs should not run. **To access: Admin** > **Universal CMDB** > **Discovery** > **Set Up Discovery Probes**. Select **Domains and Probes**.
Important Information	Jobs which have a listening functionality (that is, they do not perform discovery) are not included in a policy.
Useful Links	"Job Execution Policies" on page 67 "Domain Credential References" on page 83

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Move the policy up or down. DDM executes all the policies in the list with the first policy taking priority. If a job is included in two policies, DDM executes the first policy only for that job.
	Add a policy.
	Remove a policy.
	Edit a policy. Click to open the **Edit Policy** dialog box.
Jobs	The jobs that are affected by the policy.
Probes	The Probes that are affected by the policy.
Time	The schedule of the policy.

Description Pane

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
Description	The description that was entered during domain creation.
Domain Type	For details, see Domain Type in "Add New Domain Dialog Box" on page 73.

Details Pane

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
Last time Probe accessed	The last time that the Probe was accessed on the server machine.
Probe IPs	The IP of the Probe machine.
Status	➤ **Connected.** The Probe has successfully connected to the server (the Probe connects every few seconds). ➤ **Disconnected.** The Probe is not connected to the server.

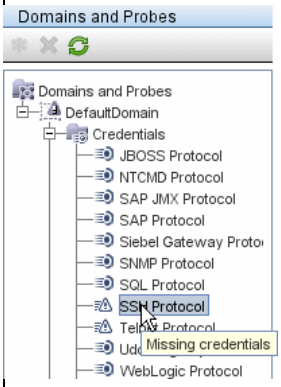
Ranges Pane

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Click to open the **Add IP Range** dialog box.
	Click a range and click the button to remove a range from the list.
	Click to open the **Edit IP Range** dialog box.

GUI Element (A–Z)	Description
Excluded	Displays the IP addresses that have been excluded from the range that the Probe uses to discover CIs. For details, see "Add/Edit IP Range Dialog Box" on page 70.
Range	The network IP addresses that the Probe uses to discover CIs. For details, see "Add/Edit IP Range Dialog Box" on page 70.

Domains and Probes Pane

Description	Enables you to view, define, or edit a domain, a Probe or a Probe's credentials. **To access: Admin > Universal CMDB > Discovery > Set Up Discovery Probes.**
Important Information	A missing credential is represented by the following icon: 
Useful Links	"Job Execution Policies" on page 67

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Adds a domain or Probe, depending on what is selected.
	Deletes a domain or Probe, depending on what is selected.
	Updates all domain and Probe information.

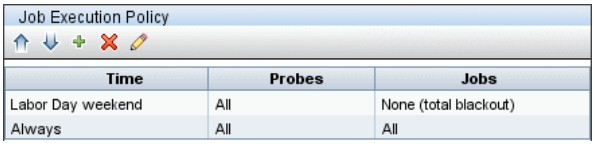
Edit Related Probes Dialog Box

Description	Enables you to select specific Probes.  **To access:** Click the Related Probes button in the Edit Policy dialog box.
Useful Links	"Job Execution Policies" on page 67

Edit Timetable Dialog Box

Description	Enables you to set the times when a Probe must run a job execution policy.  **To access:** Click the **Edit** button in the Edit Policy dialog box.
Useful Links	"Add/Edit Policy Dialog Box" on page 72

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
Description	Add a description of the specific policy. This field is mandatory. **Tip**: The text you enter here appears in the Time box in the Job Execution Policy pane, so it is recommended that the description be informative: 
Time Definition	Click a cell for a day and time to be included in the policy. To add more than one time unit, drag the pointer over the cells. **Note:** To clear a time unit, click the cell a second time.

Protocol Parameters Dialog Box

Description	Displays the attributes that can be defined for a protocol. **To access: Set Up Discovery Probes** > **Domains and Probes** > **Domain** > **Credentials**, select a protocol and click the **Add** or **Edit** button.
Important Information	For the description of each protocol, see "Domain Credential References" on page 83.

Scope Definition Dialog Box

Description	Enables you to set the range that a protocol must discover. **To access:** Click the **Edit** button in the Protocol Parameters dialog box.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
Selected Probes	To select specific Probes whose IP range must be changed, click **Edit**. For details, see "Choose Probe Dialog Box" on page 111.
Selected Ranges	➤ **All.** The protocol runs discovery on all ranges for the domain. ➤ **Selected Range.** For the procedure to select a specific range on which the protocol runs discovery or to define an excluded range, see "Add/Edit IP Range Dialog Box" on page 70.

Set Up Discovery Probes Window

Description	Enables you to define a new domain or to define a new Probe for an existing domain. Also, to define the connection data for each protocol. **To access: Admin > Universal CMDB > Discovery > Set Up Discovery Probes**.
Important Information	➤ For details on the Domains and Probes pane, see "Domains and Probes Pane" on page 79. ➤ For details on the Details pane, see "Details Pane" on page 75.
Useful Links	"Domain Credential References" on page 83

Selecting Probes

The Choose Probe to Filter, Edit Probe Limitations for TQL Output, and Edit Related Probes dialog boxes include the following elements (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	**Add Selected Probe.** Click to add a Probe to the Selected Probes column.
	**Remove Selected Probe.** Click to remove a Probe from the Selected Probes column.
All Discovery Probes	➤ Select to add all Probes in the Non-selected Probes list. ➤ Clear to add a specific Probe from the Non-selected Probes list.
Non-selected Probes	Probes that are not included in the policy/filter/limitations.
Selected Probes	Probes that are included in the policy/filter/limitations.

Domain Credential References

This section explains protocol credentials. You can edit credential attributes. For details, see "Protocol Parameters Dialog Box" on page 81.

When a protocol is selected in the Domains and Probes Pane, the following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A-Z)	Description
	Click to add new connection details, to open the **Add Protocol Parameters** dialog box.
	Select a protocol and click to remove connection details. Answer **OK** to the message.
	Select a protocol and click to edit connection details in the **Protocol Parameters** dialog box.

GUI Element (A-Z)	Description
	Select a protocol and click an arrow to move the protocol instance up or down. The order of the policies in the list defines which policy is checked first: a job starts running and checks the policy list from top to bottom. If the job name exists in a policy, the job runs. For details on adding jobs to a protocol, see "Add/Edit Policy Dialog Box" on page 72. For details on job execution policies, see "Example of Policy Ordering" on page 68.
<Right-click a credential>	Choose from the following options: ➤ **Edit.** Choose this option to enter protocol parameters, such as user name and password, that enable DDM to connect to an application on a remote machine. ➤ **Edit using previous interface.** Choose this option if, in a previous version, you added parameters to this protocol that do not exist in this version. ➤ **Check credentials.** In the box that opens, enter the IP address of the remote machine on which the protocol must run. The Probe attempts to connect to this IP and returns an answer whether the connection succeeded or not.
<Right-click a title>	Choose from the following options: ➤ **Hide Column.** Displayed when a column is shown. ➤ **Show All Columns.** Displayed when a column is hidden. ➤ **Customize.** Select to change the display order of the columns. ➤ **Auto-resize Column.** Select to change the column width to fit the contents.

All protocol credentials include the following parameters:

Parameter	Description
Network Scope	To change the range that a protocol must discover or to select a Probe, click **Edit**. For details, see "Scope Definition Dialog Box" on page 82. Default: **ALL**.
Protocol Index	Indicates the order in which protocol instances are used to make a connection attempt. The lower the index, the higher the priority. Default: **9999**. If you do not change the default, this protocol instance is used last.
User Label	Enter a label to help you identify a specific protocol credential, when you use it later. Enter a maximum of 50 characters.

This section includes the following topics:

JBoss Protocol

Parameter	Description
Connection Timeout	Timeout in milliseconds after which the Probe stops trying to connect to the JBOSS application server.
Port Number	The port number.
User Name	The name of the user needed to connect to the application as administrator.
User Password	The password of the user needed to connect to the application as administrator.

NTCMD Protocol

Parameter	Description
Connection Timeout	Timeout in milliseconds after which the Probe stops trying to connect to the NTCMD server.
User Name	The name of the user needed to connect to the host as administrator.
User Password	The password of the user needed to connect to the host as administrator.
Windows Domain	The name of the domain that includes the host where the Probe is installed.

SAP JMX Protocol

Parameter	Description
Connection Timeout	Timeout in milliseconds after which the Probe stops trying to connect to the SAP JMX console.
Port Number	The SAP JMX port number.

Parameter	Description
User Name	The name of the user needed to connect to the application as administrator.
User Password	The password of the user needed to connect to the application as administrator.

SAP Protocol

Parameter	Description
SAP Client	It is recommended to use the default value (800).
SAP Router String	A route string describes the connection required between two hosts using one or more SAProuters. Each of these SAProuters checks its Route Permission Table (http://help.sap.com/saphelp_nw04/helpdata/en/4f/992dfe446d11d189700000e8322d00/content.htm) to see whether the connection between its predecessor and successor is allowed. If it is, the SAProuter sets it up.
SAP System Number	It is recommended to use the default value (00).
Timeout/Connection Timeout	Timeout in milliseconds after which the Probe stops trying to connect to the SAP console.
User Name	The name of the user needed to log in to the SAP system.
User Password/Password	The password of the user needed to log in to the SAP system.

Siebel Gateway Protocol

Parameter	Description
Connection Timeout	Timeout in milliseconds after which the Probe stops trying to connect to the Siebel Gateway console
Path of srvrmgr	The location on the Probe server to where you copied srvrmgr. For details, see "Copy the driver Tool to the Probe Server" on page 266. **Note:** If there are several protocol entries with different srvrmgr versions, the entry with the newer version should appear before the entry with the older version. For example, to discover Siebel 7.5.3. and Siebel 7.7, define the protocol parameters for Siebel 7.7 and then the protocol parameters for Siebel 7.5.3.
Siebel Site Name	The name of the Siebel Enterprise.
User Name	The name of the user needed to log on to the Siebel enterprise
User Password	The password of the user needed to log on to the Siebel enterprise.

SNMP Protocol

Parameter	Description
Community	(For SNMP v1 and SNMP v2 only) Enter the authentication password you used when connecting to the SNMP service community (which you defined when configuring the SNMP service—for example, a community for read-only or read/write).
Connection Timeout	Timeout in milliseconds after which the Probe stops trying to connect to the SNMP agent.
Port Number	(For SNMP versions v1, v2, and v3) The port number on which the SNMP agent listens.
Retry	The number of times the Probe tries to connect to the SNMP agent. If the number is exceeded, the Probe stops attempting to make the connection.

Parameter	Description
SNMP version	The options are: ➤ version 1 or 2 ➤ version 3
User Name	(For SNMP v3 only) The name of the user authorized to log on to the management application.
User Password	(For SNMP v3 only) The password used to log on to the management application.
V3 – Authentication algorithm	(For SNMP v3 only) Two algorithms are supported: ➤ MD5 ➤ SHA
V3 – Authentication method	(For SNMP v3 only) Select one of the following options for securing the access to management information: ➤ **NoAuthNoPriv.** Using this option provides no security, confidentiality, or privacy at all. It may be useful for certain applications, such as development and debugging to turn security off. This option requires only a user name for authentication (similar to requirements for v1 and v2). ➤ **AuthNoPriv.** The user logging on to the management application is authenticated by the SNMP v3 entity before the entity allows the user to access any of the values in the MIB objects on the agent. Using this option requires a user name, password, and the authentication algorithm (HMAC-MD5 or HMAC-SHA algorithms). ➤ **AuthPriv.** The user logging on to the management application is authenticated by the SNMP v3 entity before the entity allows the user to access any of the values in the MIB objects on the agent. In addition, all of the requests and responses from the management application to the SNMP v3 entity are encrypted, so that all the data is completely secure. This option requires a user name, password, and an authentication algorithm (either (HMAC-MD5 or HMAC-SHA).

Parameter	Description
V3 – Privacy algorithm	(For SNMP v3 only) The following algorithm is supported: DES.
V3 – Privacy key	(For SNMP v3 only) The secret key used to encrypt the scoped PDU portion in an SNMP v3 message.

SQL Protocol

Parameter	Description
Connection Timeout	Timeout in milliseconds after which the Probe stops trying to connect to the database.
Database Name	The database name.
Database SID	The database SID (Oracle, DB2).
Database Type	The database type. Select the appropriate type from the box.
Port Number	The port number on which the database listens.
User Name	The name of the user needed to connect to the database as administrator.
User Password	The password of the user needed to connect to the database as administrator.

SSH Protocol

Parameter	Description
Authentication Mode	These are the following authentication options: ➤ Password ➤ Key ➤ Keyboard Interactive
Connection Timeout	Timeout in milliseconds after which the Probe stops trying to connect to the remote machine. For UNIX platforms: If your server is slow, it is recommended to change Connection Timeout to 40000.

Parameter	Description
Key Path	Location of the authentication key. (In certain environments, the key path is required to connect to an SSH agent.)
Network Address	The discovered IP network address or the network address range.
Port Number	By default a an SSH agent uses port 22. If you are using a different port for SSH in your environment, enter the required port number.
User Name	The name of the user needed to connect to the host as administrator.
User Password	The password of the user needed to connect to the host as administrator.

Telnet Protocol

Parameter	Description
Connection Timeout	Timeout in milliseconds after which the Probe stops trying to connect to the remote machine. For UNIX platforms: If your server is slow, it is recommended to change Connection Timeout to 40000.
Port Number	The port number. By default a Telnet agent uses port 23. If you are using a different port for Telnet in your environment, enter the required port number.
User Name	The name of the user needed to connect to the host as administrator.
User Password	The password of the user needed to connect to the host as administrator.

UDDI Registry Protocol

Parameter	Description
Connection Timeout	Timeout in milliseconds after which the Probe stops trying to connect to the UDDI Registry.
UDDI inquiry URL	The URL where the UDDI Registry is located.

WebLogic Protocol

Parameter	Description
Certificate PEM File Path	Enables the WebLogic gateway to verify the user who is connecting to it. Enter the path to the certificate file.
Connection Timeout	Timeout in milliseconds after which the Probe stops trying to connect to the WebLogic application server.
Key PEM File Path	Enables the WebLogic gateway to verify the user who is connecting to it. Enter the path to the key file.
Password	The password of the user needed to connect to the application as administrator.
Port Number	The port number.
Protocol	An application-level protocol that determines whether DDM should connect to the server securely. Enter either **http** or **https**.
Trust File Path	Enables the WebLogic server to verify the user who is connecting to it. Enter the name of the Trust File, including the jks extension. For example, **Demotrust.jks**.
Trust File Password	Enter the password for the Trust File user.
User Name	The name of the user needed to connect to the application as administrator.

WebSphere Protocol

Parameter	Description
Password	The password of the user needed to connect to the application as administrator.
Port	The protocol port number as provided by the WebSphere system administrator. You can also retrieve the protocol port number by connecting to the Administrative Console using the user name and password provided by the WebSphere system administrator. In your browser, enter the following URL: **http:/<host>:9090/admin**, where: ➤ **<host>** is the IP address of the host running the WebSphere protocol ➤ **9090** is the port used to connect to the WebSphere console Access **System Administration** > **Deployment Manager** to retrieve the required port number.
Timeout	Timeout in milliseconds after which the Probe stops trying to connect to the WebSphere server.
Trust Store File Name	The name of the SSL trust store file. Business Availability Center contains a default trust store file called **DummyClientTrustFile.jks**. It is located in the External resources folder in the J2EE package. To use a file other than the default trust store file provided by the system, enter the name of your trust store file and place it in the External resources folder. When the package is updated, the trust store file is copied to the relevant location.

Parameter	Description
Trust Store Password	The SSL trust store password. The default password for the default trust store file in Business Availability Center is WebAs. If you are not using the default trust store file provided by the system, enter the password for the file you are using.
User Name	The name of the user needed to connect to the application as administrator.

WMI Protocol

Parameter	Description
Connection Timeout	Timeout in milliseconds after which the Probe stops trying to connect to the WMI agent.
User Name	The name of the user needed to connect to the host as administrator.
User Password	The password of the user needed to connect to the host as administrator.
Windows Domain	The name of the domain that includes the host where the Probe is installed.

6

Run Discovery

This chapter provides information on running discovery.

This chapter includes:

Concepts

Tasks

Reference

Run Discovery – Overview

A Discovery and Dependency Mapping (DDM) job enables reuse of the same pattern for different discoveries, without the need to change the pattern itself. (To activate DDM, you activate jobs—organized in modules—and not patterns.)

Each pattern includes default configuration parameters and scheduling information that define how to perform DDM. A job can either override the default configuration (by associating a specific set of Trigger CIs with each pattern) or can run what is declared in the pattern.

Packages contain default job definitions as well as the appropriate patterns. For details, see "Package Manager" in *Model Management*.

For details on defining a new job, see "Job Editor Dialog Box" on page 157.

Working in Basic Mode or Advanced Mode

You can activate DDM with one of the following methods:

- Use **Basic Mode** to run DDM for a specific component (for example, the infrastructure, J2EE applications, or databases), using configurable, default preferences.

  For details on the workflow, see "Run Discovery – Basic Mode Workflow" on page 98.

  For details on the Discovery wizard, see "Basic Mode Window" on page 108.

Note: Basic Mode is displayed by default when accessing Run Discovery.

- Use **Advanced Mode** to run DDM when you want to customize a run by making changes to a job, pattern, and so on.

  For details on the workflow, see "Run Discovery – Advanced Mode Workflow" on page 99.

For details on the Discovery wizard, see "Advanced Mode Window" on page 107.

Note: To view Help on Run Discovery components:

- For details on the Discovery Modules pane, see "Discovery Modules Pane" on page 135.
- For details on the Details tab, see "Details Tab" on page 124.
- For details on the Properties tab, see "Properties Tab" on page 158.
- For details on the Dependency Map tab, see "Dependency Map Tab" on page 122.

Managing Problems With Error Reporting

During DDM, many errors may be uncovered, for example, connection failures, hardware problems, exceptions, timeouts, and so on. DDM displays these errors in Run Discovery, in both Basic and Advanced Mode. You can drill down from the Trigger CI that caused the problem to view the error message itself.

DDM differentiates betweeen errors that can be ignored (for example, an unreachable host) and errors that must be dealt with (for example, credential problems or missing configuration or DLL files). Moreover, DDM reports errors once, even if the same error occurs on successive runs, and reports an error even it if occurs once only.

For details on severity levels, see "Severity Levels" on page 53.

Error Table in Database

All DDM errors are saved to the discovery_problems table in the Probe Manager database schema. (The error information is saved to the database—and is not handled in the Probe's memory—to guarantee delivery to the server.) The Probe holds the latest list of problems for each Trigger CI. After each run, the Probe checks for changes and reports them in the Discovery Status pane. For details, see "Discovery Status Pane" on page 126.

Run Discovery – Basic Mode Workflow

This task describes how to begin mapping your system and its components, using the Discovery wizards. You use this workflow when you want to use default values for the components needed in a network, database, or J2EE discovery.

Note: For details of running DDM in Advanced Mode, see "Run Discovery – Advanced Mode Workflow" on page 99.

This task includes the following steps:

1 Prerequisites

Verify that the Probe is installed. For details on installing the Probe, see "Install the Probe" on page 14.

For details on licensing, see Chapter 2, "Licensing Models."

2 Access the Discovery Wizard

For details, see the relevant wizard: "Infrastructure Wizard" on page 142, "J2EE Wizard" on page 150, or "Database Wizard" on page 116.

Run Discovery – Advanced Mode Workflow

This task describes how to begin mapping your system and its components. You would use this workflow when you want to customize the components of a module.

Note: For details of running discovery in Basic Mode, see "Run Discovery – Basic Mode Workflow" on page 98.

This task includes the following steps:

1 Prerequisites

a Verify that the Probe is installed. For details on installing the Probe, see "Install the Probe" on page 14.

For details on licensing, see Chapter 2, "Licensing Models."

b Verify that the packages are deployed.

For details, see Chapter 10, "Package Manager."

2 Determine Network Range

You must define the network range of the network to be discovered. For details, see "Add/Edit IP Range Dialog Box" on page 70.

Note: Patterns try to connect to every IP in a range. Therefore, if a range is wide, network performance may be affected.

---

3 Set Relevant Credentials

To enable DDM to connect to servers or applications using specific protocols, you must set the relevant credentials (for example, NTCmd, SNMP, TTY, or WMI). For details on protocol parameters, see "Domain Credential References" on page 83. For details on the Details pane in the Set Up Discovery Probes window, see "Details Pane" on page 75.

Note: DDM tries to connect to a host by using each credential in turn. DDM then saves the successful credential. The next time DDM connects to this host, it first tries to connect using the successful credential.

---

4 Activate Relevant Jobs

Once you have defined the network range and set credentials, you can run discovery on specific jobs. For details, see Chapter 9, "Discovery and Dependency Mapping Content."

Tip: You can view a full description of a job in the Run Discovery Properties tab, under the DDM pattern name.

Example – Finding SNMP Connections

You can search for all jobs that discover SNMP connections: in the **Run Discovery** > **Discovery Modules** pane, click the **Search Discovery Job** icon. In the **Find Job**s dialog box, enter **SNMP** in the **Name** box and click **Find All**. For details, see **Add Rule** in "Discovery Modules Pane" on page 135 and "Find Jobs Dialog Box" on page 141.

5 Make Changes to Relevant Patterns

You can customize patterns to discover infrequent system components. For details on pattern writing, see Chapter 10, "Content Development and Pattern-Writing."

Important: Do not make changes to default patterns without consulting HP Software Support.

---

6 Monitor the DDM Process

For details on monitoring the CIs that are discovered by the run, see "Statistics Results Pane" on page 133.

a Define a TQL

You create a TQL query that retrieves information about CIs and CITs from the CMDB. For details, see "Define a TQL Query" in *Model Management*.

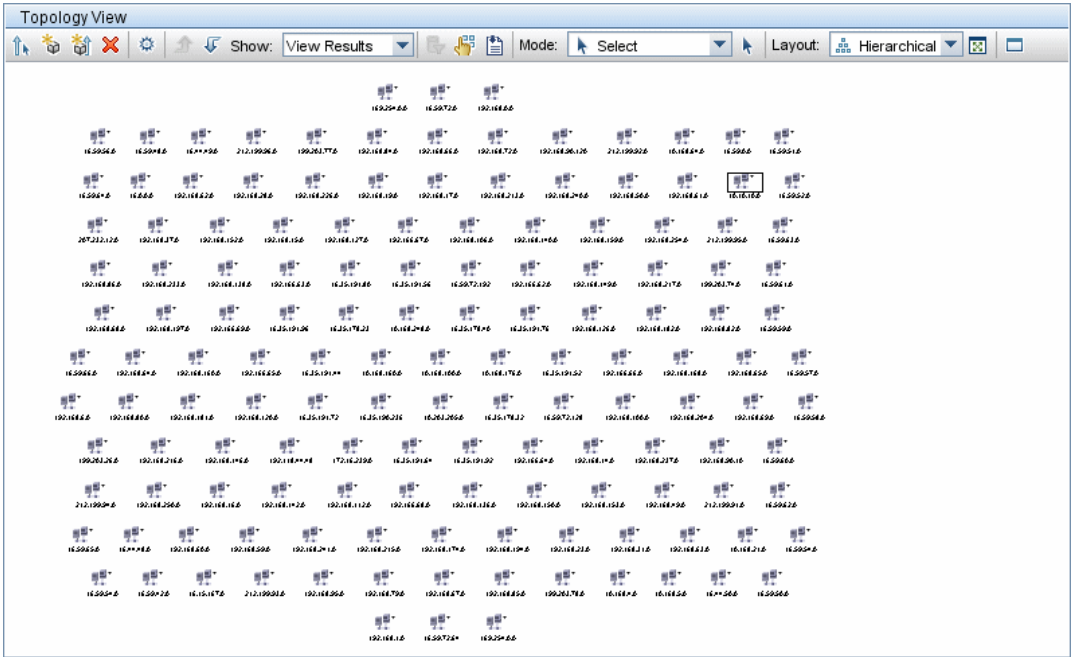
If necessary you can trigger TQLs to manually discover objects. For details, see "Trigger TQLs Pane" on page 162.

b Build a View for each TQL

A view enables you to build a subset of the overall IT universe model, containing only those CIs in the CMDB that relate to a specific discovery. For details, see "View Manager Window" in *Model Management*.

Example – Creating a View to Display Discovered CI Instances

To view the number of instances found by Business Availability Center, select **Admin** > **Universal CMDB** > **Modeling** > **IT Universe Manager**, and display the view you created.



7 View Result Statistics

You can display overall statistics for a job or you can filter the results by time range or by Probe. Each time you log in to Business Availability Center and access Run Discovery, the statistical data is updated so that the data displayed is the latest for the selected module or job.

For details on working with the statistical data, see “Statistics Results Pane” on page 133.

You can view discovered CIs also by accessing the Show Status Snapshot pane. For details, see Chapter 8, “Show Status Snapshot.”

8 Troubleshoot the Results

You can check DDM results to see which errors are being reported. For details, see "Manage Errors" on page 104.

Run an Ad-Hoc Discovery to Rediscover CIs

You use the View Discovery wizard to perform an ad-hoc discovery of a view. The ad-hoc discovery runs the jobs relevant for the CIs in the selected view, to find recent changes to the configuration.

This task describes how to run an ad-hoc discovery.

This task includes the following steps:

- "Access View" on page 103
- "Display View Discovery Wizard" on page 103
- "Choose Jobs to Run" on page 103

1 Access View

In IT Universe Manager, you access the view to be checked in the View Explorer. For details, see "View Explorer User Interface" in *Model Management*.

2 Display View Discovery Wizard

In the Topology Map pane, select the CIs to be discovered, and click the Rediscover button to display the View Discovery Wizard. For details, see "View Discovery Wizard" in *Model Management*.

A list of the jobs related to the discovered CIs in the view is displayed.

3 Choose Jobs to Run

Business Availability Center displays the Choose Jobs for View dialog box and you choose which jobs should be run. For details, see "Choose Jobs for View Dialog Box" in *Model Management*.

The Probe immediately runs the jobs that originally discovered the CIs and sends the results back to the server.

Business Availability Center displays statistics about the CIs (for example, how many new CIs have been discovered) in a table. For details, see "Discovered Changes Details Dialog Box" in *Model Management*.

Business Availability Center rebuilds the view and displays updated information.

Manage Errors

This task describes how to investigate problems that arise during a run.

Note: For details about severity levels and so on, see "Managing Problems With Error Reporting" on page 97.

This task includes the following steps:

- "Prerequisites" on page 104
- "Run the Discovery Wizard or Select the Job" on page 105
- "Locate the Problem CI" on page 105
- "Troubleshoot the Problem" on page 105

1 Prerequisites

Set up DDM. For details, see "Run Discovery – Basic Mode Workflow" on page 98 or "Run Discovery – Advanced Mode Workflow" on page 99.

2 Run the Discovery Wizard or Select the Job

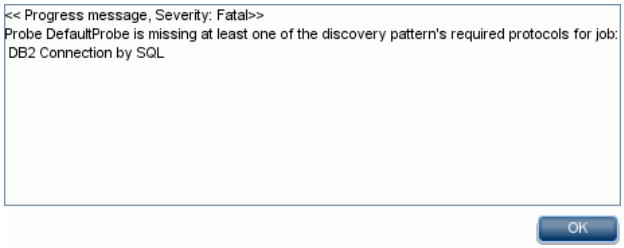
In Basic Mode, you can view error messages for a default job. In Advanced Mode, you can view error messages for one job, one module, or all modules. For details on running a wizard in Basic Mode, see "Run Discovery – Basic Mode Workflow" on page 98. For details on running a job, see "Run Discovery – Advanced Mode Workflow" on page 99.

3 Locate the Problem CI

Use the Discovery Status pane to drill down to the error messages. For details, see "Discovery Status Pane" on page 126.

Example

DDM displays the error message:



4 Troubleshoot the Problem

- For Fatal errors, you should contact HP Software Support.
- For other errors, check the CIs. For example, a Trigger CI that does not fall within the Probe's range may show an error.
- For details on setting communication logs, see "Execution Options Pane" on page 190.
- For details on managing problems, see "Managing Problems With Error Reporting" on page 97.

Run Discovery User Interface

This section describes:

Advanced Mode Window

Description	Enables you to view and manage modules and jobs, to activate jobs, and to follow job progress. Advanced mode includes the following panes: ➤ Discovery Modules pane. Each module includes jobs. You activate a module or job to discover a specific group of CIs. For details, see "Discovery Modules Pane" on page 135. **Note**: Basic Mode is displayed by default when accessing Run Discovery. ➤ Details tab. Enables you to manage a module's CIs and view CI statistics. For details, see "Details Tab" on page 124. ➤ Properties tab. Enables you to view and administer the properties of modules and jobs. For details, see "Properties Tab" on page 158. ➤ Dependency Map. Displays a visual representation of the real-time progress of the process. For details, see "Dependency Map Tab" on page 122. **To access: Admin** > **Universal CMDB** > **Discovery** > **Run Discovery**.
Important Information	Each change you make in Run Discovery is delivered to and stored in the CMDB. From there, the changes are sent to the Probe. You can verify that changes have been sent to the Probe by opening the **wrapperProbe.log** file located in <**DDM Probe root directory**>**DiscoveryProbe****root****logs**\ and searching for the following lines: processing document domainScopeDocument.bin Processing document domainScopeDocument.bin is done.
Important Information	**Note**: Basic Mode is displayed by default when accessing Run Discovery.
Included in Tasks	"Run Discovery – Advanced Mode Workflow" on page 99

Basic Mode Window

Description	Enables you to use a Discovery wizard to discover networks, databases, and J2EE applications. Basic mode includes the following panes: ➤ **List of wizards**. Enables you to choose the wizard to run. For details, see "Infrastructure Wizard" on page 142, "Database Wizard" on page 116, or "J2EE Wizard" on page 150. ➤ **Summary pane**. Enables you to run the wizard and to stop DDM running. For details, see "Summary Pane" on page 109. ➤ **Discovery Progress pane**. Enables you to view a brief run status and to drill down to problematic Trigger CIs. This pane is displayed once discovery has been run for a component. For details on managing errors, see "Discovery Status Pane" on page 126. **To access: Admin > Universal CMDB > Discovery > Run Discovery**
Important Information	**Note**: Basic Mode is displayed by default when accessing Run Discovery. For details on Advanced Mode, see "Advanced Mode Window" on page 107.
Included in Tasks	"Run Discovery – Basic Mode Workflow" on page 98
Useful Links	"Working in Basic Mode or Advanced Mode" on page 96

Summary Pane

Description	Enables you to run a Discovery wizard. **To access: Admin > Universal CMDB > Discovery > Run Discovery**
Important Information	Depending on whether a wizard has already run, the Summary pane displays the following information: ➤ If a wizard has not yet run, the Summary pane displays the steps to be performed in the wizard and the **Configure and Run** button. ➤ If a wizard has run, the Summary pane displays a summary of the run parameters, the **Configure** and **Stop Discovery** buttons, and the results of the previous run in the Discovery Progress pane. To run a discovery, select a wizard in the left pane and click **Configure** or **Configure and Run** to open the Discovery wizard. To stop a discovery run, click **Stop Discovery**.
Included in Tasks	"Run Discovery – Basic Mode Workflow" on page 98

Choose CIs to Add Dialog Box

Description	Enables you to choose CIs to run with selected jobs. **To access:** ➤ Click the **Add CI** button in the Discovery Status pane. ➤ In the Oracle TNS File page of the Database Wizard, click the **Add CI** button.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
Add button	**Note:** If you choose CIs with an error status to add to the trigger list, a message is displayed when you click the **Add** button.

GUI Element (A–Z)	Description
Search CIs	Contains filters with which you can limit the number of CIs that appear in the Search Results pane. ➤ **By Discovery TQL**. Select a Discovery TQL to search for those CIs that match the TQL. ➤ **Show only CIs containing.** To search for CIs that include a certain text, enter the text here. ➤ **Exact match.** Select to search for CIs with the exact match of the text label. (By default, you search by entering part of a text. For example, searching for **10** within the IP CIs finds all the IPs that contain 10 in their address. Entering **10** then selecting **Exact match** finds no results.) ➤ **Search.** Click to display the search results.
Search Results	Displays a list of triggered CIs answering to the criteria set in the filter. To add the CIs to the list in the triggered CIs pane, select the CIs. You can make multiple selections. ➤ **CIT**. The CI type of the selected triggered CI. ➤ **CI**. The label of the triggered CI. ➤ **Related Host**. The label for the host related to the triggered CI. ➤ **Related IPs**. The IPs of the related host. **Page.** The list of CIs is divided into pages. The number in the Page box indicates which page is currently displayed. To view other pages, use the up and down arrows, or type the page number, and press **Enter**. To determine the number of CIs that appear on a page, right-click either the up or down button and choose the required number. The default is 25.

Choose Discovery TQL Dialog Box

Description	Enables you to add a trigger TQL to a job. **To access:** Click the **Add TQL** button in the Trigger TQLs pane.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
<Discovery TQL name>	The TQLs that can query the CMDB for the selected CIT.
TQL Preview	Hold the cursor over an element to view details.

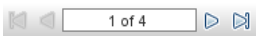
Choose Probe Dialog Box

Description	Enables you to filter the Probe list. **To access:** Click a Filter button in the **Run Discovery** > **Details** tab: ➤ **Triggered CIs** pane Filter button. For details on the menu options, see "Discovery Status Pane" on page 126. ➤ **Statistics** pane Filter button. For details on the menu options, see "Statistics Results Pane" on page 133.

CIs Discovered by [Module or Job Name] Dialog Box

Description	Enables you to view CI instances for a CIT. **To access:** ➤ Click the **View instances** button in the Statistics pane. ➤ Select **Show discovered CIs** or **Show all CI instances** in the Dependency Map.
Important Information	Depending on whether you select **Show discovered CIs** or **Show all CI instances** in the Dependency Map, you can view either all CIs discovered by a selected job or all CIs of a selected type. The number of displayed CIs appears at the top of the dialog box: 

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Click to define a filter to shorten the list of results. For details, see "Filter CI Instances Dialog Box" in *Model Management*.
	Clears the filter. All results are displayed in the list.
	Click to select the columns to be displayed. For details, see "Select Columns Dialog Box" in *Reference Information*.
	Click to refresh the list of CI instances.
	Set rows per page.
	Display another page.
Display Hidden Columns button	Click to display all CI attributes. Click **Show Filtered Attributes** to display the CI attributes selected in the Columns dialog box.
Show Filtered Attributes	Click to hide attributes.

Right-click a Row

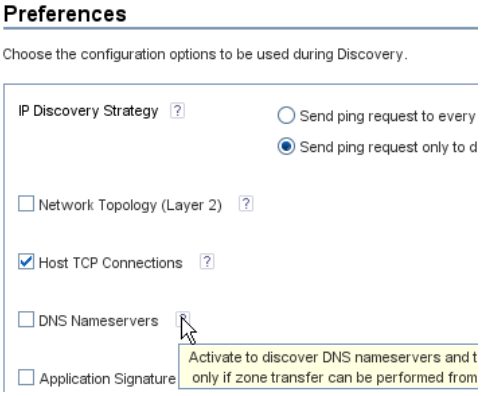
Right-click any CIT to view the following options:

GUI Element (A–Z)	Description
Actions	Select one of the following options: ➤ **Add CI to Discovery Job**. Displays the Add CI to Discovery Job dialog box, which enables you to manually invoke a job for the selected CI. You can use this option to discover additional information about the CI through one of the available jobs. ➤ **Remove CI from Discovery Job**. Displays the Remove CI from Discovery Job dialog box, which enables you to manually remove a CI from the job. ➤ **Open Credentials Information.** Displays the Protocol Parameters dialog box, in view only mode, for the selected CI (that must be of the Software Element CI type). DDM uses the credential's user name and password to access the Protocol Parameters dialog box. **Note**: DDM retrieves the credentials from the credentials_id attribute of the relevant CI. For details, see "Protocol Parameters Dialog Box" on page 81.
CI History	Displays the CI History dialog box. For details, see "CI/Relationship History Dialog Box" in *Model Management*.
Delete from CMDB	Enables you to delete the selected CI from the CMDB. **Note:** When you delete a parent CI, the CI and its children are removed from a view, but only the CI is removed from the CMDB.

GUI Element (A–Z)	Description
Get Related CIs	Related CIs are CIs that are related to a selected CI, arranged in tree format with the selected CI as the root. The child CIs are arranged by CIT. The following options are available: ➤ **From View.** Opens a window displaying the selected CI and its immediate neighbors within the current view. ➤ **From Database without Filter.** Opens a window displaying the selected CI and its immediate neighbors in the CMDB. ➤ **From Database with Filter.** Opens a dialog box that enables you to filter the related CITs so that only they are displayed in the Topology Map. The dialog box also displays the number of CIs found for each CIT. ➤ **Reveal From Database without Filter.** Displays all CIs in any layer from the database that are connected to the selected CI, on the same Topology Map. For an explanation of layers, see folding rule in the glossary. ➤ **Reveal From Database with Filter.** Opens a dialog box which displays the number of related CIs by type. Select the CI type to display in the Topology Map results. CIs of the selected types that are connected to the selected CI, from any layer of the database, are displayed in the results. ➤ **Reveal from View.** Displays all CIs that are connected to the selected CI, in any layer from the current view, on the same Topology Map. (This option is active only when viewing related CIs in IT Universe.)
Label	Select one of the following options: ➤ **Edit Label.** Opens the Edit Label dialog box which enables you to edit the name of the CI. ➤ **Reset Label.** Resets the CI name to its default value taken from the CMDB.

GUI Element (A–Z)	Description
Note	Select one of the following options: ➤ **Add Note.** Opens an editing box where you can type a note to be attached to the CI. ➤ **Delete Note.** Deletes all text saved in a note.
Properties	Displays the Properties page for the selected CI.
Relate to CI	Opens the Attach Related CI Wizard. For details, see "Attach Related CIs Wizard" in *Model Management*.
View Sublayer	Displays a window showing the CIs in the layer beneath the selected CI. **Note:** This option is only active for CIs with children.

Database Wizard

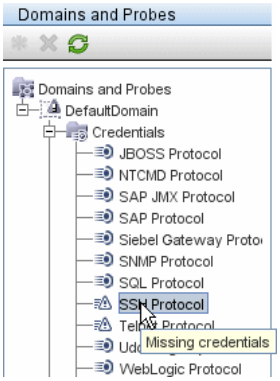
Description	Enables you to discover databases such as DB2, Oracle, Microsoft SQL, and Sybase. **To access: Admin > Universal CMDB > Discovery > Run Discovery > Basic Mode**. Select the Database wizard from the list in the left pane. Click **Open Wizard**.
Important Information	For more information, hold the pointer over a question mark icon: 
Wizard Map	The Database Discovery wizard contains: **Database Wizard** > Define Network Credentials > Configure Ports > DB2 JDBC Driver > Oracle TNS File > Schedule Discovery > Summary (Click a link to view Help for that page.)

Define Network Credentials

Description	Enables you to configure connection data for each protocol.
Important Information	You configure protocols depending on what you want to discover and on which protocols are supported on your site's network. For a list of protocols, see "Domain Credential References" on page 83. General information about the wizard is available in "Database Wizard" on page 116.
Wizard Map	The Database Discovery wizard contains: Database Wizard > **Define Network Credentials** > Configure Ports > DB2 JDBC Driver > Oracle TNS File > Schedule Discovery > Summary

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Add new connection details for selected protocol type.
	Remove a protocol.
	Edit a protocol. Click to open the Protocol Parameters dialog box.

GUI Element (A–Z)	Description
	Move a protocol up or down. DDM executes all the protocols in the list with the first protocol taking priority.
Protocol	Click to view details on the protocol, including user credentials. **Note:** A missing credential is represented by the following icon: 

Configure Ports

Description	Enables you to choose the port number and port type through which to connect to the database.
Important Information	General information about the wizard is available in “Database Wizard” on page 116.
Wizard Map	The Database Discovery wizard contains: Database Wizard > Define Network Credentials > **Configure Ports** > DB2 JDBC Driver > Oracle TNS File > Schedule Discovery > Summary

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Click to add a port to the port list. The Add New Port dialog box opens. Select the ports and click OK. To edit existing system ports, in the Add New Port dialog box, click **Edit Known System Ports**. The Edit Known System Ports dialog box opens. Select the port and click the **Edit** button. In the dialog box that opens, make changes to the entries and click **OK**. To add a port to the list, in the Edit Known System Ports dialog box, click the **Add** button. Enter details of the port name, number and type and click **OK**.
	Select a port and click the button to remove the port from the list.

DB2 JDBC Driver

Description	Enables you to select the jar file for the DB2 JDBC driver.
Important Information	General information about the wizard is available in “Database Wizard” on page 116.
Wizard Map	The Database Discovery wizard contains: Database Wizard > Define Network Credentials > Configure Ports > **DB2 JDBC Driver** > Oracle TNS File > Schedule Discovery > Summary

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
DB2 JDBC Driver	Select the check box and click **Import file...** to locate the appropriate jar file in the DB2 JDBC installation.

Oracle TNS File

Description	Enables the discovery of Oracle databases. You provide the location of the TNSNames.ora configuration file that contains database information needed to discover Oracle databases, such as port, host, SID, and so on.
Important Information	General information about the wizard is available in "Database Wizard" on page 116.
Wizard Map	The Database Discovery wizard contains: Database Wizard > Define Network Credentials > Configure Ports > DB2 JDBC Driver > **Oracle TNS File** > Schedule Discovery > Summary

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
Server Host	Select the hosts on which the TNSNames.ora file is located. Click the **Add CI** button to choose the Trigger CIs that represent these hosts. For details, see "Choose CIs to Add Dialog Box" on page 109. ➤ **CIT**. The CI type of the selected triggered CI. ➤ **CI**. The label of the triggered CI. ➤ **Related Host**. The label for the host related to the trigger CI. ➤ **Related IPs**. The IPs of the related host.
TNSNames.ora file location	Enter the location of the TNSNames.ora file in the server host system. You can enter several locations (separate the locations by commas). If you terminate the path with a delimiter (for example, c:\temp\), DDM assumes that the file name is **tnsnames.ora**.

Schedule Discovery

Description	Enables you to define a schedule for a specific job.
Important Information	General information about the wizard is available in "Database Wizard" on page 116.
Wizard Map	The Database Discovery wizard contains: Database Wizard > Define Network Credentials > Configure Ports > DB2 JDBC Driver > Oracle TNS File > **Schedule Discovery** > Summary

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	You define a time template in the Discovery Scheduler pane of the Properties tab. For details, see "Discovery Scheduler Pane" on page 161.
Allow Discovery to run at	Choose the time at which the schedule should run.
Repeat Every	Select how often the schedule should run.

Summary

Description	Enables you to review the wizard definitions before running a discovery.
Important Information	To make changes to the run, click the **Back** button. General information about the wizard is available in "Database Wizard" on page 116.
Wizard Map	The Database Discovery wizard contains: Database Wizard > Define Network Credentials > Configure Ports > DB2 JDBC Driver > Oracle TNS File > Schedule Discovery > **Summary**

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
Run	Click the button to run a discovery.

Dependency Map Tab

Description	Displays a visual representation of the real-time progress of the discovery process. The map displays: ➤ CIs that were triggered by a job ➤ CIs that were discovered as a result of the activated job. **To access:** Click the **Dependency Map** tab in the Run Discovery window.
Important Information	Depending which level you select in the Discovery Modules pane, different information is displayed in the Dependency Map tab. If you select: ➤ **The Discovery Modules root**, and select the **Show only active Discovery jobs** check box, the Dependency Map displays only active jobs and their interdependencies. ➤ **The Discovery Modules root**, and clear the **Show only active Discovery jobs** check box, the Dependency Map displays all Discovery jobs and their interdependencies. ➤ **A module**, a topology map is displayed showing the module's active and inactive jobs. ➤ **A job**, the topology map highlights the job in the module's map.
Useful Links	"CIs Discovered by [Module or Job Name] Dialog Box" on page 111

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
<right-click menu>	Use the right-click menu to view details for a job, CI, or link, for example, the number of CI instances (of a specific type) in the CMDB or the number of CI instances created by a specific job. Depending on which object is selected, the following menu options are displayed: ➤ When a job is selected: **Show discovered CIs.** Click to view the CIs discovered by the job. To filter the query, select a CIT from the menu. **Show trigger CIs.** Click to view the CIs that triggered the job. ➤ When a CI is selected: **Show all CIT instances.** Click to view all CIs of this CI type. ➤ When a link from a CI to a job is selected: **Show trigger CIs for job.** Click to view CIs (of the selected type) that triggered the job. ➤ When a link from a job to a CI is selected: **Show discovered instances.** Click to view CIs (of the selected type) that were discovered by the job.
<Toolbar>	For details, see "Toolbar Options" in *Model Management*.
<Tooltip>	Hold the pointer over a CI or job to display a description.
Show only active Discovery jobs	When the Discovery Modules root is selected in the Discovery Modules pane, this check box is displayed. Select to display all active jobs (from any module).

Details Tab

Description	Enables you to view and administer modules and jobs, to follow the progress of the DDM process, and to manage errors during discovery. **To access:** Click the **Details** tab in Run Discovery.
Important Information	Depending which level you select in the Discovery Modules pane, different information is displayed in the Details tab. If you select: ➤ **The Discovery Modules root** or a **Discovery module**, the Discovery Status and Statistics Results panes are displayed with information and statistics about all active jobs and errors discovered during a run. For details, see "Discovery Status Pane" on page 126 and "Statistics Results Pane" on page 133. ➤ **A job**, the Discovery Job Details, Discovery Status, and Statistics Results panes are displayed. For details, see "Discovery Job Details Pane" on page 125, "Discovery Status Pane" on page 126, and "Statistics Results Pane" on page 133. ➤ Several jobs or modules, the Selected Items pane is displayed. For details, see "Selected Items Pane" on page 132.
Included in Tasks	"Managing Problems With Error Reporting" on page 97

Discovery Job Details Pane

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	**View Map.** You can choose to view a map of the CITs and links that are discovered by the pattern, instead of a list. Click the button to open the Discovered Classes Map window. The selected pattern is shown together with its CIs and relationship links. Hold the cursor over a CIT to read a description in a tooltip.
Discovered CIs	The CIs that are discovered by this job.
Input CI Type	The CIT that triggers the CIs for this job.
Job Name	The name and description of the job and the package in which it is located.

Discovery Status Pane

Description	Enables you to view a brief run status and to drill down to problematic Trigger CIs, to uncover specific problems that DDM encountered during the run, for example, incorrect credentials. In **Basic Mode**, enables you to view the results of the previous run for the selected job type (that is, infrastructure, database, or J2EE application). In **Advanced Mode**, enables you to view the results of the previous run for a selected module or job, or for all modules. **To access:** ➤ In Basic Mode, locate the Discovery Overview pane. ➤ In Advanced Mode, select a module or job, click the **Details** tab, and locate the Discovery Status pane.
Important Information	➤ You can use the SHIFT and CTRL keys to select adjacent and non-adjacent CIs in a list. ➤ Depending which level you select in Advanced Mode in the Discovery Modules pane, information is displayed in the Discovery Status pane for all modules, for a specific module, or for a specific job. ➤ The information in this pane is automatically refreshed every thirty seconds.
Included in Tasks	"Manage Errors" on page 104
Useful Links	"Managing Problems With Error Reporting" on page 97

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A-Z)	Description
	Click to return to the upper pane.
	Click to refresh the status view.
	Click to add a newly-discovered CI.

GUI Element (A-Z)	Description
	Click to remove a CI from the list, if the CI is no longer of interest. The CI is deleted from the specific job.
	Click to drill down to the Trigger CI that includes the problem. **Note**: This icon is displayed only when you can drill down from error or warning links.
	Click and choose an option from the menu: ➤ **By Status.** Displays a list of Trigger CIs: ➤ **All.** Displays all the Trigger CIs. ➤ **Waiting for Probe.** Displays the Trigger CIs that are ready to be dispatched and are waiting for the Probe to retrieve them. ➤ **In Progress.** Displays the Trigger CIs that are active and are running on the Probe. ➤ **In progress (being removed).** Displays the Trigger CIs that are being removed from the **Trigger CIs** list. ➤ **Success**, **Failed**, **Warning**. Displays only those CIs that have the selected status. ➤ **Discovery Errors.** Displays CIs with an error status. ➤ **By Probe.** Displays only the CIs triggered by a selected Probe. Click to open the Choose Probe dialog box. ➤ **By Dispatch Type.** Displays a list of CIs according to one of the following options: ➤ **All**. Displays both CIs that are used to manually activate the job and Discovery TQLs that are used to automatically activate the job. ➤ **Manually added.** Displays the CIs that are used to manually activate the job. ➤ **By Discovery TQL**. Displays the CIs that are used to automatically activate the job. ➤ **Reset.** Click to remove any filters.
	Click to display a message box containing an explanation of the failure. (You can also view messages by right-clicking the CI and selecting **Show error details**.)

GUI Element (A-Z)	Description
	Click to open the Source CIs dialog box with additional information about the CI.
	Show statistics for a specific Trigger CI.
	Find a CI.
	Click to run the job.
<drill down>	You can drill down from a job or a module. ➤ Drill down from a job to view a list of Trigger CIs that are included in the job. ➤ Drill down from a module to view a list of the jobs in the module and the number of CIs returned by each job. Drill down from a job to its Trigger CIs. **Note:** A Trigger CI can be present in more than one job.

GUI Element (A-Z)	Description
<right-click CI menu>	Right-click a CI to: ➤ **Show error details.** Opens the Message dialog box with details of all errors for the selected Trigger CIs. ➤ **Remove.** Select to delete the CI from the job. The CI is removed from that job only, even if it appears in more than one job. ➤ **Run now.** To run a specific CI or set of CIs, select the CIs. They are added to the list of CIs that the Probe is going to run (**Waiting for Probe**). ➤ **Show results for triggered CIs.** DDM sends an ad-hoc request to the Probe and retrieves the latest results of the job (CIT name and number of discovered CIs) that is running on a specific trigger CI. This ad-hoc request does not run the job, but brings the results of the previous job run that are stored in the Probe's database. If the job has not yet run for this trigger CI, a message is displayed. For details, see "Show Results for Triggered CI Page" on page 164. If no communication log exists in the Probe, a message is displayed. You can choose that DDM always creates communication logs. For details, see "Execution Options Pane" on page 190.

GUI Element (A-Z)	Description
<right-click CI menu> (cont'd)	➤ **Debug.** Choose between: ➤ **View communication log for triggered CI.** Opens the log that includes information about the connnection between the Probe and the remote machine. This is on condition that you have set the **Create communication log** to either **Always** or **On failure**. For details, see "Execution Options Pane" on page 190. The communication log can reside in two places: in the Probe's memory or on the file system (as a record file). When requested, DDM displays the log retrieved from the Probe's memory. If no log exists in the memory, DDM displays the log that resides on the file system. If no log exists on the file system, a message is displayed. Click **Yes** to set this option to **Always**. Click **No** to make no changes to this option. ➤ **Go to pattern.** Displays the pattern that is included in the job in Manage Discovery Resources. ➤ **Go to job.** Displays the job in which the CI is included. ➤ **Undispatch.** Removes the Trigger CI.
Failed	Displays those CIs that returned an Error or Fatal severity. **Show errors.** Displays a list of the different types of errors returned by these CIs. For details on severity, see "Severity Levels" on page 53.
In progress	Displays the number of Trigger CIs that are awaiting their turn to be run. Click to view the jobs that are waiting to be run. Double-click a job to view the CIs that are waiting to be run.
Look for	To search for a specific Probe, related host, or related IP, enter part of its name in the box and click **Search**.
Progress	The indicator shows a summary of the current discovery run, that is, since the specific run was activated.

GUI Element (A-Z)	Description
Success	DDM displays the number of CIs that have been run successfully, that is, without errors. Click to view the jobs (and the number of CIs in each job) that completed successfully. Double-click a job to view information about all its CIs. Select a CI and use the <right-click CI menu> to view information **With warnings.** Click to view a warning message for each job. Double-click a message to view the CIs that finished successfully with a warning. Right-click a message to view the <right-click CI menu>.
Total	Displays the status of all of a job's Trigger CIs. Double click a **Warning** or **Error** status to open the Message dialog box.
Waiting for Probe	The Trigger CIs that are either waiting for the Probe or are waiting to run.

Selected Items Pane

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
<right-click menu>	Edit Scheduling. Click to open the Discovery Scheduler to define a schedule for a specific job. For details, see "Discovery Scheduler Pane" on page 161.
invoke immediately	➤ A check mark signifies that the Discovery job runs as soon as the triggered CI reaches the Probe. In this case, the Invoke on new triggered CIs immediately check box is selected in the Properties tab. ➤ If this column does not contain a check mark, the job runs according to the schedule defined in the Schedule Manager.
Job name	The name of the job.
Schedule info	The scheduling information of the job as defined in the Discovery Scheduler.
Trigger TQLs	The name of the TQL that activated the job. For details, see "Trigger TQLs Pane" on page 162.

Statistics Results Pane

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Select a CI and click this icon to view CI instances and their attributes. The CIs Discovered by [Module or Job Name] Dialog Box opens. Under the following conditions, a message is displayed: ➤ All the CIs that were discovered by this job were already discovered by another job. ➤ All the CIs that this job discovered have been deleted. ➤ The CI instances were discovered in a previous version. (In version 7.0, you cannot view instances of CIs discovered in a previous version.) **Note**: You can also view CI instances by double-clicking a row.
	Select the time range or Probe for which to display statistics about the CITs. ➤ **By Time Range**: ➤ **All.** Displays statistics for all job runs. ➤ **Last Hour/Day/Week/Month.** Choose a period of time for which to display statistics about the CITs. ➤ **Custom Range.** Click to open the Customize Statistics Time Range dialog box. Enter the date or click the arrow to choose a date and time from the calendar, for the **To** and **From** dates. To delete a date, click **Reset**. ➤ **By Probe**: To view statistics for a specific Probe, select to open the Choose Probe dialog box.
	Click to retrieve the latest data from the server (job results are not automatically updated in the Statistics pane).

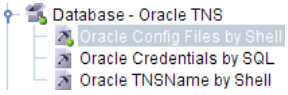
GUI Element (A–Z)	Description
<Column title>	Click a column title to change the order of the CITs from ascending to descending order, or vice versa.
<right-click a title>	Choose from the following options: ➤ **Hide Column.** Select to hide a specific column. ➤ **Show All Columns.** Displayed when a column is hidden. ➤ **Customize.** Select to display or hide columns and to change the order of the columns in the table. Opens the Columns dialog box. ➤ **Auto-resize Column.** Select to change a column width to fit the contents. For details, see "Select Columns Dialog Box" in *Reference Information*.
CIT	The name of the discovered CIT.
Created	The number of CIT instances created in the period selected or for the selected Probe.
Deleted	The number of CIT instances deleted in the period selected or for the selected Probe.
Discovered CIs	The number of CIs that were discovered for each CI type.
Filter	The time range set with the Set Time Range button.
Last updated	The date and time that the statistics table was last updated for a particular job.
Total	The total number of CIs in each column.
Updated	The number of CIT instances that were updated in the period selected.

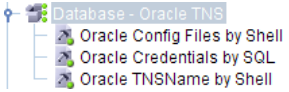
Discovery Modules Pane

Description	Enables you to view and manage modules and jobs. Each module includes the jobs necessary to discover specific CIs. **To access: Admin** > **Universal CMDB** > **Discovery** > **Run Discovery**. The default view is called **Basic Mode** and displays the Discovery Wizard. You can run the J2EE, database, or network discovery. Click **Advanced Mode** to view all modules.
Important Information	**Caution:** Only administrators with an expert knowledge of the DDM process should delete modules. **Obsolete.** Contains several modules that are no longer relevant but remain for backward compatibility and upgrade purposes. Do not use these modules on new installations. **No module**. Contains jobs that are not included in any other module.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	**Refresh All.** Updates the modules.
	**Find Job.** Click to open the Find Jobs dialog box. For example, to search for all jobs that discover SNMP connections, click the Filter icon. In the **Find Jobs** dialog box, enter **SNMP** in the **Name** box and click **Find All**. For details, see “Find Jobs Dialog Box” on page 141.
	Click to run a module or job. You can choose to activate all jobs in a module or some of them: ➤ To activate all jobs in a module, right-click the module and select **Activate**. ➤ To activate specific jobs in a module, right-click the job and select **Activate**.

GUI Element (A–Z)	Description
	Select the jobs or modules to be stopped and click **Deactivate**.
	Represents the module root. To create a module, right-click to enter the name of the module you are creating. **Note:** A name is case sensitive. Names beginning with an upper case letter appear in the Discovery Modules list before names beginning with a lower case letter.
	Represents a module. Click the key to display the jobs in a module.
	Represents a job. Click to display information about the job. To view a pattern description, hold the pointer over a job. Jobs contain configuration information derived from patterns and other resources and are the entities controlled by users, for example, when activating or deactivating a module. For details on the right-click menu, see “Right-Click Menu” on page 137.
	One green dot signifies that some of a module’s jobs are activated: 

GUI Element (A–Z)	Description
	Three green dots signify that all of a module's jobs are activated: 
	An exclamation mark signifies that one or more of the jobs is experiencing a problem that could affect the DDM process, for example, a protocol connection failure. To view the reason for the problem, click the Handle errors button in the Triggered CI pane. For details, see "Discovery Status Pane" on page 126. **Note**: If a problem is resolved by clicking the **Refresh All** button, the Problem Indicator disappears.

Right-Click Menu

GUI Element (A–Z)	Description
Activate	Click a module to run all its jobs. To run a specific job, select and activate it. The Discovery Module discovers CITs and relationships of the types that are described in each job, and places them in the CMDB. For example, the Class C IPs by ICMP job discovers the Depend, IP, and Member CITs and relationships.
Create New Job	Click to open the Job Editor dialog box. For details, see "Job Editor Dialog Box" on page 157.
Create New Module	Click to define a new name for the module root.
Deactivate	Stop the module or job from running.
Delete	Click and answer **Yes** to the warning message.
Delete job	Click and answer **Yes** to the warning message.
Edit Pattern	Click to edit the pattern in the Manage Discovery Resources window.

GUI Element (A–Z)	Description
Edit Scheduling	Click to open the Discovery Scheduler to define a schedule for a specific job.
Rename job	Click to open the Choose Name dialog box. Enter a new name for the job. **Note:** You cannot rename active jobs.
Run Now	Click to run the job again using the selected Trigger CIs.
Save as...	Click to clone the job.

Discovery Scheduler Dialog Box

Description	Enables you to define a schedule for a specific job, for example, every day DDM starts running an IP ping sweep on class C networks at 6:00 AM. **To access:** ➤ Right-click a job and choose **Edit scheduling**. ➤ Click the **Edit Scheduler** button in the Discovery Scheduler pane of the Properties tab in the Run Discovery window.
Important Information	The Discovery Scheduler defines the frequency of the discovery whereas the time template defines when the job should run, for example, during the day, at night, at weekends only, and so on. You can run the same schedule with different time templates. For example, you can define a schedule that runs every day and you can define a time template that runs at night from 01:00 AM to 05:00 AM. A job defined in this way runs every day from 01:00 AM to 05:00 AM. You can define a second time template to run at a different time, and you can use this time template too with the same schedule. For details on creating a time template, see “Edit Time Template Dialog Box” on page 141.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Opens a calendar. Select the required date and time.
	Click to validate the Cron expression you entered.
<Frequency>	➤ **Once.** Define the task to run only once. ➤ **Interval.** Defines the interval between successive runs. ➤ **Daily.** Run a task on a daily basis. ➤ **Weekly.** Run a task on a weekly basis. ➤ **Monthly.** Run a task on a monthly basis. ➤ **Cron.** Enter a Cron expression in the correct format.
Days of month	(Appears when you select **Monthly**) Click the button  to choose the days of the month on which the action must run. The Select Days dialog box opens. Choose the required days by selecting the check boxes. You can select multiple days. ➤ **Select all**. Select all the days. ➤ **Unselect all**. Clear all the selected days.
Days of the week	(Appears when you select **Weekly**) Select the day or days on which you want the action to run.
End by	Select the date and time when you want the action to stop running by selecting the **End by** check box and clicking the **Open Calendar**  button. **Note**: This step is optional. If you do not want to specify an ending date, leave the **End by** check box unselected.

GUI Element (A–Z)	Description
Invocation Hour	(Appears when you select **Daily**, **Weekly**, or **Monthly**) Select the time to activate the action. Click the button  to open the Select Hours dialog box. Choose the required time by selecting the check boxes. You can select multiple times. ➤ **Select all**. Select all the times. ➤ **Unselect all**. Clear all the selected times. **Note**: You can also enter the time manually in the **Invocation hour** box. Separate times by a comma and enter AM or PM after the hour. The manually entered action times are not restricted to the hour and half hour only: you can assign any hour and minute combination. Use the following format: HH:MM AM, for example, 8:15 AM, 11:59 PM.
Invocation Time	 (Appears when you select **Once**) Choose the date and time you want the action to begin running by clicking the **Open Calendar** button.
Months of the year	(Appears when you select **Monthly**). Select the month or months in which you want the action to run.
Repeat every	(Appears when you select **Interval**) Type a value for the interval between successive runs and choose the required unit of time (seconds, minutes, hours, or days).
Start at	 Choose the date and time when you want the action to begin running by selecting the **Start at** check box and then clicking the **Open Calendar** button to the right.
Time Zone	Set the time zone to the server time zone by clicking the **Set server time zone** button.

Edit Probe Limitation for TQL Output Dialog Box

Description	Enables you to change the Probes on which a trigger TQL is running. For details on selecting the Probes, see “Selecting Probes” on page 83. **To access:** Select a job and click the following button: **Run Discovery** > **Properties tab** > **Trigger TQLs pane** > **Probe Limit** box.

Edit Time Template Dialog Box

Description	Enables you to define a time template to use when scheduling jobs. **To access:** Click the **Add** button in the Time Templates dialog box.
Important Information	The name of the time template must be unique.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
Every day between	Define a daily schedule when a job must run. You can also type in times. You can assign any hour and minute combination.
Time Template	Enter a unique name.
Week Time	Define a weekly schedule when a job must run. Click to select a time. To select adjacent cells, click and drag the pointer over the table. To clear a time, click a cell a second time.

Find Jobs Dialog Box

Description	Enables you to search for jobs answering to specific criteria. The results of the search are displayed in the Selected Items pane in the Details tab. **To access:** Click the **Filter** button in the Discovery Modules pane.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
Direction	Searches forwards or backwards through the modules.
Find All	All jobs meeting the search criteria are highlighted.
Find Discovery job by	Choose between: ➤ **Name.** Enter the name, or part of it, of the job. ➤ **Input type.** CIs that triggered the job. Click the button to open the Choose Configuration Item Type dialog box. Locate the CI type that you are searching for. ➤ **Output type.** CIs that are discovered as a result of the activated job.
Find Next	The next job meeting the search criteria is highlighted.

Infrastructure Wizard

Description	Enables you to run discovery on the networks in your system. **To access: Admin > Universal CMDB > Discovery > Run Discovery > Basic Mode**. Select **Infrastructure Wizard** from the list in the left pane. Click **Open Wizard**.
Wizard Map	The Infrastructure Discovery wizard contains: **Infrastructure Wizard** > Define IP Ranges > Define Network Credentials > Preferences > Schedule Discovery > Summary.

Define IP Ranges

Description	Enables you to set the network range for discovery for each Probe. The results are retrieved from the addresses in the range you define. You can also define IP addresses that must be excluded from a range.
Important Information	Any changes made here affect the global configuration. General information about the wizard is available in "Infrastructure Wizard" on page 142.
Wizard Map	The Infrastructure Discovery wizard contains: Infrastructure Wizard > **Define IP Ranges** > Define Network Credentials > Preferences > Schedule Discovery > Summary.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	For details, see "Add/Edit IP Range Dialog Box" on page 70.
	Select a range and click the button to remove the range from the list.
	Select a range and click the button to edit an existing range.
Address Ranges	➤ **Range.** For details on the rules for defining ranges, see "Range" on page 72. ➤ **Excluded.** You can exclude part of a range. Select the range and click the **Add** button. In the dialog box, click the Advanced button. For details, see "Exclude Ranges" on page 71.
Discovery Probes	Enables you to view details on the Probe, including range information. You can also add ranges to, or exclude ranges from, the Probe. For details on defining a Probe, see "Domains and Probes Pane" on page 79.

Define Network Credentials

Description	Enables you to add, remove and edit a credentials set for protocols.
Important Information	You configure a credentials set depending on what you want to discover and on which protocols are supported on your site's network. For a list of protocols, see "Domain Credential References" on page 83. General information about the wizard is available in "Infrastructure Wizard" on page 142.
Wizard Map	The Infrastructure Discovery wizard contains: Infrastructure Wizard > Define IP Ranges > **Define Network Credentials** > Preferences > Schedule Discovery > Summary.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Add new connection details for selected protocol type.
	Remove a protocol.
	Edit a protocol. Click to open the Protocol Parameters dialog box.
	Click a button to move a protocol up or down to set the order in which credential sets are attempted. DDM executes all the protocols in the list with the first protocol taking priority.
Protocol	Click to view details on the protocol, including user credentials.

Preferences

Description	Enables you to choose the configuration options to be used during discovery that are activated by the Infrastructure Discovery wizard.
Important Information	General information about the wizard is available in "Infrastructure Wizard" on page 142.
Wizard Map	The Infrastructure Discovery wizard contains: Infrastructure Wizard > Define IP Ranges > Define Network Credentials > **Preferences** > Schedule Discovery > Summary.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A-Z)	Description
Application Signature	Select to discover applications running in your environment. Application Signature also discovers processes and creates process CIs in the CMDB. As part of the discovery process, Application Signature searches for configuration files. For details, see "Application Signature Discovery" on page 216.
DNS Nameservers	Discovers DNS nameserver machines and the IPs they hold names for. Choose to activate only if zone transfer can be performed from the Probe machine to the nameserver machines, that is, if the appropriate permissions exist on the DNS nameserver machines. **Network implications.** DDM tries to connect to DNS nameserver servers.

GUI Element (A-Z)	Description
Host Information	Select the host resources that you want to discover. These resources can be either physically or logically part of a host. After DDM connects to a host, it discovers the following resources: ➤ For SNMP agents, the relevant Management Information Base (MIBs). ➤ For WMI agents, the relevant Windows Management Instrumentation Query Language (WQL) queries. DDM can also execute shell commands on a machine. **Network implications.** The Software and Services network resources, because of the large quantities of data that they transmit, may cause very high network traffic. For this reason, the default is not to discover them.
Host TCP Connections	Discover TCP communication channels to map dependency relationships between hosts. This discovery requires that at least one protocol has a defined set of credentials. For details, see the previous Define Network Credentials step. **Network implications.** DDM executes shell commands on a machine to find open ports.

GUI Element (A-Z)	Description
IP Discovery Strategy	Choose the strategy for discovering IPs in your environment. This discovery requires that the SNMP protocol be configured in the previous Define Network Credentials step. ➤ **Send ping request to every address in defined IP range.** Select this option when you know that most of the IP addresses will respond, the network range is small, and most of the IPs in the range are of interest to you (that is, they are part of your network). ➤ **Send ping request only to discoverable IPs in a network.** Select this option when you know that not all IP addresses will respond and the network range is large. In this case, DDM first discovers a network, then sends a ping request to all discovered IPs in that network. **Versions and Limitations.** Verify that you have the correct credentials set for all the machines between the Probe and one of the network's switches.
Network Topology	Activate to discover the connections, on a discovered switch (for example, a host), between a host and its physical port as well as between a host and its logical layout (VLANs, ELANs). This discovery requires that at least one protocol has a defined set of credentials. For details, see the previous Define Network Credentials step.

GUI Element (A-Z)	Description
Port Scanning	The TCP ports appearing in the **Choose TCP ports for port scanning** list are scanned to discover open server ports. The ports are scanned on every discovered host. You can add new ports to be scanned, and you can remove existing ports from the list. **To choose a port that does not appear in the list:** 1 Click the **Add port** button to open the Add New Port dialog box. 2 Click the **Add port** button and enter the port name and number. 3 Click **OK**. **Network implications.** Note that the scanning process may affect performance on the network. Furthermore, you may need to inform machine owners that DDM will be trying to connect to their machines.
Process Discovery	Select to discover processes running on discovered hosts. You can use a regular expression to define a process filter. You can add new processes to be scanned, and you can remove existing processes from the list. **Network implications.** In the Edit Process dialog box, to prevent a toll on performance, you should refrain from entering a search pattern that is too general. For example, do not enter a process name that consists of an asterisk (*) only, as such a filter would try and retrieve all the processes running on all machines.

Schedule Discovery

Description	Enables you to define a schedule for a specific job.
Important Information	For details on scheduling DDM, see "Discovery Scheduler Dialog Box" on page 138. General information about the wizard is available in "Infrastructure Wizard" on page 142.
Wizard Map	The Infrastructure Discovery wizard contains: Infrastructure Wizard > Define IP Ranges > Define Network Credentials > Preferences > **Schedule Discovery** > Summary.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	You define a time template in the Discovery Scheduler pane of the Properties tab. For details, see "Discovery Scheduler Pane" on page 161.
Allow Discovery to run at	Choose the time at which the schedule should run.
Repeat Every	Select how often the schedule should run.

Summary

Description	Enables you to review the definitions before running discovery.
Important Information	Click **Run** to begin DDM. General information about the wizard is available in "Infrastructure Wizard" on page 142.
Wizard Map	The Infrastructure Discovery wizard contains: Infrastructure Wizard > Define IP Ranges > Define Network Credentials > Preferences > Schedule Discovery > **Summary**

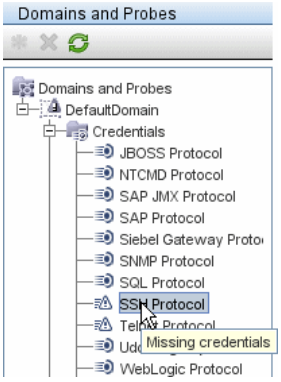
J2EE Wizard

Description	Enables you to run discovery on J2EE applications. **To access: Admin** > **Universal CMDB** > **Discovery** > **Run Discovery** > **Basic Mode**. Select the J2EE wizard from the list in the left pane. Click **Open Wizard**.
Important Information	For more information, hold the pointer over a question mark icon.
Wizard Map	The J2EE Discovery wizard contains: **J2EE Wizard** > Define Network Credentials > Configure Ports > WebLogic > WebSphere > JBoss > Oracle Application Server > Schedule Discovery > Summary.

Define Network Credentials

Description	Enables you to configure connection data for each protocol.
Important Information	You configure protocols depending on what you want to discover and on which protocols are supported on your site's network. For a list of protocols, see "Domain Credential References" on page 83. General information about the wizard is available in "J2EE Wizard" on page 150.
Wizard Map	The J2EE Discovery wizard contains: J2EE Wizard > **Define Network Credentials** > Configure Ports > WebLogic > WebSphere > JBoss > Oracle Application Server > Schedule Discovery > Summary.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Add new connection details for selected protocol type.
	Remove a protocol.
	Edit a protocol. Click to open the Protocol Parameters dialog box.
	Move a protocol up or down. DDM executes all the protocols in the list with the first protocol taking priority.
Protocol	Click to view details on the protocol, including user credentials. **Note:** A missing credential is represented by the following icon: 

Configure Ports

Description	Enables you to choose the port number and port type through which to connect to the J2EE application.
Important Information	General information about the wizard is available in "J2EE Wizard" on page 150.
Wizard Map	The J2EE Discovery wizard contains: J2EE Wizard > Define Network Credentials > **Configure Ports** > WebLogic > WebSphere > JBoss > Oracle Application Server > Schedule Discovery > Summary.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Click to add a port to the port list. The Add New Port dialog box opens. Select the ports and click OK. To edit existing system ports, in the Add New Port dialog box, click **Edit Known System Ports**. The Edit Known System Ports dialog box opens. Select the port and click the **Edit** button. In the dialog box that opens, make changes to the entries and click **OK**. To add a port to the list, in the Edit Known System Ports dialog box, click the **Add** button. Enter details of the port name, number and type and click **OK**.
	Select a port and click the button to remove the port from the list.

WebLogic

Description	Enables you to select the jar files for specific WebLogic versions.
Important Information	DDM supports the following WebLogic versions: 6.0, 6.1, 7.0, 8.1, 9.0, 9.1, and 9.2. General information about the wizard is available in "J2EE Wizard" on page 150.
Wizard Map	The J2EE Discovery wizard contains: J2EE Wizard > Define Network Credentials > Configure Ports > **WebLogic** > WebSphere > JBoss > Oracle Application Server > Schedule Discovery > Summary.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
WebLogic version	Select the check box for the versions to be discovered. Click **Import file...** to open a browse window. Browse to the appropriate WebLogic jar file.

WebSphere

Description	Enables you to select the jar files for specific WebSphere versions.
Important Information	DDM supports the following WebSphere versions: 6.0 and 6.1. General information about the wizard is available in "J2EE Wizard" on page 150.
Wizard Map	The J2EE Discovery wizard contains: J2EE Wizard > Define Network Credentials > Configure Ports > WebLogic > **WebSphere** > JBoss > Oracle Application Server > Schedule Discovery > Summary.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
WebSphere version	Select the check box for the versions to be discovered. Click **Import file...** to open a browse window. Browse to the appropriate WebSphere jar file.

JBoss

Description	Enables you to select the jar files for specific JBoss versions.
Important Information	General information about the wizard is available in "J2EE Wizard" on page 150.
Wizard Map	The J2EE Discovery wizard contains: J2EE Wizard > Define Network Credentials > Configure Ports > WebLogic > WebSphere > **JBoss** > Oracle Application Server > Schedule Discovery > Summary.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
JBoss version	Select the check box for the versions to be discovered. Click **Import file...** to open a browse window. Browse to the appropriate JBoss jar file.

Oracle Application Server

Description	Enables you to discover Oracle application servers.
Important Information	General information about the wizard is available in "J2EE Wizard" on page 150.
Wizard Map	The J2EE Discovery wizard contains: J2EE Wizard > Define Network Credentials > Configure Ports > WebLogic > WebSphere > JBoss > **Oracle Application Server** > Schedule Discovery > Summary.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A-Z)	Description
Discover Oracle Application Server (version 10g)	Select to run DDM for the Oracle Application Server, version 10g.

Schedule Discovery

Description	Enables you to define a schedule for a specific job.
Important Information	General information about the wizard is available in "J2EE Wizard" on page 150.
Wizard Map	The J2EE Discovery wizard contains: J2EE Wizard > Define Network Credentials > Configure Ports > WebLogic > WebSphere > JBoss > Oracle Application Server > **Schedule Discovery** > Summary.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	You define a time template in the Discovery Scheduler pane of the Properties tab. For details, see "Discovery Scheduler Pane" on page 161.
Allow Discovery to run at	Choose the time at which the schedule should run.
Repeat Every	Select how often the schedule should run.

Summary

Description	Enables you to review the definitions before running discovery.
Important Information	To make changes to the run, click the **Back** button. General information about the wizard is available in "J2EE Wizard" on page 150.
Wizard Map	The J2EE Discovery wizard contains: J2EE Wizard > Define Network Credentials > Configure Ports > WebLogic > WebSphere > JBoss > Oracle Application Server > Schedule Discovery > **Summary**

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
Run	Click to run DDM.

Job Editor Dialog Box

Description	Enables you to create a job. **To access:** Right-click a module in the Discovery Modules pane, and choose **Create New Job**.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Add TQL.
	Delete TQL.
	Click to define a time template in the Discovery Scheduler pane of the Properties tab. For details, see "Time Templates Dialog Box" on page 164.
	Click to open the Edit Probe Limitation dialog box. For details, see "Edit Probe Limitation for TQL Output Dialog Box" on page 140.
Description	A description of the job.
Discovery Pattern	Click the **Select** button to open the Choose Discovery Patterns dialog box. Use the SHIFT or CTRL keys to select several patterns.
Discovery Scheduler	Click the **Edit Scheduler** button to open the Discovery Scheduler dialog box. For details, see "Discovery Scheduler Dialog Box" on page 138. **Allow discovery to run at.** Choose the time at which the schedule should run. **Invoke on new triggered CIs immediately.** A check mark signifies that the job runs as soon as the triggered CI reaches the Probe. If the check box is cleared, the job runs according to the schedule defined in the Schedule Manager.
Job Name	Enter a name for the job.

GUI Element (A–Z)	Description
Parameters	**Override.** Select to override the parameter value in the pattern. **Name.** The name given to the pattern. **Value.** The value defined in the pattern.
Trigger TQLs	**TQL Name.** The name defined for the TQL. **Probe Limit.** The Probes being used for the discovery process. To add or remove Probes, click the button.

Properties Tab

Description	Enables you to view and administer the properties of modules and jobs. **To access:** Click the **Properties** tab in Run Discovery.
Important Information	Depending which level you select in the Discovery Modules pane, different information is displayed in the Properties tab. If you select: ➤ **The Discovery Modules root**, all active jobs are displayed with scheduling information. Right-click a job to edit its scheduling. For details, see "Discovery Scheduler Dialog Box" on page 138. ➤ **A Discovery module**, the Description and Module Jobs panes are displayed. To edit a description, make changes in the Description pane and click **OK**. See also "Module Jobs Pane" on page 159. ➤ **A job**, the Discovery Pattern, Discovery Scheduler, Parameters, and Trigger TQLs panes are displayed. For details, see "Discovery Pattern Pane" on page 159, "Discovery Scheduler Pane" on page 161, "Parameters Pane" on page 161, and "Trigger TQLs Pane" on page 162.

Discovery Pattern Pane

Description	Enables you to edit the pattern used by the job and to change the pattern's description. **To access:** Select a job in the Discovery Modules pane in the Run Discovery window.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
Description	To change a pattern's description, right-click the job in the Discovery Modules pane and select **Edit pattern**.
Edit	Click to show details of the pattern in the Manage Discovery Resources window. For details, see Chapter 7, "Manage Discovery Resources."

Module Jobs Pane

Description	Enables you to view the active jobs for a specific module. **To access:** Select a module in the Discovery Modules pane in the Run Discovery window.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	**Add Discovery Job to Module.** Opens the Choose Discovery Jobs dialog box where you can select jobs from more than one zip file. (Use the SHIFT or CTRL key to select several jobs.)
	**Remove Selected Discovery Job from Module.** Select the job and click the button. (No message is displayed. To restore the job, click the **Cancel** button.)
<Column title>	Click a column title to change the order of the CITs from ascending to descending order, or vice versa.

GUI Element (A–Z)	Description
<List of jobs>	All jobs included in the module. (Displayed when a specific module is selected in the Discovery Modules pane.)
<right-click menu>	Right-click a row to open the Discovery Scheduler for the selected job. For details, see "Discovery Scheduler Dialog Box" on page 138. Right-click a column title to customize the table. Choose from the following options: ➤ **Hide Column.** Select to hide a specific column. ➤ **Show All Columns.** Displayed when a column is hidden. ➤ **Customize.** Select to display or hide columns and to change the order of the columns in the table. Opens the Columns dialog box. ➤ **Auto-resize Column.** Select to change a column width to fit the contents. For details, see "Select Columns Dialog Box" in *Reference Information*.
Invoke Immediately	➤ A check mark signifies that the Discovery job runs as soon as the triggered CI reaches the Probe. In this case, the Invoke on new triggered CIs immediately check box is selected in the Properties tab. ➤ If this column does not contain a check mark, the job runs according to the schedule defined in the Schedule Manager.
Job Name	The name of the job and the package in which the job is included. (Displayed when a job is selected in the Discovery Modules pane.)
Schedule Information	The scheduling information of the job as defined in the Discovery Scheduler.
Trigger TQLs	The name of the TQL that activated the job.

Discovery Scheduler Pane

Description	Enables you to view information about the schedule set up for this job. **To access:** Select a job in the Discovery Modules pane in the Run Discovery window.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Click to add times to the Enable Discovery to run at list. The Time Templates dialog box opens. To add a time template to the list, in the Time Templates dialog box, click the **Add** button to open the Edit Time Template dialog box. For details, see "Edit Time Template Dialog Box" on page 141.
Edit Scheduler	Click to open the Discovery Scheduler. For details, see "Discovery Scheduler Dialog Box" on page 138.
Invoke on New Trigger CIs Immediately	A check mark signifies that the job runs as soon as the Trigger CI reaches the Probe. If this column does not contain a check mark, the job runs according to the schedule defined in the Schedule Manager.
Time Template	Choose a template that includes the days and times when the job should run.

Parameters Pane

Description	Enables you to override pattern behavior. For example, to change timeout, select the **Override** check box and change the value of the parameter. Click **OK** to save the change. For details on editing parameters, see "Discovery Pattern Parameters Pane" on page 196. To view a description, hold the pointer over the parameter. **To access:** Select a job in the Discovery Modules pane in the Run Discovery window.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
Name	The name given to the pattern.
Override	Select to override the parameter value in the pattern.
Value	The value defined in the pattern.

Trigger TQLs Pane

Description	Enables you to define one or more TQL queries to be used as triggers to activate the selected job. **To access:** Select a job in the Discovery Modules pane in the Run Discovery window.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	**Add TQL.** You can add one or more non-default TQL queries to be used as triggers to activate the selected job. Click to open the **Choose Discovery TQL** dialog box.
	**Remove TQL.** Select the TQL and click the button. (No message is displayed. To restore the TQL, click the **Cancel** button.) **Note:** If a TQL query is removed for an active job, DDM no longer receives new CIs coming from that TQL query. Existing Trigger CIs that originally came from the TQL query are not removed.
	Click to add or remove Probes for a specific TQL. For details, see "Edit Probe Limitation for TQL Output Dialog Box" on page 140.

GUI Element (A–Z)	Description
	Click to open the Trigger TQL Editor. For details, see "Trigger TQL Editor" on page 165.
	Click to open the Query Manager. For details, see Chapter 7, "Query Manager."
Probe Limit	The Probes being used for the discovery process. To add or remove Probes, click the button.
TQL Name	The name of the trigger TQL query that activates the job.

Related CIs Window

Description	Enables you to view, in map format, the CIs that are related to a selected CI. **To access:** In the **CIs Discovered by** dialog box, right-click a CIT and select **Get Related CIs**.
Important Information	Related CIs are CIs that are the parent, child, or sibling of an existing CI.
Useful Links	For details on all options in the Get Related CIs menu, see "Get Related CIs" on page 114.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A-Z)	Description
<menu>	For details, see "Topology Map" in *Model Management*.
<Topology Map>	For details, see "Topology Map Overview" in *Model Management*.

Show Results for Triggered CI Page

Description	Enables you to view the results of running an ad-hoc request to the Probe. DDM acquires the results by running the job on a selected trigger CI. In the case of an error, a message is displayed. **To access:** Run Discovery, select a module or job, select the Details tab. In the Discovery Status pane, drill down to a CI, right-click it, and choose **Show Results for Triggered CI**.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A-Z)	Description
	Select a CIT and click to display additional information in the Sources CIs dialog box. For details, see "Source CIs Dialog Box" on page 164.
	Click to open a topology map showing a result map for the Triggered CI. Right-click a CIT to view its properties.

Source CIs Dialog Box

The Source CIs dialog box includes the same components as the CIs Discovered by [Module Name] dialog box. For details, see "CIs Discovered by [Module or Job Name] Dialog Box" on page 111.

Time Templates Dialog Box

Description	Enables you to define a daily or weekly schedule to run selected jobs. **To access: Run Discovery > Properties tab > Discovery Scheduler pane > Time Template icon**.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Click to add a time template. Opens the Edit Time Template dialog box.
	Select a time template and click to delete.
	Select a time template and click to edit it. Opens the Edit Time Template dialog box.

Trigger TQL Editor

Description	Enables you to edit a TQL that has been defined to trigger jobs. **To access: Run Discovery** > **Properties tab** > **Trigger TQLs pane** > select a TQL and click the **Open the TQL Editor** button.
Useful Links	➤ "Trigger CITs, Trigger CIs, Input TQLs, and Trigger TQLs" on page 50 ➤ "Input TQL Editor Window" on page 185

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
<Panes>	➤ CI Types Pane ➤ Editing Pane ➤ Information Pane
TQL Name	The name of the trigger TQL query that activates the job.

CI Types Pane

Description	Displays a hierarchical tree structure of the CI Types found in the CMDB. For more details, see "CI Type Manager User Interface" in *CI Attribute Customization*. **Note**: The number of instances of each CIT in the CMDB is displayed to the right of each CIT. To create or modify a TQL query, click and drag nodes to the Editing pane and define the relationship between them. Your changes are saved to the CMDB. For details, see "Add Nodes and Relationships to a TQL Query" in *Model Management*.
Included in Tasks	➤ "Define a TQL Query" on page 267 ➤ "Pattern View Workflow" on page 141

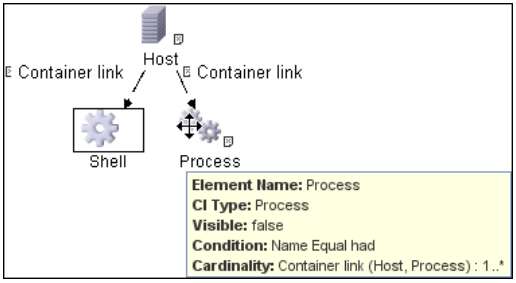
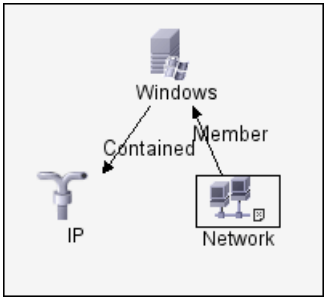
Editing Pane

Description	Enables you to edit the node selected in the Trigger TQLs pane.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
<node>	Click to display information about the node in the information pane.
<right-click menu>	For details, see "Context Menu Options" in *Model Management*.
<Toolbar>	For details, see "Toolbar Options" in *Model Management*

Information Pane

Description	Displays the properties, conditions, and cardinality for the selected node and relationship.
Important Information	Hold the pointer over a node to view information:  **Visible.** A tick signifies that the selected node or relationship is visible in the topology map. When the node/relationship is not visible, a box appears to the right of the selected node/relationship in the Editing pane:  * A small green indicator appears next to the tabs in which a change occurred.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A-Z)	Description
Edit	To view information, select a node or relationship in the Editing pane, select the tab in the Information Pane, and click the **Edit** button. For details on the Node Condition dialog box, see "Node/Relationship Properties Dialog Box" in *Model Management*.
Attributes	Displays the attribute conditions defined for the node or the relationship. For details, see "Attribute Tab" in *Model Management*.
Cardinality	Cardinality defines how many nodes you expect to have at the other end of a relationship. For example, in a relationship between host and IP, if the cardinality is 1:3, the TQL retrieves only those hosts that are connected to between one and three IPs. For details, see "Cardinality Tab" in *Model Management*.
Details	➤ **CI Type.** The CIT of the selected node/relationship. ➤ **Include subtypes.** Display both the selected CI and its descendants in the topology map.
Qualifiers	Displays the qualifier conditions defined for the node or the relationship. For details, see "Qualifier Tab" in *Model Management*.
Selected Identities	Displays the element instances that are used to define what should be included in the TQL results. For details, see "Identity Tab" in *Model Management*.

7

Manage Discovery Resources

This chapter provides information on managing Discovery and Dependency Mapping resources.

This chapter includes:

Concepts

Reference

Resource Files

The following files can be changed to enable DDM in non-default systems. The location of these files is: **Manage Discovery Resources** > **Network** > **Configuration Files**.

This section includes the following topics:

portNumberToPortName.xml

The portNumberToPortName.xml file is used by DDM as a dictionary to create Port CIs. When a port is discovered, the Probe extracts the port's name from this file, and creates the Port CI accordingly. If the port number does not appear in this file, the port name is used as the port number.

You edit this file when adding new ports to be discovered.

Note: The results of running a Network – TCP discovery appear in the Topology Map with the port names instead of the port numbers (the port title is the value of the Port Name attribute, defined in the CIT). For details, see "Add/Edit Attribute Dialog Box" in *CI Attribute Customization*.

To define a new port:

1 In the Manage Discovery Resources window, access portNumberToPortName.xml and search for the file by clicking the **Find resource** button and entering **portNumber** in the **Name** box. Click **Find Next**, then click **Close**.

The file is selected in the Discovery Resources pane and the file contents are displayed in the View pane.

2 Add another row to the file and make changes to the parameters:

```
<portInfo portProtocol="xxx" portNumber="xxx" portName="xxx" discover="0"/>
```

- **portProtocol.** The network protocol used for discovery (udp or tcp).
- **portNumber.** The port number to be discovered.
- **portName.** The name that is to be displayed for this port.
- **discover.** 1. This port must be discovered. 0: This port should not be discovered.

oidToHostClass.xml

The oidToHostClass.xml file contains a list of OID (Discovery and Dependency Mapping) numbers, for all CIs in the system that have an ID. This list is required for mapping CIs to their correct CIT, and for converting the discovered OID number of an operating system or a device into string data.

To access the oidToHostClass.xml file, in Manage Discovery Resources, search for the file by clicking the **Find resource** button and entering **oidto** in the **Name** box. Click **Find Next**, then click **Close**.

The file is selected in the Discovery Resources pane and the file contents are displayed in the View pane.

Note: If an OID is discovered and its details do not appear in the oidToHostClass.xml file, its CIT is registered in the CMDB as host.

---

oidToHostClass.xml includes the following parameters:

- **class.** The converted CIT name of the discovered OID. Under this name, the operating system or device appears in the CMDB and in Business Availability Center.
- **vendor.** The vendor of the operating system or device.
- **os.** A specific operating system, for example, Linux. This parameter is optional.
- **model.** A specific model, for example, JETDIRECT,JD30. This parameter is optional.
- **oid.** The discovered OID.

globalFiltering.xml

This file enables you to filter Probe results for all patterns, so that only results of interest to you are sent to the Business Availability Center server. (You can also filter specific patterns. For details, see "Pattern Management Tab" on page 189.)

To add a global filter:

1 Access the globalFiltering.xml file: in Manage Discovery Resources, open the Network folder and click the Configuration Files folder. Select the file to display the code in the View pane.

2 Locate the <includeFilter> and <excludeFilter> markers:

- **<includeFilter>.** When a vector marker is added to this filter, all CIs that do not match the filter are removed. If this marker is left empty, all results are sent to the server.
- **<excludeFilter>.** When a vector marker is added to this filter, all CIs that match the filter are removed. If this marker is left empty, all results are sent to the server.

  The following example shows an ip CI that has address and domain attributes:

```
<vector>
    <object class="ip">
        <attribute name="ip_address" type="String">192\.168\.82\.17.*</attribute>
        <attribute name="ip_domain" type="String">DefaultProbe</attribute>
    </object>
</vector>
```

  If this vector is defined in <includefilter>, all results not matching the filter are removed. The results sent to the server are those where the ip_address matches the regular expression **192\.168\.82\.17.*** and the ip_domain is **DefaultProbe**.

  If this vector is defined in <excludefilter>, all results matching the filter are removed. The results sent to the server are those where the ip_address does not match the regular expression **192\.168\.82\.17.*** and the ip_domain is not **DefaultProbe**.

  The following example shows a network CI that has no attributes. All network results are sent to the server:

```
<vector>
    <object class="network">
    </object>
</vector>
```

Note:

- Attributes in the filter should be of type string only. For details on attribute types, see "Attributes Page" in *CI Attribute Customization*.
- A result is considered to be a match only if all filter attributes have the same values as those in the CI. (If one of a CI's attributes is not specified in the filter, all the results for this attribute match the filter.)
- A CI can match more than one filter. The CI is removed or remains according to the filter in which it is included.
- DDM filters first according to the <includeFilter> and then applies the <excludeFilter> on the results of <includeFilter>.

Configuration Files

The following files are for internal use only and should only be changed by users with an advanced knowledge of pattern-writing:

- **discoveryPolicy.xml.** Includes the schedule when the Probe does not execute tasks. For details, see "Add/Edit Policy Dialog Box" on page 72.
- **jythonGlobalLibs.xml.** A list of default Jython global libraries that DDM loads before running scripts.

Handling Deleted CIs

DDM can automatically delete CIs that have been removed from a system. This is useful in cases where DDM can calculate directly that a CI has been removed (and does not have to rely on the aging mechanism to perform the calculation).

For example, when discovering installed software, DDM can compare the pattern result to the software that was previously discovered. Software that no longer exists on the machine is considered as removed and the software CIs that were previously discovered are automatically deleted.

You can choose to automatically delete removed CIs. For details, see “Relevant CITs Pane” on page 193.

For details on aging, see the Results Management pane in “Pattern Management Tab” on page 189.

Manage Discovery Resources User Interface

This section describes:

Choose Configuration Item Type Dialog Box

Description	Enables you to select CITs needed when running DDM on a specific pattern. **To access:**  ➤In the Find Discovery Resources dialog box, click the button, when Pattern input type or Pattern output type is selected.  ➤In the Choose Discovered CITs dialog box, click the **Links** tab. In the End 1 (or End 2) box, click the button.

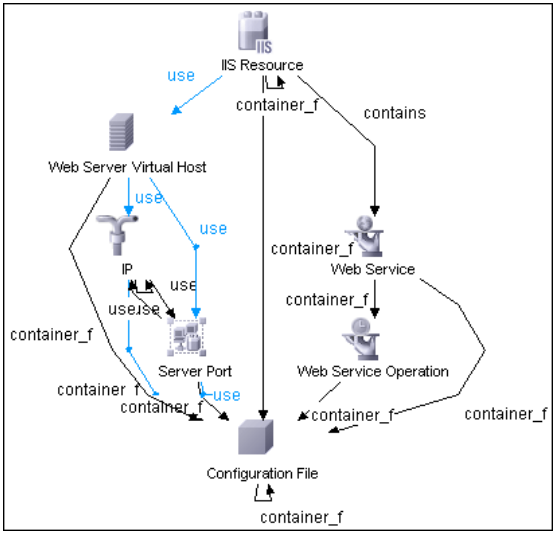
The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
<Configuration item types>	The CITs are organized in folders according to the CIT hierarchy.

Choose Discovered CITs Dialog Box

Description	Enables you to choose CITs that are to be discovered by a selected pattern and to limit links so that they are mapped only when they connect specific CITs. **To access: Admin** > **Universal CMDB** > **Discovery** > **Manage Discovery Resources**. In the Discovery Resources pane, select a pattern. In the **Pattern Signature tab** > **Discovered CITs** pane, click the **Add Discovered CIT** button.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

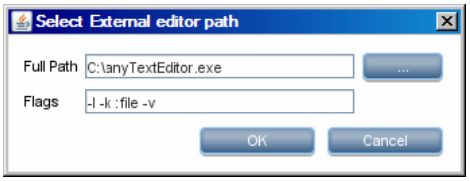
GUI Element (A–Z)	Description
Link	Select a link type from the list and click the ellipsis button  in the End 1 and End 2 boxes to open the Choose Configuration Item Type dialog box. Choose the CITs that DDM should map when they are linked by the selected link type. **Note:** DDM automatically recognizes the links between CIs and adds them to the map of discovered CITs. However, during pattern writing, you may want to exclude links between certain CITs. For example, both hosts and IPs and hosts and ports are linked by **use**. You may want to receive results only for hosts and IPs that are connected by the **use** link, and not hosts and ports. The End 1 and End 2 links determine the result received from the pattern, and this result is reflected in the map, as can be seen in the following example: 
Object	Select a CIT to be added to the list of CITs that a pattern is to discover. Save the changes by clicking the **Save** button at the bottom of the Pattern Signature pane.

Configuration File Pane

Description	Enables you to edit a specific configuration file that is part of a package. For example, you can edit the **portNumberToPortName.xml** file so that specific port numbers, names, or types are discovered. **To access:** Click a specific configuration file in the Discovery Resources pane.
Important Information	The following files are for internal use only and should only be changed by users with an advanced knowledge of pattern-writing: ➤ discoveryPolicy.xml ➤ jythonGlobalLibs.xml For details, see "Resource Files" on page 169 and "Configuration Files" on page 173.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Find specific text in the configuration file. For details, see "Find Text Dialog Box" on page 184.
	Click to go to a specific line in the configuration file. In the Go To Line dialog box, enter the line number.
	Click to open the file in an external editor.

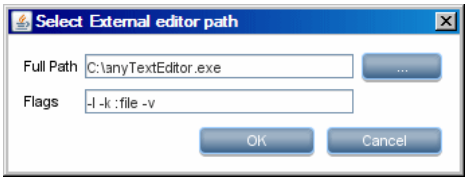
GUI Element (A–Z)	Description
	Click to edit the external editor preferences. You can run the editor by adding flags to the path. In the following example:  **:file** sets the place of the file in relation to the flags. The user cannot set the file name.
	For XML files, signifies that the code is valid.
	For XML files, signifies that the code is not valid.

Discovery Pattern Source Editor Window

Description	Enables you to edit a pattern script. **To access:** Right-click a pattern in the Discovery Resources pane and select **Edit Pattern Source**.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Find specific text in the pattern script. For details, see “Find Text Dialog Box” on page 184.
	Click to go to a specific line in the pattern script. In the **Go To Line** dialog box, enter the line number.
	Click to open the pattern script in an external text editor.

GUI Element (A–Z)	Description
	Click to edit the external editor preferences. You can run the editor by adding flags to the path. In the following example:  **:file** sets the place of the file in relation to the flags. The user cannot set the file name.
	Signifies that the code is valid.
	Signifies that the code is invalid.

Discovery Resources Pane

Description	Enables you to locate a specific package, pattern, script, configuration file, or external resource. **To access: Admin > Universal CMDB > Discovery > Manage Discovery Resources**
Important Information	Depending which level you select in the Discovery Resources pane, different information is displayed in the View pane. If you select: ➤ One of the following folders: Discovery Packages root, a specific package, a pattern, script, configuration file, or external resource: a list of the resources in that folder is displayed. To access a resource directly, double-click the resource in the View pane. ➤ A specific pattern: The Pattern Signature and Pattern Management panes. For details, see "Pattern Signature Tab" on page 194 and "Pattern Management Tab" on page 189. ➤ A script or configuration file: The script editor. For details, see "Script Pane" on page 202. ➤ An external resource: Information about the file.
Useful Links	"Package Manager User Interface" in *Model Management*.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Click to: ➤ **Create a pattern.** Enter the pattern name and click **OK**. The new pattern is added to the <<No Package>> folder. Edit the pattern. For details, see "Pattern Signature Tab" on page 194 and "Pattern Management Tab" on page 189. For details on moving a pattern to a package, see "Create New Package/Edit Package Wizard" in *Model Management*. ➤ **Create a Jython script.** Enter the script name. For details, see "Script Pane" on page 202. ➤ **Create a configuration file.** Enter the configuration file name. By default, the file takes an .xml extension. To give the file another extension, for example, *.properties, name the file and include the extension. Add the appropriate XML code or other content. For XML files, you can save the file only if it is valid. For details, see "Configuration File Pane" on page 177. ➤ **Import an external resource.** In the browser that opens, locate the resource to be imported and click **Open**.
	Click to delete the resource.
	Click to refresh the list of packages.
	Click to open the Find Discovery Resource dialog box. For details on filtering, see "Filtering Results" on page 33.
	Discovery packages root. Displays a list of all packages.
	Package root. Displays a list of all resources included in the package. You can view any of these resources by clicking the resource in the Discovery Resources pane.

GUI Element (A–Z)	Description
<Configuration files>	Right-click a file to: ➤ **Save as.** Save the file under a new name. Use this option to clone an existing file. The new file includes all attributes of the existing file. Make any necessary changes to the file and save it. ➤ **Open in Frame.** Select to open the file in a new window.
<External resource files>	An external resource is any file needed by DDM to perform discovery. For example, the nmap.exe file is needed for credential-less discovery. ➤ Right-click a file to: ➤ **Save as.** Save the resource under a new name. Use this option to clone an existing resource. The new resource includes all attributes of the existing resource and is saved to the same location in the file system. Make any necessary changes to the new resource and save it. ➤ Select the file to display information in the View pane. You can open an external resource or export it.
<Pattern files>	Right-click a file to: ➤ **Save as.** Save the pattern under a new name. Use this option to clone an existing pattern. The new pattern includes all attributes of the existing pattern. Give a name to the new pattern, and change the necessary attributes. ➤ **Go to Discovery job.** When enabled, click to open the Run Discovery window with the job selected. This option is enabled if the pattern is included in a job. ➤ **Edit pattern source.** Opens the pattern source editor where you can make changes to the pattern. For details, see "Discovery Pattern Source Editor Window" on page 178.

GUI Element (A–Z)	Description
<Script files>	Right-click a file to: ➤ **Save as.** Save the script under a new name. Use this option to clone an existing script. The new script includes all attributes of the existing script. Make any necessary changes to the script and save it. ➤ **Open in Frame.** Select to open the script in a new window. For details on editing the script, see "Discovery Pattern Source Editor Window" on page 178.

Find Discovery Resource Dialog Box

Description	Enables you to build a search query to find a particular resource. **To access:** Click the **Find Job** filter button in the Discovery Resources pane.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Click to open the Choose Configuration Item Type dialog box. **Note:** This button is not accessible when **Name** is selected.
Direction	Searches forwards or backwards through the packages.
Find All	All resources meeting the search criteria are highlighted in the Discovery Resources pane.

GUI Element (A–Z)	Description
Find Discovery resource by	Choose between: ➤ **Name.** Enter the name, or part of it, of the resources. ➤ **Pattern input type.** CIs that trigger the job. Click the button to open the Choose Configuration Item Type dialog box. Locate the CI type that you are searching for. ➤ **Pattern output type.** CIs that are discovered as a result of the activated job.
Find Next	The next resource meeting the search criteria is highlighted in the Discovery Resources pane.

Find Text Dialog Box

Description	Enables you to find text in a script or configuration file. **To access:** Click the **Find in text** button.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
Direction	Search up or down the script.
Options	Select **Case Sensitive** to narrow your search.
Text to find	Enter the text you are searching for.

Input TQL Editor Window

Description	Enables you to define which CIs can be Trigger CIs for jobs that run a specific pattern. **To access: Manage Discovery Resources** > select a pattern > **Pattern Signature** tab > click the **Edit** button next to the Input TQL box.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
<Panes>	➤ CI Types Pane ➤ Editing Pane ➤ Information Pane
TQL Name	The name of the trigger TQL query that activates the job.

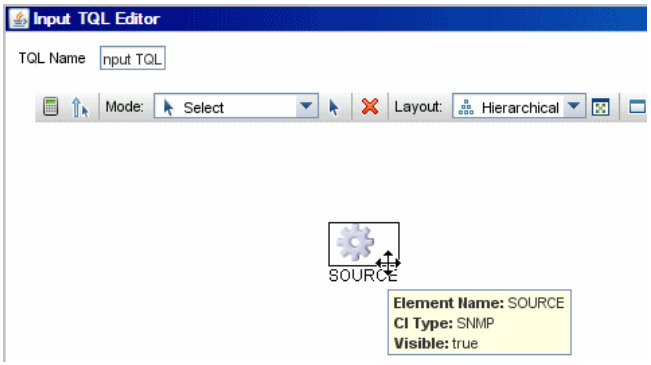
CI Types Pane

Description	Displays a hierarchical tree structure of the CI Types found in the CMDB. For more details, see "CI Type Manager User Interface" in *CI Attribute Customization*. **Note**: The number of instances of each CIT in the CMDB is displayed to the right of each CIT. To create or modify a TQL query, click and drag nodes to the Editing pane and define the relationship between them. Your changes are saved to the CMDB. For details, see "Add Nodes and Relationships to a TQL Query" in *Model Management*.

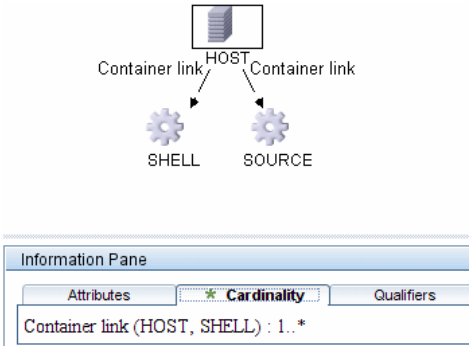
Editing Pane

Description	Enables you to edit the node.

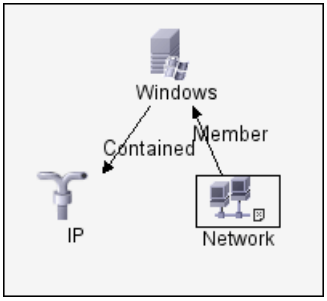
The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
<node>	Hold the cursor over a node to view information about the node: 
<Right-click menu>	For details, see "Context Menu Options" in *Model Management*
<Toolbar>	For details, see "Toolbar Options" in *Model Management*

Information Pane

Description	Displays the properties, conditions, and cardinality for the selected node and relationship.
Important Information	✱ A small green indicator appears next to the tabs that include items that can be viewed: 

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A-Z)	Description
	Click **Edit** to open the relevant dialog box for the selected tab.
Attributes	Displays the attribute conditions defined for the node or the relationship. For details, see "Attribute Tab" in *Model Management*.
Cardinality	Cardinality defines how many nodes you expect to have at the other end of a relationship. For example, in a relationship between host and IP, if the cardinality is 1:3, the TQL retrieves only those hosts that are connected to between one and three IPs. For details, see "Cardinality Tab" in *Model Management*.
Details	➤ To open the Node or Relationship Properties dialog box, select a node or relationship in the Editing pane and click the **Edit** button. For details, see "Node/Relationship Properties Dialog Box" in *Model Management*. ➤ **CI Type.** The CIT of the selected node/relationship. ➤ **Visible.** A tick signifies that the selected node/relationship is visible in the topology map. When the node/relationship is not visible, a box appears to the right of the selected node/relationship in the Editing pane:  ➤ **Include subtypes.** Display both the selected CI and its descendants in the topology map.

GUI Element (A-Z)	Description
Qualifiers	Displays the qualifier conditions defined for the node or the relationship. For details, see "Qualifier Tab" on page 324.
Selected Identities	Displays the element instances that are used to define what should be included in the TQL results. For details, see "Identity Tab" on page 326.

Pattern Management Tab

Description	Enables you to define a pattern by specifying which CITs the pattern should discover and the protocols needed to perform discovery. **To access:** Select a specific pattern in the Discovery Resources pane.
Important Information	Click the **Save** button to save any changes to this tab.
Useful Links	"The DiscoveryProbe.properties File" on page 36.

Execution Options Pane

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
Create communication logs	Choose to create a log file that logs the connection between the Probe and a remote machine. For details on viewing a communication log, see "Probe Logs" on page 56. For details on how the communication logs work, see "Recording DDM Code" on page 323. ➤ **Always/On Failure.** When selected, an ad-hoc request is set up to retrieve the communication log for the selected Trigger CI. **Always.** DDM always creates communication logs. **On Failure.** DDM automatically creates a log of the session in case discovery should fail (that is, DDM throws an exception or reports an error; report of a warning does not create a communication log). This is useful when you want to analyze which queries or operations take most of the time, send data for analysis from different locations, and so on. If the job completes successfully, no log is created. ➤ **Never.** No communication logs are saved on the file system or in memory. The communication log can reside in two places: in the Probe's memory or on the file system (as a record file). When requested, DDM displays the log retrieved from the Probe's memory. If no log exists in the memory, DDM displays the log that resides on the file system. If no log exists on the file system, a message is displayed. The communication log files are created on the Probe Manager under the **<DDM root directory>\DiscoveryProbe \root\lib\collectors\probeManager\record** folder.
Max. Execution Time	The maximum time allowed for a pattern to run on one Trigger CI.

GUI Element (A–Z)	Description
Max. Threads	Each job is run using multiple threads. You can define a maximum number of threads that can be used concurrently when running a job. If you leave this box empty, the Probe's default threading value is used (8). The default value is defined in DiscoveryProbe.properties in the defaultMaxJobThreads parameter. ➤ **regularPoolThreads.** The maximum number of worker threads allocated to the multi-threaded activity (the default is 50). ➤ **priorityPoolThreads.** The maximum number of priority worker threads (the default is 20). **Note:** The number of actual threads should never be higher than regularPoolThreads + priorityPoolThreads.

Probe Selection Pane

The following elements are included (unlabeled GUI elements are shown in angle brackets):

Description	Enables you to specify which Probe to use with a pattern. **To access:** Select a specific pattern in the Discovery Resources pane.
Important Information	By default, DDM automatically chooses the Probe for the Trigger CI according to the CI's related host. After obtaining the CI's related host, DDM chooses one of the host's IPs and chooses the Probe according to the Probe's network scope definitions. This may fail in the following situations: ➤ A Trigger CI does not have a related host (such as the **network** CIT). ➤ A triggered CI's host has multiple IPs, each belonging to a different Probe. To resolve these issues, you can specify which Probe to use with the pattern by: ➤ In the Probe Selection section, selecting **Override default probe selection**. ➤ In the Probe box, typing the Probe to use for the task.

Relevant CITs Pane

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
Automatically delete removed CIs	Select this check box to automatically delete CIs that should be removed. To add CITs to the list of CIs, click the **Add** button. In the Choose Configuration Item Type dialog box, choose the CITs. The changes you make here are added to the pattern file, for example: <resultMechanism isEnabled="true"> <autoDeleteCITs isEnabled="true"> <CIT>networkshare</CIT> </autoDeleteCITs> </resultMechanism> For details on handling CI deletion, see "Handling Deleted CIs" on page 173.

Result Grouping Pane

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
Grouping Interval (Seconds)	To group results in the Probe before they are sent to the server, type the value that indicates how long results are stored in the Probe before being transferred to the server. **Note**: If you enter a value in both boxes, DDM applies the value of whichever occurs first.
Max. CIs in group	Specify the number of CIs that should accumulate in the Probe before being transferred to the server.

Results Management Pane

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
Enable aging	Select this check box to run the aging mechanism that specifies how long a period must pass in which CIs are discovered, before DDM treats these CIs as no longer relevant and removes them. Aging parameters are defined in the Infrastructure Settings Manager (**Admin** > **Platform** > **Setup and Maintenance** > **Infrastructure Settings**): ➤ **Aging Scheduler Hour of the First Run.** Defines at what time aging first runs after server startup (for example, 02=2 AM). ➤ **Aging Scheduler Interval.** Defines the interval between runs. If Aging Time Unit = days, the interval value is days; if Aging Time Unit = hours, the interval value is hours. ➤ **Aging Time Unit.** The default is days. The hours option is provided to enable convenient verification checks of specific CIs.
Filter unchanged results	Select this check box for the Probe to send to the CMDB only those CIs that are unchanged since the last time results were sent to the server and that answer to the filter criteria. For details on filtering, see "Filtering Results" on page 33.

Pattern Signature Tab

Description	Enables you to define a pattern by specifying: ➤ which CITs the pattern should discover ➤ which protocols are needed to perform discovery **To access:** Select a specific pattern in the Discovery Resources pane.
Included in Tasks	"Implement a Pattern" on page 293

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Click the button to open the Input TQL editor. For details, see “Input TQL Editor Window” on page 185.
	Click to remove the Input TQL from the pattern.
Description	The description of the pattern.
Input TQL	Defines which CIs can be Trigger CIs for jobs that run this pattern. **Note:** Since this field is optional, not all patterns include an input TQL. **NA** signfies **not applicative**, that is, currently this pattern does not have an input TQL definition. Click the button to open the Input TQL Editor window. For details, see “Input TQL Editor Window” on page 185. For an explanation, see “Trigger CITs, Trigger CIs, Input TQLs, and Trigger TQLs” on page 50. For an example, see “Example of Input TQL Definition” on page 296.
Trigger CIT 	The CIT used as the trigger that activates the selected pattern. The trigger CIT is used as the pattern input. For details, see “Define Pattern Input (Trigger CIT and Input TQL)” on page 294. Click the button to open the Choose Configuration Item Type dialog box and to choose a CIT to use as the trigger.

Discovered CITs Pane

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Click to open the **Choose Discovered CIT** dialog box, to select a CIT that is to be discovered by the pattern. For details, see "Choose Discovered CITs Dialog Box" on page 175.
	Click to remove the CIT from the list of CITs that the pattern discovers.
	You can choose to view a map of the CITs and links that are discovered by the pattern, instead of a list. Click the button to open the Discovered CITs Map window. The CIs and relationship links discovered by the pattern are shown.
CITs	List of CITs that the pattern discovers.

Discovery Pattern Parameters Pane

The following elements are included (unlabeled GUI elements are shown in angle brackets):

	Click to open the Parameter Editor. Enter details on the parameter. The value you enter here is assigned to the attribute.
	Click to open the Parameter Editor and make changes.
	Click to remove a parameter.
Name	Each row represents the definitions for one parameter.
Value	Separate values with commas.

Required Discovery Protocols Pane

The following elements are included (unlabeled GUI elements are shown in angle brackets):

	Opens the **Add Required Protocol** dialog box.
	Click to remove an existing protocol.
Protocols	List of protocols required by the pattern for the task. For example, the NTCmd protocol, together with its user name, password, and other parameters, is needed for DDM to access a Windows system.

Triggered CI Data Pane

The following elements are included (unlabeled GUI elements are shown in angle brackets):

	Add Trigger CI data to the pattern.
	Remove Trigger CI data from the pattern.
	Edit the Trigger CI data in the Parameter Editor dialog box.

Name	The information that is needed to perform a task on a specific CI. This information is passed to the CI queried in the task.
Value	The attribute value. Variables are written using the following syntax: ${VARIABLE_NAME.attributeName} where **VARIABLE_NAME** can be one of three predefined variables: ➤ **SOURCE**. The CI that functions as the task's trigger. ➤ **HOST**. The host in which the triggered CI is contained. ➤ **PARAMETERS**. The parameter defined in the **Parameter** section. You can create a variable. For example, ${SOURCE.network_netaddr} indicates that the trigger CI is a network.

Used Scripts Pane

The following elements are included (unlabeled GUI elements are shown in angle brackets):

	Change the order of the scripts. DDM runs the scripts in the order in which they appear here.
	Add a script to the pattern.
	Remove a script from the pattern.
	Edit the selected script in the Script Editor that opens.
Scripts	A list of Jython scripts used by the pattern. The Jython scripts that appear in bold are the scripts that the currently selected pattern is using.

Manage Discovery Resources Window

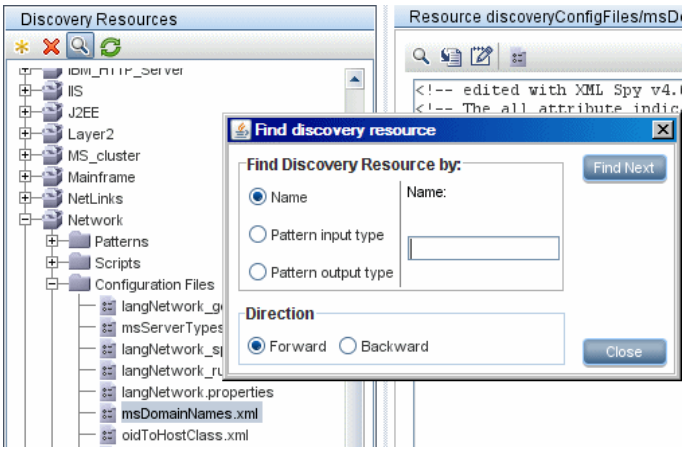
Description	Enables you to view or edit default parameter values used for the DDM process. **To access: Admin** > **Universal CMDB** > **Discovery** > **Manage Discovery Resources** or right-click a job in the Run Discovery window.
Important Information	**Note:** An asterisk (*) next to a resource (pattern, script, or configuration file) signifies that the resource has changed since the package (in which it is included) was deployed. If the original package is redeployed, the changes are deleted from the resource. To save the changes, move the resource to a new package and deploy the package (the asterisk disappears). **Caution:** Only administrators with an expert knowledge of the DDM process should delete packages.
Useful Links	"Pattern Signature Tab" on page 194 "Pattern Management Tab" on page 189 "Script Pane" on page 202 "Discovery Resources Pane" on page 180 "Configuration File Pane" on page 177

Important: The following examples include steps that modify resource files. Modification is considered a more advanced configuration and should be performed by users with an advanced knowledge of Business Availability Center only, for site-specific customization.

Example – Set Up Discovery of Microsoft Domains

To discover specific Microsoft domains on a network:

1 In the Manage Discovery Resources window, search for the msDomainNames.xml file by clicking the **Find resource** button and entering **domain** in the **Name** box:



DDM highlights the resource that includes this string.

2 Click **Find Next** until DDM highlights msDomainNames.xml in the Network package. Click **Close**.

DDM displays the file in the View pane with the results of the XML validation.

3 Locate the <**MsDomainNames all**="**true**"> marker and change **true** to **false**.

false: DDM finds only those Microsoft domain types specified in the <MsDomain> list.

true: DDM finds all existing Microsoft domain types and not only the types listed in <MsDomain>.

4 Locate the <**MsDomain**> marker and replace DISCOVERED-MS-DOMAIN with the name of the Microsoft domain.

(To add more domains to be discovered, add a <MsDomain>Domain to be discovered<MsDomain> marker for each domain.)

Example:

```
<MsDomainNames all="false">
      <MsDomain>Business Availability Center Lab</MsDomain>
      <MsDomain>Finance_servers</MsDomain>
</MsDomainNames>
```

5 Save the file. Verify that the file is valid XML by checking the Validation info.

Example – Set Up Discovery of Novell Servers

To discover the host attributes of a Novell server on a network:

1 In the Manage Discovery Resources window, search for the msServerTypes.xml file by clicking the **Find resource** button and entering **msser** in the **Name** box:

2 Click **Find Next** until DDM highlights msServerTypes.xml in the Network package. Click **Close**.

DDM displays the file in the View pane with the results of the XML validation.

3 Locate the **<MsServerType value="80" name="NOVELL" calc="false"/>** marker and change **false** to **true**.

false: DDM does not save data about this server to the database.

true: DDM saves the CIs to the CMDB (as attributes of their host).

(**value** is the discovered number of the server type. **name** is the converted name of the discovered server type. The server type appears under this name in the CMDB and in Business Availability Center.)

Example – Configure Resources for a New Package

To define a package for a pattern and configuration file that you have written for a non-default discovery:

1 To create a pattern, you can either define a new pattern or copy an existing pattern.

To define a new pattern, in the Manage Discovery Resources window, click the icon in the Discovery Resources toolbar and select **New Discovery Pattern**. For details, see "Discovery Resources Pane" on page 180.

To copy an existing pattern, select the pattern, and save the file under a new name. The new pattern is located under the << No Packages >> folder.

2 Define the configuration file in the same way, either by defining a new file or by copying an existing file.

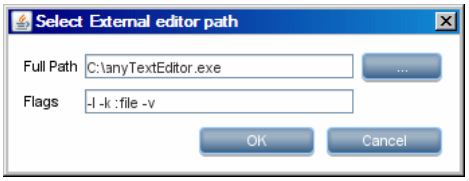
3 In Package Manager, define a new package and add the new resources to the package. For details, see "Package Manager User Interface" in *Model Management*.

Script Pane

Description	Enables you to edit a specific script that is part of a package. **To access:** Click a specific script in the Discovery Resources pane.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Find specific text in the script. For details, see "Find Text Dialog Box" on page 184.
	Click to go to a specific line in a script. In the Go To Line dialog box, enter the line number.
	Click to open the script in an external text editor.

GUI Element (A–Z)	Description
	Click to edit the external editor preferences. You can run the editor by adding flags to the path. In the following example:  **:file** sets the place of the file in relation to the flags. The user cannot set the file name.
<script>	The Jython script used by the package. For details on working with Jython, see “Step 3: Create Code” on page 304.

8

Show Status Snapshot

This chapter provides information on viewing the current status of the discovered CIs in the DDM Probes.

This chapter includes:

Concepts

Tasks

Reference

Show Status Snapshot – Overview

You use Show Status Snapshot to view the current status of the discovered CIs in the Probes. Show Status Snapshot retrieves the status from the Probes and displays the results in a view.

The view is not automatically updated; to refresh the status data, click the **Get snapshot** button.

View Current Status of Discovered CIs

This task describes how to view the current status of discovered CIs.

This task includes the following steps:

- "Prerequisites" on page 206
- "Access Show Status Snapshot" on page 206

1 Prerequisites

Verify that the Probe is enabled and is connected to the HP Universal CMDB server. For details, see "Install the DDM Probe" on page 35.

2 Access Show Status Snapshot

a Go to **Discovery** > **Show Status Snapshot**.

b Select a connected probe.

c Click the Get Snapshot button.

d Select jobs from the Progress list and click the **View Job progress** button. The Job Progress window opens.

Show Status Snapshot User Interface

This section describes:

- [Job Name] Dialog Box on page 207
- Show Status Snapshot Window on page 208

[Job Name] Dialog Box

Description	Enables you to view details abouta job, including its scheduling, as well as job statistics. **To access:** Select a job in the Progress pane of the Show Status Snapshot window and click the **View job progress** button.
Important Information	You can keep more than one dialog box open at a time. Double-click a job to open a further dialog box with details of the job.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
Job Details	**Status.** Can be **Scheduled** (the job runs according to a defined schedule) or **Running** (the job is running now). **Last updated.** The last time that the job was updated. **Threads.** The number of threads currently allocated to this job. **Progress.** The number of Trigger CIs in the job and the number of Trigger CIs that the Probe has finished working on.
Schedule	**Previous invocation.** The last time that DDM ran the job. **Next invocation.** The next time that DDM is scheduled to run the job. **Last duration.** The length of time, in seconds, taken to run the job in the previous invocation. **Average duration.** The average duration, in seconds, of the time it took the Probe to run this job. **Recurrence.** The number of times a job is run in a week. For example, if a job is scheduled to run daily, it runs 7 times in a week. If a job is scheduled to run weekly, Recurrence = 1.
Statistics Results	For details, see “Statistics Results Pane” on page 209.

Show Status Snapshot Window

Description	Enables you to view the current status of discovered CIs and all active jobs running on the Probes. **To access: Admin > Universal CMDB > Discovery > Show Status Snapshot**
Important Information	Depending on what you select in the Domains and Probes pane, different information is displayed in the View pane. If you select: ➤ a domain, you can view details and CIT statistics for the domain. For details, see "Details Pane" on page 75 and "Statistics Results Pane" on page 209. ➤ a Probe, you can view details on the Probe (such as the Probe IP), the progress of a job and you can view CIT statistics. For details, see "Details Pane" on page 208, "Progress Pane" on page 209, "Statistics Results Pane" on page 209, and "View Pane" on page 212.
Included in Tasks	"View Current Status of Discovered CIs" on page 206
Useful Links	Show Status Snapshot – Overview

Details Pane

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
Domain Type	**Customer.** A private domain used for your site. You can define several domains and each domain can include multiple Probes. Each Probe can include IP ranges but the customer domain itself has no range definition. **External.** Internet/public domain. A domain which is defined with a range. The external domain may contain only one Probe whose name equals the domain name. However, you can define several external domains in your system. For details on defining domains, see "Add New Domain Dialog Box" on page 73.

Progress Pane

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Select a CI and click this icon to view details of a job. For details, see "[Job Name] Dialog Box" on page 207.
Job	The name of the job. Double click a job to open a dialog box displaying job details. For details, see "[Job Name] Dialog Box" on page 207.
Next invocation	The next time that the Probe is scheduled to run.
Previous invocation	The last time that the Probe ran.
Progress	Can be either Scheduled or Running.
Thread count	The number of threads currently allocated to this job.
Triggered CIs	The number of CIs triggered in the job.

Statistics Results Pane

Description	Enables you to view details and CIT statistics. **To access:** Click the Default Domain or Probe name in the Domains Browser pane.

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Click to retrieve the latest data from the Probe (data is not automatically updated).
	Set the time range for which to display statistics about the CITs. ➤ **All.** Displays statistics for all job runs. ➤ **Last Hour/Day/Week/Month.** Choose a period of time for which to display statistics about the CITs. ➤ **Custom Range.** Click to open the Customize Statistics Time Range dialog box. Enter the date or click the arrow to choose a date and time from the calendar, for the **To** and **From** dates. To delete a date, click **Reset**.
	Select a CI and click this icon to view CI instances and their attributes. Opens the CIs Discovered by [Module or Job Name] Dialog Box. In the following conditions, a message is displayed: ➤ All the CIs that were discovered by this job were already discovered by another job. ➤ All the CIs that this job discovered have been deleted. ➤ The CI instances were discovered in a previous version. (From version 7.0 or later, you cannot view instances of CIs discovered in a previous version.)
<Column title>	Click a column title to change the order of the CITs from ascending to descending order, or vice versa.
<double-click a row>	For details, see “CIs Discovered by [Module or Job Name] Dialog Box” on page 111. **Note:** Not available for all CITs.

GUI Element (A–Z)	Description
<**right-click a title**>	Choose from the following options: ➤ **Hide Column.** Select to hide a specific column. ➤ **Show All Columns.** Displayed when a column is hidden. ➤ **Customize.** Select to display or hide columns and to change the order of the columns in the table. Opens the Columns dialog box. ➤ **Auto-resize Column.** Select to change a column width to fit the contents. For details, see "Select Columns Dialog Box" in *Reference Information*.
CIT	The name of the discovered CIT.
Created	The number of CIT instances created by the Probe.
Deleted	The number of CIT instances deleted by the Probe.
Discovered CIs	The sum of all the CIs for all the invocations.
Filter	The time range set with the Set Filter button.
Last updated	The date and time that the statistics table has been updated for a particular Probe.
Total	The total number of CIs in each column.
Updated	The number of CIT instances that have been updated.

View Pane

The following elements are included (unlabeled GUI elements are shown in angle brackets):

GUI Element (A–Z)	Description
	Click to view the current status of the discovered CIs and jobs on the selected Probe.
Last updated	The date and time at which the Get snapshot button was last pressed (that is, the date and time of the data displayed in Status Snapshot).
Probe IPs	The IP addresses defined for the Probe.
Running jobs	The number of jobs running on the Probe.
Scheduled jobs	The number of jobs that are scheduled to run according to the settings in the Discovery Scheduler. For details, see "Discovery Scheduler Dialog Box" on page 138.
Status	The status of the Probe (either disconnected or connected).
Threads	The sum of all threads currently allocated to the running jobs.

Part III

Advanced Discovery

9

Discovery and Dependency Mapping Content

This chapter explains how to discover specific components on your system.

This chapter includes:

Tasks

Application Signature Discovery

This task describes how to discover applications.

This task includes the following steps:

1 Overview

Application Signature discovers applications running in your environment. It maps application process names against additional data such as open port processes and command line processes and creates the appropriate application CI in the CMDB for each discovered process.

Application Signature uses the information in the Probe database, and runs queries against the Probe database, retrieving the relevant information. Application Signature does not perform discovery on the network.

Application Signature also maps the relationships between applications, based on the relationships between processes. In addition, it discovers an application's configuration file and its location, as it is listed in the applicationsSignature.xml file.

2 Job Dependencies

As Application Signature takes information from the Probe database, before running the Application Signature module, you must first run the following jobs:

- **Host Resource.** To discover the processes, activate **Host Resources by SNMP**, **Host Resources by Shell**, or **Host Resources by WMI**.
- **Network – TCP Discovery**. To discover open ports, process to port relationships and TCP connections, activate **Collect Network Data By Shell or SNMP**.

3 The applicationsSignature.xml File

The applicationsSignature.xml file contains a list of all applications that Discovery and Dependency Mapping attempts to find in the environment. You can add applications to this list.

To add applications to the list:

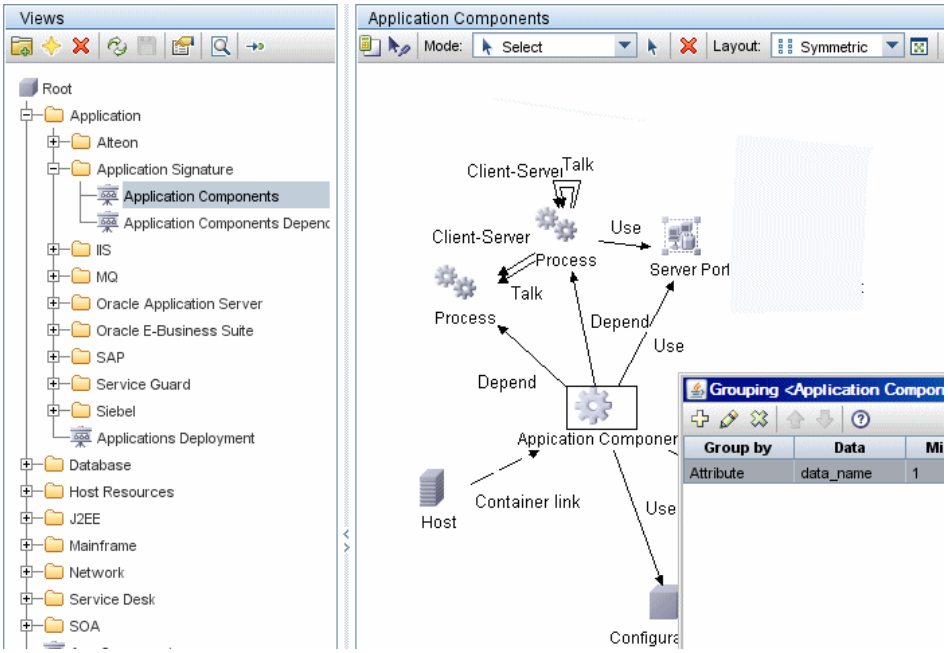
a Access the configuration file: **Discovery** > **Manage Discovery Resources** > **Discovery Resources pane** > **ApplicationSignature package** > **Configuration Files** > **applicationsSignature.xml**.

b Add a new application component to the file according to the following example:

```
<Application-Component name="HP UCMDB Discovery Probe">
    <process name="discovery_probe.exe" ports=",1977" cmdline="" />
    <configfile path=""/>
</Application-Component>
```

where:

- **Application-Component name.** The name of the application. Enter up to 100 characters. The name of the application is saved in the Application Component CI within the data_name attribute:



- **process name.** The executable file that contains the information needed to map the application.

- **ports.** The ports that the application uses. The syntax for this parameter is as follows:

  If the process has to listen on a specific port, the port should be listed. You can enter more than one port, separated by commas, for example, **ports="8888,8081,8080,81,8000,82,80"**.

  If the process does not have to listen at a specific port (that is, the application can use any port), the ports parameter should remain empty, for example, **ports=""**.

If the process does not have to listen at a specific port, but can listen on that port, enter a comma followed by the port list, for example, **ports**="**,8080,8089**".

- **cmdline.** The application can also be mapped using the process name parameter. In this case you must add a process command line (or part of it) with which the process name uniquely identifies the application.
- **configfile path.** The location of the application's configuration file.

4 Overlapping Processes

If your system includes two applications that are called by the same executable file, DDM creates CIs for both applications on the same host. For example, in the applicationsSignature.xml file, HP Business Availability Center and HP Universal CMDB are both called with MercuryAS.exe:

```
<Application-Component name="BAC">
        <process name="MercuryAS.exe" ports="" cmdline=""/>
    </Application-Component>
<Application-Component name="UCMDB">
        <process name="MercuryAS.exe" ports="" cmdline=""/>
    </Application-Component>
```

To prevent this happening, you must add a further **process name** definition to the application component. For example, to differentiate between HP Business Availability Center and HP Universal CMDB, add the following process name to each component:

```
<Application-Component name="BAC">
        <process name="MercuryAS.exe" ports="" cmdline=""/>
        <process name="bac.exe" ports="" cmdline="" required="true"/>
    </Application-Component>
    <Application-Component name="UCMDB">
        <process name="MercuryAS.exe" ports="" cmdline=""/>
        <process name="ucmdb.exe" ports="" cmdline="" required="true"/>
    </Application-Component>
```

The result is that only one application CI is created, according to which process is found, bac.exe or ucmdb.exe.

In the case that a process includes **required="false"** the process is disregarded during discovery.

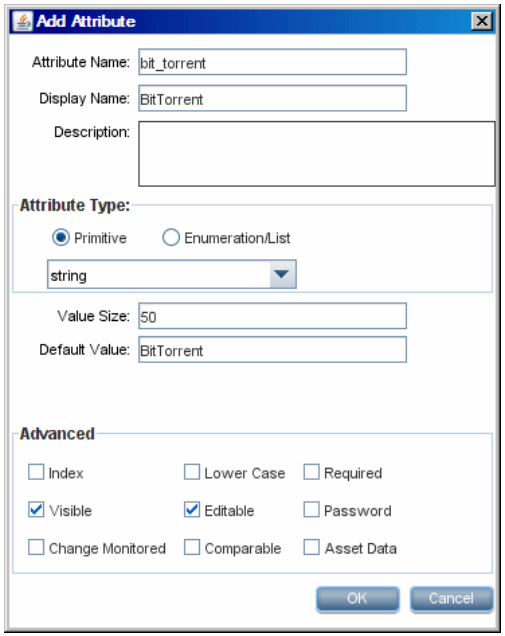
5 Automatically Set an Attribute to a Predefined Value

You can add attributes to a component so that a specific application component can be discovered. For usage, see the following example.

Example

This example explains how to discover Microsoft SQL Server databases that contain a specific BitTorrent attribute.

a In the CIT Manager, add an attribute to the Application Component CIT, as follows and save the change:



The class model is updated with the new attribute. For details on adding an attribute, see "Attributes Page" in *CI Attribute Customization*.

b In the **applicationsSignature.xml** file (**Discovery** > **Manage Discovery Resources** > **Discovery Resources pane** > **ApplicationSignature package** > **Configuration Files**), locate the Microsoft SQL Server component (or create one) and add an attribute parameter, set to **true**:

```
<Application-Component name="Microsoft SQL Server">
<process name="sqlservr.exe" ports="sql," cmdline="" />
<process name="ms-sql-s" ports="sql," cmdline="" />
<attribute name="bit_torrent" value="true"/>
</Application-Component>
```

Save the file. For details, see "The applicationsSignature.xml File" on page 217.

Note: You can add more than one attribute to the component.

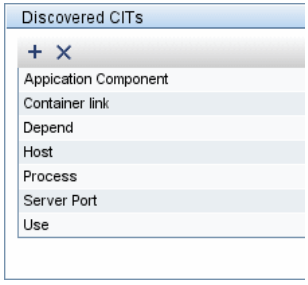
---

c In the Patterns folder, right-click the **Dis_AppComponents** pattern and choose **Go to Discovery Job** > **App Components by Network and Process Data**.

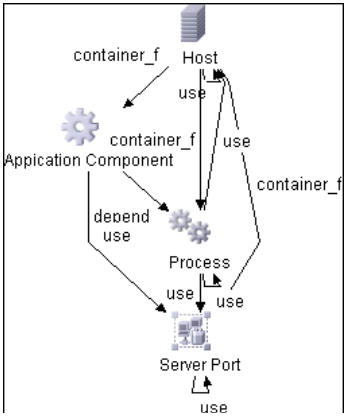
d **App Components by Network and Process** is selected in the **Discovery Modules pane**. Activate the job. For details on the Discovery Module pane, see "Discovery Modules Pane" on page 135.

e In View Manager, build a query that displays all applications where the **bit_torrent** attribute is set to **true**. For details, see "Create and Populate an Instance View" in *Model Management*.

6 Discovered CIs



7 Topology Map



8 Discovery Workflow

In the Run Discovery window, activate the Application Signature jobs in the following order:

- **App Components by Network and Process Data.** Retrieves application information with its related processes and open ports.
- **App Components CF by Shell.** Retrieves the application's configuration file and maps it to the correct application by referring to the applicationsSignature.xml file. The triggered CIs are application components that have Shell running on their host and that include a configuration file definition retrieved from applicationsSignature.xml.

9 Troubleshooting and Limitations

Problem. The error message **No data processed** is displayed.

Solution. Ensure that Processes by TTY, Processes by WMI, and TCP Discovery have already run. Either run or rerun **Host Resources by Shell/SNMP/WMI** and **Collect Network Data by Shell or SNMP**.

DNS Zones Discovery

This task describes how to discover DNS zones.

This task includes the following steps:

1 Overview

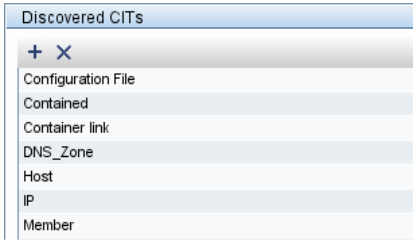
You can discover the Domain Name System (DNS) server topology, that is, the zones that each server manages, and the IPs in each zone.

- The pattern is triggered on every DNS with WMI: DDM activates the registry query to retrieve all the zones that this DNS manages.
- DDM queries each zone by activating Nslookup on the Probe against the DNS servers.

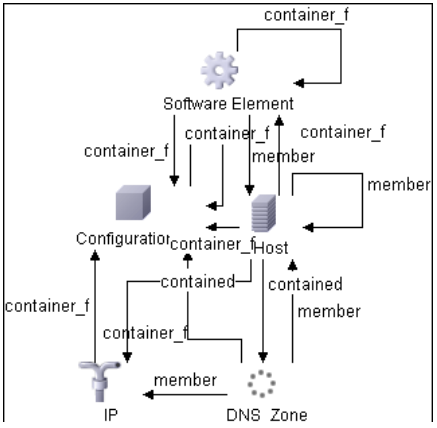
2 Supported Versions and Limitations

In many environments, Nslookup is blocked by the network administrator, in which case DDM cannot retrieve data.

3 Discovered CIs



4 Topology Map



5 Network and Protocols

Nslookup (Zone transfer)

6 Discovery Workflow

In the Run Discovery window, activate the jobs in the following order:

- DNS Zone by Nslookup.
- NTCMD connection pattern on the environment to discover the DNS servers
- Connect to these machines by WMI
- Run the pattern on the WMI that we discovered on these machines

7 Troubleshooting and Limitations

Check that Nslookup is permitted in the environment.

Host Resources Discovery

This task describes how to discover host resources.

This task includes the following steps:

1 Overview

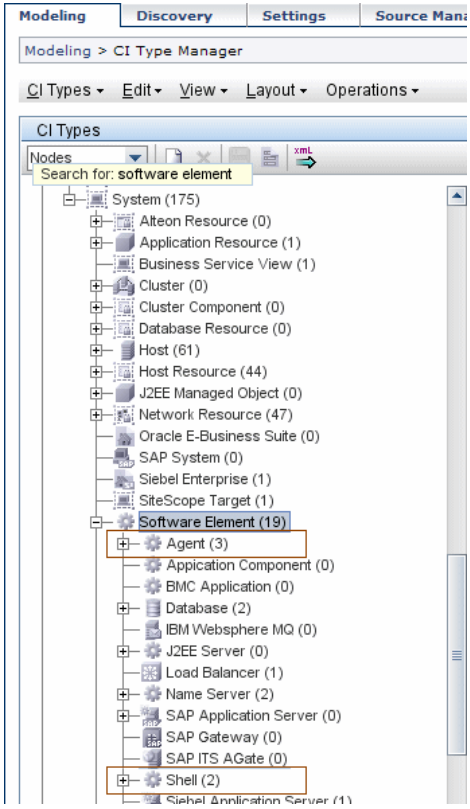
You can discover resources on Windows and UNIX hosts, for example, disk information, running processes or services, and so on.

2 Prerequisites

a Activate one of the following jobs in the **Host Resources** module:

- Host Connection by WMI
- Host Connection by SNMP
- Host Connection by Shell

b Verify that the CMDB already contains the Agent and Shell CIs: **Modeling** > **CI Type Manager**. Search for **Software Element**, and verify that Agent and Shell are present:



3 Network and Protocols

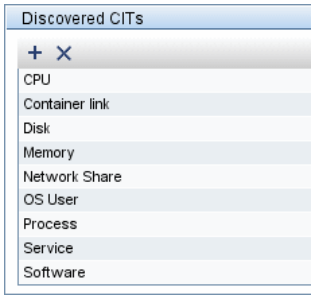
To discover host resources, define the following protocols:

- **NTCmd.** For details, see "NTCMD Protocol" on page 86.
- **SNMP**. For details, see "SNMP Protocol" on page 88.
- **SSH/Telnet.** For details, see "SSH Protocol" on page 90 and "Telnet Protocol" on page 91.
- **WMI.** For details, see "NTCMD Protocol" on page 86.

Users do not need root permissions, but do need the appropriate credentials to enable connecting to the remote machines and running the relevant commands.

4 Discovered CIs

The following CITs are discovered by the Host Resource job:

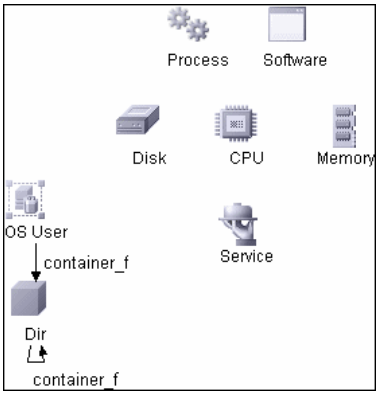


The attributes for the resource name and its root container (that is, its parent–in this case, the Host) are updated during the run.

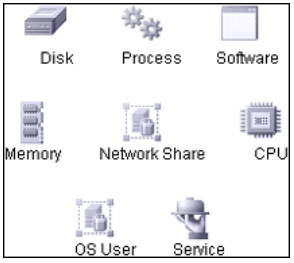
5 Topology Map

➤ Host Resources by SNMP

➤ Host Resources by Shell



➤ Host Resources by WMI



6 Discovery Workflow

In the Run Discovery window, activate the jobs in the following order:

➤ Host Resources by SNMP

➤ Host Resources by Shell

➤ Host Resources by WMI

IBM DB2 Server Discovery

This task describes how to discover IBM DB2 databases.

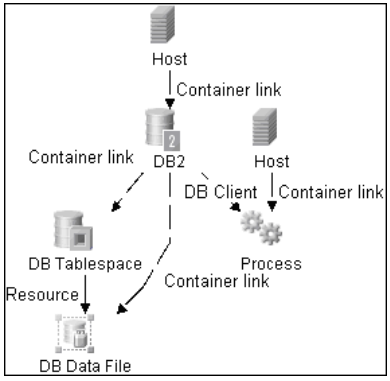
This task includes the following steps:

1 Overview

Discovers IBM DB2 Server databases and their components on the network.

2 Topology Map

The following image depicts the topology of the IBM DB2 Server view:



This view shows a host on which an IBM DB2 Server is installed, the processes that communicate with the server (connected by DB Client links), and the DB tablespaces.

3 Network and Protocols

IBM DB2 Server uses the **SQL protocol**.

4 Discovery Workflow

a Verify the user name, password, and port used by IBM DB2 Server.

b Set up the **SQL protocol**. For details, see "SQL Protocol" on page 90.In the Database Type box, choose **db2**.

c In the Run Discovery window, activate the jobs in the **Database – DB2** module in the following order:

- DB2 Connection by SQL
- DB2 Topology by SQL
- Databases TCP Ports

d For details on the CIs that are discovered, see the Statistics table in the Details tab. For details, see "Statistics Results Pane" on page 133.

5 Limitation

To perform an IBM DB2 discovery, copy the following files from the installation folder on the IBM DB2 machine to the DDM Probe machine:

- **db2java.zip**
- **db2jcc.jar**
- **db2jcc_license_cisuz.jar**
- **db2jcc_license.jar**

Place the files in the following folder: **<DDM Probe root directory>\DiscoveryProbe\root\lib\collectors\probeManager\discoveryResources\db\db2**.

Restart the Probe.

Internet Information Services (IIS) Discovery

Note: DDM supports IIS versions 5 and 6.

This task describes how to discover IIS.

This task includes the following steps:

- "Network and Protocols" on page 231
- "Discovery Workflow" on page 231
- "Discovered CIs" on page 231

1 Network and Protocols

Set up the **NTCMD protocol** and verify that the target machine running IIS lies in the Probe range. For details, see "NTCMD Protocol" on page 86.

2 Discovery Workflow

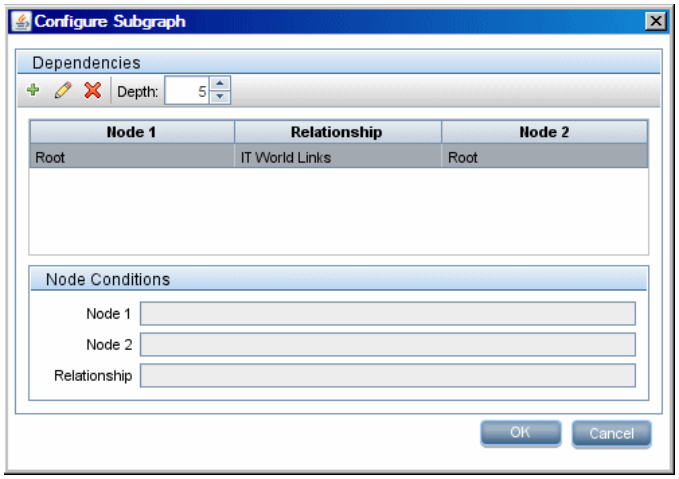
In the Run Discovery window, activate the jobs in the following order:

- Host Connection by NTCMD in the Network – Protocol Connections module
- TCP Ports in the Network – Advanced module
- Web Server Detection using TCP Ports in the Web Servers – Basic module
- IIS Applications by NTCMD in the Web Servers – IIS module

3 Discovered CIs

For details on the CIs that are discovered, see the Statistics table in the Details tab.

The dependency list for the IIS Web Site node is defined as follows (for details, see "Configure Subgraph Dialog Box" in *Model Management*):



Layer 2 Discovery

Note: Layer 2 discovery runs on Catalyst (Cisco Systems) network switches only.

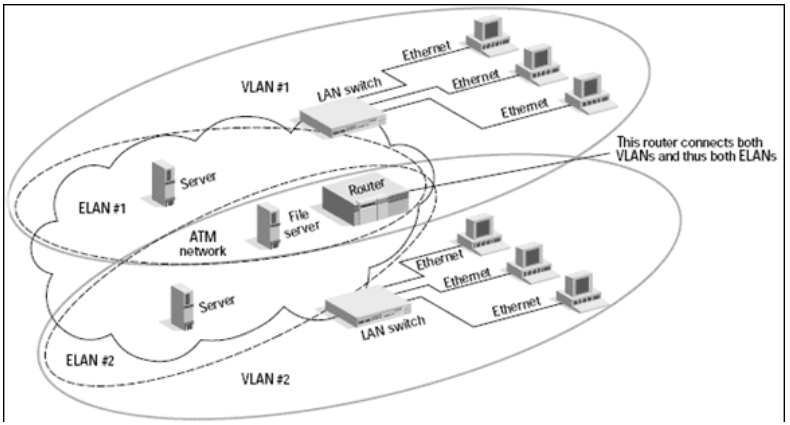
This task describes how to discover Layer 2 objects.

This task includes the following steps:

1 Overview

The Layer 2 package discovers the Layer 2 topology that includes the switches tree topology (the backbone links between the switches) and also the end user connections to the switch-ports (the Layer 2 links between a switch and a host). The Layer 2 package is based on the SNMP protocol.

The following graphic illustrates a router connecting overlapping VLANs/ELANs:



2 Prerequisites

Important:

- All network connection patterns should finish running before you activate the Layer 2 patterns.
- Make sure that there is SNMP access to all switches in the environment to be discovered, as that is a key requirement for fully discovering the Layer 2 topology.
- When defining the SNMP protocol credentials, have available the Port and Community authentication parameters.

- In the **Network – Layer 2** module, run the **Host Networking By SNMP** job. As a result of this run, DDM saves SNMP CIs to the CMDB.

Note: Layer 2 discovery is based on the connection patterns for the following reasons:

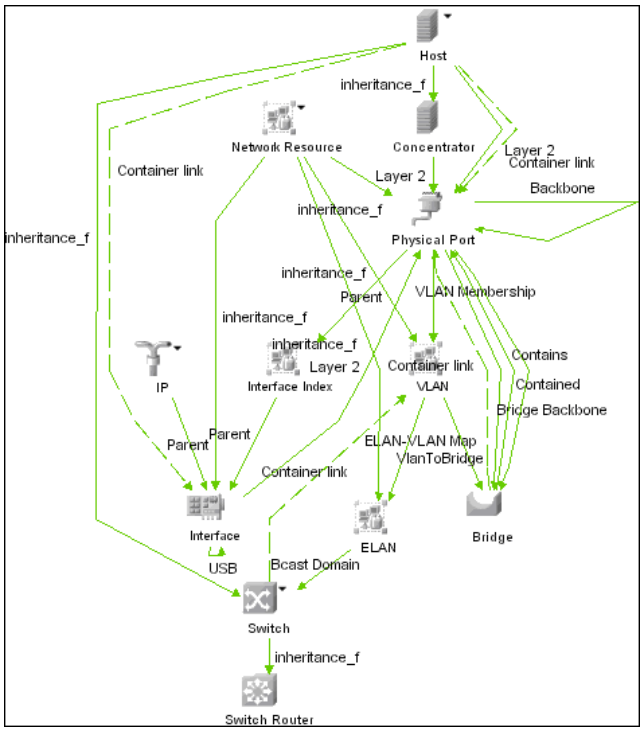
- The Layer 2 connectivity between the switch-port to the host is based on the host MAC address. These MAC addresses are discovered by the network connection jobs (Host Interfaces).
- The trigger of the Layer 2 job is dependent on the type of the discovered switch. The switch class and type is discovered by the SNMP connection job.

- **Host Networking by SNMP** job. You should run this job on all SNMP agents of the switches that were discovered in the environment. The to-be discovered Layer 2 link names are dependent on this discovery. (Layer 2 link names are the replica of the relevant interface index name and description that the host base pattern discovers.)

3 Class Model

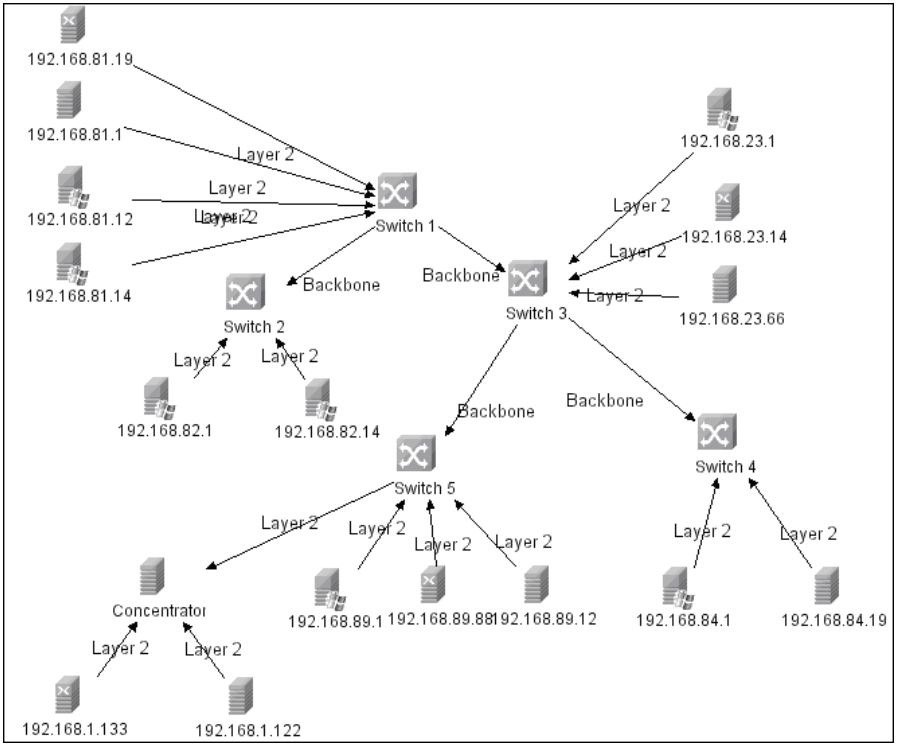
CIT Name	Description
VLAN	Virtual LAN. A logical, not physical, group of devices, defined by software. VLANs enable network administrators to resegment their networks without physically rearranging the devices or network connections.
ELAN	Emulated LAN. A set of clients and servers connected by virtual circuits over an ATM network.
Concentrator	Represents a hub or a switch to which there is no SNMP access.
Bridge	A physical unit inside the switch (like a NIC card) that holds the switch physical ports.
Physical Port	The switch physical port.
Contains	The link between the port and bridge.
VLAN Membership	The link between the port and VLAN.
VlanToBridge	The link between the VLAN and bridge.
ELAN-VLAN Map	The link between the ELAN and VLAN.
Bcast Domain	The link between the VLAN and switch.

4 Layer 2 Relationships



- A Layer 2 switch can be connected to its ports directly or through a VLAN.
- The Bridge CI represents the basic MAC address (Network Interface Card) on which the ports are located.
- Each switch-port can be connected to a host or interface object (the end user machines) by a Layer 2 link, or to a port-switch by a Backbone link.

5 Topology Map



6 Discovery Workflow

Important: The Layer 2 package includes four jobs. Each job discovers a part of the Layer 2 architecture. You should activate these jobs in the following order.

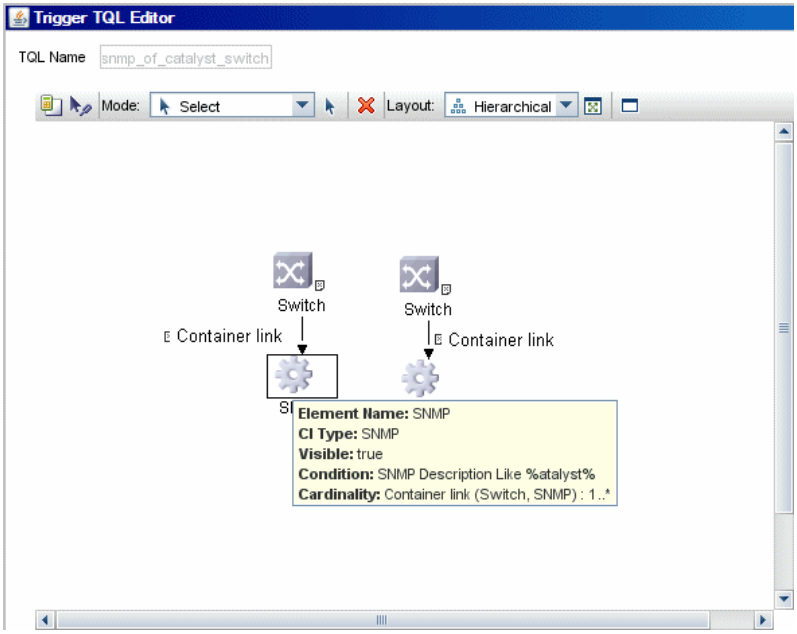
---

a Activate the **VLANS by SNMP** job.

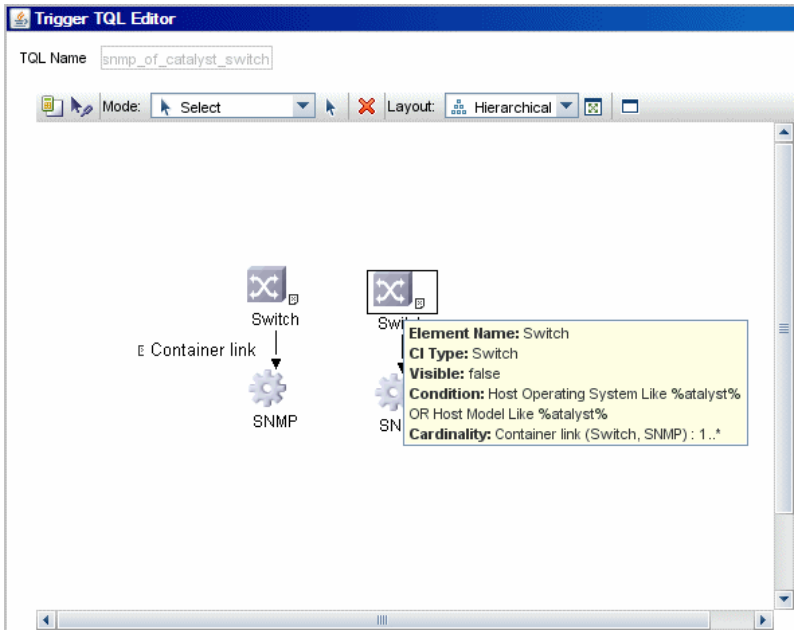
The trigger for this job is the **snmp_of_catalyst_switch** TQL. The Switch CIT is either:

- an SNMP object that holds a description containing the string **atalyst**
- an SNMP agent that is connected to a switch that holds an operating system or model attribute value containing the string **atalyst**

SNMP Description Like %atalyst%:



Host Operating System Like %atalyst% OR Host Model Like %atalyst%:



The SNMP_Net_Dis_Catalyst_Vlans.py script retrieves the VLAN, ELAN name, and VLAN number per ELAN tables.

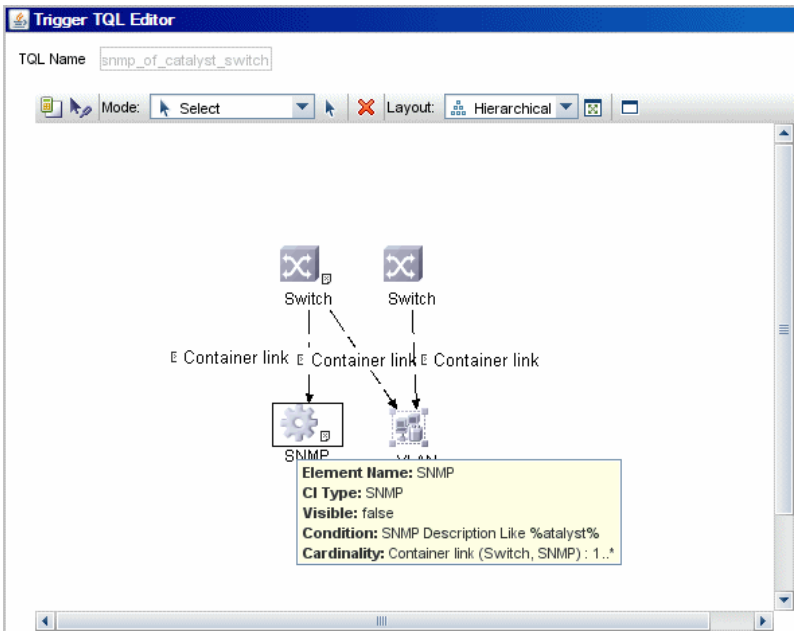
b Activate the **VLAN ports by SNMP** job.

The trigger for this job is the **catalyst_vlan** TQL. This is a VLAN object that has a connection to either:

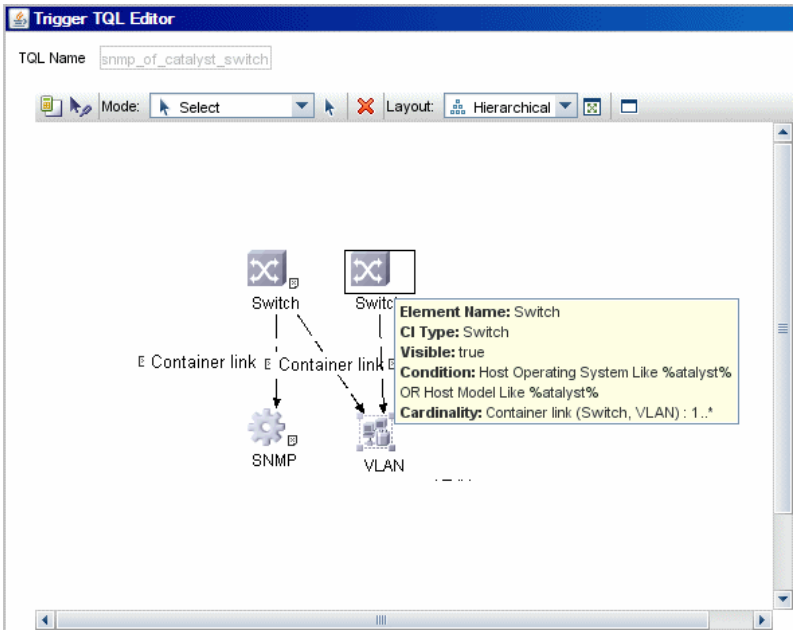
- a switch with an SNMP object that holds a description containing the string **atalyst**
- a switch that holds an operating system or model attribute value containing the string **atalyst**

The trigger is placed on the VLAN object instead of on the SNMP itself because the VLAN object must be authenticated with a special community string (and not with the regular community string that was discovered on the SNMP object on the discovered switch). This community string should hold the value <COMMUNITY>@<VLAN NUMBER>. For example, if the community string is **public** and the discovered VLAN number is **16**, the community string is **public@16**. For details on the SNMP protocol parameters, see "SNMP Protocol" on page 88.

SNMP Description Like %atalyst%:



Host Operating System Like %atalyst% OR Host Model Like %atalyst%:



The SNMP_Net_Dis_VMS_catalyst.py script retrieves the Base MAC table and Port number If Index table.

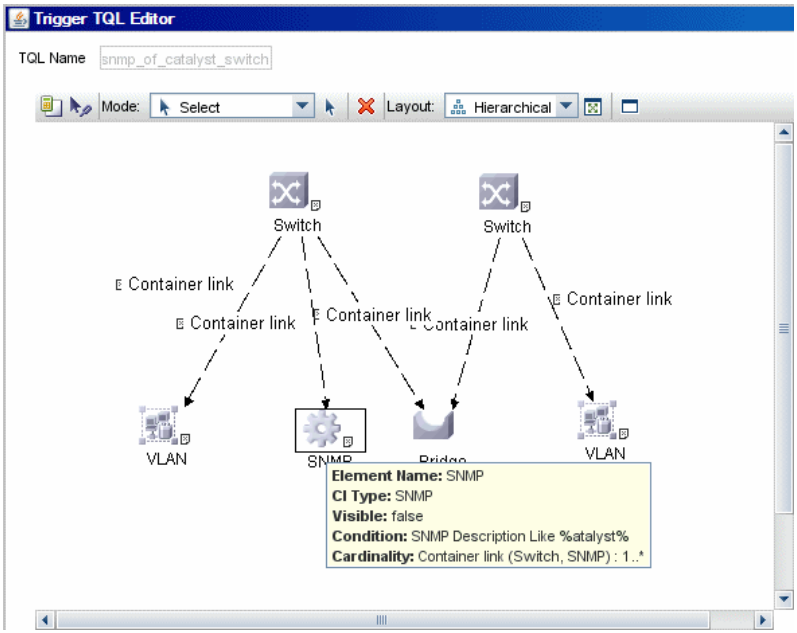
c Activate the **Layer2 Topology Bridge based by SNMP** job.

The trigger for this job is the **catalyst_bridge_no_vlan** TQL. This is a Bridge object that has a connection to:

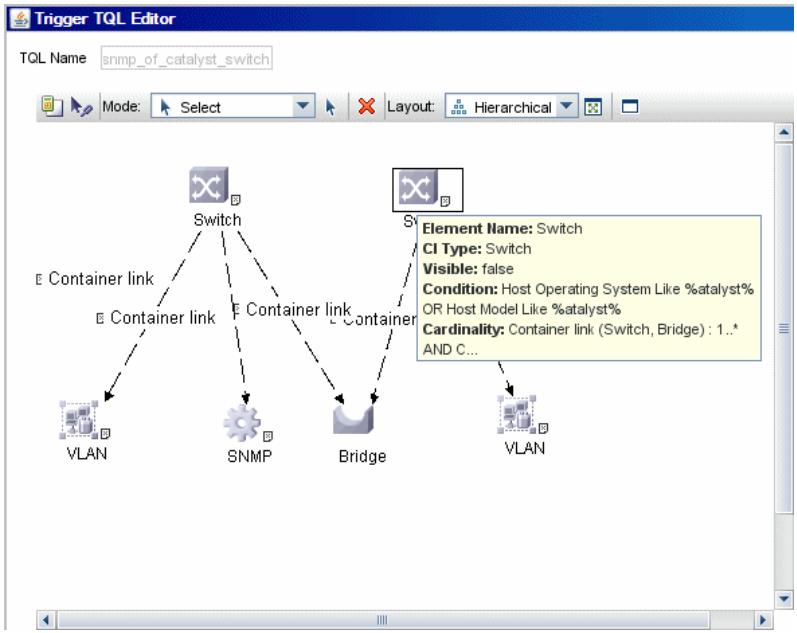
- a switch with an SNMP object that holds a description containing the string **atalyst**
- a switch that holds an operating system or model attribute value containing the string **atalyst**.
- the **NOT VLAN Bridge MAC is null** attribute

In both the first two cases, the switch should have zero connections to a VLAN.

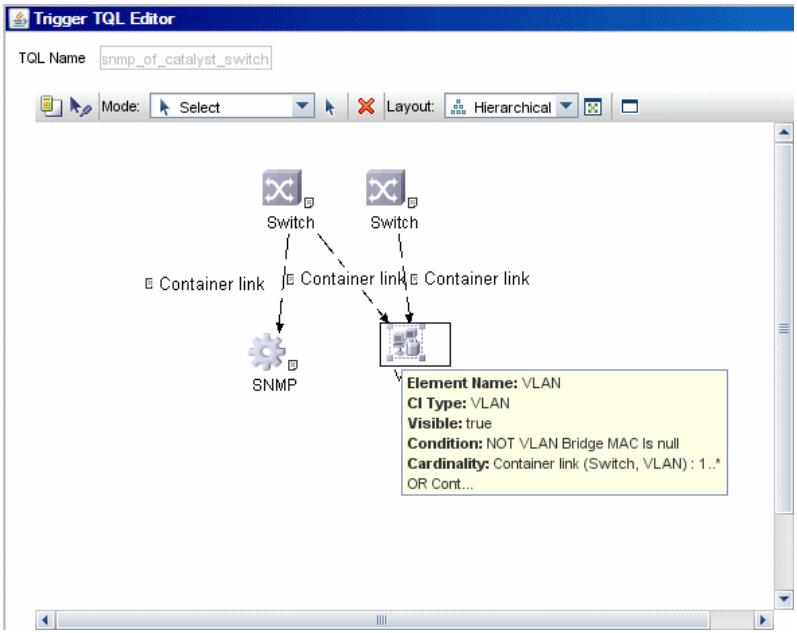
SNMP Description Like %atalyst%:



Host Operating System Like %atalyst% OR Host Model Like %atalyst%:



NOT VLAN Bridge MAC is null:



Both this job (**Layer2 Topology Bridge based by SNMP**) and the following job (**Layer2 Topology VLAN based by SNMP**) use the bridgePortDisc.py script. The difference between the jobs in this script is the way they retrieve the community string:

- **Layer2 Topology Bridge based by SNMP** uses the regular SNMP community authentication. The job is triggered on the Bridge only when the discovered switch has no VLANS.
- **Layer2 Topology VLAN based by SNMP** is triggered on each one of the VLANs discovered on the switch. This job uses the relevant special community authentication, as explained in step b, based on the triggered VLAN number.

Note:

- When the VLANs by SNMP job runs, it discovers Layer 2 topology that is relevant to the discovered VLAN only.
- Bridge Layer 2 discovery. If a machine has no VLANs, discovery is triggered on the bridge of the switch. DDM retrieves the Layer 2 topology of all the switches.
- If we dispatch the Bridge Layer 2 job on the bridge of a switch that holds VLANs only, the default VLAN Layer 2 topology is discovered.

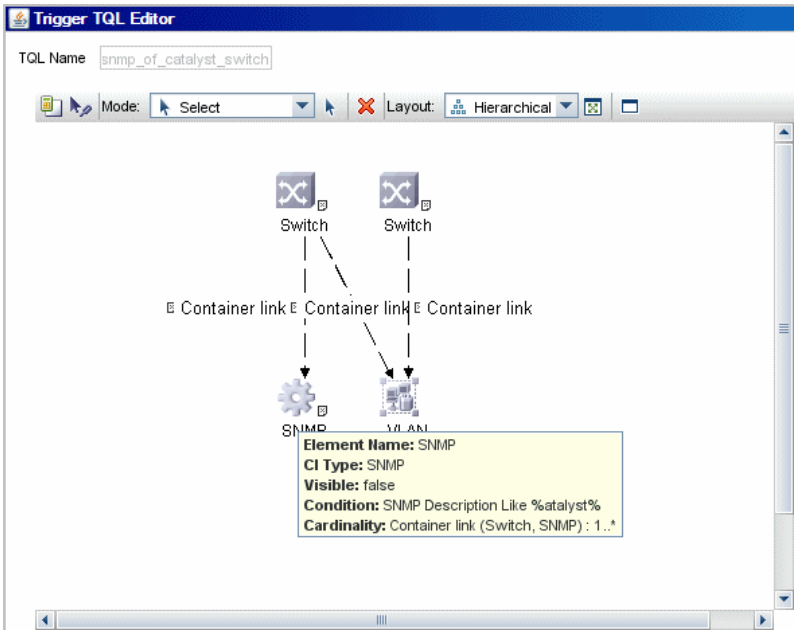
---

d Activate the **Layer2 Topology VLAN based by SNMP** job.

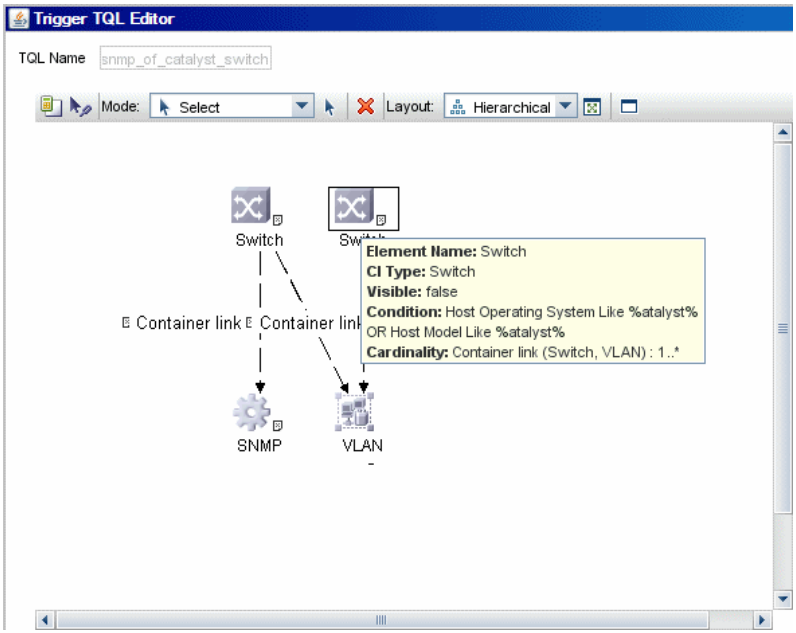
The trigger for this job is the **catalyst_vlan_with_bridge** TQL. This is a VLAN object with a value in its bridge_mac attribute. It should also have a connection to either:

- a switch with an SNMP object that holds a description containing the string **atalyst**
- a switch that holds an operating system or model attribute value containing the string **atalyst**

SNMP Description Like %atalyst%:



Host Operating System Like %atalyst% OR Host Model Like %atalyst%:



For details on the bridgePortDisc.py script, see step c.

The Backbone and Layer 2 links are created by the enrichments of the Layer 2 package, based on the data that was discovered by these jobs. After these jobs have run, job statistics do not show any Layer 2 or Backbone links as parts of the results.

7 Troubleshooting and Limitations

- If the results of the discovery return empty, verify that you have access to the discovered SNMP agent (or to the SNMP agent using the special community authentication) and that all the requested MIB tables are responding to SNMP requests from the Probe machine. For details on the MIB tables, refer to the appropriate script.
- In cases where the reported bridge MAC address is 000000000000, "", or null, the pattern does not report results.

- If the retrieved basic bridge MAC (retrieved from the 1.3.6.1.2.1.17.1.1 table) is not the same as the given bridgeId in the destination data, the pattern returns zero results.
  In the case of SNMP_Dis_L2_Bridge, bridgeId is set by bridge_basemacaddr.
  In the case of SNMP_Dis_L2_VLAN, bridgeId is set by vlan_bridgemac.

Microsoft Cluster Server Discovery

This task describes how to discover Microsoft Cluster servers.

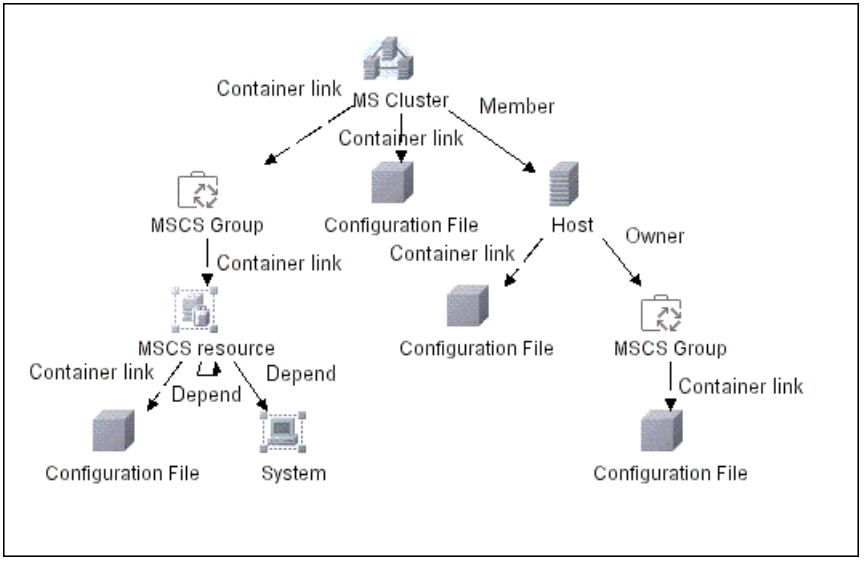
This task includes the following steps:

1 Overview

The MS Cluster discovery process enables you to discover the topology of a Microsoft Cluster Server on the network.

2 Topology Map

The following depicts the topology of the Microsoft Cluster Server view:



This Microsoft Cluster Server view shows the clusters discovered in the system. The cluster contains Microsoft Cluster groups. Each of the groups contains Microsoft Cluster resources.

3 Network and Protocols

- **WMI Protocol.** For details, see "WMI Protocol" on page 94.
- **NTCMD Protocol.** For details, see "NTCMD Protocol" on page 86.

4 Discovery Workflow

a In the Run Discovery window, activate the modules in the following order:

- Network - Protocol Connections (Host Connection by WMI, then Host Connection by NTCMD)
- Host Resources - WMI (Services by WMI)
- MS Cluster

b For details on the CIs that are discovered, see the Statistics table in the Details tab.

Microsoft SQL Server Discovery

This task describes how to discover the Microsoft SQL Server application.

This task includes the following steps:

1 Overview

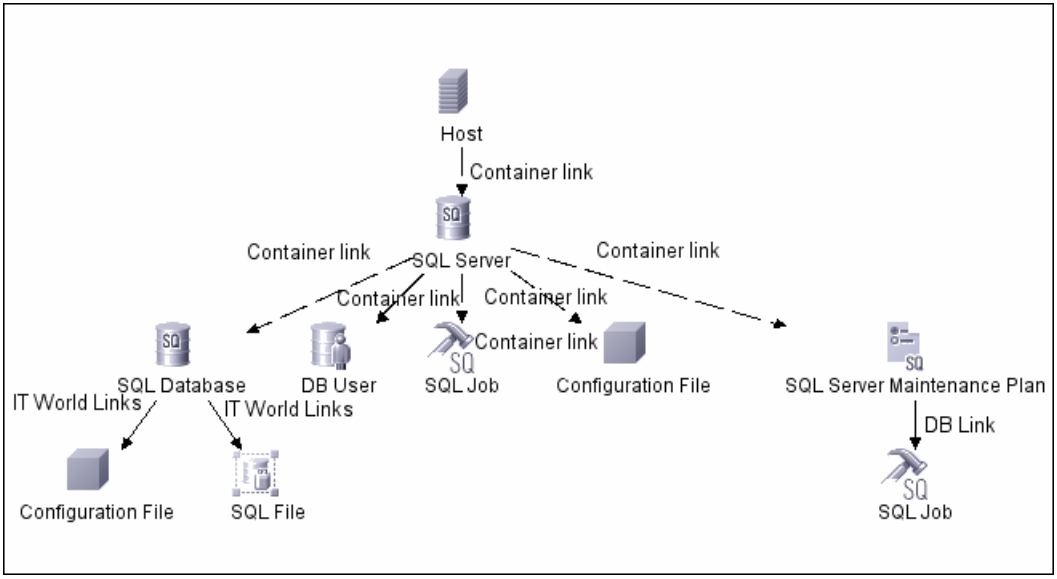
You can discover Microsoft SQL Server databases and their components on your network.

2 Supported Versions

Microsoft SQL Server 2000, 2005.

3 Topology Map

The following image depicts the topology of the Microsoft SQL Server view:



This view shows the hosts on which Microsoft SQL Server is installed. Microsoft SQL Server contains the databases, users, SQL jobs, and configuration files of this database, and maintenance plans.

4 Network and Protocols

Microsoft SQL Server uses the **SQL protocol**.

5 Discovery Workflow

a Verify the user name, password, and port used by Microsoft SQL Server.

b Set up the **SQL protocol**. For details, see "SQL Protocol" on page 90.

c In the Run Discovery window, activate the jobs in the **Database – MS-SQL** module in the following order:

- TCP Ports
- MSSQL Connection by SQL
- MSSQL Topology by SQL

d For details on the CIs that are discovered, see the Statistics table in the Details tab. For details, see "Statistics Results Pane" on page 133.

Network Discovery

You activate the jobs in the network modules to discover information about host components, for example, which hosts have open TCP/UDP ports, the ARP table of a router that is using the SNMP protocol, and so on. For details on the Infrastructure wizard to discover the network, see "Infrastructure Wizard" on page 142.

Network – TCP Discovery

This task describes how to discover TCP data.

This task includes the following steps:

1 Overview

The DDM Probe includes a built-in MySQL database so there is no need to install a separate MySQL instance for either the NetFlow or TCP discovery. Instead, data is saved to a dedicated scheme (called netflow for historical reasons).

TCP discovery consists of the following separate parts:

- Gathering the connectivity information (either via running netstat commands or by listening to netflow packets). This part is achieved by Collect Network Data by Shell or SNMP and Collect Network Data by NetFlow.
- Querying the gathered network data through dedicated DDM jobs. For details, see "TCP Query Jobs" on page 254.

2 Job Order and Scheduling

By default, all queries are scheduled to be run on a relatively frequent basis (every hour). The queries themselves are not re-run unless the data set has changed since the last run, in order not to waste CPU cycles on the Probe (note known issue #2 below which results from this requirement, not taking into account job parameter changes).

Although you can activate the Collect Network Data jobs along with the relevant queries, you would probably not see any results until at least one hour has passed to the next scheduled invocation of the query (because by the time the first set of queries are run, no data has been gathered). So a best practice (especially for POC) is to make sure data gathering is complete and only then launch the query and see the result it populates.

3 TCP Query Jobs

- **Servers by Network Data.** This is the main job. It enables the discovery of specific service names (the services parameters). The service name to port numbers are still configurable through the portNumberToPortName.xml file.

  This job, together with the appropriate connection information gathering discovery job, serves as the replacement for the DIS_TCP pattern.

  The links discovered by this job are clientserver links (between the client IP and the server port to which it connects) and dependency links between the related hosts.

  There are flags available for:

  - onlyHostDepend links. Enables discovery of dependency links only (without the clientserver links).
  - includeOutscopeClients/Servers. Prevents discovery of services or clients on machines which are out of a probe's network scope.
  - updateUtilizationInfo. Relevant only for NetFlow and can be used to prevent reporting the packets and octets count information on the clientserver links.

- **IP Traffic by Network Data.** This job discovers traffic links between all communicating IPs. The traffic links are populated between any two IPs that are seen to communicate. On these links is an attribute with the value of the top ports (the most important TCP/UDP ports) that could be found between those two IPs/hosts. The top ports are calculated according to the number of clients and the size of the network traffic between them.

  You can configure the maximum number of recognized, interesting ports through the maxPorts parameter.

- **Server Ports by Network Data.** This discovery job only discovers open server ports according to the list of specified services. This can be useful if you do not want to discover the TCP connections themselves but do want to know which ports are open, without performing any TCP port scanning (which may be dangerous in some organizations).

  This discovery job is not relevant for NetFlow data as there is no LISTEN flag in this case.

- **Potential servers by Network Data.** This job is implemented in Jython, that is, you can use it as a reference to write advanced queries against the probe's gathered TCP connections data set. It can be used in situations where you want to find clientserver links but without defining the port numbers in advance. The server port is defined according to the criteria passed as job parameters: clientCount (the minimum number of clients for the service), minPackets/minOctets (minimum packets and octets – relevant only for netflow).

Important: This job does not come out-of-the-box with a Trigger TQL because it is not intended to be used on many triggers. Rather, you should activate the job manually against specific IP instances, to find unknown server ports. It is preferable to add the triggers afterwards to the portNumberToPortName.xml file and continue discovery through the **Servers by Network Data**.

- **Collect Network Data by Shell or SNMP.** Discovers the TCP connections of the discovered machines using Telnet, SSH, SNMP, or NTCmd.
- **Process To Process by Network Data.** This pattern maps processes communications on hosts with Windows 2003 and Windows XP.

Process to Process (P2P) Discovery

This task describes how to discover processes.

This task includes the following steps:

1 Overview

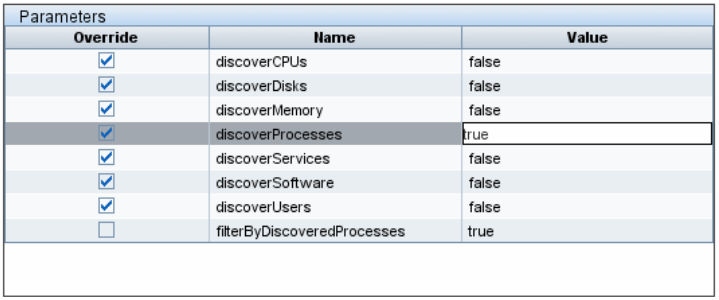
The Process to Process by Network Data job discovers processes running on Windows 2003 and Windows XP machines, on Linux machines, and on Hewlett Packard UniX (HP-UX) machines. For the HP-UX machine, you should install lsof software, which can be downloaded from the Internet, for example, from http://www.netadmintools.com/html/lsof.man.html.

2 Prerequisites

a Run **Host Connection by Shell/SNMP/WMI**. (NTCmd, SSH, Telnet, and WMI CIs are discovered.)

b Run **Collect Network Data by Shell or SNMP** (no CIs are discovered).

c Run **Host Resources By Shell/SNMP/WMI** (set discoveryProcesses to **true**, since that is the only required parameter, and all other parameters to **false**).


Parameters

Override	Name	Value
☑	discoverCPUs	false
☑	discoverDisks	false
☑	discoverMemory	false
☑	discoverProcesses	true
☑	discoverServices	false
☑	discoverSoftware	false
☑	discoverUsers	false
☐	filterByDiscoveredProcesses	true

3 Network and Protocols

To discover P2P resources, define the following protocols:

- **NTCmd.** For details, see "NTCMD Protocol" on page 86.
- **SNMP.** For details, see "SNMP Protocol" on page 88.
- **SSH.** For details, see "SSH Protocol" on page 90.
- **Telnet.** For details, see "Telnet Protocol" on page 91.
- **WMI**. For details, see "WMI Protocol" on page 94.

Note: None of these protocols is mandatory, but WMI alone does not retrieve network data.

---

4 Affected Jobs and Job Dependencies

- Host Connection by Shell/SNMP/Wmi
- Collect Network Data by Shell or Snmp
- Host Resources by Shell/SNMP/Wmi
- Process to Process

5 Discovery Workflow

In the Run Discovery window, activate the jobs in the following order:

- Host Resources
- SNMP
- Host Connections

6 Discovery Pattern Parameters

You can filter the list of processes to run only those processes that retrieve data that is of interest to you. The network connectivity of these processes is omitted.

To view the list of processes that are filtered by default, select the **Process to Process by Network Data** job in the Run Discovery window, Advanced Mode, choose **Properties**, then locate the **Parameters** pane. The values are displayed in the **Value** box and are comma-separated.

To add processes to this list, access the Manage Discovery Resources window by clicking the **Edit** button in the Discovery Pattern pane. The Pattern Signature pane is displayed. Locate the Discovery Pattern Parameters pane. For details on adding, deleting, or editing parameters, see "Discovery Pattern Parameters Pane" on page 196.

The **Process to Process by Network Data** job uses the ProcessCommunication parameter, which includes the following values:

- **filterProcessesByName.** Limits the processes to be run according to the names in the **Value** box.

  An asterisk signifies that all ports are to be found. You can also use numbers, comma-separated names, and ranges, for example, 1020-1030,5020,Oracle,HTTP,23.

- **services.** The services to be discovered.

7 Discovered CIs

- **Client-Server.** DDM determines which machine is the server and which the client:
  - If there is only 1 client, DDM identifies it as such, by checking the ports and the **portNumberToPortName.xml** file (**Manage Discovery Resources** > **Discovery Resources** > **Network** > **Configuration Files**).
  - If the port number equals, or is less than, **1024**, DDM identifies it as a server.
  - If neither of these cases is true, DDM retrieves the **Talk** link of the **use** CIT.
- **Talk.** This link is created between two processes only if DDM does not recognise the Client-Server link between the processes. The Talk link reports bidirectionally.

SAP Discovery

This task describes how to discover SAP.

This task includes the following steps:

1 Overview

The SAP discovery process enables you to discover application components, SAP transactions and transports, and SAP topology.

Note: To discover more than one SAP system, it is recommended to create a SAP Protocol with different users and passwords for each SAP system.

---

2 Prerequisites

Install Java connectors:

- Download the SAP JCo package from the Tools & Services window of SAP JCo in SAP Service Marketplace:

  https://websmp101.sap-ag.de/~form/sapnet?_SHORTKEY=01100035870000463649

- Extract **sapjco-ntintel-2.0.8.zip** to a temporary directory (for example: C:\temp) on the Business Availability Center machine.
- Create a **sap** directory (in lowercase) in the **<DDM Probe root directory>\Discovery Probe\root\ext** directory on the machine where the Probe is installed.
- Copy **sapjco.jar** from the temporary directory to the **<DDM Probe root directory>\DiscoveryProbe\root\ext\sap** directory on the machine where the Probe is installed.
- Copy **sapjcorfc.dll** from the temporary directory to the **<DDM Probe root directory>\DiscoveryProbe\root\ext\sap** directory on the machine where the Probe is installed.
- Copy **librfc32.dll** from the temporary directory to the **%winnt%\system32** directory.
- Verify that the **MSVCR71.dll** and **MSVCP71.dll** files are located in the **%winnt%\system32** directory.

3 Network and Protocols

- **SNMP Protocol.** For details, see "SNMP Protocol" on page 88.
- **WMI Protocol.** For details, see "WMI Protocol" on page 94.
- **NTCMD Protocol.** For details, see "NTCMD Protocol" on page 86.
- **SAP Protocol.** For details, see "SAP Protocol" on page 87.

4 Discovery Workflow

a To trigger the discovery of SAP System networking features, add a Network CI to the CMDB. For details, see "New CI Wizard" in *Model Management*.

b In the Run Discovery window, activate the modules in the following order:

- Network – Basic (Class C IPs by ICMP)
- Network – Protocol Connections (Host Connection by SNMP, Host Connection by NTCMD, and Host Connection by WMI)
- Host Resources – WMI (Processes by WMI)

  If the SAP system has an ITS configuration, to discover the ITS entities of the SAP system, run this pattern as a prerequisite to the SAP discovery that discovers ITS entities.
- Network – Advanced (TCP Ports)
- Web Servers – Basic (Web server detection using TCP Ports)

  If the SAP system has an ITS configuration, to discover the ITS entities of the SAP system, run this pattern as a prerequisite to the SAP discovery that discovers ITS entities.
- Application – SAP (R/3)

  SAP Site. Discovers infrastructure entities in the SAP System: hosts, R/3 Application servers, Work Processes, databases, SAP clients, configuration files, software components (discovered as configuration files), and support packages (discovered as configuration files).

  SAP ITS by NTCMD. Discovers Internet Transaction Server (ITS) entities (Application Gateway and Web Gateway).

  SAP Solution Manager. Discovers SAP Solution Manager components. **Note:** Before you run this pattern to discover application components, SAP transactions, and SAP transports, you must set the discovery mode. For details, see step d.

c For details on the CIs that are discovered, see the Statistics table in the Details tab.

d Set the discovery mode. According to the type of discovery you are running, you set the pattern parameters as follows:

- Access the SAP pattern: **Manage Discovery Resources** > **SAP_Discovery package** > **SAP_Dis_Applications**.
- Select the Pattern Signature tab and locate the Discovery Pattern Parameters pane.

➤ Set one of the following parameters, and click **OK** to save the changes.

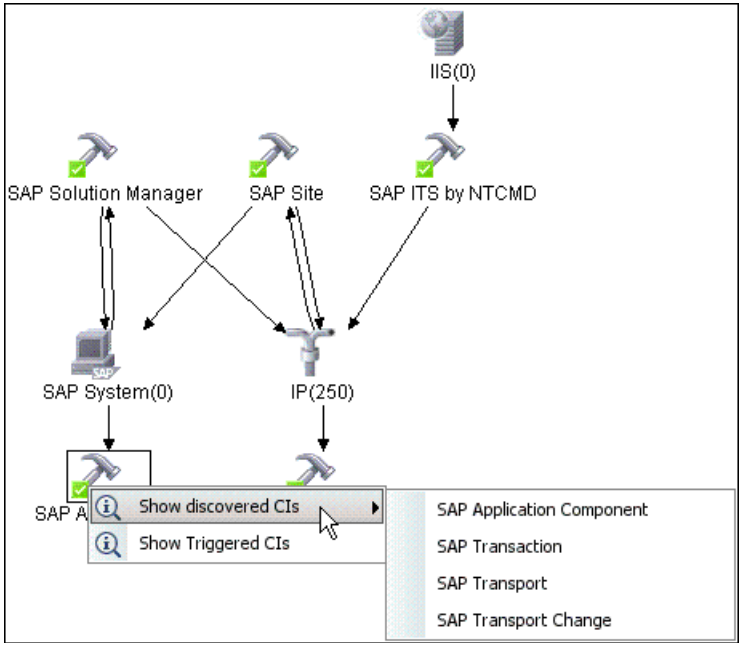
To discover all SAP transactions: Set **get_tx_all** to **false**.

To discover active SAP transactions: Set **get_tx_active** to **true**.

To discover SAP transactions that have been changed by discovered transports: Set **get_tx_change** to **true**.

e Verify that DDM discovered the appropriate components. Access the SAP_Topology view in View Manager and verify that the map displays all components.

f To view the CIs discovered by the SAP discovery, select **Run Discovery > Application SAP (R/3)**, select a job and access the Dependency Map tab. Right click the selected job and choose **Show discovered CIs**.



g To view the SAP CITs, access the CI Type Manager and select **IT Universe > System > Application Resource > SAP Resource**. Hold the cursor over a CIT to view a description.

For details on KPIs created together with the CIs, see the SAP and SAP Alerts KPIs in "KPI Repository" in *CI Attribute Customization*.

h SAP Solution Manager discovery enables you to discover the business process hierarchy. To run this discovery, in the Run Discovery window, activate the SAP Solution Manager job.

Siebel Discovery

This task describes how to discover Siebel topology.

This task includes the following steps:

1 Overview

Using the Siebel patterns, you can run an automatic Siebel discovery to create the Siebel world, together with its components, inside Business Availability Center.

During discovery:

- All Siebel-related IT entities that reside in the organization are discovered and configuration items (CIs) are written to the CMDB.
- When a new Siebel Application CI is created, two KPIs are created under it: Transactions and Locations.
- The relationships between the elements are created and saved in the CMDB.
- The newly generated CIs are displayed when the Siebel Enterprises view is selected in View Explorer under the Siebel Enterprises root CI.

- Four logical containers—Applications, Business Processes, Hosts and Locations—are created under the Siebel Enterprises root CI.
- After discovery has run, you must manually update some of the discovered CI's properties. For details, see "Configure HP Business Availability Center for Siebel Applications" in *Solutions and Integrations*.

Note: Verify that all Siebel server IP addresses are included in the range. If you do not want to cover all servers with one IP range, you can split the range into several ranges.

---

2 Network and Protocols

Set up the following protocols for the Windows platform and continue to Discovery Workflow:

- **WMI Protocol.** For details, see "WMI Protocol" on page 94.
- **NTCMD Protocol.** For details, see "NTCMD Protocol" on page 86.
- **Siebel Gateway Protocol.** For details, see "Siebel Gateway Protocol" on page 88.

Set up the following protocols for the UNIX platform and continue to Discovery Workflow:

- **SSH Protocol.** For details, see "SSH Protocol" on page 90.
- **Telnet Protocol.** For details, see "Telnet Protocol" on page 91.
- **Siebel Gateway Protocol.** For details, see "Siebel Gateway Protocol" on page 88.

3 Discovery Workflow

a For Siebel discovery to run, you must copy the driver tool to the Probe server. For details, see "Copy the driver Tool to the Probe Server" on page 266.

b For the SiteScope Siebel monitors to work correctly, copy the driver tool and the SARM Analyzer tool to the SiteScope server. On SiteScope, the driver tool is launched by the Siebel Application Server monitor to retrieve the metrics. For details, see "Copy the driver Tool and the SARM Analyzer Tool to the SiteScope Server" on page 267.

c To trigger the discovery of Siebel networking features, add a Network CI to the CMDB. For details, see "New CI Wizard" in *Model Management*.

d In the Run Discovery window, activate the modules in the following order:

- Network – Basic (Class C IPs by ICMP)
- Network – Protocol Connections (Host Connection by NTCMD, Host Connection by WMI, Host Connection by TTY)

e To discover the Web tier, activate the following modules:

- Network - Advanced (TCP Ports)
- Application - Siebel (Siebel Web Applications by NTCMD, Siebel Web Applications by TTY, Siebel DB by WMI and NTCMD)
- Web Server - Basic (WebServer Detection using TCP Ports)

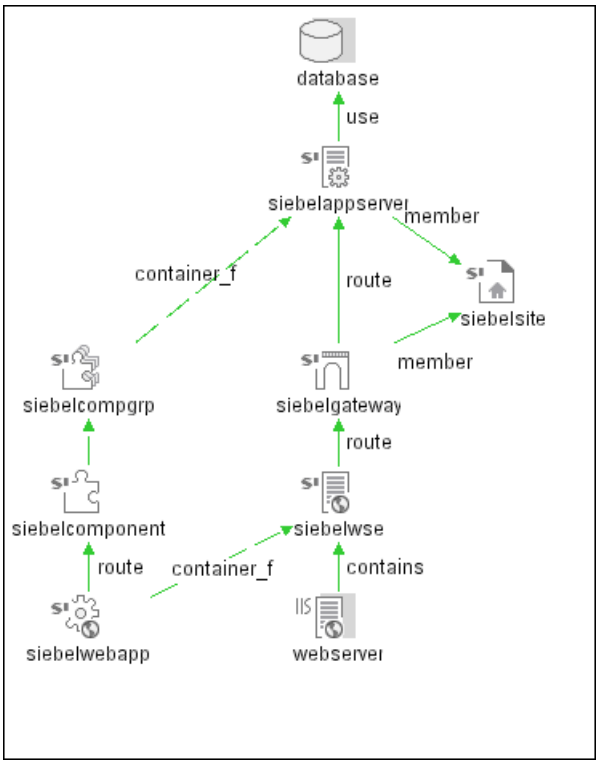
f To discover Siebel, activate all the patterns in the Application – Siebel module.

Note: The following enrichment patterns automatically run in the background during discovery:
Siebel_Route_WebApp_To_Component. Builds the route between Siebel Web Application CIs and Siebel Component CIs.
Siebel_Web_To_Middle_Tier. Builds the route between the Web tier and the middle tier when the Siebel enterprise uses a Resonate server for load balancing.

---

g For details on the CIs that are discovered, see the Statistics table in the Details tab.

4 Topology Map



5 Copy the driver Tool to the Probe Server

The driver tool is used to extract data about the enterprise structure from Siebel.

Note: If you are working with different versions of Siebel in your organization, make sure you use a driver tool with a version that is appropriate for the Siebel server.

To copy the driver tool to the Probe server:

a Copy the driver Command Line Interface (CLI) tool from the Siebel server to any folder on the Probe server.

b It is recommended to run the Siebel connection test to validate the driver installation. To run the connection test, open the command line on the Probe server and change directory to the location of the driver.exe file.

c Run from the command line:

```
>driver /e [site_name] /g [gateway_host] /u [username] /p [password]
```

If the connection is established successfully, the Command Prompt window displays the driver prompt and a status message about the number of connected servers.

6 Copy the driver Tool and the SARM Analyzer Tool to the SiteScope Server

---

Note:

- The SARM Analyzer is used for analyzing SARM data, so that it can be displayed in the Business Availability Center for Siebel SARM – User Trace Breakdown tab.
- If you are working with different versions of Siebel in your organization, make sure you use a driver and a SARM Analyzer with a version that is appropriate for the SiteScope server.
- It is recommended to run the Siebel connection test to validate the driver installation.
- SiteScope should run under a domain user name which has permissions to run server manager and SARM Analyzer and also has read access to the log folders on the Siebel servers (Web servers and application servers).
- All Siebel servers (Windows and UNIX) must be defined as remote servers on the SiteScope server. For details, refer to the *SiteScope Reference Guide*.

To copy the driver tool to the SiteScope server:

a Copy the driver Command Line Interface (CLI) tool from the Siebel server to any folder on the SiteScope server.

b To run a connection test, open the command line on the SiteScope server and change directory to the location of the driver.exe file.

c Run from the command line:

```
>driver /e [site_name] /g [gateway_host] /u [username] /p [password]
```

For the connection to work properly, verify that the user and password have the correct permissions for a remote connection.

If the connection is established successfully, you should see the driver prompt and the status message about the number of connected servers.

To copy the SARM Analyzer tool to the SiteScope server:

a Copy the SARM Analyzer tool from the Siebel server to a folder on the SiteScope server.

b If your site includes a large number of Web servers, it is preferable to use multiple SiteScopes to distribute the work done by the SARM Analyzer tool between those SiteScopes. In such a case, copy SARM Analyzer to each SiteScope.

7 Troubleshooting and Limitations

The Siebel DB by TTY job cannot discover virtual Siebel application servers (with a different name and configuration to the actual Siebel application server) running on UNIX machines.

Universal Description Discovery and Integration (UDDI) Discovery

This task describes how to discover UDDI processes.

This task includes the following steps:

1 Overview

The UDDI discovery process enables you to discover Web services from a UDDI registry.

DDM queries the UDDI registry for its Web services, including non-SOAP services, or for a specific publisher service (if defined in the UDDI Registry protocol). The Web services found in the UDDI registry are represented by a **webservice** CI in the CMDB and the registry is created as a **uddiregistry** CI.

Note: Business Availability Center supports UDDI versions 2 and 3.

---

2 Network and Protocols

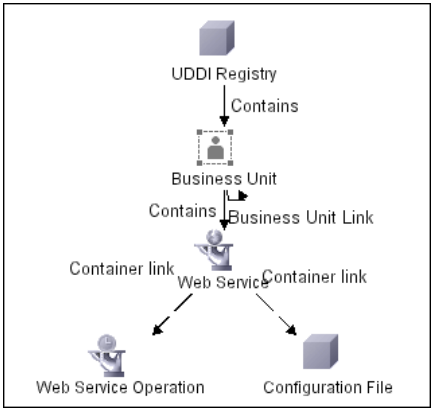
a Set up the **UDDI protocol**. For details, see "UDDI Registry Protocol" on page 92.

b In the Run Discovery window, activate the **Application – Webservices** module.

c For details on the CIs that are discovered, see the Statistics table in the Details tab.

3 Topology Map

The following depicts the topology of the **SOA_UDDI_View**:



4 Discovery Workflow

(Optional) To enter the name of the service publisher whose services you want to publish:

a Access the Manage Discovery Resources window.

b In the Discovery Resources pane, locate the Webservices package and select the **UDDI_Registry** pattern.

c In the Pattern Signature tab, in the Discovery Pattern Parameters pane, select the **organization** parameter and click the **Edit** button.

d In the Parameter Editor:

- In the **Value** box, enter the name of the service publisher.
- In the **Description** box, enter the required description of the organization.

e Save the changes.

Veritas Cluster Server Discovery

This task describes how to discover Veritas Cluster servers.

This task includes the following steps:

1 Overview

The Veritas Cluster discovery process enables you to discover Veritas Cluster Servers (VCS), and their member machines (also referred to as nodes), that activate the discovered resources provided by the cluster.

2 Network and Protocols

a Set up the **SSH protocol**. For details, see "SSH Protocol" on page 90.

b Set up the **Telnet protocol**. For details, see "Telnet Protocol" on page 91.

3 Discovery Workflow

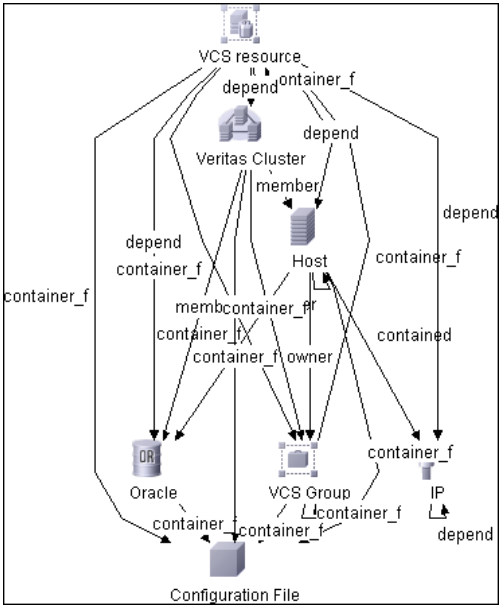
In the Run Discovery window, activate the following modules:

- Network - Protocol Connections
- Host Resources - SSH/Telnet
- Veritas Cluster

For details on the CIs that are discovered, see the Statistics Results table in the Details tab. For details, see "Statistics Results Pane" on page 133.

4 Topology Map

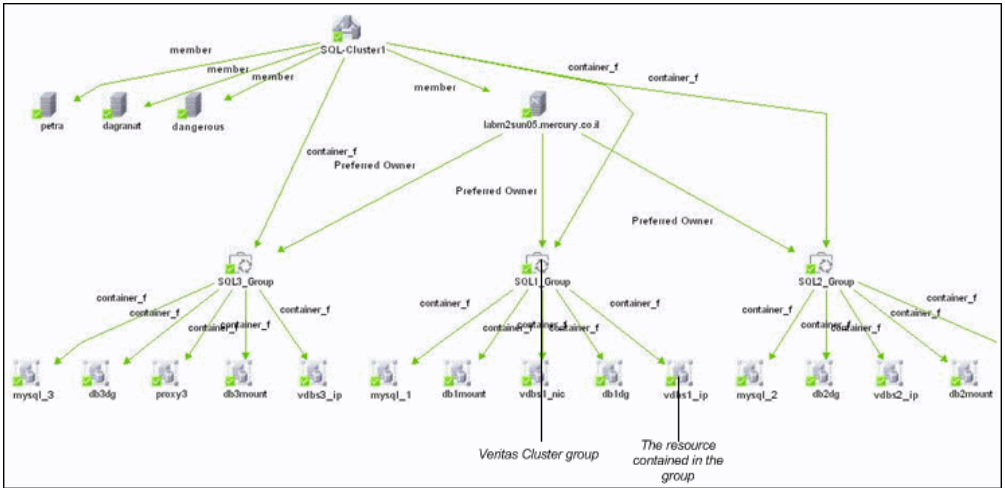
The following view depicts the Veritas Cluster Server topology.



This view shows the top layer of the Veritas Cluster topology. It displays the discovered Veritas Cluster and the nodes that are members of that cluster. Each member node is linked by a **member** relationship to the Veritas Cluster.

Veritas Clusters contain multiple nodes. Each node is responsible for running certain services and applications. The nodes are used as backups for one another. When a system components fails, another node takes over to provide the necessary service.

Double-click the required node to drill down to the CIs folded underneath.



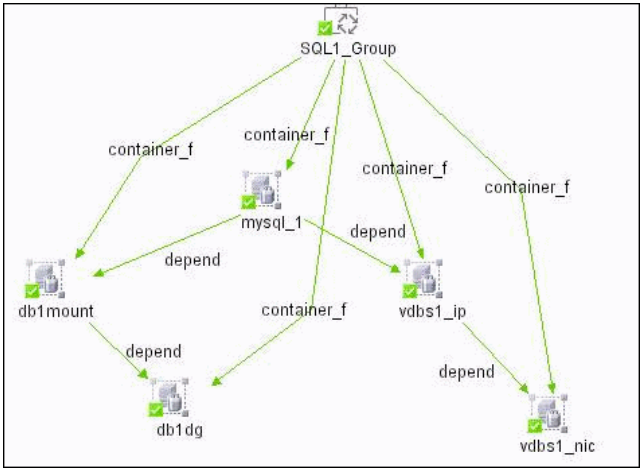
This view displays the Veritas Cluster groups and the resources contained in each group.

A Veritas Cluster group is a collection of dependent or related resources that is managed as a single unit. Each Veritas Cluster group is linked to a designated node, which is responsible for activating the resources contained in the group. The node is linked to the group by a **Preferred Owner** relationship. If a failure occurs in the designated node, the responsibility for activating the resources is switched over to a different node.

Certain resources in each group are dependent on one another. Resources that are dependent on one another are linked by a **depend** relationship.

The following figure shows these dependencies:

- **db1dg** is dependent on **db1mount**
- **db1mount** is dependent on **mysql_1**
- **vdbs1_nic** is dependent on **vdbs1_ip**
- **vdbs1_ip** is dependent on **mysql_1**



VMware Discovery

This task describes how to discover VMware.

This task includes the following steps:

- "Overview" on page 274
- "Network and Protocols" on page 275
- "Discovery Workflow" on page 275

1 Overview

The VMware discovery process enables you to discover virtual machines, processors, memory, storage, and network resources that are running on VMware version 2.5 ESX servers.

2 Network and Protocols

- Set up the **SSH protocol**. For details, see "SSH Protocol" on page 90.
- Set up the **Telnet protocol**. For details, see "Telnet Protocol" on page 91.

3 Discovery Workflow

a In the Run Discovery window, activate the following modules:

- Network - Basic
- Host Resources by Shell (Job)
- Network - VMWare

b For details on the CIs that are discovered, see the Statistics table in the Details tab.

The **VM Server** CIT inherits the base attributes from the system CIT. The data_name attribute contains the VM server type (for example, VMWare Server). The vmserver_version attribute contains version information (for example, ESX 2.5.2). The root_container attribute enables multiple VM servers with the same data_name attribute to remain unique.

The **Interface** CIT represents a virtual interface on condition that the isvirtual attribute is set to **true** (the default value is false). Since the virtual interface is linked to the physical interface on the VM server, the depend link joins the CIT to itself. The only difference between the virtual and physical interface CITs is that the virtual interface includes the isvirtual attribute. This CIT is part of the network resource CIT.

WebLogic Discovery

This task describes how to discover WebLogic.

This task includes the following steps:

1 Overview

DDM first finds WebLogic servers based on the JMX protocol, then discovers the WebLogic J2EE environment and components.

The WebLogic discovery process enables you to discover all the deployed Web services and operations deployed on a WebLogic server.

2 Supported Versions

WebLogic version 9 is supported.

3 Prerequisites

If you are using WebLogic version 7, perform the following procedure to discover this Weblogic version:

- Take the **webserviceclient.jar** and **weblogic.jar** files from the following location: **<BEA Installation root folder>\<WebLogic version number>\server\lib**.
- Place both jar files in the following location: **<DDM Probe root directory>\DiscoveryProbe\root\lib\collectors\probeManager\discoveryResources**. (This folder is created when the Probe connects to the WebLogic server. However, if you have not yet run DDM, you must manually create the folder.)
- Rename the jar files by adding a suffix that includes the WebLogic version number, as follows: **webserviceclient70.jar**, **weblogic70.jar**.

4 Network and Protocols

- Set up the **WebLogic protocol**. For details, see "WebLogic Protocol" on page 92.
- In the Run Discovery window, activate the job in the J2EE – WebLogic module.
- For details on the CIs that are discovered, see the Statistics Results table in the Details tab. For details, see "Statistics Results Pane" on page 133.

WebSphere Discovery

This task describes how to discover WebSphere.

This task includes the following steps:

1 Overview

DDM first finds WebSphere servers based on either SOAP or RMI authentication, then discovers the WebSphere J2EE environment and components.

WebSphere discovery discovers Web services that are deployed on an IBM WebSphere server. The discovered Web services are represented by the webservice CIT in the CMDB.

2 Supported Versions

Business Availability Center supports WebSphere versions 5 and 6.

3 Network and Protocols

Set up the **WebSphere protocol**. For details, see "WebSphere Protocol" on page 93.

4 Discovery Workflow

a In the Run Discovery window, activate the J2EE – WebSphere job.

b For details on the CIs that are discovered, see the Statistics Results table in the Details tab. For details, see "Statistics Results Pane" on page 133.

Web Server Discovery

The Web Servers modules discover the Apache Web server, Web servers that use a TCP port, and the Microsoft Internet Information Server (IIS).

10

Content Development and Pattern-Writing

This chapter describes the approaches, methodologies, and practices of developing new Discovery and Dependency Mapping (DDM) content (known as pattern-writing) for HP Business Availability Center.

This chapter includes:

Concepts

Tasks

Reference

Content Development and Pattern-Writing – Introduction

Prior to beginning actual planning for development of new content, it is important for you to understand the processes and interactions commonly associated with this development.

The following sections can help you understand what you need to know and do, to successfully manage and execute a discovery development project.

This chapter:

- Assumes a working knowledge of HP Business Availability Center and some basic familiarity with the elements of the DDM system. It is meant to assist you in the learning process and does not provide a complete guide.
- Covers the stages of planning, research, and implementation of new discovery content for HP Business Availability Center using DDM, along with guidelines and considerations that need to be taken into account.
- Provides information on the key APIs of the Discovery and Dependency Mapping Framework. For full documentation on the available APIs, see the *HP Discovery and Dependency Mapping API Reference*. (Other non-formal APIs exist but even though used on out of the box patterns, they may be subject to change.)

Associating Business Value with Discovery Development

The use case for developing new discovery content should be driven by a business case and plan to produce business value. That is, the goal of mapping system components to CIs and adding them to the CMDB is to provide business value.

The content may not always be used for application mapping, although this is a common intermediate step for many use cases. Regardless of the end usage of the content, your plan should answer these questions of this approach:

- Who is the consumer? How should the consumer act on the information provided by the CIs (and the relationships between them)? What is the business context in which the CIs and relationships are to be viewed? Is the consumer of these CIs a person or a product or both?
- Once the perfect combination of CIs and relationships exists in the CMDB, how do I plan on using them to produce business value?
- What should the perfect mapping look like?
  - What term would most meaningfully describe the relationships between each CI?
  - What types of CIs would be most important to include?
  - What is the end usage and end user of the map?
- What would be the perfect report layout?

Once the business justification is established, the next step is to embody the business value in a document. This means picturing the perfect map using a drawing tool and understanding the impact and dependencies between CIs, reports, how changes are tracked, what change is important, monitoring, compliance, and additional business value as required by the use cases.

This drawing (or model) is referred as the **blueprint**.

For example, if it is critical for the application to know when a certain configuration file has changed, the file should be mapped and linked to the appropriate CI (to which it relates) in the drawn map.

Work with an SME (Subject Matter Expert) of the area, who is the end user of the developed content. This expert should point out the critical entities (CIs with attributes and relationships) that must exist in the CMDB to provide business value.

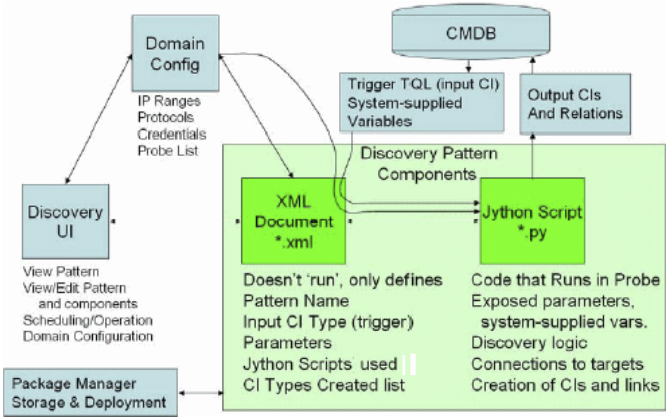
One method could be to provide a questionnaire to the application owner (also the SME in this case). The owner should be able to specify the above goals and blueprint. The owner must at least provide a current architecture of the application.

You should map critical data only and no unnecessary data: you can always enhance DDM later. The goal should be to set up a limited discovery that works and provides value. Mapping large quantities of data gives more impressive maps but can be confusing and time consuming to develop.

Once the model and business value is clear, continue to the next stage. This stage can be revisited as more concrete information is provided from the next stages.

DDM Patterns and Related Components

The following diagram shows a pattern's components and the components they interact with to execute discovery. The components in green are the actual patterns, and the components in blue are components that interact with patterns.



Note that the minimum notion of a pattern is two files: an XML document and a Jython script. The DDM Framework, including input CIs, credentials, and user-supplied libraries, is exposed to the pattern at run time. Both discovery pattern components are administered through the DDM application. They are stored operationally in the CMDB itself; although the external package remains, it is not referred to for operation. The Package Manager enables preservation of the new discovery content capability.

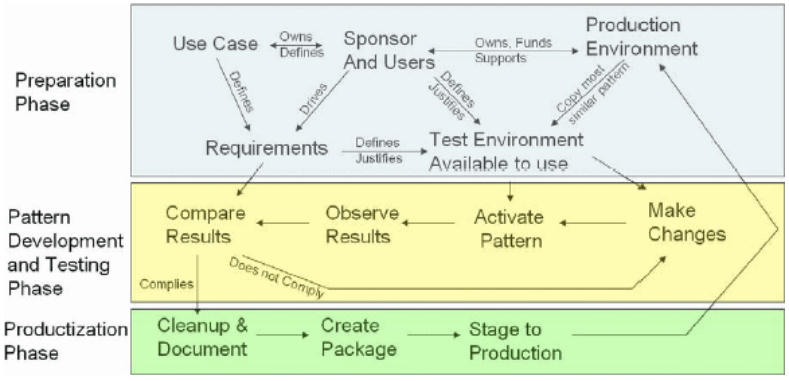
Input CIs to the pattern are provided by a TQL, and are exposed to the pattern script in system-supplied variables. Pattern parameters are also supplied as destination data, so you can configure the pattern's operation according to a pattern's specific function.

The DDM application is used to create and test new patterns. You use the job, resource, and domain configuration pages during pattern-writing.

Patterns are stored and transported as packages. The Package Manager application and the JMX console are used to create packages from newly created patterns, and to deploy patterns on new systems.

The DDM Development Cycle

The following illustration shows a flowchart for pattern-writing. Most of the time is spent in the middle section which is the iterative loop of development and testing.



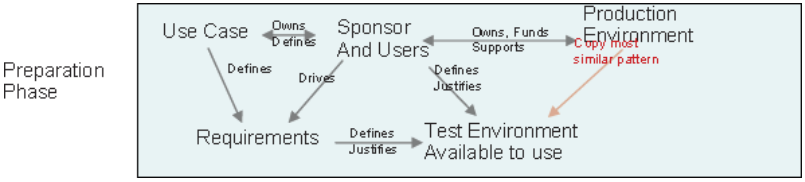
Each phase of pattern development builds on the last one:

- **The Research and Preparation Phase phase** encompasses the driving business needs and use cases, and also accounts for securing the necessary facilities to develop and test the pattern.
- **The Pattern Development and Testing phase** is a highly iterative process. As the pattern begins to take shape, you begin testing against the final use cases, make changes, test again, and repeat this process until the pattern complies with the requirements.
- **The Pattern Packaging and Productization phase** accounts for the last phase of development. As a best practice, a final pass should be made to clean up debugging remnants, documents and comments, to look at security considerations, and so on, before moving on to packaging. You should always have at least a readme document to explain the inner workings of the pattern. Someone (maybe even you) may need to look at this pattern in the future and will be aided greatly by even the most limited documentation.

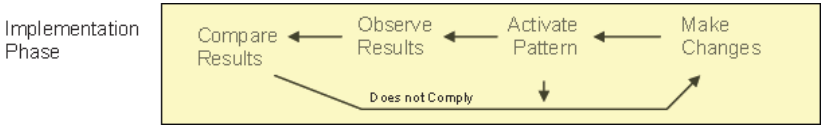
  Once you are satisfied with the way the pattern looks and works, you are ready to package it. Using either the UCMDB Package Manager or manual exporting of the components, create a DDM package *.zip file. As a best practice, you should deploy and test this package on another UCMDB system before releasing it to production, to ensure that all the components are accounted for and successfully packaged. For details on packaging, see "Package Creation" in *Model Management*.

  The following sections expand on each of the phases showing the most critical steps and best practices.

Research and Preparation Phase



1. When planning to modify an existing pattern, the first technical step is to make a backup of that pattern and ensure you can return it to its pristine state. If you plan to create a new pattern, copy the most similar pattern and save it under an appropriate name. For details, see "<Pattern files>" on page 182.
2. Research how the pattern should collect data.
   - ➤ Use External tools/protocols to obtain the data
   - ➤ Develop how the pattern should create CIs based on the data
   - ➤ You now know what a similar pattern should look like
3. Determine most similar pattern based on:
   - ➤ Same CIs created
   - ➤ Same Protocols used (SNMP)
   - ➤ Same kind of targets (by OS type, versions, and so on)
4. Copy entire package.
5. Unzip into work space and rename the pattern (XML) and Jython (.py) files.



Pattern Development and Testing

Startup and Preparation of Copy

- ➤ Modify XML parts of the pattern: Name (id) in line 1, Created CI Types, and Called Jython script name.
- ➤ Get the copy running with identical results to the original pattern.
- ➤ Comment out most of the code, especially the critical result-producing code.

Development and Testing

- Use other sample code to develop changes
- Test pattern by running it
- Use a dedicated view to validate complex results, search to validate simple results

Pattern Packaging and Productization

Cleanup and Document

- Remove debugging
- Comment all functions and add some opening comments in the main section
- Create sample TQL and view for the user to test

Create Package

- Export patterns, TQL, and so on with the Package Manager. For details, see "Package Manager" in *Model Management*.
- Check any dependencies your package has on other packages, for example, if the CIs created by those packages are input CIs to your pattern.
- Use Package Manager to create a package zip. For details, see "Package Manager" in *Model Management*.
- Test deployment by removing parts of the new content and redeploying, or deploying on another test system.

DDM and Integration

DDM discovery patterns are capable of integration with other products. Consider the following definitions:

- DDM collects specific content from many targets.
- Integration collects multiple types of content from one system.

Note that these definitions do not distinguish between the methods of collection. Neither does DDM. The process of developing a new pattern is the same process for developing new integration. You do the same research, make the same choices for new vs. existing patterns, write the patterns the same way, and so on. Only a few things change:

- The final pattern's scheduling. Integration patterns may run more frequently than discovery, but it depends on the use cases.
- Input CIs:
  - Integration: non-CI trigger to run with no input: a file name or source is passed through the pattern parameter.
  - Discovery: uses regular, UCMDB CIs for input.

For integration projects, you should almost always reuse an existing pattern. The direction of the integration (from HP Business Availability Center to another product, or from another product to HP Business Availability Center) may affect your approach to development. There are field packages available for you to copy for your own uses, using proven techniques.

From HP Business Availability Center to another project:

- Create a TQL that produces the CIs and relations you want to export.
- Use a generic wrapper pattern to execute the TQL and write the results to an XML file for the external product to read.

Note: For examples of field packages, contact HP Software Support.

To integrate another product to HP Business Availability Center: Depending on how the other product exposes its data, the integration pattern acts differently:

Integration Type	Reference Example to Be Reused
Access the product's database directly	HP ED
Read in a csv or xml file produced by an export	HP ServiceCenter
Access a product's API	BMC Atrium/Remedy

Research Stage

The prerequisite of this stage is a **blueprint** of the CIs and relationships needed to be discovered by DDM, which should include the attributes that are to be discovered. For details, see "Content Development and Pattern-Writing – Introduction" on page 280.

This section includes the following topics:

Modifying an Existing Pattern

You modify an existing pattern when an out-of-the-box or field DDM pattern exists, but:

- it does not discover specific attributes that are needed
- a specific type of target (OS) is not being discovered or is being incorrectly discovered
- a specific relationship is not being discovered or created

If an existing pattern does some, but not all, of the job, your first approach should be to evaluate the existing patterns and verify if one of them almost does what you want; if it does, you can modify the existing pattern.

You should also evaluate if an existing field pattern is available. Field patterns are discovery patterns that are available but are not out-of-the-box. Contact HP Software Support to receive the current list of field patterns.

Writing a New Pattern

A new pattern needs to be developed:

- When it is faster to write a pattern than to insert the information manually into the CMDB (generally, from about 50 to 100 CIs and relationships) or it is not a one-time effort.
- When the need justifies the effort.
- If out of the box or field patterns are not available.
- If the results can be reused.
- When the target environment or its data is available (you cannot discover what you cannot see).

Model Research

- Browse the CMDB class model (CI Type Manager) and match the entities and relations from your **blueprint** to existing CITs. It is highly recommended to adhere to the current model to avoid possible complications during version upgrade. If you need to extend the model, you should create new CITs since an upgrade may overwrite out of the box CITs.
- If some entities, relations, or attributes are lacking from the current model, you should create them. It is preferable to create a package with these CITs (which will also later hold all the discovery, views, and other artifacts relating to this package) since you need to be able to deploy these CITs on each installation of HP Business Availability Center.

Technology Research

Once you have verified that the CMDB hold the relevant CIs, the next stage is to decide how to retrieve this data from the relevant systems.

Retrieving data usually involves using a protocol to access a management part of the application, actual data of the application, or configuration files or databases that are related to the application. Any data source that can provide information on a system is valuable. Technology research requires both extensive knowledge of the system in question and sometimes creativity.

For home-grown applications, it may be helpful to provide a questionnaire form to the application owner. In this form the owner should list all the areas in the application that can provide information needed for the blueprint and business values. This information should include (but does not have to be limited to) management databases, configuration files, log files, management interfaces, administration programs, Web services, messages or events sent, and so on.

For off-the-shelf products, you should focus on documentation, forums, or support of the product. Look for administration guides, plug-ins and integrations guides, management guides, and so on. If data is still missing from the management interfaces, read about the configuration files of the application, registry entries, log files, NT event logs, and any artifacts of the application that control its correct operation.

Guidelines for Choosing Ways to Access Data

Relevance: Select sources or a combination of sources that provide the most data. If a single source supplies most information whereas the rest of the information is scattered or hard to access, try to assess the value of the remaining information by comparison with the effort or risk of getting it. Sometimes you may decide to reduce the blueprint if the value or cost does not warranty the invested effort.

Reuse: If HP Business Availability Center already includes a specific connection protocol support it is a good reason to use it. It means the DDM Framework is able to supply a ready made client and configuration for the connection. Otherwise, you may need to invest in infrastructure development. You can view the currently supported HP Business Availability Center connection protocols: **Discovery** > **Setup Discovery Probe** > **Domains and Probes pane**. For details, see "Domains and Probes Pane" on page 79.

You can add new protocols by adding new CIs to the model. For details, contact HP Software Support.

Note: To access Windows Registry data, you can use either WMI or NTCmd.

Security: Access to information usually requires credentials (user name, password), which are entered in the CMDB and are kept secure throughout the product. If possible, and if adding security does not conflict with other principles you have set, choose the least sensitive credential or protocol that still answers access needs. For example, if information is available both through JMX (standard administration interface, limited) and Telnet, it is preferable to use JMX since it inherently provides limited access and (usually) no access to the underlying platform.

Comfort: Some management interfaces may include more advanced features. For example, it might be easier to issues queries (SQL, WMI) than to navigate information trees or build regular expressions for parsing.

Developer Audience: The people who will eventually develop DDM may have an inclination towards a certain technology. This can also be considered if two technologies provide almost the same information at an equal cost in other factors.

Summary

The outcome of this stage is a document describing the access methods and the relevant information that can be extracted from each method. The document should also contain a mapping from each source to each relevant blueprint data.

Each access method should be marked according to the above instructions. Finally you should now have a plan of which sources to discover and what information to extract from each source into the blueprint model (which should by now have been mapped to the corresponding UCMDB model).

HP Discovery and Dependency Mapping API Reference

For full documentation on the available APIs, see *HP Discovery and Dependency Mapping API Reference*. These files are located in the following folder:

\\<HP Business Availability Center root directory>\AppServer\webapps\site.war\amdocs\eng\doc_lib\Discovery_and_Dependency_Mapping\DDM_JavaDoc\index.html

Implement a Pattern

A DDM task has the aim of accessing remote (or local) systems, modeling extracted data as CIs, and saving the CIs to the CMDB. The task consists of the following steps:

1 **DDM pattern.**

You configure a pattern file that holds the context, parameters, and result types for DDM by selecting the scripts that are to be part of the pattern. For details, see the following section.

2 **DDM job.**

You configure a job with scheduling information and a trigger TQL. For details, see “Step 2: Assign a Job to the Pattern” on page 302.

3 **DDM code.**

You can edit the Jython or Java code that is contained in the pattern files and that refers to the DDM Framework. For details, see “Step 3: Create Code” on page 304.

To write new DDM content, you create each of the above components, each one of which is automatically bound to the component in the previous step. For example, once you create a job and select the relevant pattern, the pattern file binds to the job.

Step 1: Create a Discovery and Dependency Mapping Pattern

A DDM pattern can be considered as the definition of a function. This function defines an input definition, runs logic on the input, defines the output, and provides a result.

Each DDM pattern specifies input and output: Both input and output are Trigger CIs that are specifically defined in the pattern. The DDM pattern extracts data from the input Trigger CI and passes this data as parameters to the DDM code. (Data from related CIs is sometimes passed to the code too. For details, see "Get Related CIs" on page 114.) A pattern's DDM code is generic, apart from these specific input Trigger CI parameters that are passed to the code.

For details on input components, see "Trigger CITs, Trigger CIs, Input TQLs, and Trigger TQLs" on page 50.

This task describes how to create a DDM pattern.

This task includes the following steps:

- "Define Pattern Input (Trigger CIT and Input TQL)" on page 294
- "Define Pattern Output" on page 299
- "Override Pattern Parameters" on page 300

Define Pattern Input (Trigger CIT and Input TQL)

You use the Trigger CIT and Input Topology Query Language (TQL) components to define specific CIs as pattern input:

- The Trigger CIT defines which CIT is used as the input for the pattern. For example, for a pattern that is going to discover IPs, the input CIT is Network.
- The Input TQL is a regular, editable TQL that defines the query against the CMDB. The Input TQL defines additional constraints on the CIT (for example, if the task requires a hostID or application_ip attribute), and can define more CI data, if needed by the pattern.

If the pattern requires additional information from the CIs that are related to the Trigger CI, you can add additional nodes to the input TQL. For details, see "Example of Input TQL Definition" on page 296 and "Add Nodes and Relationships to a TQL Query" in *Model Management*.

➤ The Trigger CI data contains all the required information on the Trigger CI as well as information from the other nodes in the Input TQL, if they are defined. DDM uses variables to retrieve data from the CIs. When the task is downloaded to the Probe, the Trigger CI data variables are replaced with actual values that exist on the attributes for real CI instances.

Example of Trigger CIT Definition

In this example, a Trigger CIT defines that IP CIs are permitted in the pattern.

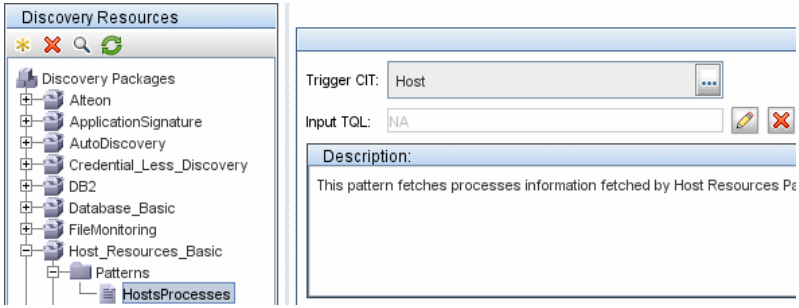
1 Access **Discovery** > **Manage Discovery Resources** > **Pattern Signature**. Select the HostProcesses pattern (**Discovery Packages** > **Host_Resources_Basic** > **Patterns** > **HostProcesses**).

2 Locate the Trigger CIT box. For details, see "Triggered CI Data Pane" on page 197.

3 Click the button to open the Choose Configuration Item Type dialog box. For details, see "Choose Configuration Item Type Dialog Box" on page 175.

4 Select the CIT.

In this example, the IP CI (Host) is permitted in the pattern:



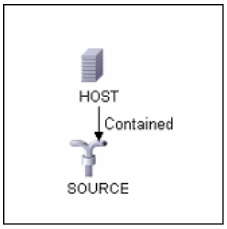
Example of Input TQL Definition

In this example, the Input TQL defines that the IP CI (configured in the previous example as the Trigger CIT) must be connected to a Host CI.

1 Access **Discovery** > **Manage Discovery Resources** > **Pattern Signature**. Locate the Input TQL box. Click the **Edit** button to open the Input TQL Editor. For details, see "Input TQL Editor Window" on page 185.

2 In the Input TQL Editor, name the Trigger CI node **SOURCE**: right-click the node and choose **Node Properties**. In the **Element Name** box, change the name to **SOURCE**.

3 Add a host and a Contained relationship to the IP CI. For details on working with the Input TQL Editor, see "Input TQL Editor Window" on page 185.



The **IP** CI is connected to a **HOST** CI. The input TQL consists of two nodes, **HOST** and **IP**, with a link between them. The **IP** CI is named **SOURCE**.

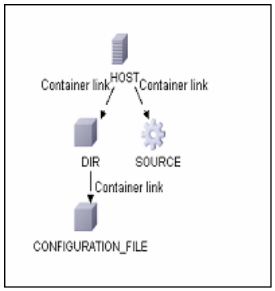
Example of Adding Variables to the Input TQL

In this example, you add DIRECTORY and CONFIGURATION_FILE variables to the Input TQL created in the previous example. These variables help to define what you want to discover, in this case, to find the configuration files residing on the hosts that are linked to the IPs you need to discover.

1 Display the Input TQL created in the previous example.

2 Access **Discovery** > **Manage Discovery Resources** > **Pattern Management**. Locate the Triggered CI Data pane. For details, see "Triggered CI Data Pane" on page 197.

3 Add variables to the Input TQL. For details, see "Value" on page 198.



Example of Replacing Variables with Actual Data

In this example, variables replace the IP CI data variables with actual values that exist on real IP CI instances in your system.

The Triggered CI data for the IP CI includes a fileName variable. This variable enables the replacement of the CONFIGURATION_FILE node in the Input TQL with the actual values of the configuration file located on a host:

Triggered CI Data

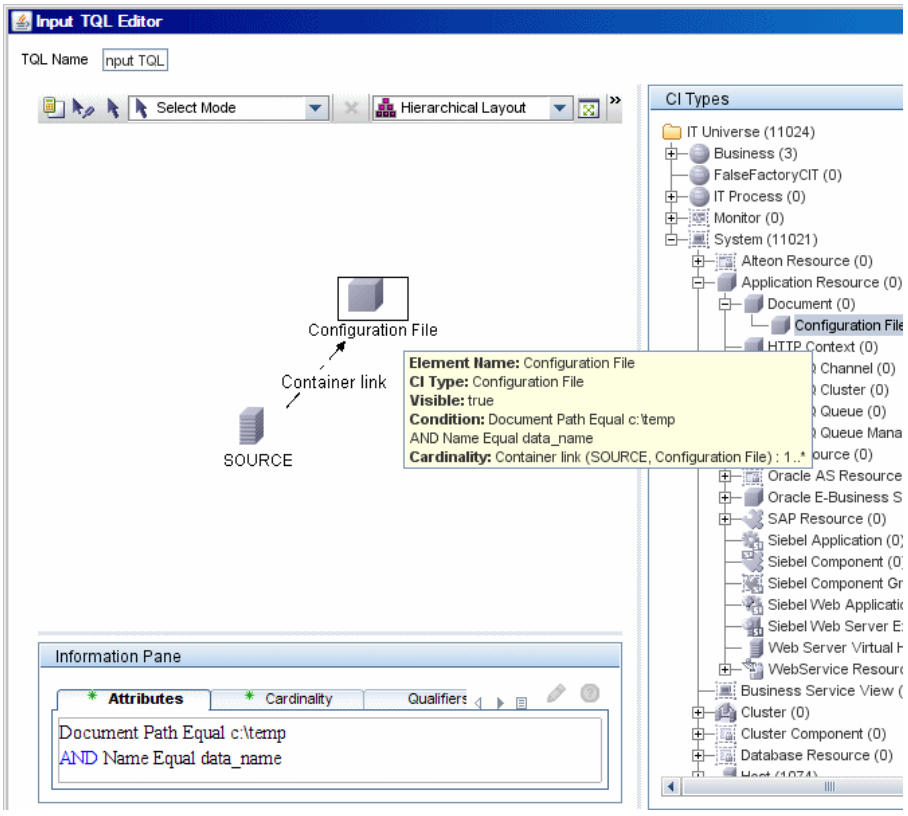
Name	Value
Protocol	${SOURCE.credentials_id}
credentialsId	${SOURCE.credentials_id}
fileName	${CONFIGURATION_FILE.data_name}
hostID	${HOST.root_id}
ip_address	${SOURCE.application_ip}
path	${CONFIGURATION_FILE.document_path}

The Trigger CI data is uploaded to the Probe with all variables replaced by actual values. The pattern script includes a command to use the DDM Framework to retrieve the actual values of the defined variables:

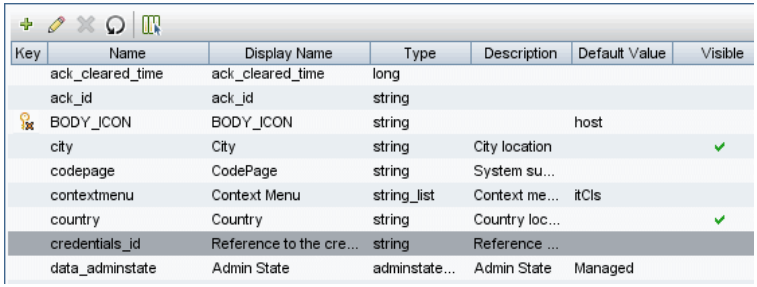
```
Framework.getTriggerCIData ('ip_address')
```

Note:

➤ The fileName and path variables use the data_name and document_path attributes from the Configuration File node (defined in the Input TQL – see previous example):



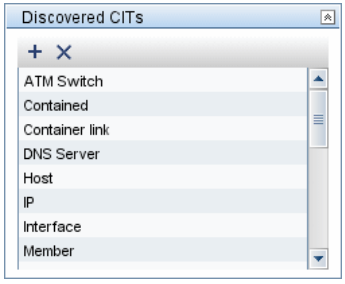
➤ The Protocol, credentialsId, and ip_address variables use the root_class, credentials_id, and application_ip attributes:



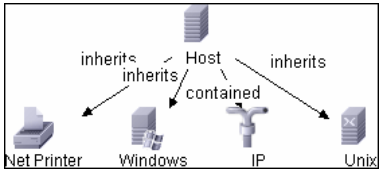
Key	Name	Display Name	Type	Description	Default Value	Visible
	ack_cleared_time	ack_cleared_time	long			
	ack_id	ack_id	string			
	BODY_ICON	BODY_ICON	string		host	
	city	City	string	City location		✔
	codepage	CodePage	string	System su...		
	contextmenu	Context Menu	string_list	Context me...	itCIs	
	country	Country	string	Country loc...		✔
	credentials_id	Reference to the cre...	string	Reference ...		
	data_adminstate	Admin State	adminstate...	Admin State	Managed	

Define Pattern Output

The output of the pattern is new CIs and links between them. That is, the output is a list of discovered CITs (**Discovery** > **Manage Discovery Resources** > **Pattern Signature tab**):



You can also view the CITs as a topology map, that is, the components and the way in which they are linked together (click the **View Discovered CITs as Map** button):



The discovered CIs are returned by the DDM code (that is, the Jython script) in the format of UCMDB's ObjectStateHolderVector. For details, see “Results Generation by the Jython Script” on page 310.

Example of Pattern Output

In this example, you define which CITs are to be part of the IP CI output.

1 Access **Discovery** > **Manage Discovery Resources**.

2 In the Discovery Resources pane, select **Network** > **Pattern** > **NSLOOKUP_on_Probe**.

3 In the Pattern Signature tab, locate the Discovered CITs pane.

4 The CITs that are to be part of the pattern output are listed. Add CITs to, or remove from, the list. For details, see "Discovered CITs Pane" on page 196.

Override Pattern Parameters

To configure a pattern for more than one job, you can override pattern parameters. For example, the pattern SQL_NET_Dis_Connection is used by both the MSSQL Connection by SQL and the Oracle Connection by SQL jobs.

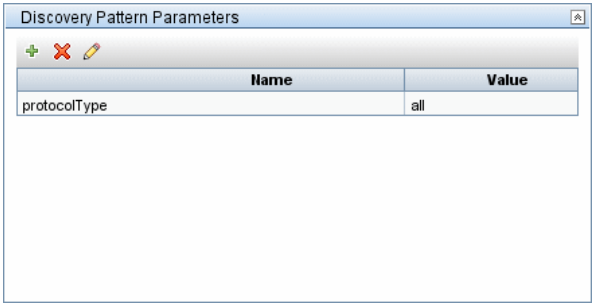
Example of Overriding a Pattern Parameter

This example illustrates overriding a pattern parameter so that one pattern can be used to discover both Microsoft SQL Server and Oracle databases.

1 Access **Discovery** > **Manage Discovery Resources**.

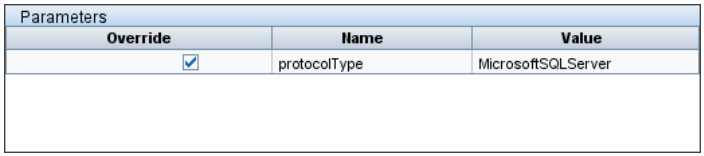
2 In the Discovery Resources pane, select **Database Basic** > **Pattern** > **SQL_NET_Dis_Connection**.

3 In the Pattern Signature tab, locate the **Discovery Pattern Parameters** pane. The protocolType parameter has a value of **all**:



4 Right-click the **SQL_NET_Dis_Connection** pattern and choose **Go to Discovery Job** > **MSSQL Connection by SQL**.

5 Display the Properties tab. Locate the Parameters pane:



The all value is overwritten with the MicrosoftSQLServer value.

Note: The Oracle Connection by SQL job includes the same parameter but the value is overwritten with an oracle value.

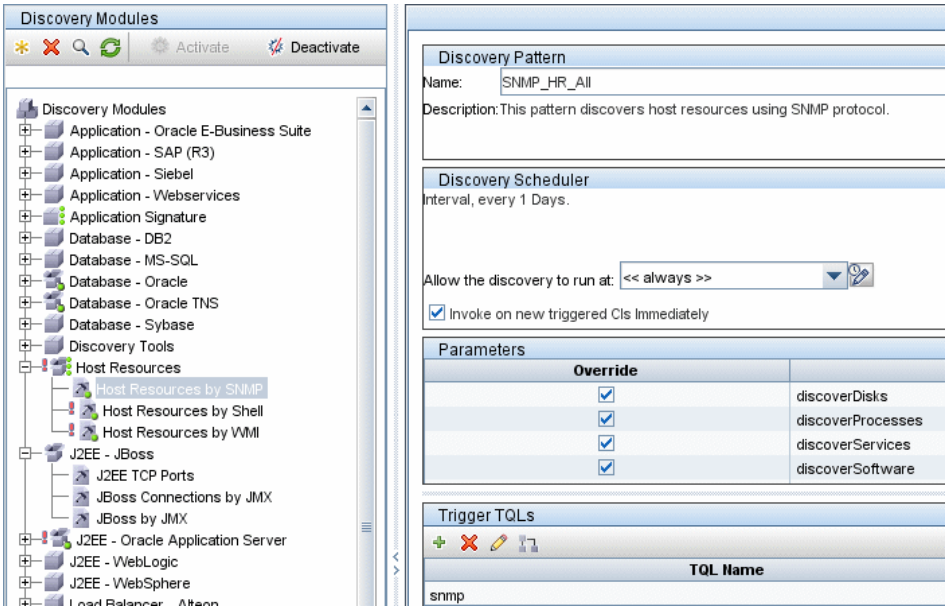
For details on adding, deleting, or editing parameters, see "Discovery Pattern Parameters Pane" on page 196.

DDM begins looking for Microsoft SQL Server instances according to this parameter.

Step 2: Assign a Job to the Pattern

Each pattern has one or more associated jobs that define the execution policy. Jobs enable scheduling the same pattern differently over different set of Triggered CIs and also enable supplying different parameters for each set.

The jobs appear in the Discovery Modules tree, and this is the entity that the user activates.



Trigger TQL

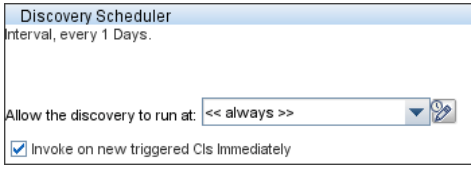
Each job is associated with Trigger TQLs. These Trigger TQLs publish results that are used as Input Trigger CIs for the pattern of this job.

A Trigger TQL can add constraints to an Input TQL. For example, if an input TQL's results are IPs connected to SNMP, a trigger TQL's results can be IPs connected to SNMP within the range 195.0.0.0-195.0.0.10.

Note: A trigger TQL must refer to the same objects that the input TQL refers to. For example, if an input TQL queries for IPs running SNMP, you cannot define a trigger TQL (for the same job) to query for IPs connected to a host, because some of the IPs may not be connected to an SNMP object, as required by the input TQL.

Scheduling

The scheduling information for the Probe specifies when to run the code on Trigger CIs. If the **Invoke on new triggered CIs Immediately** check box is selected, the code also runs once on each Trigger CI when it reaches the Probe, regardless of future schedule settings.



For each schedule occurrence for each job, the Probe runs the DDM code against all Trigger CIs accumulated for that job. For details, see “Schedule Modules to Run” on page 53 and “Discovery Scheduler Dialog Box” on page 138.

Parameters

When configuring a job you can override the pattern parameters. For details, see “Override Pattern Parameters” on page 300.

Step 3: Create Code

DDM code is mostly written in Jython. Jython is an implementation of the high-level, dynamic, object-oriented language Python written in 100% pure Java, and seamlessly integrated with the Java platform. It thus enables you to run Python on any Java platform. (From the Jython Web site http://www.jython.org/Project/index.html.)

The following section describes the actual writing of Jython code inside the DDM Framework. This section specifically addresses those contact points between the Jython script and the Framework that it calls, and also describes the Jython libraries and utilities that should be used whenever possible.

Note: Scripts written for DDM should be compatible with Jython version 2.1.

This task describes how to write code.

This task includes the following steps:

- "Execution of the Code" on page 304
- "Modifying Out of the Box Scripts" on page 305
- "Structure of the Jython File" on page 307
- "Results Generation by the Jython Script" on page 310
- "The Framework Instance" on page 312
- "Finding the Correct Credentials (for Connection Patterns)" on page 315
- "Handling Exceptions from Java" on page 317

1 Execution of the Code

After a job is activated, a task with all the required information is downloaded to the Probe.

The Probe starts running the DDM code using the information specified in the task.

The Jython code flow starts running from a main entry in the script, executes code to discover CIs, and provides results of a vector of discovered CIs.

2 Modifying Out of the Box Scripts

When making out of the box script modifications, make only minimal changes to the script and place any necessary methods in an external script. You can track changes more efficiently and, when moving to a newer HP Business Availability Center version, your code is not overwritten.

For example, the following single line of code in an out of the box script calls a method that calculates a Web server name in an application-specific way:

```
serverName = iplanet_cspecific.PlugInProcessing(serverName, transportHN,
mam_utils)
```

The more complex logic that decides how to calculate this name is contained in an external script:

```
# implement customer specific processing for 'servername' attribute of httpplugin

#

def PlugInProcessing(servername, transportHN, mam_utils_handle):

        # support application-specific HTTP plug-in naming

        if servername == "appsrv_instance":

                # servername is supposed to match up with the j2ee server name,
however some groups do strange things with their

                # iPlanet plug-in files. this is the best work-around we could find. this join
can't be done with IP address:port

                # because multiple apps on a web server share the same IP:port for
multiple websphere applications

                logger.debug('httpcontext_webapplicationserver attribute has been
changed from [' + servername + '] to [' + transportHN[:5] + '] to facilitate websphere
enrichment')

                servername = transportHN[:5]

        return servername
```

Save the external script in the External Resources folder. For details, see “Discovery Resources Pane” on page 180. If you add this script to a package, you can use this script for other jobs, too. For details on working with Package Manager, see “Package Manager” in *Model Management*.

During upgrade, the change you make to the single line of code is overwritten by the new version of the out of the box script, so you will need to replace the line. However, the external script is not overwritten.

3 Structure of the Jython File

The Jython file is composed of three parts in a specific order:

- 1. Imports
- 2. Functions definitions (optional)
- 3. Main Function - DiscoveryMain

The following is an example of a Jython script:

```
#   imports section
from appilog.common.system.types import ObjectStateHolder
from appilog.common.system.types.vectors import ObjectStateHolderVector

# Function definition
def foo:
 # do something

#   Main Function
def DiscoveryMain(Framework):
 OSHVResult = ObjectStateHolderVector()

 ## Write implementation to return new result CIs here...

 return OSHVResult
```

Imports

Jython classes are spread across hierarchical namespaces. In version 7.0 or later, unlike in previous versions, there are no implicit imports, and so every class you use must be imported explicitly. (This change was made for performance reasons and to enable an easier understanding of the Jython script by not hiding necessary details.)

- To import a Jython script:

```
import logger
```

- To import a Java class:

```
from appilog.collectors.clients import ClientsConsts
```

Main Function – DiscoveryMain

Each Jython runable script file contains a main function: DiscoveryMain.

The DiscoveryMain function is the main entry into the script; it is the first function that runs. The main function may call other functions that are defined in the scripts:

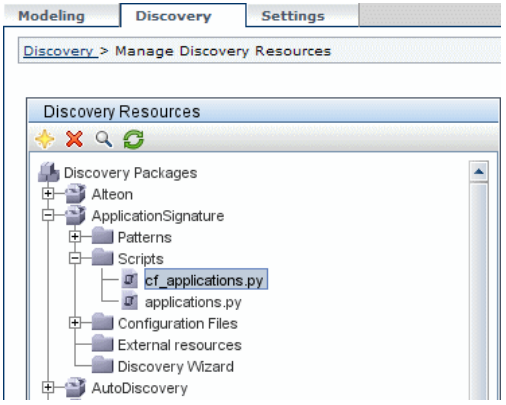
```
def DiscoveryMain(Framework):
```

The Framework argument must be specified in the main function definition. This argument is used by the main function to retrieve information that is required to run the scripts (such as information on the Trigger CI and parameters) and can also be used to report on errors that occur during the script run.

You can create a Jython script without any main method. Such scripts are used as library scripts that are called from other scripts.

Functions Definition

Each script can contain additional functions that are called from the main code. Each such function can call another function, which either exists in the current script or in another script (use the import statement). Note that to use another script, you must add it to the Scripts section of the package:



Example of a Function Calling Another Function

In the following example, the main code calls the doQueryOSUsers(..) method which calls an internal method doOSUserOSH(..):

```
def doOSUserOSH(name):
 sw_obj = ObjectStateHolder('winosuser')

 sw_obj.setAttribute('data_name', name)
 # return the object
 return sw_obj

def doQueryOSUsers(client, OSHVResult):
 _hostObj = modeling.createHostOSH(client.getIpAddress())
 data_name_mib = '1.3.6.1.4.1.77.1.2.25.1.1,1.3.6.1.4.1.77.1.2.25.1.2,string'
 resultSet = client.executeQuery(data_name_mib)
 while resultSet.next():
       UserName = resultSet.getString(2)
       ########## send object ###############
       OSUserOSH = doOSUserOSH(UserName)
       OSUserOSH.setContainer(_hostObj)
       OSHVResult.add(OSUserOSH)

def DiscoveryMain(Framework):
 OSHVResult = ObjectStateHolderVector()
 try:
       client =
Framework.getClientFactory(ClientsConsts.SNMP_PROTOCOL_NAME).createClient()
 except:
       Framework.reportError('Connection failed')
 else:
       doQueryOSUsers(client, OSHVResult)
       client.close()
 return OSHVResult
```

If this script is a global library that is relevant to many patterns, you can add it to the list of scripts in the jythonGlobalLibs.xml configuration file, instead of adding it to each pattern (**Discovery** > **Manage Discovery Resources** > **Discovery Packages** > **AutoDiscovery** > **Configuration Files**).

4 Results Generation by the Jython Script

Each Jython script runs on a specific Trigger CI, and ends with results that are returned by the return value of the DiscoveryMain function.

The script result is actually a group of CIs and links that are to be inserted or updated in the CMDB. The script returns this group of CIs and links in the format of ObjectStateHolderVector.

The ObjectStateHolder class is a way to represent an object or link defined in the CMDB. The ObjectStateHolder object contains the CIT name and a list of attributes and their values. The ObjectStateHolderVector is a vector of ObjectStateHolder instances.

The ObjectStateHolder Syntax

This section explains how to build the DDM results into a CMDB model.

Example of Setting Attributes on the CIs

The ObjectStateHolder class describes the DDM result graph. Each CI and link (relationship) is placed inside an instance of the ObjectStateHolder class as in the following Jython code sample:

```
# siebel application server
1 appServerOSH = ObjectStateHolder('siebelappserver' )
2 appServerOSH.setStringAttribute('data_name', sblsvrName)
3 appServerOSH.setStringAttribute ('application_ip', ip)
4 appServerOSH.setContainer(appServerHostOSH)
```

- Line 1 creates a CI of type **siebelappserver**.
- Line 2 creates an attribute called **data_name** with a value of **sblsvrName** which is a Jython variable we set with the value we discovered for the server name.
- Line 3 sets a non-key attribute that is updated in the CMDB.
- Line 4 is the building of containment (the result is a graph). It specifies that this application server is contained inside a host (another ObjectStateHolder class in the scope).

Note: Each CI being reported by the Jython script must include values for all the key attributes of the CI's CI Type.

Example of Relations (Links)

The following link example explains how the graph is represented:

```
1 linkOSH = ObjectStateHolder('route')
2 linkOSH.setAttribute('link_end1', gatewayOSH)
3 linkOSH.setAttribute('link_end2', appServerOSH)
```

- Line 1 creates the link (that is also of the ObjectStateHolder class. The only difference is that route is a link CI Type).
- Lines 2 and 3 specify the nodes at the end of each link. This is done using the **end1** and **end2** attributes of the link which must be specified (because they are the minimal key attributes of each link). The attribute values are ObjectStateHolder instances. For details on End 1 and End 2, see "Link" on page 176.

Important: A link is directional. You should verify that End 1 and End 2 nodes correspond to valid CITs at each end. If the nodes are not valid, the result object fails DDM validation and is not reported correctly. For details, see "CI Type Relationships" in *CI Attribute Customization*.

Example of Vector (Gathering CIs)

After creating objects with attributes, and links with objects at their ends, you must now group them together. You do this by adding them to an ObjectStateHolderVector instance, as follows:

```
oshvMyResult = ObjectStateHolderVector()
oshvMyResult.add(appServerOSH)
oshvMyResult.add(linkOSH)
```

For details on reporting this composite result to the Framework so it can be sent to the CMDB server, see the sendObjects method.

Once the DDM result graph is assembled in an ObjectStateHolderVector instance, it must be returned to the DDM Framework to be inserted into the CMDB. This is done by returning the ObjectStateHolderVector instance as the result of the DiscoveryMain() function.

Note: For details on creating **OSH** for common CITs, see **modeling.py** in "Jython Libraries and Utilities" on page 320.

---

5 The Framework Instance

The Framework instance is the only argument that is supplied in the main function in the Jython script. This is an interface that can be used to retrieve information required to run the script (for example, information on trigger CIs and pattern parameters), and is also used to report on errors that occur during the script run. For details, see "HP Discovery and Dependency Mapping API Reference" on page 292.

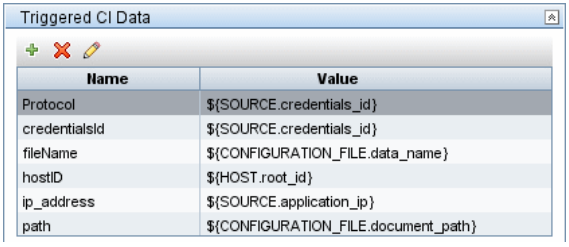
This section describes the most important Framework usages.

Framework.getTriggerCIData(String attributeName)

This API provides the intermediate step between the Trigger CI data defined in the pattern and the script.

Example of Retrieving Credential Information

You request the following Trigger CI data information:



To retrieve the credential information from the task, use this API:

```
credId = Framework.getTriggerCIData('credentialsId')
```

Framework.getClientFactory(String protocol)

A connection to a remote machine is made by creating a client object and executing commands on that client. To create a client, you retrieve the ClientFactory class. The getClientFactory() method receives the type of the requested client protocol. The protocol constants are defined in the ClientsConsts class. For details on credentials and supported protocols, see "Domain Credential References" on page 83.

Example of Creating a ClientFactory Instance for the NTCmd Protocol

To create a ClientFactory instance for the NTCmd protocol:

```
clientFactory =
Framework.getClientFactory(ClientsConsts.NTCMD_PROTOCOL_NAME)
```

You can now use the ClientFactory instance to create the required client (corresponding to the protocol supplied for ClientFactory).

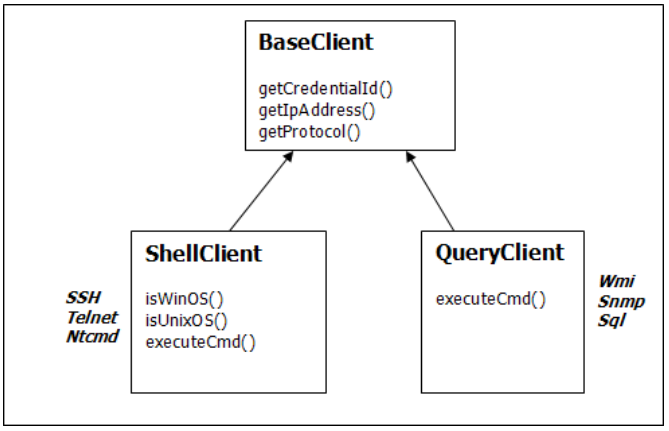
Example of Creating a WMI Client and Running a WMI Query

To create a WMI client and run a WMI query using the client:

```
wmiClient = Framework.createClient(credential)
resultSet = wmiClient. executeQuery("SELECT TotalPhysicalMemory
                                    FROMWin32_LogicalMemoryConfiguration")
```

Note: To make the createClient() API work, add the following parameter to the Trigger CI data parameters: **credentialsId = ${SOURCE.credentials_id}** in the Triggered CI Data pane. Or you can manually add the credentials ID when calling the function:
wmiClient = clientFactory().createClient(credentials_id).

The following diagram illustrates the hierarchy of the clients, with their commonly-supported APIs:



For details on the clients and their supported APIs, see BaseClient, ShellClient, and QueryClient in the *HP Discovery and Dependency Mapping API Reference*.

Framework.getParameter (String parameterName)

In addition to retrieving information on the Trigger CI, you often need to retrieve a pattern parameter value. For example:

Parameters

Override	Name	Value
☑	protocolType	MicrosoftSQLServer

Example of Retrieving the Value of the protocolType Parameter

To retrieve the value of the protocolType parameter from the Jython script, use the following API:

```
protocolType = Framework.getParameterValue('protocolType')
```

Framework.reportError(String message) and Framework.reportWarning(String message)

Some errors (for example, connection failure, hardware problems, timeouts) can occur during a script run. When such errors are detected, Framework can report on the problem. The message that is reported reaches the server and is displayed for the user.

Example of a Report Error and Message

The following example illustrates the use of the reportError(<Error Msg>) API:

```
try:
      client = Framework.getClientFactory(ClientsConsts.SNMP_PROTOCOL_NAME)
      createClient()
except:
      strException = str(sys.exc_info()[1]).strip()
      Framework. reportError ('Connection failed: %s' % strException)
```

You can use either one of the APIs—Framework.reportError(String message), Framework.reportWarning(String message)—to report on a problem. The difference between the two APIs is that when reporting an error, the Probe saves a communication log file with the entire session's parameters to the file system. In this way you are able to track the session and better understand the error.

6 Finding the Correct Credentials (for Connection Patterns)

A pattern trying to connect to a remote system needs to try all possible credentials. One of the parameters needed when creating a client (through ClientFactory) is the credentials ID. The connection script gains access to possible credential sets and tries them one by one using the clientFactory.getAvailableProtocols() method. When one credential set succeeds, the pattern reports a CI connection object on the host of this trigger CI (with the credentials ID that matches the IP) to the CMDB. Subsequent patterns can use this connection object CI directly to connect to the credential set (that is, the patterns do not have to try all possible credentials again).

The following example shows how to obtain all entries of the SNMP protocol. Note that here the IP is obtained from the Trigger CI data (# Get the Trigger CI data values).

The connection script requests all possible protocol credentials (# Go over all the protocol credentials) and tries them in a loop until one succeeds (resultVector). For details, see the **two-phase connect paradigm** entry in "Separating Patterns" on page 325.

```
import logger
from appilog.collectors.clients import ClientsConsts
from appilog.common.system.types.vectors import ObjectStateHolderVector

      def mainFunction(Framework):
resultVector = ObjectStateHolderVector()

      # Get the Trigger CI data values
      ip_address = Framework.getDestinationAttribute('ip_address')
      ip_domain  = Framework.getDestinationAttribute('ip_domain')

      # Create the client factory for SNMP
      clientFactory = framework.getClientFactory(ClientsConsts.SNMP_PROTOCOL_NAME)
      protocols = clientFactory.getAvailableProtocols(ip_address, ip_domain)

      connected = 0

      # Go over all the protocol credentials
      for credentials_id in protocols:
            client = None
            try:
                  # try to connect to the snmp agent
                  client = clientFactory.createClient(credentials_id)

                  // Query the agent
                  ….

                  # connection succeed
                  connected = 1
            except:
                  if client != None:
                        client.close()

      if (not connected):
            logger.debug('Failed to connect using all credentials')
      else:
            // return the results as OSHV
            return resultVector
```

7 Handling Exceptions from Java

Some Java classes throw an exception upon failure. It is recommended to catch the exception and handle it, otherwise it causes the pattern to terminate unexpectedly.

When catching a known exception, in most cases you should print its stack trace to the log and issue a proper message to the UI, for example:

```
try:
    client = Framework.getClientFactory().createClient()
except Exception, msg:
    Framework.reportError('Connection failed')
    logger.debugException('Exception while connecting: %s' % (msg))
    return
```

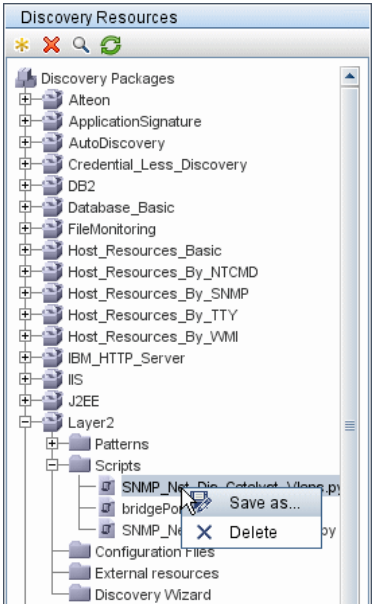
If the exception is not fatal and the script can continue, you should omit the call for the reportError() method and enable the script to continue.

Discovery and Dependency Mapping Code

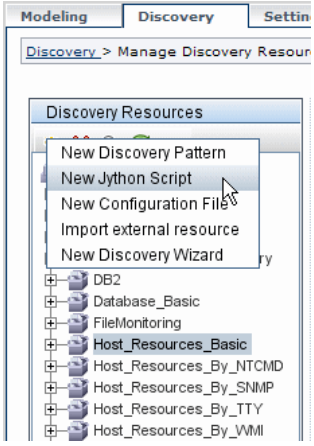
The actual implementation of connecting to the remote system, querying its data, and mapping it as CMDB data is performed by the DDM code. For example, the code contains the logic for connecting to a database and extracting data from it. In this case, the code expects to receive a JDBC URL, a user name, a password, a port, and so on. These parameters are specific for each instance of the database that answers the TQL query. We define these variables in the pattern (in the Trigger CI data) and when the job runs, these specific details are passed to the code for execution.

The pattern can refer to this code by a Java class name or a Jython script name. In this section we discuss writing DDM code as Jython scripts.

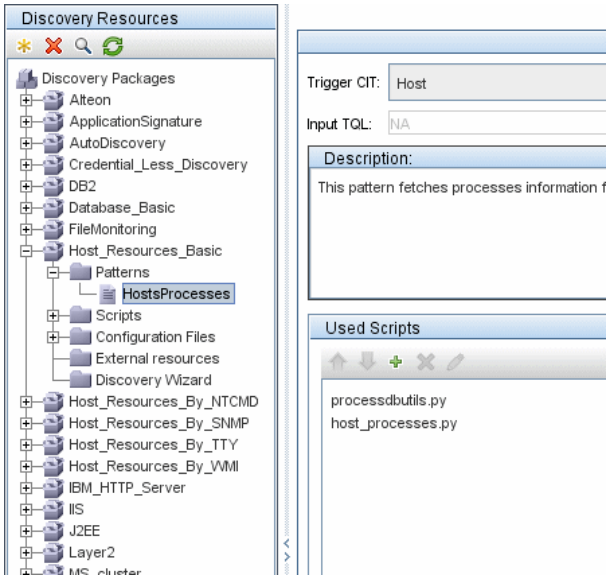
A pattern can contain a list of scripts to be used when running DDM. When creating a new pattern, you usually create a new script and assign it to the pattern. A new script includes basic templates, but you can use one of the other scripts as a template by right-clicking it and selecting **Save as**:



For details on writing new Jython scripts, see "Step 3: Create Code" on page 304. You add scripts through the Manage Discovery Resources window:



The list of scripts are run one after the other, in the order in which they are defined in the pattern:



Note: A script must be specified even though it is being used solely as a library by another script. In this case, the library script must be defined before the script using it. In this example, the processdbutils.py script is a library used by the last host_processes.py script. Libraries are distinguished from regular runable scripts by the lack of the DiscoveryMain() function.

Jython Libraries and Utilities

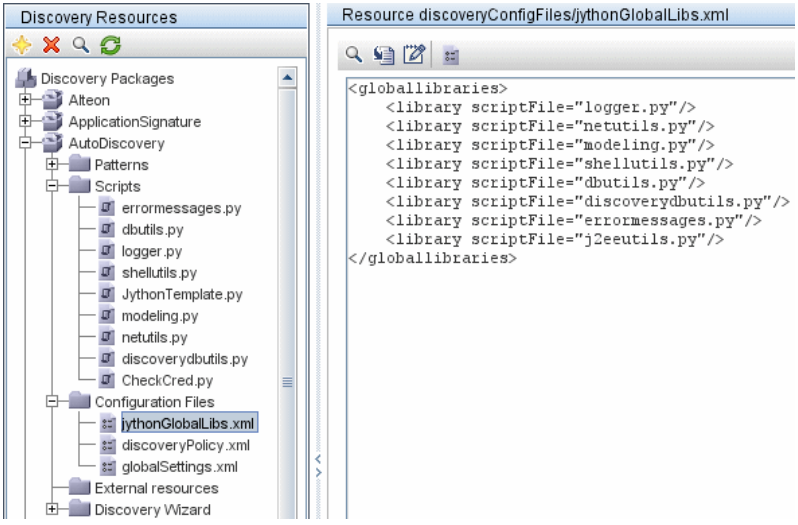
Several utility scripts are used widely in DDM patterns. These scripts are part of the AutoDiscovery package and are located under: **<DDM Probe root directory>\DiscoveryProbe\root\lib\collectors\probeManager\discoveryScripts** with the other scripts that are downloaded to the Probe.

Note: The discoveryScript folder is created dynamically when the Probe begins working.

To use one of the utility scripts, add the following import line to the import section of the script:

```
import <script name>
```

The AutoDiscovery Python library contains Jython utility scripts. These library scripts are considered DDM's external library. They are defined in the jythonGlobalLibs.xml file (located in the **Configuration Files** folder).



Each script that appears in the jythonGlobalLibs.xml file is loaded by default at Probe startup, so there is no need to use them explicitly in the pattern definition.

This section includes the following topics:

logger.py

The **logger.py** script contains log utilities and helper functions for error reporting. You can call its debug, info, and error APIs to write to the log files. Log messages are recorded in **C:\hp\DDM\DiscoveryProbe\root\logs\probeMgr-patternsDebug.log**.

Messages are entered in the log file according to the debug level defined for the PATTERNS_DEBUG appender in the **C:\hp\DDM\DiscoveryProbe\root\lib\collectors\probeManager\probeMgrLog4j.properties** file. (By default, the level is DEBUG.) For details, see "Severity Levels" on page 53.

```
########################################################################
#################       PATTERNS_DEBUG log              ################
########################################################################
log4j.category.PATTERNS_DEBUG=DEBUG, PATTERNS_DEBUG
log4j.appender.PATTERNS_DEBUG=org.apache.log4j.RollingFileAppender
log4j.appender.PATTERNS_DEBUG.File=C:/hp/DDM/DiscoveryProbe/root/logs/probe
Mgr-patternsDebug.log
log4j.appender.PATTERNS_DEBUG.Append=true
log4j.appender.PATTERNS_DEBUG.MaxFileSize=15MB
log4j.appender.PATTERNS_DEBUG.Threshold=DEBUG
log4j.appender.PATTERNS_DEBUG.MaxBackupIndex=10
log4j.appender.PATTERNS_DEBUG.layout=org.apache.log4j.PatternLayout
log4j.appender.PATTERNS_DEBUG.layout.ConversionPattern=<%d> [%-5p] [%t] -
%m%n
log4j.appender.PATTERNS_DEBUG.encoding=UTF-8
```

The info and error messages also appear in the Command Prompt console.

There are two sets of APIs:

- logger.<debug/info/warn/error>
- logger.<debugException/infoException/warnException/errorException>

The first set issues the concatenation of all its string arguments at the appropriate log level and the second set issues the concatenation as well as issuing the stack trace of the most recently-thrown exception, to more easily understand the exception reason. For example:

```
logger.debug('found the result')
logger.errorException('Error in discovery')
```

modeling.py

The **modeling.py** script contains APIs for creating hosts, IPs, process CIs, and so on. These APIs simplify the creation of common objects and make the code more readable. For example:

```
ipOSH= modeling.createIpOSH(ip)
host = modeling.createHostOSH(ip_address)
member1 = modeling.createLinkOSH('member', ipOSH, networkOSH)
```

netutils.py

The **netutils.py** library is used to retrieve network and TCP information, such as retrieving operating system names, checking if a MAC address is valid, checking if an IP address is valid, and so on. For example:

```
dnsName = netutils.getHostName(ip, ip)
isValidIp = netutils.isValidIp(ip_address)
address = netutils.getHostAddress(hostName)
```

shellutils.py

The **shellutils.py** library provides an API for executing shell commands and retrieving the end status of an executed command, and enables running multiple commands based on that end status. The library is initialized with a Shell Client, and uses the client to run commands and retrieve results. For example:

```
ttyClient = clientFactory.createClient(Props)
clientShUtils = shellutils.ShellUtils(ttyClient)
if (clientShUtils.isWinOs()):
        logger.debug ('discovering Windows..')
```

Using External Java jar Files Within Jython

When developing new Jython scripts, external Java Libraries (JAR files) or third party executable files are sometimes needed as either Java utility archives, connection archives such as JDBC Driver JAR files, or executable files (for example, such as **nmap.exe** is used for credential-less discovery).

These resources are bundled in the package under the External Resources folder. Any resource put in this folder is automatically sent to any Probe that connects to your HP Universal CMDB server.

In addition, when discovery is launched, any JAR file resource is loaded into the Jython's class path making all the classes within it available for import and use.

Recording DDM Code

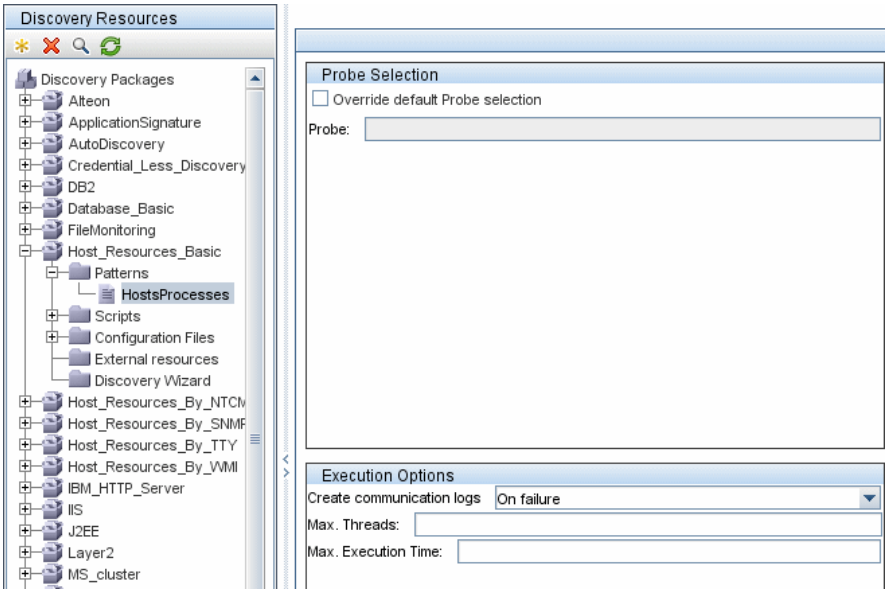
It can be very useful to record an entire execution, including all parameters, for example, when debugging and testing code.

The Execution Options functionality enables you to record the entire execution with all relevant variables. Furthermore, you can view extra debug information that is usually not printed to log files even at the debug level.

To activate the recording:

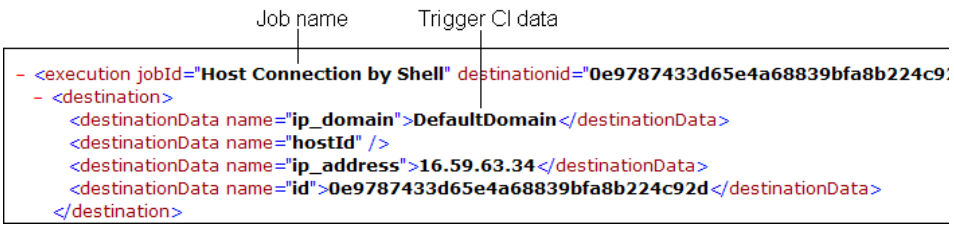
1 Access **Discovery** > **Run Discovery**. Right-click the job whose run you want to log and select **Edit pattern** to open the Manage Discovery Resources application.

2 Locate the **Execution Options** pane in the Pattern Management tab:

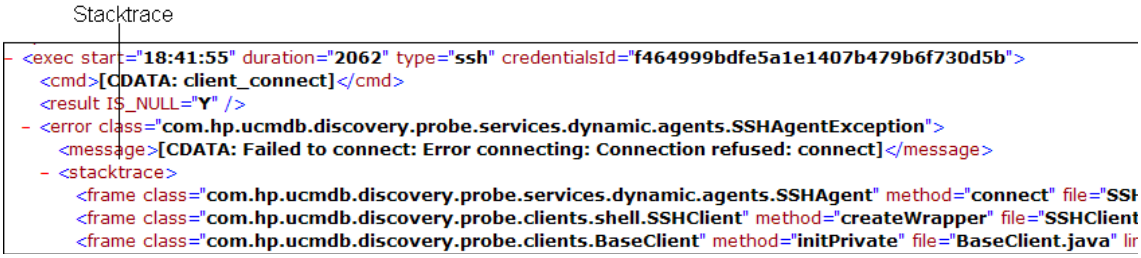


3 Change the **Create communication logs** box to **Always**. For details on setting logging options, see "Execution Options Pane" on page 190.

The following example is the XML log file that is created when the Host Connection by Shell job is run and the **Create communication logs** box is set to **Always** or **On Failure**:



The following example shows the message and stacktrace parameters:



Separating Patterns

Technically, an entire discovery could be defined in a single pattern. But good design demands that a complex system be separated into simpler, more manageable components.

The following are guidelines and best practices for dividing the DDM process:

- Discovery should be done in stages. Each stage should be represented by a pattern that should map an area or tier of the system. Patterns should rely on the previous stage or tier to be discovered, to continue discovery of the system. For example, Pattern A is triggered by an application server TQL result and maps the application server tier. As part of this mapping, a JDBC connection component is mapped. Pattern B registers a JDBC connection component as a trigger TQL and uses the results of pattern A to access the database tier (for example, through the JDBC URL attribute) and maps the database tier.
- **The two-phase connect paradigm**: Most systems require credentials to access their data. This means that a user/password combination needs to be tried against these systems. The DDM administrator supplies credentials information in a secure way to the system and can give several, prioritized login credentials. This is referred to as the **Protocol Dictionary**. If the system is not accessible (for whatever reason) there is no point in performing further discovery. If the connection is successful, there needs to be a way to indicate which credential set was successfully used, for future discovery access.

These two phases lead to a separation of the two patterns in the following cases:

- **Connection Pattern**: This is a pattern that accepts an initial trigger and looks for the existence of a remote agent on that trigger. It does so by trying all entries in the Protocol Dictionary which match this agent's type. If successful, this pattern provides as its result a remote agent CI (SNMP, WMI, and so on), which also points to the correct entry in the Protocol Dictionary for future connections. This agent CI is then part of a trigger for the content pattern.
- **Content Pattern**: This pattern's precondition is the successful connection of the previous pattern (preconditions specified by the TQLs). These types of patterns no longer need to look through all of the Protocol Dictionary since they have a way to obtain the correct credentials from the remote agent CI and use them to log in to the discovered system.

- Different scheduling considerations can also influence discovery division. For example, a system may only be queried during off hours, so even though it would make sense to join the pattern to the same pattern discovering another system, the different schedules mean that you need to create two patterns.
- Discovery of different management interfaces or technologies to discover the same system should be placed in separate patterns. This is so that you can activate the access method appropriate for each system or organization. For example, some organizations have WMI access to machines but do not have SNMP agents installed on them.

Job and Pattern XML Formats

Jobs and patterns are saved in the CMDB in an XML format. The job name appears in the job's XML file and is referred to by the **id** attribute. Each job has a corresponding pattern which is referred to by the **patternId** attribute.

Example of Job XML

A job name is **CPUs by TTY**. Its corresponding pattern is **TTY_HR_CPU**:

```
<job id="CPUs by TTY" displayName="CPUs by TTY">
      <patternId>TTY_HR_CPU</patternId>
      <triggers>
            <trigger>shell_on_unix</trigger>
      </triggers>
      <schedule>
            <offsetElem>0</offsetElem>
            <generatedFromElem>1</generatedFromElem>
            <schedularType>scheduler_simple_Schedule</schedularType>
            <timeExpElem>Days_1</timeExpElem>
      </schedule>
</job>
```

Example of Pattern XML

The pattern name is **TTY_HR_CPU**. The pattern input is defined by the <**inputClass**> tag. The pattern output is defined by the <**discoveredClasses**> tag.

```
<pattern id="TTY_HR_CPU" description="Discover CPU on Unix boxes."
schemaVersion="7.0">
 <discoveredClasses>
      <discoveredClass>container_f</discoveredClass>
      <discoveredClass>cpu</discoveredClass>
 </discoveredClasses>
 <parameters />
 <taskInfo className="appilog.collectors.services.dynamic.core.DynamicService">
      <destinationInfo className="appilog.collectors.tasks.BaseDestinationData">
           <destinationData
name="ip_address">${SOURCE.application_ip}</destinationData>
           <destinationData
name="Protocol">${SOURCE.root_class}</destinationData>
           <destinationData
name="credentialsId">${SOURCE.credentials_id}</destinationData>
           <destinationData
name="language">${SOURCE.language:NA}</destinationData>
           <destinationData
name="codepage">${SOURCE.codepage:NA}</destinationData>
      </destinationInfo>
      <params
className="appilog.collectors.services.dynamic.core.DynamicServiceParams">
           <script>TTY_HR_CPU_Lib.py</script>
           <script>TTY_HR_CPU.py</script>
      </params>
 </taskInfo>
 <inputClass>shell</inputClass>
</pattern>
```

11

Working with the HP Discovery and Dependency Mapping Web Service

This chapter explains how third-party or custom tools can use the HP Discovery and Dependency Mapping Web Service to manage Discovery and Dependency Mapping (DDM).

For full documentation on the available operations, see *HP Discovery and Dependency Mapping Schema Reference*. These files are located in the following folder:

\\<HP Business Availability Center root directory>\AppServer\webapps\site.war\amdocs\eng\doc_lib\Discovery_and_Dependency_Mapping\DDM_Schema\webframe.html

This chapter includes:

Concepts

Tasks

Reference

Conventions

This chapter uses the following conventions:

- This style, Element, indicates that an item is an entity in the database or an element defined in the schema, including structures passed to or returned by methods. Plain text indicates that the item is being discussed in a general context.
- DDM elements and method arguments are spelled in the case in which they are specified in the schema. This usually means that a class name or generic reference to an instance of the class is capitalized. An element or argument to a method is not capitalized. For example, a credential is an element of type Credential passed to a method.

The HP Discovery and Dependency Mapping Web Service

The HP Discovery and Dependency Mapping Web Service is an API used to integrate applications with HP Universal CMDB (UCMDB). The API provides methods to:

- Manage credentials: view, add, update, and remove
- Manage jobs: view status, activate, and deactivate
- Manage probe ranges: view. add, and update
- Manage triggers: Add or remove a trigger CI, and add, remove, or disable a trigger TQL
- View general data on domains and probes

Users of the HP Discovery and Dependency Mapping Web Service should be familiar with:

- The SOAP specification
- An object-oriented programming language such as C++, C# or Java
- HP Universal CMDB
- Discovery and Dependency Mapping

This section includes the following topics:

Permissions

The administrator provides login credentials for connecting with the Web service. The required credentials depend on whether you are using DDM with a standalone version of HP Universal CMDB or from within HP Business Availability Center.

When permissions are assigned through DDM, the permission levels are View, Update, and Execute. When they are assigned using the HP Business Availability Center, the levels are View and Update, where Update also includes Execution. To view the permissions required for each operation, see each operation's request documentation in the *HP Discovery and Dependency Mapping Schema Reference*.

To assign permissions:

➤ **DDM with HP Universal CMDB standalone.** Log in using the credentials of a DDM user who has been granted permissions on the discovery resources.

➤ **DDM embedded in HP Business Availability Center.** Log in using the credentials of a Business Availability Center user. The user must have been granted the relevant permissions on the UCMDB Web Service resource in Business Availability Center.

Call the Web Service

The HP Discovery and Dependency Mapping Web Service enables calling server-side methods using standard SOAP programming techniques. If the statement cannot be parsed or if there is a problem invoking the method, the API methods throw a SoapFault exception. When a SoapFault exception is thrown, the service populates one or more of the error message, error code, and exception message fields. If there is no error, the results of the invocation are returned.

SOAP programmers can access the WSDL at:

http://<server>[:port]/axis2/services/DiscoveryService?wsdl

The port specification is only necessary for non-standard installations. Consult your system administrator for the correct port number.

The URL for calling the service is:

http://<server>[:port]/axis2/services/DiscoveryService

Discovery and Dependency Mapping Methods

This section contains a list of the Web service operations and a brief summary of their use. For full documentation of the request and response for each operation, see *HP Discovery and Dependency Mapping Schema Reference*.

This section includes the following topics:

Managing Discovery Job Methods

- **activateJob**

  Activates the specified job.

- **deactivateJob**

  Deactivates the specified job.

- **dispatchAdHocJob**

  Dispatchs a job on the probe ad-hoc. The job must be active and contain the specified trigger CI.

- **getDiscoveryJobsNames**

  Returns the list of job names.

- **isJobActive**

  Checks whether the job is active.

Managing Trigger Methods

- **addTriggerCI**

  Adds a new trigger CI to the specified job.

- **addTriggerTQL**

  Adds a new trigger TQL to the specified job.

- **disableTriggerTQL**

  Prevents the TQL from triggering the job, but does not permanently remove it from the list of queries that trigger the job.

- **removeTriggerCI**

  Removes the specified CI from the list of CIs that trigger the job

- **removeTriggerTQL**

  Removes the specified TQL from the list of queries that trigger the job.

- **setTriggerTQLProbesLimit**

  Restrict the probes in which the TQL is active in the job to the specified list.

Domain and Probe Data Methods

- **getDomainType**

  Returns the domain type.

- **getDomainsNames**

  Returns the names of the current domains.

- **getProbeIPs**

  Returns the IP addresses of the specified probe.

- **getProbesNames**

  Returns the names of the probes in the specified domain.

- **getProbeScope**

  Returns the scope definition of the splecified probe.

- **isProbeConnected**

  Checks whether the specified probe is connected.

- **updateProbeScope**

  Sets the scope of the specified probe, overriding the existing scope.

Credentials Data Methods

- **addCredentialsEntry**

  Adds a credentials entry to the specified protocol for the spedified domain.

- **getCredentialsEntriesIDs**

  Returns the IDs of the credentials defined for the specified protocol.

- **getCredentialsEntry**

  Returns the credentials defined for the specified protocol. Encrypted attribues are returned empty.

- **removeCredentialsEntry**

  Removes the specified credentials from the protocol.

➤ **updateCredentialsEntry**

Sets new values for properties of the specified credentials entry.

Index

M

N

O

P

U

V

W