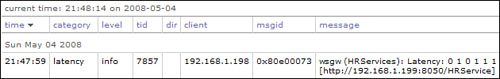
**Chapter 14: Logging and Monitoring**

Add a note hereCare and responsibility for DataPower SOA Appliances tend to cross operational boundaries, as they function at a network level, but perform security and application-specific roles. This separation of concerns is most prominent in the logging and monitoring configurations, as various enterprise teams require vastly different information about device health, processing actions, and performance metrics. This chapter discusses various options for both logging and monitoring schemes. We discuss the power of a publish-subscribe logging system, the configuration of various log delivery methods, and some common delineations between hardware-level and application-level events. This includes both shortcuts for common notifications and the creation of custom log categories and messages for maximum flexibility. On the monitoring side, we discuss the configuration of SNMP-based monitoring tools for maximum device visibility, suggested items for polling, and options for Web Services monitoring.

Add a note hereThe power of each of these systems allows for some amazing opportunities for real-time modification of running configurations and error remediation. Log recipients can send SOAP messages to update a service level agreement based on current conditions, dynamically provision new backend servers to a load balancer configuration, and enable or disable entire services. Monitoring stations can gather real-time Web service metrics to identify troublesome servers before they compromise guaranteed service levels, notify an operations team at the first sign of a bottleneck, and provide detailed chargeback statistics.

**Add a note here****Anatomy of a Log Message**

Add a note hereWe’ll get to the fun stuff—the capabilities of a flexible publish-subscribe logging system—shortly, but first we should discuss the anatomy of a log message. No matter which system generates a message, which protocol delivers it, or where it is read, each message has very similar fields. The first place you’re likely to see these fields is in the default log, which we discuss in detail later; for now, you can see some of the fields defined across the top, as shown in Figure 14-1.

[](javascript:PopImage('IMG_310','http://images.books24x7.com/bookimages/id_30903/14fig01_alt.jpg','630','100'))  
Add a note hereFigure 14-1: Viewing the fields available to a log message.

**Add a note here****Domain**

Add a note hereThe domain field represents the application domain from which the log message originated. Depending on your domain usage strategy, this can narrow down the event to a particular project team, application, or even user. On devices with multiple development environments (development, QA, pre-production), this field will likely specify the environment as well. This field is not visible in the figure; when not present, the message came from the current working domain.

**Add a note here****Timestamp**

Add a note hereThe event timestamp is used for sorting and correlation of log events. Each event is recorded as it enters the log subsystem and receives a timestamp. Depending on the configuration, this can be expressed in a syslog timestamp format, which provides only the date and time expressed to the second, or in a numeric format that is expressed as milliseconds since the January 1, 1970 (known as the epoch). The latter is obviously more precise, but is not supported by syslog and syslog-ng logging servers, nor is it available on the default log.

**Add a note here****Type**

Add a note hereThe event type signifies the category of message. These categories are preconfigured on the device but can be extended through the use of custom log categories, which we discuss later in this chapter. Some message types include aaa, which represents events dealing with authentication, authorization, and auditing decisions; mq, which deals with WebSphere MQ interaction; and wsrr, which notes interactions between DataPower and the WebSphere Services Registry and Repository. High-level services have their own categories (xmlfirewall, ws-proxy, and mpgw), as do low-level system operations (crypto, system, and xmlparse).

**Add a note here****Class**

Add a note hereIf the event originates with a configuration object on the device, the class field represents the type of object that creates the log message. This can be similar to the type, but it is limited to objects; if the message is logged by the system itself (for a hardware issue or something similar), this field will not be present. Example values for this field include wsgw (for a Web Service Proxy), xmlfirewall, xmlmgr (for an XML manager), and source-http (for an HTTP front-side handler).

**Add a note here****Object**

Add a note hereEvery object created on the device has a name, and this name populates the object field if it has sent a message to the log system. This is closely tied to the class field, as the name refers to an object of that type. For instance, there may be a domain, a Web Service Proxy, and a Processing Policy, and all of them are called HRServices. However, if the log entry contains wsgw (HRServices), then you know the message is from the Web Service Proxy.

**Add a note here****Priority**

Add a note herePriority is a particularly important log field, as this is the one most commonly used for message filtering. This is also sometimes called severity, and the eight possible values correspond closely to those defined by the syslog specification. In ascending order of severity, the levels are debug, information, notice, warning, error, critical, alert, and emergency. These priorities are relatively self-explanatory; some examples are Power Supply Failure at a critical level, File Downloaded at a notice level, and Unable to open URL <http://badurl/> at an error level.

**Add a note here****Transaction Type**

Add a note hereThe transaction type represents the current processing mode when the log message is created. If a service is currently processing a response rule, the transaction type would be response. Possible options are request, response, and error. If the event occurred outside the scope of a Processing Policy, this field will be absent.

**Add a note here****Transaction**

Add a note hereEach execution of a service is assigned a transaction identifier (ID), as are some internal processes. This ID is unique to the device and can be used for event correlation, as every log message generated during an execution will share the same transaction ID. The transaction field contains this value. Please note that these identifiers are not simply incremented by one for each transaction, nor are they necessarily sequential."Transaction Identifiers"

**Transaction Identifiers**

Add a note hereIt is important to note that transaction identifiers are not sequential, nor are they unique across device reboots. Log correlation should also depend on additional fields such as timestamps.

**Add a note here****Client IP**

Add a note hereThis field contains the IP address of the client that initiated the transaction. Though IP is not necessarily a sure indicator of a unique user, especially when segmented at an ISP level, the subnets in use can often provide some additional information about the caller and can usually differentiate one client from the next.

**Add a note here****Event Code**

Add a note hereThe event code is a unique identifier for a particular error message. There can be many instances of this message, but the event itself is the same. For instance, event code 0x00e4000a represents the error message No NTP servers are configured. These event codes can be viewed directly from the WebGUI by navigating to Administration→Debug→View List of Event Codes in the left-side menu. They are also grouped by category so that you can use filters with wildcard values to select whole ranges of events. For example, all the MQ-related event codes contain the prefix 0x0133.

**Add a note here****Message**

Add a note hereLast but not least, the text of the message itself is included in the message field. This might be generic, such as Power Supply #1 Failed, or it might be specific to a particular processing action, such as Schema validation failed: cvc-simple-type 1: element product-id value ‘1234’is not a valid instance of the element type. This is the least structured of the fields and can contain any text, including messages custom-defined during development.

**Add a note here****Log Message Example**

Add a note hereHere is an example log message with each of the preceding fields defined:

Add a note hereSun May 04 2008 15:55:35 [HRServices][multistep][info] wsgw(HRServices):

tid(20018)[request][192.168.1.198]: rule (HRServices\_default\_request-rule):

#2 results: 'generated from INPUT results stored in OUTPUT' completed ok.

Add a note hereEach field can be separated out for explanation, as seen in Table 14-1.

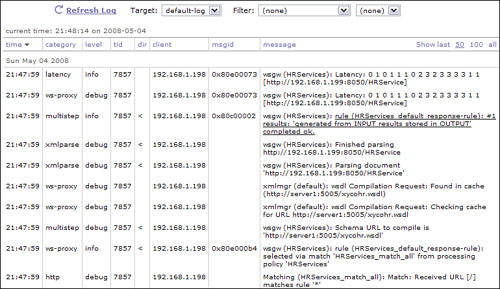
| Add a note hereTable 14-1: Parsing a Log Message into Its Component Fields  [[http://www.books24x7.com/images/b24-bluearrow.gif](http://www.books24x7.com/outputobject.asp?bookid=30903&chunkid=452692263&objectid=ch14table01&objecttype=spreadsheet)Open table as spreadsheet](http://www.books24x7.com/outputobject.asp?bookid=30903&chunkid=452692263&objectid=ch14table01&objecttype=spreadsheet) | |
| --- | --- |
| **Add a note hereField** | **Add a note hereValue** |
| Add a note hereTimestamp | Add a note hereSun May 04 2008 15:55:35 |
| Add a note hereDomain | Add a note hereHRServices |
| Add a note hereType | Add a note heremultistep |
| Add a note herePriority | Add a note hereinfo |
| Add a note hereClass | Add a note herewsgw |
| Add a note hereObject | Add a note hereHRServices |
| Add a note hereTransaction | Add a note here20018 |
| Add a note hereTransaction Type | Add a note hererequest |
| Add a note hereClient IP | Add a note here192.168.1.198 |
| Add a note hereMessage | Add a note hererule (HRServices\_default\_request-rule):#2 results: ‘generated from INPUT results stored in OUTPUT’ completed ok. |

**Default Logs**

Add a note hereEach domain has a default log defined that provides event information from the WebGUI. To access the current log, click on the View Logs link from the Control Panel, as seen in Figure 14-2, or navigate to Status→View Logs→System Logs from the left-side panel.

[](javascript:PopImage('IMG_311','http://images.books24x7.com/bookimages/id_30903/14fig02.jpg','484','143'))  
Add a note hereFigure 14-2: Accessing the default log from the Control Panel.

Add a note hereEither method of navigation takes you to the system log page, which by default shows the last 50 entries in the default log. The interface enables you to filter the entries by category and/or priority, in order to limit the number of lines. These options are available from the drop-down menus at the top of the screen, as shown in Figure 14-3.

[](javascript:PopImage('IMG_312','http://images.books24x7.com/bookimages/id_30903/14fig03_alt.jpg','770','445'))  
Add a note hereFigure 14-3: Viewing the default log.

Add a note hereFor additional filtering, you can click a transaction number, client IP address, or event code. Each of these opens a new window with messages related to the selected value; for example, clicking a transaction ID displays only messages from that transaction. To view more entries on a single page, you can choose to show the last 100 entries or to show all entries since the last log rotation. Click one of these options in the upper-right to see more; just because you don’t see a message in the last 50 entries doesn’t mean it is gone.

Add a note hereThe [priority](http://www.books24x7.com/assetviewer.aspx?bkid=30903&destid=2508#2508) of messages gathered by the default log can be modified from the Troubleshooting screen. From the Control Panel, click the Troubleshooting icon seen in Figure 14-2. From the Logging section shown in Figure 14-4, choose a new [priority](http://www.books24x7.com/assetviewer.aspx?bkid=30903&destid=2508#2508) level and click Set Log Level. In development environments, it is fine to go all the way down to debug-level logging; in production environments, however, default logging should be set to a warning or error level to avoid any impact on performance.

  
Add a note hereFigure 14-4: Changing the default log level.

Add a note hereDuring development or while attempting to debug a series of transactions, it is often helpful to click the View System Logs link from this page. As opposed to viewing the log from the Troubleshooting page or Navigation menu, this opens the log page in a new window, enabling you to modify configurations in one window and refresh the logs in another. This log is also available from the file system; the logtemp: directory contains the default log in both text and XML formats.

Add a note hereAll services and many other objects have a View Log link at the top of their configuration screen. Each of these links is a simple filter of the same default log file, extracting only messages that pertain to that object."System Logs in the Default Domain"

**System Logs in the Default Domain**

Add a note hereLog configuration in the default domain is somewhat omniscient, in that the messages logged there are gathered from all domains on the device as well as device-specific events. For example, if the log is configured to record all information level events, then events occurring at that level (or higher) in all domains will be inserted into the default log.

Add a note hereThe default domain often captures log events for other domains that are not in the logs for that domain. These are usually low-level network events, such as SSL failures and Tivoli Access Manager configuration messages.

Add a note hereIn the end, the default system log is special only in that it is predefined and its behavior (aside from filtering on priority) cannot be modified. In reality, it is just one of many possible targets for log messages on (or off) the device. We continue with a discussion of how to enable additional log destinations.

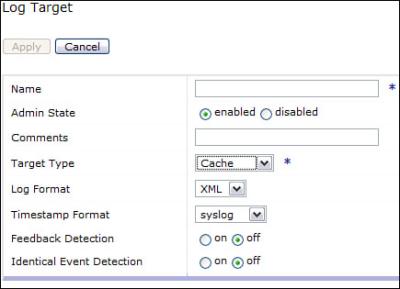
## Log Targets

Add a note hereThe logging subsystem on DataPower SOA Appliances is based on a publish-subscribe concept that enables distribution of selected messages to various protocols and destinations; message selection is a user-configurable process that can select broad message categories and priorities but can also be very granular if necessary. Event messages can be generated by anything on the appliance, including a hardware-level device monitor, a service processing an application transaction, or a stylesheet logging a custom error message. Obviously, some of these items are more critical than others, and some may only be of interest to a select group of users; some messages may not be of any interest at all. The goal of a reasonable logging configuration is to make sure that appropriate events are being received by the team that would be interested in those messages, are being delivered using the most suitable protocol, and are not impacting the flow of production traffic. Get those messages to the people that need to see them! This can initiate active configuration management or error remediation, depending upon what information is being delivered.

### Add a note hereTarget Fields

Add a note hereDirecting a subset of all log messages to a particular destination/protocol is accomplished through the use of log targets. A log target describes both the endpoint to which the logs will be sent and the events that should be selected for sending. They include both protocol-level specifications (hostnames for remote log servers) and subscriptions to events in certain categories at user-defined levels. These subscriptions can also include filters for granular selection of events logged by specific objects.

Add a note hereTo create a new log target in a domain, navigate to Administration→Miscellaneous→Manage Log Targets in the left-side menu. You’ll notice there is always one predefined log target (default-log) that populates the system log; this target can be modified only in terms of the priority of events gathered. Click Add to configure a new log target, resulting in the log target configuration page seen in Figure 14-5.

[](javascript:PopImage('IMG_314','http://images.books24x7.com/bookimages/id_30903/14fig05.jpg','435','315'))  
Add a note hereFigure 14-5: Defining a new log target.

Add a note hereWe’ll soon define the events that will be subscribed to by this log target, but first we need to make some basic format choices and define a destination. Choose a log format from the drop-down menu; some target types require a particular format (XML for a SOAP target), but many targets (File, for instance) will accept various formats.

#### Log Formats

Add a note hereThe default log format, XML, places each log entry into an <entry/> node containing elements and attributes with all the log data. The CSV (comma separated values) format is similar but uses commas as delimiters between each field. The text format outputs each entry in a one-line string of text without any delimiting structure. The CBE (Common Base Event) format is a specification-defined XML standard designed to be common across various hardware and software products. Lastly, the Raw format outputs the message field in each entry without any of the other fields (timestamp, category, priority, and so on).

#### Common Fields

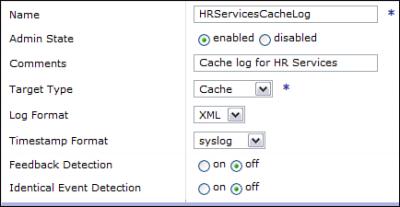
Add a note hereThe timestamp format (syslog or numeric) defines the version of the timestamp used for each message; you must choose syslog for a remote syslog or syslog-ng server, and can use numeric for all others. Feedback detection prevents any loops caused by messages from the log system by suppressing all events related to the logs, while identical event detection prevents an identical event from occurring for a user-defined time period. Some target types also allow a backup log; if the logs cannot reach their intended destination, the backup target will receive any ensuing messages.

### Add a note hereTarget Types

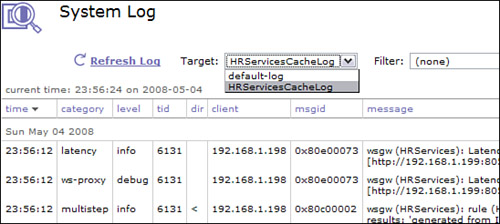
Add a note hereEvery target requires a name and a type. The type represents the log storage format or protocol over which events are sent; choose one from the drop-down menu. Depending on the target chosen, additional options are provided. This is an opportunity to integrate DataPower with the logging infrastructure chosen by your enterprise, be it syslog, SNMP, or a remote drive mounted using NFS. The following sections discuss the details of each option.

#### Cache Target

Add a note hereThe cache option is used for short-term viewing of logs on the device. Upon creation of the log target, as shown in Figure 14-6, entries are collected in memory.

[](javascript:PopImage('IMG_315','http://images.books24x7.com/bookimages/id_30903/14fig06.jpg','434','225'))  
Add a note hereFigure 14-6: Creating a cache-based log target.

Add a note hereThe messages gathered by a cache-based log target are available for viewing from the system logs page. Click View Logs from the Control Panel and then choose the target name from the drop-down menu, as seen in Figure 14-7.

[](javascript:PopImage('IMG_316','http://images.books24x7.com/bookimages/id_30903/14fig07_alt.jpg','548','246'))  
Add a note hereFigure 14-7: Viewing a cache log target.

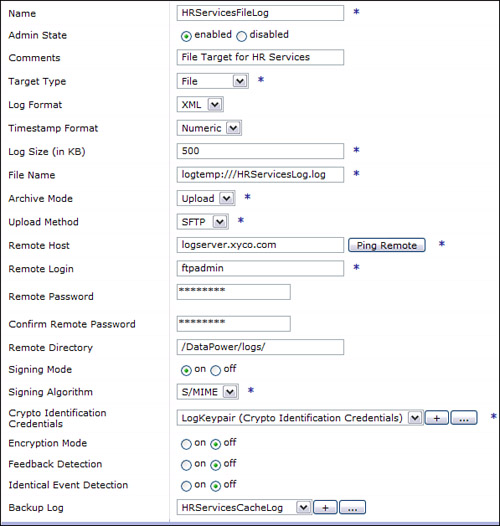
Add a note hereAfter you view the messages from the log page, the cache on those messages expires. That is, the next time you open this page, these events no longer appear. Even clicking Refresh Log makes the displayed events disappear. For this reason, the cache target should only be used for quick peeks at short-term events.

#### Console Target

Add a note hereThe console log type is for short-term viewing, similar to the cache target, but messages are written to the Command Line Interface (CLI), rather than to an in-memory cache. To view the messages generated by this target type, you must use SSH (or TELNET) to log in to the CLI for the device. Switch to the domain in which the target is defined; any applicable log messages will flow across the screen as traffic flows through the device.

#### File Target

Add a note hereA file log target keeps event messages in a file on the local device. This file grows as log messages are added. When the file size reaches a user-defined threshold, it is either rotated or uploaded to a remote server, as defined by an administrator. Figure 14-8 shows the full configuration; depending on the options chosen, additional fields need to be completed.

[](javascript:PopImage('IMG_317','http://images.books24x7.com/bookimages/id_30903/14fig08_alt.jpg','597','628'))  
Add a note hereFigure 14-8: Defining a local file for log messages.

Add a note hereFile targets require a name and location for the local file. These files can be stored in either the logtemp: or logstore: directories. Logtemp files are available from the file system but are technically stored in volatile memory and will not persist during a device reboot; logstore files will stay on the device even after it is rebooted. Files written to the hard drive or flash drive on devices with persistent storage will also survive a reboot.

Add a note hereThese targets also require a size at which the file will be acted upon; the default is 500KB, but it can range from 100KB to 5MB. When the log file reaches the configured size, it will be archived by either rotating to a new file or being uploaded to a remote server. Choosing rotate allows you to specify how many times the file rotates to an archived version (in the same directory) before the oldest files begin to be deleted. Choosing upload asks for additional protocol details.

Add a note hereLog file uploading can be accomplished through a number of different protocols. The FTP and Secure FTP (SFTP) options require a remote FTP host, login, password, and directory. The Secure Copy (SCP) option requires the same fields and uses SSH as the underlying protocol to copy files remotely. The SMTP server takes sender and destination email addresses, and an SMTP server and client domain.

Add a note hereBefore logs are uploaded from the device, they can be signed and/or encrypted. Turn on Signing Mode or Encryption Mode to enable these options; they each require a signing algorithm and cryptographic material. The example in Figure 14-8 shows the identification credentials used for file signing.

#### NFS Target

Add a note hereAn NFS target writes log events to a file on a remotely hosted NFS mount. These targets work similarly to file targets, including the option to rotate or upload files when they reach a size threshold. Instead of a local directory, this target requires a static NFS mount to be defined. Because NFS requires a network connection and may introduce some latency, it also allows a Rate Limit to restrict the number of transferred events per second.

#### SMTP Target

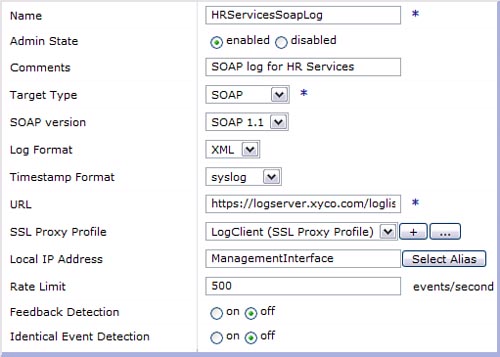
Add a note hereThe SMTP target allows an email to be sent with each log message. This obviously requires significant overhead and could produce a large number of emails when used more than sparingly. However, this option is commonly used for critical events that require on-call personnel to be immediately notified. Email targets require an SMTP server and client domain, as well as sender and receiver email addresses.

#### SNMP Target

Add a note hereAn SNMP target uses preconfigured SNMP settings to deliver log events as Notification traps. An SNMP server such as HP OpenView or Tivoli NetView® can configure a trap listener to receive these notifications. For access to the appliance MIBs and assistance in configuring the SNMP settings, see the “[Device Monitoring](http://www.books24x7.com/assetviewer.aspx?bkid=30903&destid=2684#2684)” section later in this chapter.

#### SOAP Target

Add a note hereSystem logs can be sent to a remote log server using a predefined SOAP interface. The example in Figure 14-9 shows events being routed to an HTTPS URL and includes an SSL proxy profile for securing the transport.

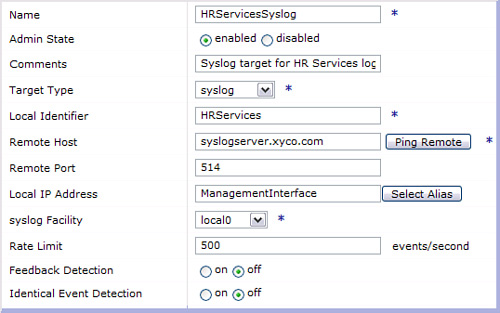
[](javascript:PopImage('IMG_318','http://images.books24x7.com/bookimages/id_30903/14fig09_alt.jpg','511','365'))  
Add a note hereFigure 14-9: Configuring a SOAP log target.

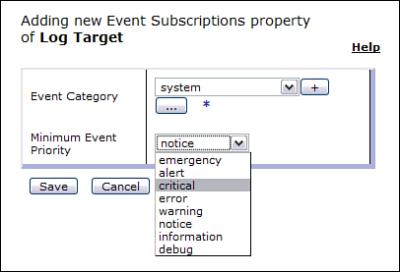
Add a note hereIn order to make creation of a SOAP listener as easy as possible, DataPower provides a WSDL file that describes the service interface and a schema file that describes the log structure. Both of these files (log-soap.wsdl and log-soap.xsd) are available from the store: directory.

#### Syslog/Syslog-ng Targets

Add a note hereThe most common integration between DataPower and a logging infrastructure is the use of syslog or syslog-ng to deliver application-level messages to a common logging server, where events from multiple systems are parsed, correlated, and filtered for importance. Syslog uses the UDP protocol for fast, efficient delivery of messages, while syslog-ng uses TCP for reliability; this allows syslog-ng to use SSL for secure transport.

Add a note hereThe example in Figure 14-10 shows the definition of a syslog target that delivers event messages to a remote server.

[](javascript:PopImage('IMG_319','http://images.books24x7.com/bookimages/id_30903/14fig10_alt.jpg','538','337'))  
Add a note hereFigure 14-10: Configuring a syslog target.

[](javascript:PopImage('IMG_320','http://images.books24x7.com/bookimages/id_30903/14fig11.jpg','406','277'))  
Add a note hereFigure 14-11: Subscribing to all critical system events.

### Add a note hereEvent Subscriptions

Add a note hereLog targets have only two requirements: a fully configured target (on the Main tab) and at least one subscription (on the Event Subscriptions tab). Now that we’ve defined a target protocol and destination for the delivery of log messages, we need to configure DataPower to select the appropriate messages for that target. The Event Subscriptions tab defines all messages that will be initially selected for delivery; these events can later be filtered using the other tabs. By default, there are no subscriptions defined; click Add to create a new subscription. In the pop-up that appears, select an event category and a minimum event priority, as shown in Figure 14-11. In this example, all system events at a critical level or above are gathered by this log target.

Add a note hereEach saved event subscription will be added to the list so that the end result is the union of all applicable messages. In Figure 14-12, multiple subscriptions create a policy that might be used for an operations team monitoring device-level errors. For a more generic log, the event category all can be used to retrieve all messages at a given priority.

[](javascript:PopImage('IMG_321','http://images.books24x7.com/bookimages/id_30903/14fig12.jpg','443','123'))  
Add a note hereFigure 14-12: Compiling multiple subscriptions.

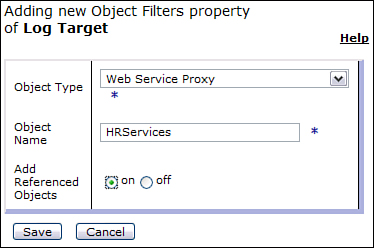
Add a note hereNow that we’ve defined the categories and priorities for subscription, we may want to filter these messages to a more specific set. Depending on the options chosen, these targets may produce a large number of error messages, many of which are of no interest to the team that is monitoring the destination log server."Subscriptions Are Required"

**Subscriptions Are Required**

Add a note hereThough the categories and priorities chosen for a particular logging mechanism can be modest, there *must* be at least one subscription defined for each target. This is a required field, and you will receive zero messages at your log server if no subscriptions have been created. There is no WebGUI warning for an empty subscription list, so keep this in mind if your log is unexpectedly empty.

### Add a note hereObject Filters

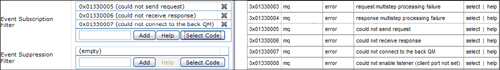
Add a note hereMessages selected by a log target subscription can be filtered by the objects that created them; for example, you can select only events from a Multi-Protocol Gateway named MarketingService. Click the Object Filters tab to begin configuring the objects you care to receive information about. Click Add to create a new filter selection. Objects are categorized by type; you can filter on high-level objects such as XML Firewalls and Web service proxies, or low-level objects, such as a particular AAA policy. Choose the type of object and type the name of the particular object on which to filter. If you choose a category and leave the object name blank, all objects of that type will be selected. Keep in mind that many messages written to the logs for a given service are actually generated by lower-level objects contained within those services. If you turn on Add Referenced Objects, every lower-level object referenced by the named object will also be included in log collection. In the example shown in Figure 14-13, log messages are limited to those generated by the HRServices Web Service Proxy or objects referenced by that proxy. This log target may be useful for the application development team that created this particular service.

  
Add a note hereFigure 14-13: Filtering on a specific object.

Add a note hereObject filters are entirely optional and limit the messages being collected. For a broader range of log events, leave the object filters tab empty.

### Add a note hereEvent Filters

Add a note hereFor the ultimate in log granularity, messages can be filtered by their event codes. This filtering can be either negative or positive; that is, you can select which codes you would like to see, or you can select which codes you would like to suppress. Click the Event Filters tab to view the current filters. To add an event code to the list, click Select Code to view a sortable/selectable list of codes that can be added to the filter. Often, the easiest way to choose the actual codes generated by your service configurations is to view the default system log at a debug level and select which events are important. Copy and paste the event code from the log to the filter to prevent any potential typographical errors. In the example in Figure 14-14, MQ connectivity errors are being selected for a log target that may be appropriate for MQ administrators. Alternatively, you could specify the single filter 0x0133\*, where the asterisk acts as a wildcard to match all MQ events at once.

[](javascript:PopImage('IMG_323','http://images.books24x7.com/bookimages/id_30903/14fig14_alt.jpg','1016','143'))  
Add a note hereFigure 14-14: Selecting specific event codes for a log target.

Add a note hereThe need to suppress events is readily apparent if there are too many log messages of one particular type. These events may be repetitive due to constant polling, such as an error that an MQ queue manager is down in a development environment. The event selection process is exactly the same; click Select Code and choose your events, or copy the event codes from an existing set of logs. Keep in mind that event suppression may complicate problem determination and should be used judiciously; consider using Identical Event Detection (on the Main tab) instead."Selective Suppression"

**Selective Suppression**

Add a note hereOne event that is commonly suppressed in development environments is the message that “Power supply #2 has failed.” This message can be caused by the second power supply being unplugged for cost savings or simple oversight. However, even a development box is important, and physical access to the device (often in a remote datacenter) is hard to come by. Instead of suppressing this message, just plug in the second power supply.

**Special Log Objects**

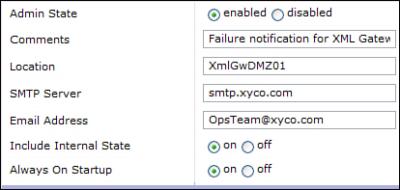
Add a note hereThere are a few special log objects that simplify selection and delivery of important messages. They are configured using wizards to define the message destinations.

**Add a note here****Email Pager**

Add a note hereAn email pager is specifically configured to send all domain events (or device events if created in the default domain) that have a priority defined as critical or above (critical, alert, and emergency) to an email address. Navigate to the email pager configuration at Administration→Miscellaneous→New Email Pager in the left menu. The simple wizard asks for a log target name, some optional comments, an SMTP server hostname, and a destination email address. This wizard generates a new log target subscribing to all critical events in this domain and delivering them to the defined email address.

**Add a note here****Failure Notification**

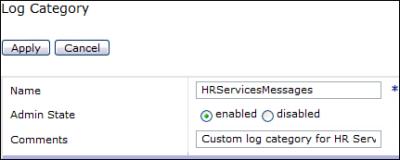
Add a note hereA failure notification is generated when a device experiences an unscheduled outage. As the device obviously cannot send a notification while it is down, the message (if configured) is sent via email upon resumption of service. Power failures (affecting both power supplies), device throttle/reboots due to memory thresholds, and accidental power-downs are examples of potential notification triggers. Failure notification must be configured in the default domain, as it deals with a system-level problem. Navigate to Objects→System→Failure Notification to configure the notification destination. Fill in the appropriate email address and server for failure notification, as seen in Figure 14-15.

[](javascript:PopImage('IMG_324','http://images.books24x7.com/bookimages/id_30903/14fig15.jpg','414','197'))  
Add a note hereFigure 14-15: Defining a failure notification.

Add a note hereThe location field defines the device in question and is inserted into the subject line of the outgoing email. You can also include an email report of the current state of the device or choose to send a notification on every startup (even during planned reboots) by selecting the appropriate radio buttons.

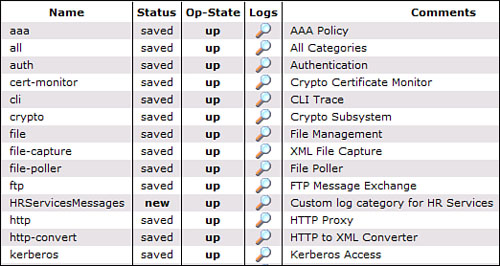
**Custom Log Categories**

Add a note hereTo this point, all the log categories and event priorities we’ve discussed have been enumerated by DataPower and have correlated directly with syslog-style severities and built-in objects/processes on the device. However, you may want to institute custom log categories for easier filtering or for correlation of device logs with application logs. These custom categories are often created for custom logging within a service defined for a particular application, which we discuss shortly. To create a new custom log category, navigate to Administration→Miscellaneous→Configure Log Categories in the left menu. Click on Add to create a new category, and then provide a name as shown in Figure 14-16.

[](javascript:PopImage('IMG_325','http://images.books24x7.com/bookimages/id_30903/14fig16.jpg','431','173'))  
Add a note hereFigure 14-16: Defining a custom log category.

Add a note hereThere are a couple of caveats to be aware of when using custom log categories. A custom log category is available only within the domain in which it is created, and that same category name cannot be used in another domain. That means that custom category names must be unique across the entire device.

Add a note hereThe example in Figure 14-17 shows the categories for an HRServices domain, along with the new custom category HRServicesMessages.

[](javascript:PopImage('IMG_326','http://images.books24x7.com/bookimages/id_30903/14fig17_alt.jpg','522','278'))  
Add a note hereFigure 14-17: Viewing built-in and custom log categories.

Add a note hereIf all events generated by the device are already within a given category, why bother defining a custom category? You may want to have log messages that reference application data specific to the transaction, rather than simply listing DataPower processing steps/errors. In the [next section](http://www.books24x7.com/assetviewer.aspx?bkid=30903&destid=2648#2648), we discuss how to define those custom log messages.

**Custom Log Messages**

Add a note hereMost information regarding DataPower custom development is discussed in [Part VI](http://www.books24x7.com/assetviewer.aspx?bkid=30903&destid=4077#4077), “[DataPower Development](http://www.books24x7.com/assetviewer.aspx?bkid=30903&destid=4077" \l "4077" \t "_parent).” However, one point bears repeating, as creation of custom log messages requires a stylesheet that takes advantage of the appliance’s XSLT extension functions.

Add a note hereDataPower has overloaded the xsl:message element to take additional attributes. As seen in Listing 14-1, dp:type chooses a log category and dp:priority adds a message severity. In the example, a date is written to a custom log message at an information priority in the HRServicesMessages category.

Add a note hereListing 14-1: Using Extension Elements to Define a Custom Log Message

Add a note here<xsl:message dp:type="HRServicesMessages" dp:priority="info">

Requested holiday: <xsl:value-of select="//requestedDate"/>

</xsl:message>

Add a note hereThese log messages do not have to use custom log categories or user-defined priorities; any xsl:message elements are written to the DataPower logs anyway, but the customization is powerful. Both attributes (if used) must be defined in the DataPower namespace (<http://www.datapower.com/extensions>). More information on programming in a DataPower environment is provided in [Part VI](http://www.books24x7.com/assetviewer.aspx?bkid=30903&destid=4077#4077).

## Separation of Concerns (Division of Log Traffic)

Add a note hereNow that you’ve seen the various options for configuring new log targets, let’s put that knowledge to work. First, you need to define various targets for the disparate enterprise groups that may need to receive information from production devices. Each group should review the log categories, event codes, and delivery protocol options that are available, as well as the logs generated during development and QA. Customize the following suggestions with your own additions; for example, if your services require connectivity to a messaging system (such as MQ, Tibco EMS, or WebSphere JMS), your networking and/or messaging administrators may desire additional logs in those areas. Keep in mind that any target defined in the default directory gathers subscriptions (appropriately filtered) from all the other domains; targets defined within domains are restricted to that domain’s messages.

### Add a note hereOperations Targets

Add a note hereLog targets created for a production operations department generally deal with the health of the device from a hardware perspective. These departments often have 24/7 device/server monitoring, so these issues are most likely to be noticed immediately and escalated to internal teams or IBM support. The targets are usually located in the default domain and often contain some version of the following:

* Add a note hereSNMP target subscribing to all events at a critical level
* Add a note hereSNMP target subscribing to network events at an error level (optional)
* Add a note hereSMTP target subscribing to all events at a critical level
* Add a note hereFailure notification (SMTP) with internal state and “Always on Startup” turned on

Add a note hereThe critical event threshold also gathers alert and emergency messages for complete “serious problem” coverage. The network-focused target is generally applicable only when the operations team is also in charge of network monitoring, rather than being solely hardware-focused. If this is required, it can be added as an additional subscription to the all-critical target. The identical SMTP target is used as a backup, in case there are SNMP connectivity issues. The SMTP failure notification provides coverage for unexpected outages (for which there will obviously be no warning).

### Add a note hereApplication Targets

Add a note hereLogs directed at application development/administrative groups are often the most disparate, as they depend considerably on the processing requirements of each application. These are usually defined at a domain level to gather information only about a set of services within a project deployment. The vast majority of application level log targets are either syslog/syslog-ng targets for real-time collection or file-based targets for delayed upload via ftp or scp. A sample set of targets could include the following:

* Add a note hereSyslog-ng target using SSL transport and subscribing to all events at an error level
* Add a note hereIndividual file targets uploaded via ftp and subscribing to error events filtered on a specific service
* Add a note hereSMTP target subscribing to error level events and filtered to event codes signifying loss of connectivity to a critical business system (queue manager, SQL data source, and so on).

Add a note hereIf your enterprise doesn’t have a syslog server in place, the first target can be replaced by a file target being uploaded to a central log repository, or a SOAP target that can send real-time logs to a remote listener that logs to a file or database. The individual file targets can be uploaded to the server hosting a backend application (service), to be used for later log correlation. These individual targets can be customized to include additional subscriptions (at perhaps a warning or information level) in specific log categories. It also enables custom messages in user-defined log categories to be logged separately from device-generated messages.

### Add a note hereSecurity Targets

Add a note hereSecurity is a broad category, as application security policies may be different from IT security policies. Desired log messages for an enterprise security group can vary widely depending on the sphere of influence of the team. The method of delivery is also less well-defined than for other groups, as security monitoring is less often standardized to syslog or SNMP. The exact definition of event subscriptions and log destinations is entirely up to you, but some common realms of interest include the following:

* Add a note hereSOAP target to an intrusion detection server (IDS), subscribing to error events and filtered by IDS-related event codes
* Add a note hereSyslog target subscribing to AAA events at a warning level
* Add a note hereSMTP target subscribing to cert-monitor objects at a warning level
* Add a note hereFile target signed and uploaded via FTP, subscribing to auth, cli, and file events at an information level

Add a note hereThe first two targets check for intrusions or invalid service authentication or authorization. The cert-monitor target alerts an administrator by email when a certificate on the device is moving toward expiration; configuration of how often and how early these expirations are checked is available from the Objects→Crypto→Crypto Certificate Monitor in the left-side menu. Last, the file target becomes an audit log for a particular domain, so that administrative modifications will generate a record for nonrepudiation purposes. It is important to create separate administrative user accounts to ensure useful audit information; don’t share the admin account or you’ll lose your oversight!

**Transaction Logging**

Add a note hereTo this point we have been discussing only event logging—that is, events that happen on the device either during processing or because of some internal process or hardware status change. This is very different from transaction logging, which takes a copy of a particular message payload and sends it to a log destination for audit purposes. Adding transaction logging mechanisms can double the amount of network traffic, as each message flows to the backend as well as to the log; keep this in mind when defining your logging requirements. There are several different methods for performing transaction logging, all of which operate on an existing Processing Policy within a service. For a full explanation of how actions are added to processing rules and configured, see [Part III](http://www.books24x7.com/assetviewer.aspx?bkid=30903&destid=754#754), “[DataPower Services](http://www.books24x7.com/assetviewer.aspx?bkid=30903&destid=754" \l "754" \t "_parent).” For now, we can discuss the various logging options."Privacy Concerns"

**Privacy Concerns**

Add a note hereWhen undertaking any form of transaction logging, be aware of what sort of personal data will be recorded. Depending on the privacy and security laws surrounding your particular industry and the contents of incoming messages, persisting this information may result in lawsuits or significant fines. Check the HIPAA, SOx, PCI[[1](http://www.books24x7.com/assetviewer.aspx?bookid=30903&chunkid=789500974&noteMenuToggle=0&hitSectionMenuToggle=0&leftMenuState=1" \l "ftn.ch14fn01)], or other compliance regulations before configuring this type of logging.

**Add a note here****Log Action**

Add a note hereThe Log action is available from the Advanced Action icon. This action requires a destination URL and will send the input context to the defined endpoint. The message is wrapped in the same schema used for the SOAP log target; it contains all the standard log metadata fields (timestamp, category, client IP, and so on), and the payload is inserted into the message field. The category and priority applied are configurable from the Log action interface. The URL must be an HTTP or HTTPS endpoint.

**Add a note here****Results Action**

Add a note hereThe Results and Results-Async actions, when inserted within a processing rule and defined with a URL, send the current message payload to a defined URL. These actions, as opposed to the Log action, have no wrappers or additional metadata (SOAP envelope, timestamp, and so on) and simply pass the entire message. The URL can use any of the protocols supported by DataPower (such as HTTPS, MQ, JMS); more information on generating these URLs is available in [Chapter 9](http://www.books24x7.com/assetviewer.aspx?bkid=30903&destid=1276#1276), [“Multi-Protocol Gateway.”](http://www.books24x7.com/assetviewer.aspx?bkid=30903&destid=1276#1276) The Results-Async action performs an asynchronous fire-and-forget to the endpoint, while the Results action wait for a valid response.

Add a note hereUsing the Results action with a destination for logging is different from the common practice of ending a processing rule with an empty Results action. In the latter case, the action simply provides a method for getting the expected output into the OUTPUT context for delivery to the service backend (on a request) or client (on a response). As this is the most common appearance of the Results action, it is sometimes disconcerting (but perfectly reasonable) to have multiple Results actions spread throughout the processing rule, each sending its current context to a remote destination. They can even be adjacent to one another, stacked to send to multiple targets.

**Add a note here****Custom Transaction Log**

Add a note hereA custom transaction log consists of an XSL stylesheet that gathers transaction variables and payload information and creates a message for delivery to a log server. Using DataPower extension functions to write custom stylesheets will be discussed in the development chapters; building a transaction log payload is as simple as assembling an XML message containing the desired data. This can contain the transaction ID, client IP, client credentials, service/operation being called, incoming URLs, and many other pieces of metadata—this provides the most flexibility in defining your own logging schema to match your current infrastructure.

Add a note hereOnce the transaction log message has been defined, there are a few options for message delivery. From within the custom stylesheet, you can use the dp:url-open extension element to open a connection to a logging server via any supported protocol. Examples of using dp:url-open are provided in the DataPower development chapters. Similarly, you can output the message to the next step in the processing rule, and then use a Results or Results-Async action to perform the same sort of delivery. If you require more information than the standard event log metadata but would like some portion of the payload delivered through a log target, a custom stylesheet can be used to reduce the message to a size that is suitable for an event log; the entire log message must be less than 2KB. Use the “[Custom Log Messages](http://www.books24x7.com/assetviewer.aspx?bkid=30903&destid=2648#2648)” section earlier in this chapter to submit the custom message to the logging system.

Add a note here[[1](http://www.books24x7.com/assetviewer.aspx?bookid=30903&chunkid=789500974&noteMenuToggle=0&hitSectionMenuToggle=0&leftMenuState=1" \l "ch14fn01)]HIPAA, Sarbanes-Oxley, and PCI are regulations for the healthcare, public corporation, and credit card industries.

**Device Monitoring**

Add a note hereIn earlier chapters, we’ve discussed a few different ways to retrieve data from the appliance using common interfaces. In the WebGUI, the Status menu contains most of the information that would be used for monitoring; in the CLI and the XML Management Interface, the show command and the get-status operation provide the same information. However, most common enterprise monitoring solutions use the SNMP[[2](http://www.books24x7.com/assetviewer.aspx?bookid=30903&chunkid=660434808&noteMenuToggle=0&hitSectionMenuToggle=0&leftMenuState=1" \l "ftn.ch14fn02)] standard to retrieve information from network devices. DataPower appliances provide an SNMP interface that can be used to poll the device for data or receive traps generated by the device. Any event message on the device can be sent to an SNMP listener via a log target.

Add a note hereAt a high level, monitoring tools that support SNMP poll the appliance (and other devices on the network) for key indicators of health using strictly defined GET[[3](http://www.books24x7.com/assetviewer.aspx?bookid=30903&chunkid=660434808&noteMenuToggle=0&hitSectionMenuToggle=0&leftMenuState=1" \l "ftn.ch14fn03)] requests. Potential status data points are referenced using a complex object identifier (OID); DataPower will recognize incoming OID requests and respond with the appropriate values. As a corollary, enterprise monitoring tools can also listen for “traps,” which are messages that can be arbitrarily sent by devices to provide additional information; these traps can be filtered and acted on.

Add a note herePrecise configuration depends upon the SNMP versions supported by your enterprise monitoring tools. DataPower supports all major versions, including v1, v2c, and v3.

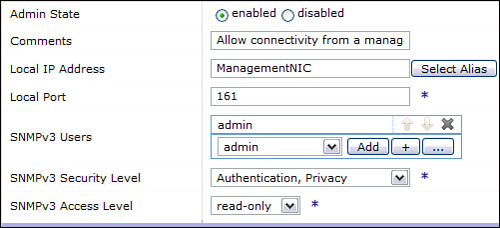
**Add a note here****Where Are the MIBs?**

Add a note hereThis is likely the first question that will be asked by any operations personnel trying to monitor the appliances. To download the MIB[[4](http://www.books24x7.com/assetviewer.aspx?bookid=30903&chunkid=660434808&noteMenuToggle=0&hitSectionMenuToggle=0&leftMenuState=1" \l "ftn.ch14fn04)] files and perform all other SNMP configuration, navigate to Administration→Access→SNMP Settings from the left menu. The second tab contains the three device MIBs, which can be downloaded by right-clicking on the filenames and saving them to a local machine. They can then be uploaded to a monitoring console such as Tivoli NetView, HP OpenView, or BMC PATROL, which will understand the available monitoring points and can display them to the administrator for selection in various formats. Each of the three files represents a different mode of SNMP interaction:

* Add a note heredrConfigMIB.txt describes a method for retrieving the configuration via SNMP.
* Add a note heredrStatusMIB.txt provides the interface for current status retrieval.
* Add a note heredrNotificationMIB.txt describes the traps sent by log targets on the device.

**Add a note here****Configuring SNMP Polling of the Device**

Add a note hereThe Main tab on the SNMP Settings page defines how a monitoring server can connect to the appliance for polling. Define a local IP (or host alias) and SNMP port on the device, as seen in Figure 14-18; the default port is 161. Turn on the listener for this port by choosing the enabled radio button and clicking Apply.

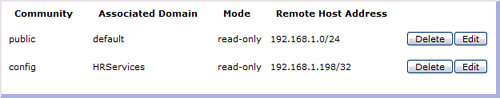
[](javascript:PopImage('IMG_327','http://images.books24x7.com/bookimages/id_30903/14fig18_alt.jpg','501','228'))  
Add a note hereFigure 14-18: Enabling SNMP polling of the device.

Add a note hereIf you’ll be using SNMPv3, you can also define users, a security level, and an access level, depending on the permissions you want the poller to have. For version 3, you’ll also have to switch to the SNMPv3 Contexts tab and map between contexts and application domains.

Add a note hereFor SNMP version 1 or 2c, you’ll have to define communities that have access to certain information through polling. A community is defined in external SNMP tooling as a group that sends or receives SNMP traps; the names chosen in DataPower have to match those defined in your monitoring tools. Switch to the SNMPv1/v2c Communities tab and click Add to define a new community as shown in Figure 14-19.

  
Add a note hereFigure 14-19: Defining a new SNMP community.

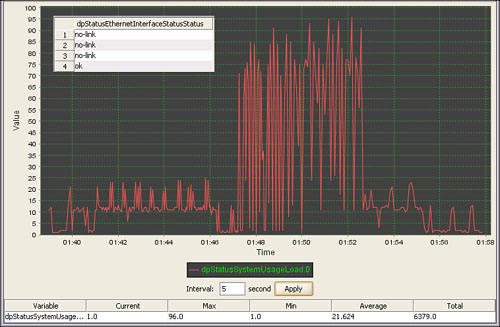
Add a note hereAdding a community for a domain allows members of that community to poll for information from that domain. For device-level status information, select the default domain. You also need to define a range of IP addresses that are allowable as device pollers. Using a class C address as shown in the first entry in Figure 14-20 allows a range of IPs, while the distinguished IP shown in the second entry limits access to a single poller.

[](javascript:PopImage('IMG_329','http://images.books24x7.com/bookimages/id_30903/14fig20_alt.jpg','625','123'))  
Add a note hereFigure 14-20: Viewing multiple defined communities.

Add a note hereAfter communities have been defined and polling has been enabled, you can begin to poll the device from your monitoring server. Some suggested status parameters to monitor include

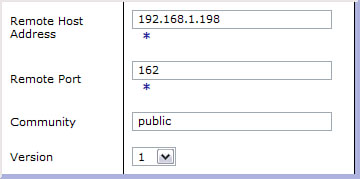
* Add a note here**dpStatusEthernetInterfaceStatusStatus—** Shows the current state of each Ethernet interface. (OK means a successful connection and no-link means a lost connection.)
* Add a note here**dpStatusSystemStatusLoad—** Shows the current device load percentage; this is the best measure of overall system usage.
* Add a note here**dpStatusFilesystemStatusFreeEncrypted—** Shows the available space on the file system.
* Add a note here**dpStatusMemoryStatusFreeMemory—** Shows the available memory available for processing.
* Add a note here**dpStatusCPUUsagetenSeconds—** Shows the CPU usage percentage for the last ten seconds.

Add a note hereThese values can be polled in several different ways; you can retrieve individual values at certain intervals for auditing purposes, graph the numbers over time, or create tables of parameter values. These values should be monitored and the appropriate team should be alerted if these values are out of your acceptable ranges. In Figure 14-21, you can see a graph of system load over time and a table with the current Ethernet interface status.

[](javascript:PopImage('IMG_330','http://images.books24x7.com/bookimages/id_30903/14fig21_alt.jpg','758','496'))  
Add a note hereFigure 14-21: Viewing various status parameters using SNMP.

**Add a note here****Configuring Sending of SNMP Traps**

Add a note hereAside from proactive polling for device health, monitoring solutions can also receive SNMP traps from the DataPower appliance. This functionality is enabled from the SNMP Settings page; the Trap and Notification Targets tab enables you to define the off-device trap listeners. Choose the IP and port of the remote listener and choose a version and community that will be supported by the monitoring server, as shown in Figure 14-22.

  
Add a note hereFigure 14-22: Defining an off-device trap listener.

Add a note hereAdd the listeners that will receive each log message that has been defined to send to an SNMP target. As a shortcut to send specific events, you can select the Trap Event Subscriptions tab and define events that will always send messages to the defined trap targets. For more flexible trap definitions, define an SNMP log target as described earlier in this chapter. A log target gives you the flexibility of subscribing to whole categories of events, filtering on objects that created those events, and selecting specific event codes to include or suppress.

Add a note here[[2](http://www.books24x7.com/assetviewer.aspx?bookid=30903&chunkid=660434808&noteMenuToggle=0&hitSectionMenuToggle=0&leftMenuState=1" \l "ch14fn02)]Simple Network Management Protocol, a polling/trapping standard for monitoring network-accessible devices.

Add a note here[[3](http://www.books24x7.com/assetviewer.aspx?bookid=30903&chunkid=660434808&noteMenuToggle=0&hitSectionMenuToggle=0&leftMenuState=1" \l "ch14fn03)]SNMP GET requests should not be confused with GETs for any other protocol (MQ, etc.). The SNMP specification defines both the format and use of these commands.

[[4](http://www.books24x7.com/assetviewer.aspx?bookid=30903&chunkid=660434808&noteMenuToggle=0&hitSectionMenuToggle=0&leftMenuState=1" \l "ch14fn04)]A Management Information Base (MIB) file describes the database of objects (and their OIDs) that can be monitored by a network management system via SNMP. Each product that allows SNMP monitoring comes with a MIB file to explain its interfaces to any monitoring product.



## Service Monitoring

Add a note hereDevice monitoring is all about device health; service monitoring, on the other hand, is all about gathering operational metrics for the Web services that are present within an infrastructure. Because DataPower acts as a proxy to all Web services traffic, it represents an ideal location in which to gather statistical information about the execution of those services. However, DataPower is not suited to graphing or reporting on those statistics, aside from simple SLM graphs and basic execution numbers. For a full monitoring solution, products such as ITCAM for SOA will correlate service executions, produce graphs, and monitor the entire lifecycle of a service transaction. DataPower can interoperate with ITCAM for SOA using the WSDM standard for exchanging data. This data can be aggregated across domains and appliances, or can be split to see the performance of individual devices. It can also contain information about service clients and can serve as the basis for dynamic distribution of future requests.

Add a note hereFor retrieval of simpler statistics that may not need additional processing and multidimensional analysis, we have already discussed several methods for status retrieval. Most recently, we demonstrated monitoring of the appliance using SNMP polling; these status parameters include Web service and SLM metrics. From the CLI the show command works on Web services statistics, as does the get-status operation on the SOAP management interface.

## Summary

Add a note hereWe have seen that there are many different ways to gather information from a DataPower SOA Appliance. Information can be gathered on the device and offloaded at regular intervals, polled from the device using a standards-based monitoring console, and pushed from the device to various targets in a real-time environment. At every opportunity, the appliances will integrate with the best-of-breed tools that have become standard in the enterprise, including tight integration to IBM’s monitoring and logging products. The flexibility and granularity provided by the logging system on the device allows for separation of concerns, so that each group receives only the information to which they have subscribed. This allows for complete integration into the enterprise, similar to any other network device.