CHAPTER 7

Using the CiscoWorks Home Page

The CiscoWorks Home Page (CWHP) is the user interface for CWCS-based network management applications. It offers an improved user interface and navigation paradigm for multiple CiscoWorks applications. It provides:

- Launch points for local and remote Cisco Works applications and their major functions.
- Easier navigation between CWHP and the applications presented on CWHP.
- Support for applications that are based on the User Interface Infrastructure (UII) and that have their own home pages.
- Support for Common Services administration.
- Launch points for external resources, including URLs on Cisco.com, custom tools and third-party applications
- Launch points for Cisco product updates.

The following topics describe CWHP and how to integrate it with your application:

- Understanding CWHP
- Integrating Your Application with CWHP

For basic information on CWHP, see the "About the CiscoWorks Home Page Component" section on page 6-5.

For more information about CWHP, see:

- *CiscoWorks Home Page Functional Specification, EDCS-281825l*
- *Use Cases and Behavior of CWHP*, EDCS-284828
- *Triveni UII Integration with Core Security,* EDCS-199291
- *Mjollnir CMF 2.3 PRD, EDCS-2634301*
- *CMF 2.3 System Functional Specification*, EDCS-283137

Understanding CWHP

CWHP is the primary user interface and launch point for all local and remote CWCS administration application tasks. It is a Cisco User Interface Infrastructure (UII) application that displays banner information and tool bar items in standard locations in a web browser.

The following topics provide essential background information about CWHP and how it works:

- About the CWHP Interface

- How CWHP Works
- How CWHP Uses CMIC
- How CWHP Handles Security
- About the CWHP Runtime Structure

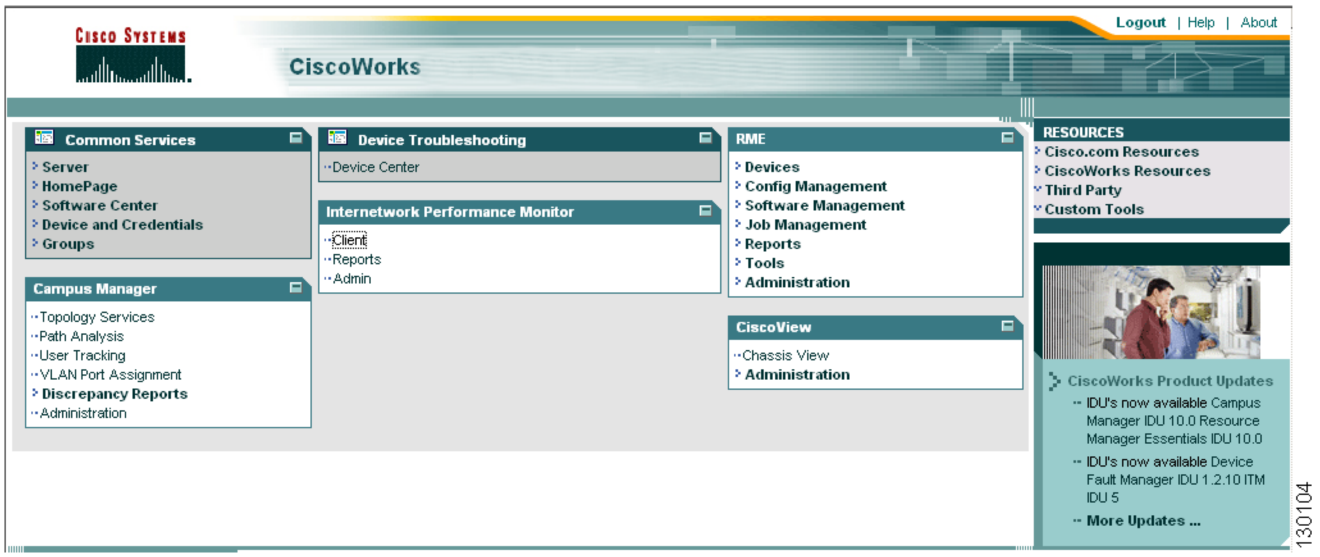
About the CWHP Interface

CWHP divides the standard UII Content Area into six main areas:

- Common Services panel: Provides links to all CWCS-specific functions, including server and group administration, device and credentials maintenance This comes built-in with Common Services.
- Device Troubleshooting panel: Provides a launch point to the CWCS Device Center. This comes built-in with Common Services.
- Application panels: Display launch points for all CWCS-based applications installed on local or remote servers. The name of each application panel serves as a link to the relevant application homepage, which is launched in a new window when the user selects it.
- LMS Setup Center panel: Provides a launch point to the LMS Setup Center where you can configure the system settings for all the applictaions in one stop. This comes built-in with Common Services.
- Resources panel: Provides launch points for external resources (including other CiscoWorks resources, Cisco.com resources, third-party application links, and web-based custom tool links).
- CiscoWorks Product Updates panel: Displays informative messages about CiscoWorks product announcements, and help-related topics.
- Tool Bar: Allows users to logout, access online help, or display “About” information (this includes license information, version and patch level, installation date and copyright information).

Figure 7-1 shows the layout of a typical CWHP:

Figure 7-1 Typical CWHP Layout



CWHP uses the first three columns of the content area to display application panels. Applications get onto application panels by being registered with CMIC. The CWHP application queries the CMIC registry for all registered applications, both local and remote, at user login. It displays them based on the order they appear in the CMIC MST file.

CWHP supports applications installed on multiple servers. If the same application is found on more than one server, the instance of the application running on the local server is listed first. Instances of the application found on other servers then appear in alphabetical order by the server name.

The title of each application panel displays the application's name. The title also serves as a link to the relevant application's home page.The square expand/collapse icon shown to the right of the application's title expands/collapses the entire panel.

Application tasks are displayed as labels, in a hierarchical manner. Clicking on the label will launch the URL associated with the item in a popup window. First- level tasks for each application are listed below the title, and are shown in collapsed mode by default. Lower- level task labels are displayed only if the user manually expands the first-level item by clicking on the ">" icon. The content of each panel is limited to two levels, but there is no limit on the number of entries per level. These levels are mapped to application home-page navigation levels.

When the user clicks a task label from the hierarchy, the CWHP link to the task launches the application's home page in a new window and then selects the task by default. If an instance of the application home page is already displayed in a window, that window will be given focus and the new task will be selected.

CWHP does not perform authorization of contents displayed in application panels. It is the responsibility of applications to display proper error messages in cases where the user selects a task for which he is not authorized.

CWHP uses the last column to display information about external resources. These include CiscoWorks Resources, Cisco.com Resources, Custom Tools, Third Party Applications and Cisco Product Updates. Using the CWHP Admin UI, this section can be turned off and used for application panels.

CWHP uses the Tree Window Component to render the UI for applications. This component uses the The xbTreeWidgetStatic JavaScript tree implementation. The xbTreeWidgetStatic is open-source software from Netscape and is a cross-browser JavaScript tree widget which allows flexible construction of HTML tree widgets using a simple API and HTML. DHTML manipulation of the tree is possible in browsers which support the Microsoft Internet Explorer-proprietary HTMLElement.innerHTML property. For browsers which do not support this property, xbTreeWidgetStatic will generate a simple hierarchical layout. XbTreeWidgetStatic is designed to be used before a page load event fires by writing the HTML into the document. XbTreeWidgetStatic is static because it cannot modify the contents of the Tree Widget after it has been written to the document. XbTreeWidgetStatic requires the use of JavaScript 1.2 in the implementation of its data structures.

How CWHP Works

CWHP processing involves the following steps:

1. A CWCS-based application is installed on the same server with an instance of CWCS Server. The installation script calls the CMIC Register API, passing with the call the application's MST template URN, and its host, port, and protocol. This registers the application and its home page with CMIC.

   CiscoWorks applications installed on other servers, third-party applications, and custom home-grown tools are registered with CMIC by end users, using the CMIC Administration User Interface.

2. Based on the template URN, the CMIC registry fetches the application's MST file from the MST template store and updates it with the host, port, and protocol. It then replaces it in the MST template store in serialized form for easy search.

3. During CWCS Server startup, CWHP loads a servlet that searches the CMIC registry for the CWHP-related integration tags shown in Table 7-1. The servlet caches the CWHP-related CMIC application and task information it finds.
4. Each successful user login after that will access CWHP and invoke the CWHP servlet. The CWHP servlet, in turn, checks that it is still in sync with CMIC and, if not, recreates the CWHP cache. The servlet then fetches the application and task information from its cache.

How CWHP Uses CMIC

The CMIC registry (see Chapter 9, "Integrating Applications with CMIC") stores data about all registered applications in the network. By querying the registry as needed, CWHP gets all the information it needs to provide launch points to these registered applications.

Applications cannot be integrated unless they have been registered with CMIC. To do this, you must create a CMIC MST template and tag your application URLs appropriately, using the task tags defined by CWHP, and register the template with CMIC. Table 7-1 lists the integration tags used in the MST file to identify application tasks to CMIC.

You need not create different templates for CWHP or Device Center. All your application tasks can be part of a single template and registered with CMIC at one time.

Registration is a one-time process. You do not need to register your application again any time after initial registration. Ideally, the registration code should be part of a post-installation script executed immediately after your application is installed.

All CMIC MST templates are stored under *$NMSROOT/*data/cmic/mst-templates. You should copy your template to this location and then pass the file name during CMIC registration.

Table 7-1 CWHP Integration Tags

CWHP Function	CMIC MST Integration Tag	Use this tag to show the task in
Cisco Works Applications installed on the same server	CWHP_APP_TASK	Relevant CWHP Application panel
Cisco Works Resources	CWHP_CW_RSRC	CWHP Cisco Works Resources panel
Cisco.com Resources	CWHP_CSCO_RSRC	CWHP Cisco.com Resources panel
Cisco Works Applications installed on other servers	CWHP_OTHR_APPS	CWHP Cisco Works Other Servers panel
Third Party Applications	CWHP_THRD_PRTY	CWHP Third Party Applications panel
Custom Tools	CWHP_CSTM_TOOL	CWHP Custom Tools panel
Common Services Administration	CWHP_TASK	Common Services Panel
Device Center	DC_TOOLS	Device Troubleshooting Panel
Setup Center	systemSetup	LMS Setup Center Panel
	securitySetup	
	dataCollectionSettings	
	dataCollectionSchedule	
	dataPurge	

How CWHP Handles Security

CWHP uses the UII security integration component, which integrates UII security with the CWCS security system . Using this component, CWHP can redirect security requests to the CWCS security system.

The request flow for authentication:

- User points the browser to the CWCS URL.
- The HTML login page is displayed.
- Once login is successful, redirection to the CWHP web application (running under Tomcat) happens through the help of the CMFLiaisonServlet. This servlet enables single sessions across servlet engines. If the user bookmarks his CiscoWorks Home Page and then later goes to that URL, the UII security authentication implementation will redirect the request to the HTML login page.

Authorization is required for rendering CWHP navigation menu items and this is handled by the following UII Authorization guidelines:

- Provide taskid information in the CWHP site map xml file
- Implement the isAuthorized method of UIIAuthorizeImpl to handle the authorization relevant for the CWCS security infrastructure

CWHP does not verify authorization for rendering application launch points; it assumes the user has been verified by the login. CWHP also does not perform license verification while displaying application links.

The HTML based login panel is generated through a JSP page. A JSP page renders the login panel based on the login module that is selected. SSL is used as the secure transport mechanism. SSL port will be open in non-SSL mode also, to accept login requests. The login page is served off the Tomcat servlet engine and CWCS Web Server. The SSL certificate must be generated for the CWCS Web Server at install time, as is done for CORE.

About the CWHP Runtime Structure

All CWCS shared components (such as cmic.jar) are installed under *$NMSROOT/MDC/tomcat/lib/apps*. All shared class files are placed under *$NMSROOT/MDC/tomcat/lib/apps/classes directory.*

All CWCS modules that provide a user interface, including CWHP, are installed in folders below the CWCS *$NMSROOT/*MDC/tomcat/webapps/cwcs folder under Tomcat. Table 7-2 shows where to find all CWHP-specific runtime files.

Table 7-2 CWHP Runtime Files

For all CWHP-related	**See this folder under *$NMSROOT/*MDC/tomcat/webapps/cwhp**
JSP files	/screens/cwhp
UII action classes	/WEB-INF/classes/com/cisco/nm/cmf/cwhp /ui/action
UII form bean classes	/WEB-INF/classes/com/cisco/nm/cmf/cwhp /ui/form

Integrating Your Application with CWHP

To integrate your application with the CiscoWorks Home Page, you must:

- Register your application and its tasks with CMIC.
- Implement security features.
- Implement any special license checks you may require
- Customize the content of the message area as needed

The following topics describe how to perform each of these tasks, as well as tips on how to proceed if you cannot fully implement UII in your application:

- Registering Your UII-based Application with CWHP
- Implementing CWHP Security
- Implementing Special License Checks
- Handling CWHP Messages
- Migrating to CWHP

Registering Your UII-based Application with CWHP

In most cases, users will use the CWHP application panels to access your application tasks. Therefore, your first task in registering your application is to create a CMIC MST file that identifies the major tasks in your application using the integration tag CWHP_TASK. Each task that you identify and tag in this manner will appear in the application panel for your application.

If you are using UII sub-area bar menu items with screenids, you do not have to identify every subtask on the menu in the MST. Instead, specify the sub-area bar menu item as a TASKGROUP and have the TaskURL value set to the screenid of the sub-area bar menu item. When the application is launched in the new window, the UII will take care of selecting the sub-area bar menu item and displaying the content associated with it. This also works for TOC menu items.

We recommend that you also define the custom attribute “window_name”, with a consistent window name as its value, within the CWHP_TASK integration tag. If you specify this custom attribute, CWHP will use the same window instance for all of the application’s tasks. If you do not specify this attribute, CWHP will assign unique window names for each task automatically, and each task will be displayed in a different window.

Example 7-1 shows a snippet from the MST file for Campus Manager that identifies tasks to be displayed on the Cisco Works Home Page in RME window. It uses both TASKGROUP and TaskURLs.

Example 7-1 Sample MST File

```
<TASKGROUP GroupName=”Devices”>
    <TASKINFO TaskName=”Group Management” TaskIdentity=”t001” TaskDescription=”Group
Management” TaskCategory=”C” TaskSubCategory=”C/admin” SecurityTag=”nm.cm.admin”
TaskURL=”/rme/groupmanagement.do” SubmitMethod=”GET” IsAPI=”false”>
    <INTEGRATIONTAG TagName=”CWHP_TASK”>
    <ATTRIBUTE Name=”window_name” Value=”rmewin” />
    </INEGRATIONTAG>
</TASKINFO>
```

```
</TASKGROUP>
```

Once you have completed the application MST file:

- Modify your application installation script to copy the MST file to *$NMSROOT*/data/cmf/cmic/mst-templates. For information on installation scripts, see Chapter 21, "Using the Installation Framework."
- Modify your application post-installation script to call the CMIC Register API and register this MST file with CMIC. This requires writing a java wrapper to call the Register API in a way that allows you to pass the MST file name to it. For information on the CMIC APIs, see Chapter 9, "Integrating Applications with CMIC."

Implementing CWHP Security

Applications using CWHP can handle user authorization and authentication via the Common Services UII Security integration component, which provides a UII security implementation based on the CWCS security system. This component is packaged as cs-uii-security.jar, and is included in the CWCS build as an SDK tar file. It contains the following implementation classes:

- UIIAuthenticateImpl (com.cisco.nm.cwcs.cwhp.uii.security: Redirects authentication requests to CWCS security system.
- UIIAuthorizeImpl: Redirects authorization requests to CWCS security system.
- CAMSystem: A wrapper for the CWCS security system that provides UII application access to CAM security APIs.
- securityDBLoader: A servlet that loads UII Tasks into memory.
- Task: The object representation for UII Tasks.

To provide full authentication and authorization services for your CWHP-based application:

1. Place cs-uii-security.jar in the root of the runtime structure for your application (i.e.,*$NMSROOT*/MDC/Tomcat/webapps/*$myapp*) .
2. Specify UIIAuthenticationImpl and UIIAuthorizeImpl as your security implementation classes in your application's web.xml file.
3. Define TaskDefinition, RoleDefinition and *CMFRoleDefinition* RoleMap XML files required by the CWCS security system's CAM infrastructure. For details, see Chapter 10, "Using the Security System."
4. Register these TaskDefinition, RoleDefinition and *CMFRoleDefinition* RoleMap XML files with CCR. For details, see Chapter 13, "Using the Core Client Registry."
5. In the struts-config.xml or site map xml file, define page-level or component-level security and use the task ID string from the TaskDefinition XML file.

Note that:

- Your team must extend UIIAuthenticationImpl to support any special license requirements you may have (see).
- An additional UII TaskDefinition XML file is no longer needed. If you have provided a UII TaskDefinition file, you must change the page-level and component-level definitions in the struts-config.xml and site map.xml files, and the "taskid" attribute of <uii:button...> tags in JSP pages, to use the task id string from your UII TaskDefinition file. If not, simply specify the CAM infrastructure version instead.

- Logout.jsp is no longer part of the UII security integration component. In CWCS, the Logout toolbar item is available only on the CiscoWorks Home Page. All application home pages have only a Close toolbar item.

Implementing Special License Checks

CWCS provides a complete licensing framework. This framework offers FLEXlm based licensing, which allows for a very wide range of licensing models and approaches, including feature and task-based licensing. For more information on the licensing framework, see Chapter 34, "Using the Licensing APIs."

To integrate this smoothly with authentication, the UIIAuthenticationImpl class supports a license check method that can be performed before generating the lightweight tree navigation. If the license is not valid, the tree will not be generated.

Since the license-check is application specific, the method has been added to the UIIAuthenticationImpl class:

```
public boolean doLicenseCheck(HttpServletRequest request, HttpServletResponse response,
ServletConfig config) {
        // default implementation provided by integration component is true
        return true;
}
```

The isAuthenticated method will call doLicenseCheck after successful authentication.

Please note that it is the application team's responsibility to extend the UIIAuthenticationImpl class and override this function based on your licensing requirements. Also be aware that, in this particular case, your team must specify this classname as the value for the init-param "UIIAuthenticateImpl" in the UIIController definition section of the web.xml file.

Handling CWHP Messages

The Product Updates panel displays product upgrades, tips, and other support information. The panel displays two types of messages:

- Standard messages: These messages are either read from the default message file or downloaded from CCO.
- Urgent messages: Urgent messages are read every 60 seconds and put at the beginning of the message queue.

Customers can add urgent messages to the Product Updates panel. Bear in mind that user messages are intended to be used by end users, not development teams. Cisco business units who want to add their marketing messages for CWCS-compatible applications or suites to the standard message file should forward their requirements to NMTG marketing.

Messages displayed in the Product Updates panel are read from files in the ***NMSROOT*/lib/classpath/com/cisco/nm/cmf/servlet/msgdir** runtime directory.

The message servlet determines the type of message based on the name of the message file. Table 7-3 shows the files that can be contained in the message directory.

Table 7-3 Message Directory Files

File name	Description
DefaultCcoMsgFile	Used if CCO cannot be contacted to download live messages. This holds good for CWCS installations on networks that are not connected to the internet. **Note** To include messages about your application in the panel, work with the NMTG marketing department to get your messages added to this file.
DownloadedCcoMsgFile	Used to store messages downloaded from the CCO web site. The file is updated with a fresh download from CCO every 24 hours by default. If there is an error downloading this file, then the previously downloaded file is preserved and continues to supply messages to the panel. The DefaultCcoMsgFile is used only if DownloadedCcoMsgFile has *not* been downloaded from CCO, earlier.
.urgent	Urgent messages are read by the message servlet every 60 seconds by default and then deleted. Messages read from this file are given message-queue priority and show up in the panel immediately. There can be a delay of up to 60 seconds between the time an .urgent file is written and the time its messages appear in the message window. This is caused by the 60-second polling interval in the browser and the 30-second scanning interval for .urgent files on the server.
UserMessageFile	TBD

Migrating to CWHP

CWHP supports full integration at the task level only with web-based applications that are UII-compliant. If your application is web based (e.g., an applet, or JWS), but was not created using the User Interface Infrastructure (UII), you can only integrate with CWCS by registering your application's base URL with CWHP.

We strongly recommend that your team consider a full implementation of UII, including redesign and conversion of existing application pages, if you want to integrate with CWHP. If this is not possible, you should at least try to implement the UII navigation features. Doing so will allow you to define tasks and their navigation in the manner required to launch them directly from CWHP.