Ignite-UX with ServiceControl Manager Applications

Author: Scot Greenidge
Hewlett-Packard
3404 E. Harmony Road, M.S. 99
Ft. Collins, Colorado 80528-9599
970-898-3387 (office), 970-898-2838 (fax)
scot_greenidge@hp.com

Abstract

Since its introduction three years ago, Ignite-UX has enjoyed a high degree of success as a viable and important tool for the HP-UX Systems Administrator. Due to both its flexibility and targeted management domain, Ignite-UX can be employed to solve a variety of problems that are common to many HP-UX computing environments. That said, a certain amount of finesse and creativity is required to properly take advantage of the most applicable part of Ignite-UX for a given management task.

This paper attempts to communicate information about Ignite-UX in three areas. First, a brief description of Ignite-UX, its primary use models, and features. Second, a description of how Ignite-UX is integrated with ServiceControl Manager and the specific benefits this integration provides to the administrator. Lastly, a scenario based discussion on how to apply Ignite-UX to solve a HP-UX management task using ServiceControl Manager.

Introduction to Ignite-UX

So what is Ignite-UX? Ignite-UX is a tool set (i.e. a collection of individual tools) that is designed to work cooperatively towards a stated goal. What is the goal? The primary goal of Ignite-UX is to boot a system from an independent device (i.e. the network, a CD-ROM, or a tape) into a self-contained environment. From that self-contained environment, file systems are configured and fresh copy of the HP-UX operating system is laid down along with a configurable set of system customizations. While this definition is rather terse, it

describes the core capability of Ignite-UX. Each of the tools that are contained by Ignite-UX, in one way or another, support this primary goal.

With this general definition in mind, let's examine some of the primary use models supported by Ignite-UX.

The most basic use model is the installation of a new system from some sort of media. Typically, an administrator will boot system from a CD-ROM then control the entire installation session via a terminal user interface running on the system's console. Decisions about the layout of the file systems and basic system configuration are made. Software is loaded from one or more CD's.

A second use model involves the use of something called an Ignite-UX server. An Ignite-UX server is a system onto which the Ignite-UX product (i.e. B5724AA_APZ) is installed. As an example, in our network you would install Ignite-UX with the following command:

```
# swinstall –s iuxbld.fc.hp.com:/release/Ignite-UX B5724AA_APZ
```

Once the product is installed, you can execute "**/opt/ignite/bin/ignite**" which, in turn, leads you through the additional steps required to use the Ignite-UX server for system installations. Briefly, the two primary steps are to supply one or more temporary IP addresses to be used by client systems when performing a network boot. Secondly, to specify the location of a software depot (used to supply the desired version of the HP-UX operating system).

Once the Ignite-UX server is configured, you are prepared to perform client installations using the server. The installation process can be controlled from either the client's console or the Ignite-UX server.

When controlling the installation from the client console, the client is first re-booted and the boot sequence interrupted using the escape key. At the prompt, you issue the command:

```
boot lan.172.34.5.6 install
```

The IP address that is supplied is that of your Ignite-UX server. The command itself instructs the client's primary loader to contact the Ignite-UX server and perform a network boot. Part of the ensuing process includes copying over an installation kernel and an installation file system. The install kernel is then executed and a RAM based file system is constructed from which the remaining portion of the installation will be controlled.

Since the installation is being controlled from the client (referred to as a *pull installation*), you will interact with the terminal based user interface in the same way you would during the media based installation. The only difference is that the software source has shifted from a CD-ROM or tape to a network connection with the Ignite-UX server.

If the installation is being controlled from the Ignite-UX server (referred to as a *push installation*), you will invoke the command **`/opt/ignite/bin/ignite`** on the Ignite-UX server**.**

This command brings up the Ignite-UX console, which in turn can be used to both control and monitor a particular client or set of clients. The user interface is modeled on an object/action paradigm. The clients (represented as system icons) are the "objects" and the set of "actions" which may be performed on the clients are available through various menu items.

Finally, Ignite-UX also provides the ability to initiate the installation process on a remote client(s) with the **bootsys(1M***)* command. This command allows the administrator to reference a specific HP-UX configuration and then instruct a client or group of clients to reboot and install using the desired configuration. The point here is the ability to craft a "hands-off" HP-UX installation process, a process that is both reliable and repeatable.

A secondary, but very important component of the Ignite-UX tool set, is the ability to construct a recovery archive for a system. In fact, for some customers, this represents the primary use of Ignite-UX. A recovery archive is a snapshot of a system at a given point in time. It can be used to fully recover the system in the case of failure (such as a failed system disk, software problem etc..). The command, **make_net_recovery(1M)** (along with its companion **make_recovery(1M)**) is the workhorse for this type of operation. **make_net_recovery(1M)** can either be executed by hand on an individual system or launched from the Ignite-UX console (i.e. the **ignite(1M)** command).

make_net_recovery(1M) involves two primary modes of operation. The first, which we will call interactive mode, allows the administrator to view the file system configuration of a given client. From this view, the administrator can then make decisions regarding what should or should not be included in the archive. The decision points range from entire volume groups down to individual files. Once the archive contents are defined, construction of the recovery archive begins and a record of what is required to reconstruct the archive is saved (/var/opt/ignite/clients/<lanic-id>/recovery/archive_content).

The second mode of operation utilizes the description of what the recovery archive contains and allows the administrator to construct a new archive. This is effectively a refresh operation. A configurable number of archives can be saved before the oldest one is replaced by the next execution instance of **make_net_recovery(1M***)*. Once the archive is complete, the administrator can also use additional Ignite-UX tools to construct a bootable archive tape.

Here is a summary of the available Ignite-UX commands:

- **add_new_client(1M)** – Add a new client to an Ignite-UX server without requiring a reboot of the client
- **add_release(1M)** – Add a new software release to an Ignite-UX server
- **bootsys(1M)** – Reboot and install systems using Ignite-UX
- **check_recovery(1M)** – Compare current system to recovery status file
- **ignite(5)** – Control/monitor clients remotely from the Ignite-UX graphical user interface (GUI)
- **instl_adm(1M),instl_adm(4)** – Edit/maintain Ignite-UX configuration files
- **instl_combine(1M)** – Combine a LIF file and a file system such that it can be written to a CD-ROM
- **instl_bootd(1M)** – Boot protocol daemon for Ignite-UX clients
- **make_bundles(1M)** – Construct a **S**oftware **D**istributor bundle from products and/or filesets in an **SD** depot
- **make_depots (1M)** – Copy bundles from an **SD** source into a depot which can be used by other Ignite-UX tools
- **make_boot_tape(1M)** – Construct a bootable tape which can be used to connect to an Ignite-UX server
- **make_recovery(1M)** – Construct a bootable system recovery tape for a whole disk or LVM system while the system itself is up and running
- **make_net_recovery(4)** – Construct recovery archives for a running system over the network with flexibly defined content
- **make_sys_image(1M)** – Construct a compressed system archive of a running system
- **manage_index(1M)** – Manipulate/manage Ignite-UX INDEX files
- **make_config(1M)** – Construct Ignite-UX config files which correspond to an **SD** depot

- **pkg_rec_depot(1M)** – Construct an **SD** depot containing the Ignite-UX recovery tools filesets
- **print_manifest(1M)** – Print a system manifest containing information about hardware, software, file system layout etc
- **remove_release(1M)** – Remove all the Ignite-UX files associated with a single release of HP-UX
- **setup_server(1M)** – Perform administration tasks for an Ignite-UX server
- **save_config(1M)** – Extract disk and file system structures along with certain system and networking parameters and write them to a configuration file. This effectively builds a config file that describes the systems current disk and file system layout

Briefly then, this introduces some of the core concepts and capabilities of Ignite-UX. Listed in the appendices of this paper are additional resources for learning about Ignite-UX. In particular, we mention one here that contains under one cover many of the individual sources. It is the: "Ignite-UX Administrator Guide" and can be downloaded from the Ignite-UX web site at:

http://www.software.hp.com/software/HPsoftware/IUX

Alternatively, it can be ordered directly from Hewlett-Packard using part number: B2355-90677.

In the next section, we will take a closer look at some of the new functionality that is being introduced as a part of Ignite-UX's integration of ServiceControl Manager.

Using Ignite-UX with ServiceControl Manager

To one degree or another many computing environments are forced to deal with the problem of distributed systems management. ServiceControl Manager is an attempt to address some of the common issues faced by a Systems Administrator in a distributed HP-UX environment. One way of thinking about the relationship between ServiceControl Manager and Ignite-UX is that Ignite-UX works well for initially configuring many similar (or identical) systems. ServiceControl Manager enables you to maintain consistency on those same systems over time. Several papers are being presented during INTERWORKS 2000 that look at ServiceControl Manager in detail. Our purpose here will be to discuss some of the broad ServiceControl management concepts and then to look a bit more carefully at the integration points that relate to Ignite-UX. Initially, then, let's look at the ServiceControl Manager Architecture.

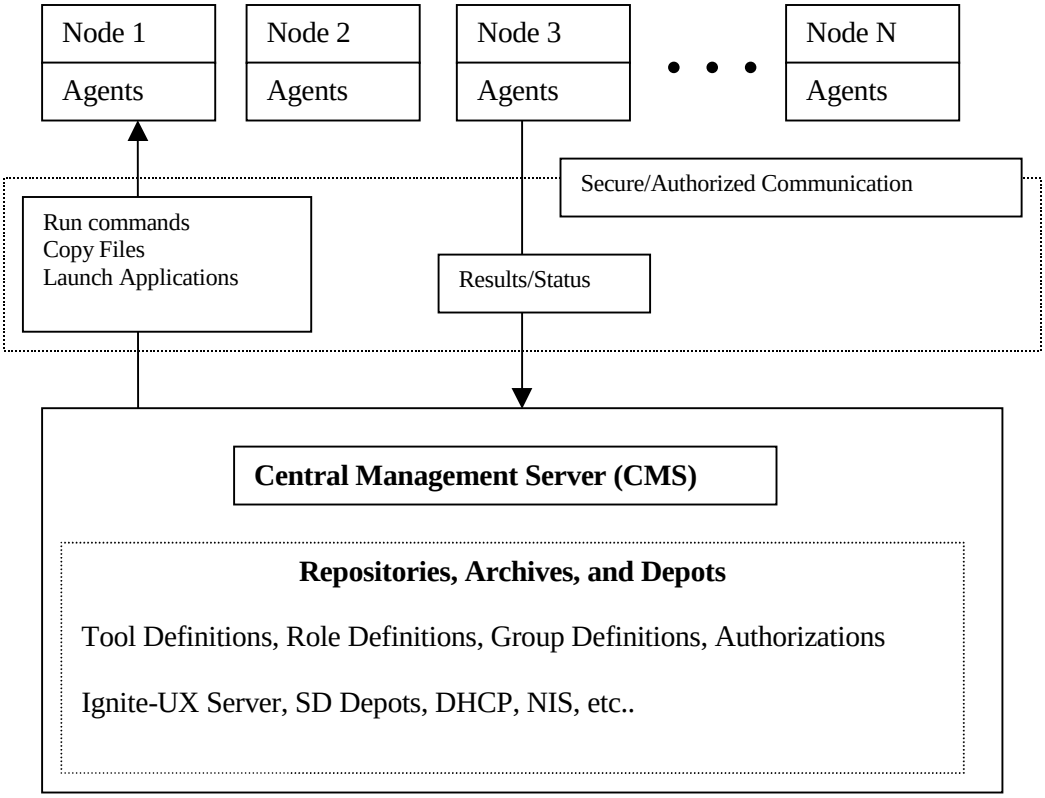

Figure 1: Basic ServiceControl Manager Architecture

ServiceControl Manager (SCM) increases system administration effectiveness by distributing the execution of existing tools across a set of systems (referred to as "managed nodes"). The ServiceControl Manager's central operations are performed on the Central Management Server (CMS). The CMS functionality can be accessed from the command line (through eleven related commands) or through a Java based Graphical User Interface (GUI). On HP-UX the GUI is displayed using X-Windows, locally, on the console or remotely, on another system. The Java application can also display on a PC-compatible system using web browser technology. Command line access is available through the system console or through a telnet session.

Role based management, an organizational concept, is built into ServiceControl Manager. Role based management increases both efficiency and security by allowing senior administrators to delegate specific management activities to less experienced individuals across one or more managed nodes. More specifically, ServiceControl Manager supports

the concept of role definitions. A role definition is a collection of one more tools whose grouping defines a particular management function. An example of a simplistic role from an Ignite-UX perspective might be that of "Ignite-UX Operator". In this case, you want to allow the "Ignite-UX Operator" to only perform management activities which do not risk actual changes to a managed node. Some of the tasks you might want the operator to perform are:

1. Construct or Modify a Recovery Archive

2. Refresh a Recovery Archive

3. Monitor the status of a recovery archive

4. View or print system manifest information

Under ServiceControl Manager, each of these tasks are "implemented" by one or more tools. The grouping of the tools defines our "Ignite-UX Operator" role. The initial release of ServiceControl Manager includes the following tools in support the above tasks. They are:

1. "Construct or Modify Recovery Archive"

2. "Refresh Recovery Archive"

3. "Ignite-UX Restricted Console"

Once a role has been defined, a Systems Administrator may now form a three-way association of a role, a user, and a group of one or more managed nodes. Forming this association is referred to as an "authorization". In our example, the authorization would be that of granting a user the role of "Ignite-UX Operator" on a set of managed nodes. Once this authorization is established, ServiceControl Manager guarantees that the user cannot perform activities outside of this role on the target set of managed nodes.

Extending our example one step further, we could construct another role, which we will call "Ignite-UX Administrator". This role will include those components of Ignite-UX which in fact do make changes to a managed node. Some of these tasks are:

1. Install HP-UX onto system

2. Recover a system using a recovery archive

3. Modify the Ignite-UX Server default configuration

4. Setup an Ignite-UX Server

5. Reboot a system instructing it to connect to the Ignite-UX Server

As in the “Ignite-UX Operator” role, each of these tasks must be implemented by one or more tools under ServiceControl Manager. In the initial release of ServiceControl Manager, these tools are:

1. “Install or Recover System”

2. “Ignite-UX Console”

In our example, a user would need to be assigned both the “Ignite-UX Operator” role and the “Ignite-UX Administrator” role for a set of managed nodes in order to have full use of Ignite-UX under ServiceControl Manager.

The essential point here is that the user can be anyone in the standard passwd database. Granting root access to either the Ignite-UX server or the managed nodes is no longer a requirement to access Ignite-UX functionality.

At this point, is might be useful to examine the concept of a tool in more detail. A tool, as it is defined by ServiceControl Manager, is the encapsulation of a HP-UX command line. Included in this encapsulation are instructions regarding any required parameters to the command line, as well as the ability to copy files from a source to a destination prior to invoking the command line. With this as a basic definition, tools are divided into two main types.

Single System Aware (SSA) Tools

Single system aware tools are designed to execute on a single node. Commands such as bdf(1M), or sam(1M), or ps(1M) are candidates for a single system aware tool. One of the Ignite-UX tools included in ServiceControl Manager, “Refresh Recovery Archive”, is another example of a single system aware tool. The point here is that when the tool is run against a group of systems, the effect is that of a secure remsh(1M). ServiceControl Manager runs the tool on each of the destination nodes with results being routed back to the Central Management Station (CMS). The ServiceControl Manager component that accomplishes the remote execution is referred to as the Distributed Task Facility (or DTF). One further note is that, under the first release of ServiceControl Manager, no more than sixteen processes will be active at any given time. If you were to launch a tool like “Refresh Recovery Archive” against twenty managed nodes, sixteen processes would start with the remaining four queuing up waiting for one of the process slots to become available. An obvious enhancement to this model is to make the number of processes a configurable attribute.

Multiple System Aware (MSA) Tools

Multiple system aware tools are those that already know how to deal with multiple systems, such as Software Distributor (with push capability) and Ignite-UX. In this case, managed nodes are selected within ServiceControl Manager in the same manner as Single-system

aware tools, but the tool operates on the target nodes using its own internal mechanisms rather than using the Distributed Task Facility. The following is a list of the Ignite-UX tools that fall into the MSA category:

- "Create or Modify Recovery Archive"
- "Install or Recover System"
- "Ignite-UX Console"
- "Ignite-UX Restricted Console"

If the division of Ignite-UX functions into separate roles seems interesting or helpful, the ability to limit interaction to a particular group of systems should also be compelling. Environments which already have an established Ignite-UX server which in turn serves a large number of systems (possibly including multiple management groups) are aware of the fact that when using the **ignite(1M)** console, access (at least visually) is granted to all systems contained in /var/opt/ignite/clients.

With ServiceControl Manager, management domains defined by groups of systems can be established. Thus, when a user with the "Ignite-UX Administrator" role (to borrow from the example above) launches the "Ignite-UX Console" tool, he sees only the systems within his management domain (though many more may be using the Ignite-UX Server). This not only simplifies what you are viewing, but it also helps prevent accidental interaction with systems for which another Systems Administrator is responsible.

If breaking up systems along boundaries of responsibility is not a requirement, one could envision other groupings along application type, system function etc.

Finally, the Ignite-UX tools that are provided with the initial release of ServiceControl Manager are not exhaustive. They represent some of expected tasks that a Systems Administrator would want to perform using Ignite-UX within ServiceControl Manager. One of the additional benefits of the ServiceControl Manager environment is that a Systems Administrator can build tools locally to fit his own process or needs. The initial tools that are supplied provide both an entry point as well as an example from which other tools can be built.

Additional sources of information on the topic of building ServiceControl Manager tools can be found in the: *ServiceControl Manager Technical Reference***.** The location of the actual tool definitions supplied with ServiceControl Manager is: **/var/opt/mx/tools.** Also, see the **mxtool(4)/mxtool(1M)** man pages.

We now move on to look at some of the recent enhancements to Ignite-UX as well as examining a specific scenario that utilizes both ServiceControl Manager and Ignite-UX.

Ignite-UX in Practice

Recent Feature Review

make_net_recovery(1M)/make_net_recovery(1M)

Interacting with the recovery process no longer changes the behavior. The recovery_mode flag remains active and any changes made using the user interface are now "intelligently" merged into the original system files. In addition, two new options were added to the **make_recovery(1M)** command. The first, **-i**, allows you to construct an interactive tape. Normally, when booting a system from a recovery tape, you have ten seconds to interrupt the boot process in order to make modifications. Specifying the **–i** option means, that the interactive menus are always presented. In addition, this option can also be used to prevent a system recovery when booting from the tape device by mistake. The **–t** option allows you to supply a custom title/message which, in turn, will be displayed when booting from the tape. This can be helpful in identifying the tape. The default message written to the tape is: **Recovery tape created from system:** <system name> **on** <date>.

Both **make_recovery(1M)** and **make_net_recovery(1M)** now correctly re-import volume groups that contain all raw (or unmounted) logical volumes. However, they still do not backup any data from raw logical volumes that are re-constructed during the recovery process.

With regard to MC/Service Guard, both **make_recovery(1M)** and **make_net_recovery(1M)** will correctly archive and restore the shared volume group configurations. Previously this had been a problem in that **vgdisplay(1M)** was used to discover the volume group information. **vgdisplay(1M)** will fail if it is run against a shared volume, which is locked by a peer node in the cluster.

The LVM commands that are run to re-import volume groups when using a **make_recovery(1M)** tape are now done using the post_config_cmd keyword (instead of the final_cmd keyword). This allows any output/errors from these commands to be logged to the /var/opt/ignite/clients/<lanic-id>/install.log for later analysis.

The "A" version now supports system archives greater than 2GB using the NFS access method. Both client and server must be running 10.20 (or for the "B" version, which already had this feature, the server must be running 11.00 and the client either 11.00 or 10.20) and have the NFS PV3 software (networking ACE) loaded. See the Ignite-UX release notes (/opt/ignite/share/doc/release_note) for more detail on required patches.

ignite(1M)

The **ignite(1M)** GUI now supports the ability to hand in a list of systems on the command line which in turn will serve as a filter. Only systems in the filter list will be presented in the Ignite-UX console although many more may exist in /var/opt/ignite/clients.

The **ignite(1M)** GUI also supports two new short cut options, which allow you to jump directly to either the "Install" action or the "Create Recovery Archive" action directly from command line invocation. For example, "ignite –install <client>" will automatically select the client and take you directly into the configuration editor.

Miscellaneous

When using the **bootsys(1M)** command to reboot a client in order to install to a disk other than the current root disk, the AUTO file in the boot area on the original root disk is now restored to the original setting. Booting from the original boot disk will automatically remove the INSTALL kernels left behind by the **bootsys(1M)** command.

The **bootsys(1M)** command may now be executed on the client as well as on the server (the precondition is that the Ignite-UX.MGMT-TOOLS fileset has been loaded onto the client).

In order to better support NIS+, the **bootsys(1M)** command now uses **nsquery(1)** (if possible) instead of **nslookup(1)** for hostname resolution.

The **make_medialif(1M)** command now has a "-a" option to construct media that contains all INSTALL kernels (32 bit and 64 bit). This allows you to construct media that can be used on all Hewlett-Packard 9000 systems (32 bit, N-class, V-class, etc..).

Automating Recovery Archives with ServiceControl

In the previous sections, we have noted a tool called **make_net_recovery(1M)**, which can be used to produce system recovery archives over the network. We have also examined (briefly) a new management environment called **ServiceControl Manager.** In this section, we will combine both the Ignite-UX tool and the management environment to construct a process for automating the production of recovery archives.

To begin with, let's assume that we will establish a ServiceControl management cluster that consists of a CMS (Central Management Server) and twenty-five managed nodes. The process of setting up the CMS is documented in the *Installing ServiceControl Manager* manual (see the Appendices of this paper for reference information) The main tasks that you would need to work through are:

1. swinstall the B8337BA product onto the CMS system

2. execute /opt/mx/bin/mxsetup

3. swinstall the B8338BA product onto each of the managed nodes

4. swinstall the AgentConfig fileset from <the CMS>:/var/opt/mx/depot11 onto each of the managed nodes

5. add the managed nodes to the CMS using: mxnode –a <system name>

The next step involves the construction of a role that will accomplish the Ignite-UX tasks. The **scmgr(1M)** GUI provides access to these steps, but for the purpose of illustration, we will use the command line interface. The basic steps are:

1. Construct an Ignite-UX Operator role using **mxrole(1M)**

2. List the Ignite-UX tool definitions and save them to a file

3. Edit the file and add the Ignite-UX Operator role to the tool definition

4. Load the tool definitions back in with the updated role information

5. Add a user to ServiceControl who will be working under the Ignite-UX Operator role

6. Authorize the user on each of the managed nodes with the Ignite-UX Operator role

ServiceControl Manager ships with sixteen predefined roles. In the first release, a new role cannot be constructed, but one of the existing roles can be renamed to represent its intended use. For our example, we will use the **mxrole(1M)** command to "construct" the Ignite-UX Operator role. First, we will list the roles that are currently defined on the CMS using:

mxrole –lt

This will produce the following listing:

```
NAME      ENABLED?  DESCRIPTION
role16    true      For use by ServiceControl administrators
role15    true      For use by ServiceControl administrators
role14    true      For use by ServiceControl administrators
role13    true      For use by ServiceControl administrators
role12    true      For use by ServiceControl administrators
role11    true      For use by ServiceControl administrators
role10    true      For use by ServiceControl administrators
role9     true      For use by ServiceControl administrators
role8     true      For use by ServiceControl administrators
role7     true      For use by ServiceControl administrators
role6     true      For use by ServiceControl administrators
lvmadmin  true      A role for LVM Administrators
```

```
operator  true      A read-only role for operators
webadmin  true      A role for WEB Server Administrators
dbadmin   true      A role for Database Administrators
Master Role true      The ServiceControl Master Role
```

Selecting “role6” as our target role, you will then execute the following command:

mxrole –m “role6” –N “IUX Operator”

We have now renamed “role6” to be “IUX Operator”. Next, we modify the description field for the role using:

mxrole -m "IUX Operator" -d "Ignite-UX daily maintenance tasks"

Listing the roles a second time using **mxrole –lt** shows the following:

```
NAME            ENABLED?  DESCRIPTION
role16          true      For use by ServiceControl administrators
role15          true      For use by ServiceControl administrators
role14          true      For use by ServiceControl administrators
role13          true      For use by ServiceControl administrators
role12          true      For use by ServiceControl administrators
role11          true      For use by ServiceControl administrators
role10          true      For use by ServiceControl administrators
role9           true      For use by ServiceControl administrators
role8           true      For use by ServiceControl administrators
role7           true      For use by ServiceControl administrators
IUX Operator    true      Ignite-UX daily maintenance tasks
lvmadmin        true      A role for LVM Administrators
operator        true      A read-only role for operators
webadmin        true      A role for WEB Server Administrators
dbadmin         true      A role for Database Administrators
Master Role     true      The ServiceControl Master Role
```

Now that the role information is updated, we can assign the required tools to the role. Since we are going to use an existing Ignite-UX tool definition, we will first need to make the existing tool definition available for editing. One method of doing this is to use the **mxtool(1M)** command to query the repository for all tools in the “Ignite and Recovery” category. The following command performs just that:

mxtool -lf -c "Ignite and Recovery" >/tmp/foo

The “-lf” option instructs **mxtool(1M)** to list a formatted tool definition (suitable for editing and then re-loading). The “-c” option acts as a filter against the domain of tool definitions contained in the repository.

We now edit /tmp/foo using a text editor such as emacs or vi(1). For the purpose of our example, we only want to enable the "Refresh Recovery Archive" tool for the "IUX Operator" role. In actuality, you may want to include additional tools (either the built in tools that shipped with ServiceControl Manager or tools that you develop locally). The modified tool definition for "Refresh Recovery Archive" would now look as follows:

```
//
// This tool definition was automatically
// generated by the ServiceControl Manager
//
// Tool originally created on: Thursday, January 6, 2000 9:02:53 AM
// Tool last modified on: Thursday, January 6, 2000 9:02:53 AM MST
// File Generated: Tuesday, January 18, 2000 1:55:14 PM MST
//
SSA tool "Refresh Recovery Archive" {
    description "Refresh the managed node's recovery archive from
existing template"
    comment "The template file for the managed node is found in
/var/opt/ignite/clients/<client-name>/recovery  on  the  Ignite-UX
server."
    category "Ignite and Recovery"
    owner root
    execute {
        command "/opt/ignite/bin/make_net_recovery -v -s $MX_CMS"
        nolaunch
        log
        user root
    }
    roles {
        "Master Role",
        "IUX Operator"
    }
}
```

Note the addition of the "IUX Operator" role to the roles grammar construct. Saving this change, you would then reload the tool definitions using **mxtool(1M)** and the modify flag. The command to accomplish this is:

mxtool –m –f /tmp/foo

Now that the IUX Operator role is established, our remaining work lies in authorizing a user with the IUX Operator role, generating the initial recovery archive definitions, and setting up **cron(1M)** to perform the scheduled updates.

Recall that one of the benefits of using ServiceControl Manager is the fact that we avoid the need of opening up "root" access (through something like /.rhosts) on the managed nodes. With that in mind, let's assume that we have the user "operator" already defined in our

NIS/NIS+ passwd database. We can then let ServiceControl Manager know about the user. The command to accomplish this is:

mxuser –a –u operator

Once the user has been added on the CMS, a role authorization can be established. To enable the user with the "IUX Operator" role, you must execute the following for each managed node:

mxauth –a –u operator –M "IUX Operator" –n <managed node>

Now that the authorization has been established, the user "operator" can execute the tool "Refresh Recovery Archive" on any of the managed nodes. The tool will actually run as user "root", but the user "operator" is not required to have "root" access to the managed nodes themselves. From this point, we move on to the task of constructing the initial archive definitions. As root on the CMS, you would execute the following:

mxexec –t "Ignite-UX Console"

This will bring up the **ignite(1M)** console with all of the managed nodes visible. For each of the managed nodes, you will need to select the node and run the "Create Recovery Archive" action. The action will lead you through a brief task wizard to define the attributes and content of the archive. At the end of the wizard, the construction of the initial archive will begin.

Once this process has been performed for each of the managed nodes, you are in a position to initiate the scheduled update process for the archives. You will have noticed that task wizard gives you an opportunity to keep a configurable number of archives per managed node on the Ignite-UX server. This allows the **cron(1M)** task to perform a rolling replacement of the oldest archive.

Prior to enabling the cron task for "Refresh Recovery Archive", you must verify that the user "operator" is authorized to run the **crontab(1M)** command. This can be handled by two different mechanisms. The first is the file /usr/lib/cron/cron.allow. If this file exists, the "operator" user name must appear in the file. If this file does not exist, "operator" is permitted to use **crontab(1M)** as long as its name does not appear in the file /usr/lib/cron/cron.deny. If neither file exists, only the user "root" can use **crontab(1M)**.

After you have enabled **crontab(1M)** access for "operator", you will need to edit a file to contain an entry similar to the following:

```
0 0 * * 6 /opt/mx/bin/mxexec –t "Refresh Recovery Archive" –n
<node1> … <node25>
```

Saving the file to /tmp/operator.crontab and then running **crontab /tmp/operator.crontab** will move the file in /var/spool/cron/crontabs thus preparing it to be executed by **cron(1M)**. The schedule as specified above is every Saturday night at midnight, the "Refresh Recovery Archive" tool will be launched on the specified managed nodes.

One further note regarding network bandwidth is that the ServiceControl management environment will not execute any more than sixteen tasks in parallel. If you are concerned about the network utilization of sixteen simultaneous processes writing over NFS back to the CMS, you may want to consider breaking up the **crontab(1M)** entry so that blocks of systems will run at different times.

Conclusion

In this paper, we have attempted to introduce some of the main concepts and capabilities of Ignite-UX both as a standalone tool set and as an integrated tool set within ServiceControl Manager. The references listed in the Appendix will provide some help on where to go for more information.

If, after reading the paper and/or experimenting with Ignite-UX, you discover deficiencies (which of course you will) and areas where we could change the product to better support your use of it, please do not hesitate let us know. While defects are usually reported through your support contact, enhancement requests or comments can always be sent to: iux_enhance@fc.hp.com

Appendices

- Ignite-UX Administrator Guide

  This guide describes installing, configuring and using Ignite-UX. It is written for experienced HP-UX systems administrators who are responsible for setting up and maintaining HP-UX workstations and servers. The administrator must be familiar with installing HP 9000 hardware and software, upgrading software, applying patches and troubleshooting system problems.

  Further, it attempts to pull together under one cover much of the pertinent information related to system administration with Ignite-UX.

  Note: This guide was not available as at the time of writing of this paper. The link above takes you to the Ignite-UX web site. Look in the documentation section for a link to a PDF version of the guide. (also, HP Part Number: B2355-90677)

- Ignite-UX Cold Installations

  This paper, presented at Interworks 97, introduces the basics of Ignite-UX.

- Ignite-UX Startup Guide for System Administrators

  This is a cookbook approach to setting up and using an Ignite-UX server. The focus is on handling large replicated sites.

- Customized Install Media

  This paper, presented at Interworks 98, describes how you go about creating your own customized install media with Ignite-UX.

- Ignite-UX and Mirrored Disks

  This paper describes how to use Ignite-UX in mirrored disk situations.

- Ignite-UX System Recovery

  This paper, presented at Interworks 98, describes the basics of make_recovery.

- Frequently Asked Questions About Ignite-UX

  This is a list of frequently asked questions about Ignite-UX.

- Installing ServiceControl Manager

  This document contains all of the required information for both installing and configuring ServiceControl Manager. The web site information listed below will also contain links to access the ServiceControl Manager product.

  To access this document over the Web, browse to: www.software.hp.com. From this web site, you will want to look for the link taking you to ServiceControl or simply search for "ServiceControl Manager". There will be able to find PDF versions of all of the ServiceControl Manager documentation. Unfortunately, at the time of writing this paper the final link address was not established.

- ServiceControl Manager Technical Reference

  This document contains detailed information on how to use ServiceControl Manager.

  To access this document over the Web, browse to: www.software.hp.com. From this web site, you will want to look for the link taking you to ServiceControl or simply search for "ServiceControl Manager". There will be able to find PDF versions of all of the ServiceControl Manager documentation. Unfortunately, at the time of writing this paper the final link address was not established.