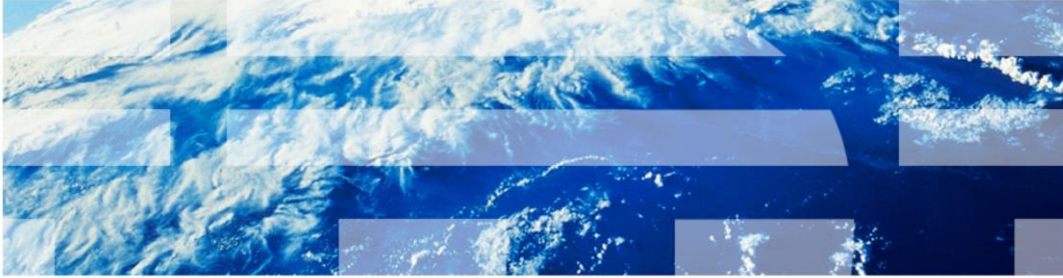


# IBM PureApplication System

## Virtual systems– Add-ons



This presentation covers the virtual system add-on support.

---

## Table of contents

- Overview
- Defining add-ons
- Deploying add-ons
- Using add-ons with the command-line

Virtual machine add-ons provide a mechanism to include advanced customizations in your virtual system patterns. Additionally, add-ons support customizing virtual hardware in your deployed virtual machines. You will see defining add-ons, deploying add-ons and the command line interface support for add-ons.

## **Overview**

The overview section introduces concepts and terminology for virtual machine add-ons and explains the default add-ons that come pre-loaded on IBM PureApplication™ System.

## Virtual machine add-ons

- Besides script packages, this is an alternate mechanism to customize virtual machine configuration
  - Fine tune hardware and OS configuration
  - New type of resource available in the appliance catalog
  - Drop add-ons onto parts while editing virtual system patterns
  - Provide add-on customization parameters at deploy-time
- Four default add-ons types available:
  - **Network:** creates and initializes a network interface
  - **Disk:** creates a new virtual disk, formats the file system and mounts the disk
  - **User account:** defines additional user – you specify the user name, password
  - **Raw disk:** creates a new virtual disk, added without partitions or formatting



4

Virtual systems – Add-ons

© 2013 IBM Corporation

Add-ons are specialized scripts that customize virtual machine configuration. Add-ons provide fine tuning for hardware and operating system configuration. Initial configuration for add-ons is provided in the administrative console from the Catalog > Add-Ons menu. You can use default add-ons, or create and clone new ones.

Add-ons are generally scripts that can apply to all VMs independent of the middleware or software on the VM.

After creating an add-on, it can be dropped onto a topology pattern part from the Patterns window. When the pattern is deployed, you provide the customization parameters for the add-ons to customize the hardware and operating system configuration.

There are four add-on types – disk, network, user, and raw disk.

The disk add-on adds a virtual disk to the virtual machine and optionally formats the file system and mounts the disk.

The network add-on adds a virtual network interface controller, or NIC, to the virtual machine. The default NIC add-on defines and initializes a new network interface. NIC add-ons must be deployed with environment profiles.

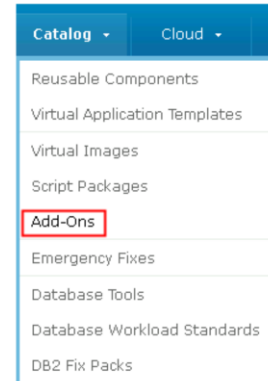
The user add-on defines an additional user on the virtual machine. The default add-on script runs a simple add user command. No additional account configuration is performed.

Finally, the raw disk add-on adds a virtual disk to the virtual machine, the disk is added raw without partitions or formatting.

The disk and NIC add-ons are not available for use with patterns that deploy to IBM PowerVM® hypervisors.

## Using add-ons (1 of 2)

- Add-ons work similar to script packages with some differences (discussed later)
- Add-ons can be viewed from the Catalog in the Workload console
- Add-ons can be cloned and modified, and new add-ons can be created from the catalog

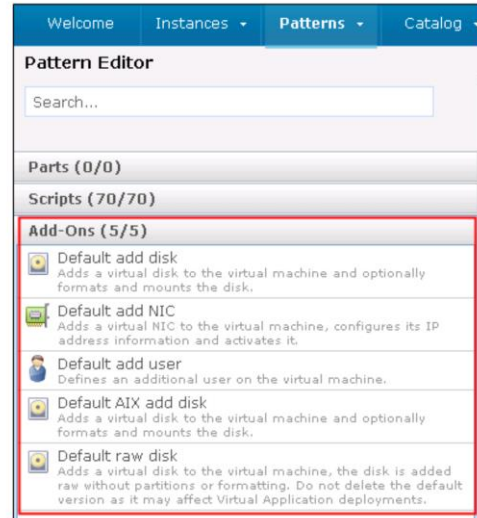


To provide user interface consistency, add-ons have been implemented modeling script packages.

Add-ons can be viewed, modified, or cloned from the Workload console “Add-ons” menu option.

## Using add-ons (2 of 2)

- Using add-ons in a pattern
  - Drag add-ons onto pattern parts in the editor
  - Specify the parameters needed for the add-on
    - during pattern creation or deployment time
      - For example, a mount point and file system format for a disk add-on, or user name and password for add-user add-on.



There is a section in the pattern editor palette called Add-ons that allows you to drag an add-on onto the parts in a pattern. Like script packages, add-ons have custom configuration parameters that are configured either during pattern creation or at deployment time.

## ***Available add-ons***

The overview section introduces concepts and terminology for virtual machine add-ons and explains the default add-ons that come pre-loaded on IBM PureApplication System.

## Add user add-on

Default add user	
Description:	Defines an additional user on the virtual machine.
Type:	User
Created on:	Mar 7, 2011 10:11:04 AM
Current status:	Read-only
Updated on:	Mar 7, 2011 10:11:04 AM
Add-on package files:	The script package is in defaultadduser.zip. <a href="#">Download</a>
Environment:	USERNAME = (no default value) PASSWORD = (no default value)
Working directory:	<a href="#">/tmp/defaultadduser</a>
Logging directory:	<a href="#">/tmp/defaultadduser</a>
Executable:	<a href="#">sh /tmp/defaultadduser/adduser</a>
Arguments:	None provided
Timeout:	120000
Access granted to:	Administrator [owner] Everyone [read] [remove] <a href="#">Add more...</a>

- There are two parameters for defining a user:
  - **User name:** user name to be added; required
  - **Password:** password for the user to be added; if not provided, no password is set
- The default add-on script runs a simple add user command
  - No additional account configuration is performed
  - Good starting point for additional customization
  - Guaranteed to run before user-level script packages

8

Virtual systems—Add-ons

© 2013 IBM Corporation

The user add-on creates a user account in the operating system with a user name and password, if one is specified. No additional configuration is done. IBM PureApplication System provides a defaultadduser.zip package that can be downloaded and customized. You can perform additional actions on the user account, for example, granting access to a specific portion of the file system or changing user permissions.

One use for this add-on is to ensure the user account is created and accessible before user-level script packages run. This is important if you require the user to be created and you are not using explicit script package ordering.

Notice the add-on interface mimics the script package interface. All the parameters are identical except the TYPE parameter near the top.

The TYPE parameter is unique to add-ons and is assigned when the add-on is created. This field cannot be changed after the creation of the add-on. In the case of the user add-on, no hardware definition operations are required.



## Add disk add-on

- There are three parameters for defining a new disk:
  - Disk size: the size of the new disk, in gigabytes
    - Default is 10
  - File system type: the format of the file system being created; must be a valid option for the `–t` argument of `mkfs`
    - Default is `ext3`
  - Mount point: location to mount the disk; the directory is created if it does not exist
    - Disk is not mounted by default

Default add disk	
Description:	Adds a virtual disk to the virtual machine and optionally formats and mounts the disk.
Type:	Disk
Created on:	May 23, 2011 9:57:30 AM
Current status:	Read-only
Updated on:	May 23, 2011 9:57:31 AM
Add-on package files:	The script package is in defaultadddisk.zip. <a href="#">Download</a>
Environment:	DISK_SIZE_GB = 10 FILESYSTEM_TYPE = ext3 MOUNT_POINT = (no default value)
Working directory:	/tmp/defaultadddisk
Logging directory:	/tmp/defaultadddisk
Executable:	sh /tmp/defaultadddisk/adddisk
Arguments:	None provided
Timeout:	1800000
Included in patterns:	(none)
In the cloud now:	(none)
Access granted to:	Administrator [owner] Everyone [read] [remove] <a href="#">Add more...</a>

The disk add-on creates a new virtual disk of size “DISK\_SIZE\_GB” and formats it to file system format specified in the “FILESYSTEM\_TYPE” parameter. The default is `ext3`. The disk is mounted at the file system location in the “MOUNT\_POINT” parameter. Notice that the Type field for disk add-on is assigned the value “Disk.”

At the hardware definition level, the disk is defined using the size specified in the “DISK\_SIZE\_GB” parameter, the default is 10 gigabytes. Remember this process is internal to the appliance, run at deployment time, and not exposed to the user. The add-on package script formats and mounts the disk. You can customize the add-on package by downloading the `defaultadddisk.zip` script and altering it to fit your environment.

One possible scenario for this add-on is to use a Logical Volume Management, or LVM. The default implementation script needs to be customized to define an LVM.

## Add raw disk add-on

- There are two parameters for defining a raw disk:
  - **Disk size:** the size of the new disk, in gigabytes
    - Default is 30
  - **File system type:** the format of the file system being created
    - Default is raw

Default raw disk	
Description:	Adds a virtual disk to the virtual machine, the disk is added raw without partitions or formatting. Do not delete the default version as it may affect Virtual Application deployments.
Type:	Disk
Created on:	May 23, 2011 9:57:32 AM
Current status:	Read-only
Updated on:	May 23, 2011 9:57:33 AM
Add-on package files:	The script package is in defaultaddrawdisk.zip. <a href="#">Download</a>
Environment:	DISK_SIZE_GB = 30 FILESYSTEM_TYPE = raw
Working directory:	/tmp/defaultaddrawdisk
Logging directory:	/tmp/defaultaddrawdisk
Executable:	sh /tmp/defaultaddrawdisk/addrawdisk
Arguments:	None provided
Timeout:	5000
Included in patterns:	(none)
In the cloud now:	d-de13e90d-c0fd-43fa-bad3-bb02a7825786
Access granted to:	Administrator [owner] Everyone [read] [remove] <a href="#">Add more...</a>

The raw disk add-on initializes the disk by invoking the hypervisor-level APIs to define the new device. This process is internal to the appliance and runs at deployment time. The disk is not formatted or mounted.

As with the disk add-on, the raw disk add-on creates a new virtual disk of size “DISK\_SIZE\_GB” but the disk is not formatted nor mounted. The default size is 30 gigabytes and the file system is raw. The Type field for raw disk add-on is assigned the value “Disk.”

You can customize the raw disk add-on package by downloading the defaultaddrawdisk.zip script and altering it to fit your environment.

## Add NIC (Network interface card) add-on

Default add NIC	
Description:	Adds a virtual NIC to the virtual machine, configures its IP address information and activates it.
Type:	NIC
Created on:	Mar 7, 2011 10:11:05 AM
Current status:	Read-only
Updated on:	Mar 7, 2011 10:11:05 AM
Add-on package files:	The script package is in defaultaddnic.zip. <a href="#">Download</a>
Environment:	(none)
Working directory:	/tmp/defaultaddnic
Logging directory:	/tmp/defaultaddnic
Executable:	sh /tmp/defaultaddnic/addnic
Arguments:	None provided
Timeout:	120000
Access granted to:	Administrator [owner] Everyone [read] [remove] <a href="#">Add more...</a>

- **Default NIC add-on defines and initializes a new network interface**
  - Not supported on PowerVM
  - Must be deployed with environment profiles
- **No parameters are exposed by default**
  - If you are using IP pinning, IP address fields are exposed for NIC add-ons during deployment
  - Otherwise, NIC is initialized and IP assigned using normal placement process

11

Virtual systems – Add-ons

© 2013 IBM Corporation

The network interface controller, or NIC, add-on defines and initializes a new network interface. NIC add-ons must be deployed using environment profiles. The rationale for requiring environment profile usage is to enable network isolation. Having two different NICs allows network isolation for two different applications in the same application server, for example, one managing internet server traffic and the other managing internal intranet server traffic.

Unlike the other add-ons, there are no environment variables associated with the NIC add-on package. The default script package can be downloaded from IBM PureApplication System and is called defaultaddnic.zip. The IP and network information for the add-on NIC is selected the same way as the default NIC that is included in all the virtual machines. If you are not using IP pinning in your environment profile, the placement algorithm will choose the network information. If you are using IP pinning, then you will set the IP address for the add-on NICs at deployment time.

IBM PowerVM hypervisors do not support NIC add-ons.

## Example: Deployment dialog for NIC add-on with IP pinning

Default VM network interface, eth0

Fill in the required values for this part of the pattern.

Name:

\* In cloud group:

\* IP Group (virtual machine 1 network interface 0):

\* Addresses (virtual machine 1 network interface 0):

\* IP Group (virtual machine 1 network interface 1):

\* Addresses (virtual machine 1 network interface 1):

\* Virtual CPUs:

\* Memory size (MB):

Network interface being created for the add-on, eth1

12      Virtual systems—Add-ons      © 2013 IBM Corporation

Here is an example of the virtual part configuration page that you must fill out at deployment. If your environment profile specifies deployer-assigned IP addresses, then you are required to provide the IP addresses for all NICs. The first IP address section is for “virtual machine 1 network interface 0” and will correspond to the eth0 interface inside the virtual machine. This is the default NIC address and is always present.

The second highlighted section of IP information is for the add-on NIC. This will correspond to eth1 in the deployed virtual machine. You are required to provide this information for each NIC that you have added.

If you are not using IP pinning in your environment profile, then you are not required to provide the IP addresses and the IP assignment will happen using the standard placement algorithm.

## Example: Hardware configuration performed by the appliance

- Add-ons use hypervisor-level APIs to configure new hardware in VMs at deployment time
- **Example:** Multiple NICs defined on the VM page

Hardware and network	
Virtual CPU count:	1 (You must stop this virtual machine in order to change this value.)
CPU shares on host:	1000
CPU shares consumed on host:	0.0
Virtual memory (MB):	2048 (You must stop this virtual machine in order to change this value.)
SSH public key:	id_rsa.pub
Network interface 0:	aimcp154.austin.ibm.com (9.3.75.154)
MAC address 0:	00:50:56:84:00:c7
Network interface 1:	aimcp153.austin.ibm.com (9.3.75.153)
MAC address 1:	00:50:56:84:00:c8

Default NIC, eth0

Add-on NIC, eth1

13

Virtual systems—Add-ons

© 2013 IBM Corporation

Once a system is deployed, you can view the NIC information on the virtual systems page, under the virtual machine hardware and network section. The network interface 0 field is the eth0 interface in the virtual machine and corresponds to the default NIC. The network interface 1 field is the add-on NIC and corresponds to eth1.

## Differences between add-ons, script packages

- Add-ons always run at system creation time, never user-initiated or at deletion
- Add-ons are not orderable
- Add-ons have a type field that triggers special processing in the deployment code to add the additional hardware to the VM
- Add-ons and scripts show up in different places in the web console
  - Menus
  - Pattern editor
  - Palette
- The command-line interface object is different
  - `deployer.addons` versus `deployer.scripts`

Although add-ons and script packages have very similar configuration steps, there are some notable differences. While script packages have multiple execution-time options that are specified by the user, such as deploy time, delete time, or user initiated, add-ons do not. Add-ons only run at deployment time. You do not specify the order of execution of the add-ons in the pattern editor. However, add-ons always run before any user-supplied script package. The add-on definition has a special TYPE field that triggers the appropriate hypervisor-level API calls to be invoked to create the relevant hardware for disk or NIC creation.

Finally, although the interface for add-ons mimics script packages, they are their own entities and are located in different locations in the user interface. Both have different command sets.

## ***Defining add-ons***

This section will discuss step-by-step instructions for defining add-ons in the appliance.

## Default add-on implementation

- Add-ons are implemented at two levels:
  - Deployment logic to provision new hardware
    - For disk, NIC only; not user-customizable
  - Deployment logic to configure the provisioned resources
    - Packaged as default implementation scripts; fully customizable
    - **Example:** Default user add-on has a defaultadduser.zip package that contains a script to define a user account
- Create custom add-ons by downloading and modifying the add-on package to use in new or cloned add-ons:

Add-on package files:      The script package is in defaultadduser.zip. [Download](#)

Before discussing default implementation customization, it is important to remember there are two levels of add-on implementation. The first is the deployment logic used to provision new hardware. This is transparent to you and is not customizable. This is for disk or NIC creation.

The second is the deployment logic used to configure the provisioned resources. This piece of the configuration is customizable. IBM PureApplication System exposes the add-on script packages for each pre-loaded add-on. There is one script package per add-on that can be downloaded from the administrative console and used as a starting point for customization. You can modify the default add-on script package and upload it to your own add-on, resulting in a customized add-on implementation. The default script packages are documented to assist you during the customization process.



## Default add-on implementation package content – Configuration

- Each default add-on package contains:
  - **cbscript.json file that defines configuration data and environment variables**
  - Script file for the default add-on implementation

Sample  
cbscript.json  
content for the  
default disk  
add-on

```
[{
  "description": "RM05013",
  "command": "sh /tmp/defaultaddisk/adddisk",
  "log": "/tmp/defaultaddisk",
  "location": "/tmp/defaultaddisk",
  "timeout": 1800000,
  "type": "ADDON_DISK",
  "keys": [
    {
      "scriptkey": "DISK_SIZE",
      "scriptvalue": "10",
      "scriptdefaultvalue": "10"
    },
    {
      "scriptkey": "FILESYSTEM_TYPE",
      "scriptvalue": "ext3",
      "scriptdefaultvalue": "ext3"
    },
    {
      "scriptkey": "MOUNT_POINT",
      "scriptvalue": "",
      "scriptdefaultvalue": ""
    }
  ]
}]
```

Special parameter to  
identify the add-on  
type (disk, NIC, user)

Deploy-time  
parameters for  
the add-on

The next few pages step through an example of the default implementation package for the disk add-on. Each add-on has its own default implementation package.

The format for the add-on implementation package is the same as for script packages, each package contains a cbscript.json file and the script itself. On this page you will examine the cbscript.json file.

You will notice the cbscript.json file uses the same general format as script packages, with the exception of the TYPE parameter. Again, the TYPE parameter is specific to add-ons. If the add-on has additional environment variables, they are listed here as keys. These keys are exposed at deployment time.

## Default add-on implementation package content – Script (1 of 2)

- Each default add-on package contains:
  - cbscript.json file that defines configuration data and environment variables
  - **Script file for the default add-on implementation**

### First part of default **adddisk** script

Locate the uninitialized disk that the appliance created

```
echo "**** searching for uninitialized disk of size
${DISK_SIZE_GB}GiB"
DEVICE=$(sfdisk -s | grep $((($DISK_SIZE_GB * 1024 * 1024))'$' | sed -e
's/:.*//'| sort | while read dev do if ! parted $dev print >/dev/null
then echo $dev break fi fi
done)
```

There is a script package associated with each add-on implementation package.

Shown here is the first part of the add-on script.

For this example, the disk add-on, the script initially searches for the uninitialized disk that the appliance created. The amount of space required was specified in the DISK\_SIZE\_GB parameter.

## Default add-on implementation package content – Script (2 of 2)

Second part of default **adddisk** script

```

if [ -n "$DEVICE" ]
then
  echo "*** partitioning disk at ${DEVICE}"
  parted $DEVICE mktable msdos
  LAST_SECTOR=$(fdisk -l -u $DEVICE | sed -e '/total [0-9][0-9]* sectors!/d' -e
's/. *total \([0-9][0-9]*\) sectors.*\/\1/')
  parted $DEVICE mkpart primary 63s $((LAST_SECTOR - 1))s
  sync
fi

if [ -n "$FILESYSTEM_TYPE" ]
then
  echo "*** formatting ${DEVICE}1 with ${FILESYSTEM_TYPE} filesystem"
  mkfs -t "$FILESYSTEM_TYPE" ${DEVICE}1
fi

if [ -n "$MOUNT_POINT" ]
then
  echo "*** mounting ${DEVICE}1 at ${MOUNT_POINT}"
  mkdir -p "$MOUNT_POINT"
  echo "${DEVICE}1 ${MOUNT_POINT}          ${FILESYSTEM_TYPE} auto 1 2" >>/etc/fstab
  mount "$MOUNT_POINT"
fi
fi

else
  echo "*** could not determine disk device" >&2
  exit 1
fi

```

Partition the disk

Format the file system

Mount the disk

Once the disk space has been found, the script continues by partitioning the disk, formatting the file system, and mounting the file system at the mount point.

The file system type and the mount point are parameters specified in the add-on `cbscript.json` configuration file. The file system type given must be valid for the `-t` parameter of the `mkfs` command. These can vary across images and patterns based on the virtual machine configuration.

## Cloning an add-on

- Go to the add-on details page for the add-on you want to clone
  - Catalog->Add-ons
- Use **Clone** button to make a copy of the add-on

Default add user <span style="float: right;">1</span>	
Description:	Defines an additional user on the virtual machine.
Type:	User
Created on:	Mar 7, 2011 10:11:04 AM
Current status:	Read-only
Updated on:	Mar 7, 2011 10:11:04 AM
Add-on package files:	The script package is in defaultadduser.zip. <a href="#">Download</a>
Environment:	USERNAME = (no default value) PASSWORD = (no default value)

A new add-on will be created with all of the same files and fields.

\* Name:  2

Second, provide a name for the cloned add-on

The first option to create a new add-on is to clone an existing one. From the administrative console select Catalog-> Add-ons. A list of valid add-ons is displayed on the left. From the add-on list, select the add-on you want to clone, click the clone button and provide a name for your new add-on.

Since the default add-ons are locked and cannot be modified, you must make a copy of the add-on before you can customize for your environment.

## Customizing a cloned add-on

**Custom user add-on**

Description: Defines an additional user on the virtual machine.

Type: User

Created on: Mar 30, 2011 3:03:20 PM

Current status: Draft

Updated on: Mar 30, 2011 3:03:21 PM

Add-on package files:

The script package is in defaultadduser.zip.

Environment: USERNAME = (no default value) [remove]  
PASSWORD = (no default value) [remove]  
Add variable  =

Working directory: /tmp/defaultadduser

Logging directory: /tmp/defaultadduser

Executable: sh /tmp/defaultadduser/adduser

Arguments: None provided

Timeout: 120000

Access granted to:

**Cloned add-ons are pre-populated with the original add-on configuration data; you can:**

- Provide a custom add-on package, using the same format as script packages
- Optionally download and modify the default add-on implementation
- Remove existing environment variables or create new ones
- Configure standard script package parameters for working and logging directories, executable, arguments, timeout
- Set add-on permissions

21 Virtual systems - Add-ons © 2013 IBM Corporation

When an add-on is cloned, all the configuration parameters from the original add-on are carried forward. Shown here is an example of a cloned user add-on.

The newly cloned add-on is unlocked and ready to be modified. The add-on package files can be downloaded, modified for your environment, and uploaded to your cloned add-on. Once the modified add-on package files have been uploaded, click the refresh button in the administrative console to view your changes.

The environment variables are populated from the cbscript.json file, but can be modified manually from the administrative console. All manual modifications will override the values from the cbscript.json file.

## Create an empty add-on

- From the **Add-Ons** page in the **Catalog**, click the **+** create a new empty add-on
- Provide an **Add-on name** and **Type** (Disk, NIC, User)

Describe the add-on you want to load into the catalog.

\* Add-on name:

\* Type:   
Disk  
NIC  
User

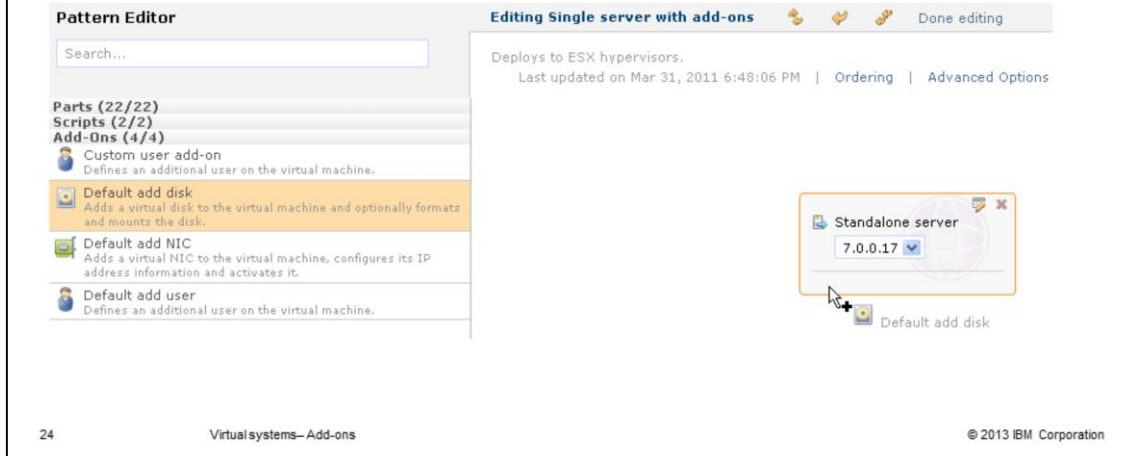
You can create a new add-on by selecting the green plus sign from the Add-ons page. You must specify the name of your add-on and the type. There are three types to choose from, disk, NIC, or user. The type you choose will determine the underlying hardware-level definition actions that the add-on performs.

## ***Deploying add-ons***

This section will discuss step-by-step instructions for deploying add-ons in the appliance.

## Using add-ons in a pattern

- In the **Pattern Editor**, there is a new **Add-ons** section in the palette
- Drag and drop add-ons from the palette to the pattern part on the canvas
  - Same interaction model as script packages



In order to deploy the add-on you just created, you must add it to your pattern. There is a new section in the pattern editor palette for add-ons.

As with script packages, you drag the add-on from the palette on the left to the pattern part on the right.



## Optionally define add-on parameters in the pattern

**Note:** The default NIC add-on does not have any configurable parameters

Parameters for Default add disk

DISK\_SIZE\_GB:

FILESYSTEM\_TYPE:

MOUNT\_POINT:

OK Cancel

Parameters for Default add user

USERNAME:

PASSWORD:

Verify password:

OK Cancel

25 Virtual systems—Add-ons © 2013 IBM Corporation

Once the add-on has been added to your pattern, you have the option to configure any environment parameters associated with that add-on. Click the pencil icon located in the add-on graphic and customize the values to represent your environment. Once a value is entered, you have the option to lock the parameter for your add-on. Shown here are the parameters for the default disk and user add-ons.

The NIC add-on does not have any configurable parameters. The network information is assigned according to the placement algorithm for the deployment target.

Notice that each add-on has a different graphical representation: a disk device for the disk add-on, a network card for the NIC add-on, and a person for the user add-on.

Finally, notice in the palette editor that the add-ons are located above the user-defined script packages. In the example above, the add disk and add user add-on icons are located above the HTTP server script package. This graphical representation is a reminder that the add-ons are invoked before the user-defined script packages.

## Deployment dialog

When deploying this pattern, you are prompted for add-on parameters in the deployment dialog

Standalone server  
7.0.0.17

Fill in the required values for this part of the pattern.

- WebSphere administrative user name: virtuser
- WebSphere administrative password: .....
- Verify password: .....
- DISK\_SIZE\_GB:** 7
- FILESYSTEM\_TYPE:** xfs
- MOUNT\_POINT:** /my-fs
- USERNAME:** addon\_user
- PASSWORD:** .....
- Verify password: .....

OK Cancel

26 Virtual systems—Add-ons © 2013 IBM Corporation

In this example, you are deploying a single server that contains a user add-on and a disk add-on. The parameters for the add-ons are exposed in the deployment dialog. For the user add-on, you must specify a user name, in this case `addon_user`, and a password.

For the disk add-on, you provide a disk size, a file system format, and a mount point. In this case, you are creating a 7-gigabyte disk with the file system format “xfs”, mounted at `/my_fs`.

These add-on parameter values are referenced in the rest of the examples in this section.

## Empty add-on configuration

Empty add-ons do not contain any default configuration data; they must be customized before use

My custom add-on	
Description:	<a href="#">None provided</a>
Type:	Disk
Created on:	Mar 30, 2011 2:59:45 PM
Current status:	Draft
Updated on:	Mar 30, 2011 2:59:45 PM
Add-on package files:	<input type="text" value="Browse..."/> <input type="button" value="Upload"/> There are no files for this add-on.
Environment:	(none) Add variable <input type="text" value="name"/> = <input type="text" value="value"/> <input type="button" value="Add"/>
Working directory:	<a href="#">/tmp</a>
Logging directory:	<a href="#">None provided</a>
Executable:	<a href="#">None provided</a>
Arguments:	<a href="#">None provided</a>
Timeout:	<a href="#">6000</a>
Access granted to:	Administrator [owner] <input type="text" value="Add more..."/>

27

Virtual systems - Add-ons

© 2013 IBM Corporation

When you create an add-on from scratch, there are no default parameters and no default implementation package. The add-on configuration is truly empty. The add-on must be customized before use.

Once you have configured your add-on, the add-on can be locked – that is, marked read-only - and published.

## Deployment results

History		The virtual system has been deployed and is ready to use
	The virtual system has been deployed and is ready to use	Mar 31, 2011 7:30:12 PM
	Executing script package Default add user on virtual machine Single server with add-ons aimcp143 Standalone	Mar 31, 2011 7:29:42 PM
	Executing script package Default add disk on virtual machine Single server with add-ons aimcp143 Standalone	Mar 31, 2011 7:29:27 PM
	Starting virtual machine Single server with add-ons aimcp143 Standalone	Mar 31, 2011 7:21:24 PM
	Starting virtual machines	Mar 31, 2011 7:21:24 PM
	Registering virtual system Single server with add-ons	Mar 31, 2011 7:20:55 PM
	Transferring virtual images to hypervisors	Mar 31, 2011 7:20:39 PM
	Generating model for topology and network	Mar 31, 2011 7:20:24 PM
	Reserving cloud resources	Mar 31, 2011 7:20:09 PM
	Deployment has been queued	Mar 31, 2011 7:19:38 PM

Virtual system History shows when add-on scripts are invoked

Script Packages			
Default add user	✓	Mar 31, 2011 7:29:48 PM	remote_std_out.log remote_std_err.log cloudburst_collect1301617788140.zip
Default add disk	✓	Mar 31, 2011 7:29:35 PM	remote_std_out.log remote_std_err.log cloudburst_collect1301617774806.zip
WebSphere Hypervisor Edition Startup Logs	✓	Mar 31, 2011 7:30:06 PM	remote_std_out.log remote_std_err.log cloudburst_collect1301617801801.zip

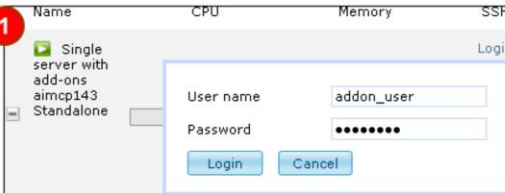
Virtual machine details show add-on results in Script Packages section

During the deployment process, add-on script packages are executed. The virtual system history lists each step as it runs. Shown here is the history for the invocation for the disk and user add-on script packages.

Once complete, the add-on script execution results and log files are available. Expand the virtual machine section, scroll to the bottom and you will see the results of your add-on script packages. In this case, the green check mark status means that the add-ons ran successfully and should be working correctly. If the add-on script packages fail, a red X status indicator is displayed. In this case you can open the related log files to view details of the error.


## Verify user configuration – Log in to the operating system

Open an SSH session, using the credentials defined in the user add-on



The screenshot shows a table with columns for Name, CPU, Memory, and SSH. A red circle with the number '1' is next to the first row. A modal dialog box is open over the table, titled 'Login'. It contains a 'User name' field with 'addon\_user' entered, a 'Password' field with masked characters, and 'Login' and 'Cancel' buttons.

Verify the identity of the current user



```
IBM WebSphere Application Server Hypervisor Edition
addon_user@aimcp143: ~->
addon_user@aimcp143: ~-> whoami
addon_user
addon_user@aimcp143: ~->
```

A red circle with the number '2' is next to the terminal output.


Now that the virtual system is deployed and the add-on script packages have started successfully, you can go to the virtual machine section of your virtual system and verify that your add-ons are working.

For the user account, log in to an SSH session using the credentials that you specified in the user add-on configuration. Shown here is the login for the browser-based SSH console available from the IBM PureApplication System user interface. Log in with the user account you created, in this case `addon_user`. Once authenticated, run the `whoami` command to verify that you really are logged in with the add-on user account.

## Verify disk configuration – List raw devices

- Inside this WebSphere® Application Server Hypervisor Edition VM for ESX, there are four default disks:
  - /dev/sda is the operating system disk (12GB)
  - /dev/sdb is the application server binaries disk (8GB)
  - /dev/sdc is the application server profiles disk (2GB)
  - /dev/sdd is the IBM HTTP Server disk (2GB)
- The device /dev/sde is the new disk created by the disk add-on (7GB)

```
aimcp143:~ # sfdisk -s
/dev/sda: 12582912
/dev/sdb: 8388608
/dev/sdc: 2097152
/dev/sdd: 2097152
/dev/sde: 7340032
total: 32505856 blocks
```



To verify the add-on disk configuration, it is important to understand the default configuration. For WebSphere Application Server Hypervisor Edition, there are four default disks, one for the operating system, one for the application server binary files, one for profiles, and one for the HTTP server. The device names and default sizes are shown here.

The first new disk you create with the disk add-on is created on device /dev/sde. Shown here, you can see that /dev/sde exists and is 7 gigabytes in size, which is what was specified during the configuration.

To view the list of all the disk devices, along with their sizes, enter the command: `sfdisk -s`. This validates that your disk exists and it is the correct size.

## Verify disk configuration – Show mounted disks

- The /dev/sde disk is mounted on /my\_fs and has a file system type of xfs

```
aimcp143:~ # mount
/dev/sda2 on / type ext3 (rw,ac1,user_xattr)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
debugfs on /sys/kernel/debug type debugfs (rw)
devtmpfs on /dev type devtmpfs (rw,mode=0755)
tmpfs on /dev/shm type tmpfs (rw,mode=1777)
devpts on /dev/pts type devpts (rw,mode=0620,gid=5)
/dev/sdb1 on /opt/IBM/WebSphere/AppServer type ext3 (rw,ac1,user_xattr)
/dev/sdc1 on /opt/IBM/WebSphere/Profiles type ext3 (rw,ac1,user_xattr)
/dev/sdd1 on /opt/IBM/HTTPServer type ext3 (rw,ac1,user_xattr)
fusectl on /sys/fs/fuse/connections type fusectl (rw)
/proc on /var/lib/ntp/proc type none (ro,bind)
gvfs-fuse-daemon on /home/virtuser/.gvfs type fuse.gvfs-fuse-daemon
(rw,nosuid,nodev,user=virtuser)
/proc on /var/lib/ntp/proc type none (ro,bind)
/dev/sde1 on /my_fs type xfs (rw)
```




During this configuration, you asked for a 7-gigabyte disk of file system format “xfs”, mounted at /my\_fs. You have verified you have a 7-gigabyte disk with device name /dev/sde. The mount command will list all the mounted disks, not just virtual disks, and display some configuration information. Shown here are the results of the mount command. Notice the disk that was created is mounted at /my\_fs and is of type “xfs”.

## Verify disk configuration – Check available space

- Go into the directory where the disk is mounted and list the available space
  - During deployment, **7GB** was requested at **/my\_fs**

```
aimcp143:~ # cd /my_fs
aimcp143:/my_fs # df -h .
Filesystem Size Used Avail Use% Mounted on
/dev/sde1 7.0G 4.2M 7.0G 1% /my_fs
aimcp143:/my_fs #
```



Finally, navigate to the mount point, in this case `/my_fs`, enter the `df -h` command and verify that there is 7 gigabytes of usable space.



## ***Using add-ons with the command-line***

This section walks through a command line interface example showing how to create and use add-ons.

## Creating an add-on with the command-line interface

- **Example:** Create an add-on, then list attributes

```
>>> myaddon = deployer.addons << { 'name':'CLI add-on', 'type':
deployer.DISK_ADDON }
>>> myaddon
{
  "acl": (nested object),
  "archive": (nested object),
  "command": "",
  "commandargs": "",
  "created": Apr 8, 2011 5:38:07 PM,
  "currentstatus": "RM01027",
  "currentstatus_text": "Draft",
  "description": None,
  "environment": (nested object),
  "id": 74,
  "label": "CLI add-on",
  "location": "/tmp",
  "log": "",
  "name": "CLI add-on",
  "owner": (nested object),
  "timeout": 60000000,
  "type": "ADDON_DISK",
  "updated": Apr 8, 2011 5:38:07 PM
}
```

All command line interface operations are based on the `deployer.addons` object. In the example shown here, all lines that start with three greater-than symbols represent commands that run in the command line interface. The command shown here creates a new add-on called “CLI add-on” and is of type `DISK_ADDON`. The attributes for the disk add-on are then listed.

## Modifying an add-on with the command-line interface

- **Example:** Modify add-on attributes and set the add-on package

```
>>> myaddon.description
>>> myaddon.description = 'New add-on from the CLI'
>>> myaddon.description
'New add-on from the CLI'
>>> myaddon.archive.set(open('adddisk.zip'))
>>> myaddon.environment
{
  "DISK_SIZE_GB": "10",
  "FILESYSTEM_TYPE": "ext3",
  "MOUNT_POINT": ""
}
>>> myaddon.environment['FILESYSTEM_TYPE'] = 'raw'
>>> myaddon.environment
{
  "DISK_SIZE_GB": "10",
  "FILESYSTEM_TYPE": "raw",
  "MOUNT_POINT": ""
}
>>>
```

Once the add-on is created, you will need to configure the environment.

The first command shown here lists the add-on description. Since you did not include a description during creation, the initial description is blank. You are able to set the description field to “New add-on from the CLI”, re-run the description command and see the newly-defined value displayed.

The `myaddon.archive.set(open('adddisk.zip'))` command defines the implementation package for the add-on.

Now that you have defined an implementation package for the add-on, all the default parameters in the implementation package configuration file are automatically pulled into the add-on definition. These come from the `cbscript.json` file. The environment command displays the current values for the disk size, file system and mount point. You can override the default values by assigning new values from the command line.

The example shown changes the `FILESYSTEM_TYPE` to “raw” instead of the default value “ext3”. After displaying the current values, notice that the file system type is now raw.

## Using add-ons in a pattern with the command-line interface

- **Example:** Create a pattern that includes an add-on

```
>>> mypattern = deployer.patterns << { 'name' : 'CLI add-on pattern' }
>>> myimage = deployer.virtualimages.WebSphere[0]
>>> standalone = mypattern.parts << myimage.parts.Standalone[0]
>>> standalone.scripts << myaddon
{
  "description": "Adds a virtual disk to the virtual machine and optionally
formats and mounts the disk.",
  "executionOrder": 0,
  "id": 1,
  "isdeletable": True,
  "label": "CLI add-on",
  "parameters": (nested object),
  "part": (nested object),
  "startsafter": (nested object),
  "type": "ADDON_DISK"
}
>>>
```

Now that you have created and configured an add-on, you will need to add the add-on into a pattern.

The first command shown here, `deployer.patterns`, creates a new empty pattern.

The second command selects the virtual images that are used in the pattern.

The third command adds the stand-alone server part from the image to the empty pattern. In this example, “standalone” contains a reference to the pattern with a single server part.

Finally, the `scripts` command associates the add-ons with the part.

The pattern with the add-on is ready to deploy.

Section

## ***Summary***

This section summarizes the add-on feature.

## Summary

- Virtual machine add-ons offer a new mechanism to customize virtual machine configuration
  - Augment hardware and OS configuration
  - New type of resource available in the appliance catalog – create from scratch or clone / customize
  - Drop add-ons onto parts while editing infrastructure patterns
  - Provide add-on customization parameters at deploy-time
- Four built-in add-on types: disk, raw disk, NIC, user
- Work with add-ons using the web interface or command-line interface

This presentation discussed virtual machine add-ons that are included in a pattern to provide a mechanism for advanced customizations. There are four supported add-ons, disk, raw disk, NIC, and user. You covered defining add-ons, deploying add-ons, and the command line interface support for add-ons.

---

## Trademarks, disclaimer, and copyright information

IBM, the IBM logo, ibm.com, PowerVM, PureApplication, and WebSphere are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of other IBM trademarks is available on the web at "[Copyright and trademark information](http://www.ibm.com/legal/copytrade.shtml)" at <http://www.ibm.com/legal/copytrade.shtml>

Other company, product, or service names may be trademarks or service marks of others.

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM'S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS OR SOFTWARE.

© Copyright International Business Machines Corporation 2013. All rights reserved.