HP CloudSystem Matrix: Collecting Usage Data for Show-back, Charge-back, and Billing Purposes

Technical white paper

Table of contents

Abstract 2
Overview 2
CloudSystem Matrix 4
The CloudSystem Matrix Service Catalog and Charge-back Data 4
Operations on a Service and Charge-back 6
The pull approach using the CloudSystem Matrix API/CLI 8
The push approach using CloudSystem Matrix Workflows 9
Optional Enhancements 10
Pull versus Push? 10
Pulling charge-back data from CloudSystem Matrix using the CLI/API 10
Accessing service information through APIs and CLIs 11
Working with the listServices call 11
Working with the getService call 12
Pushing charge-back data from Matrix 14
Customizing service deployment behavior with workflows 14
Service-specific workflows 15
Global workflows 16
Service specific or global workflows for charge-back? 18
Extracting utilization data for a service from Matrix 18
Augmenting service information with utilization information 19
Summary 19
For more information 21

Abstract

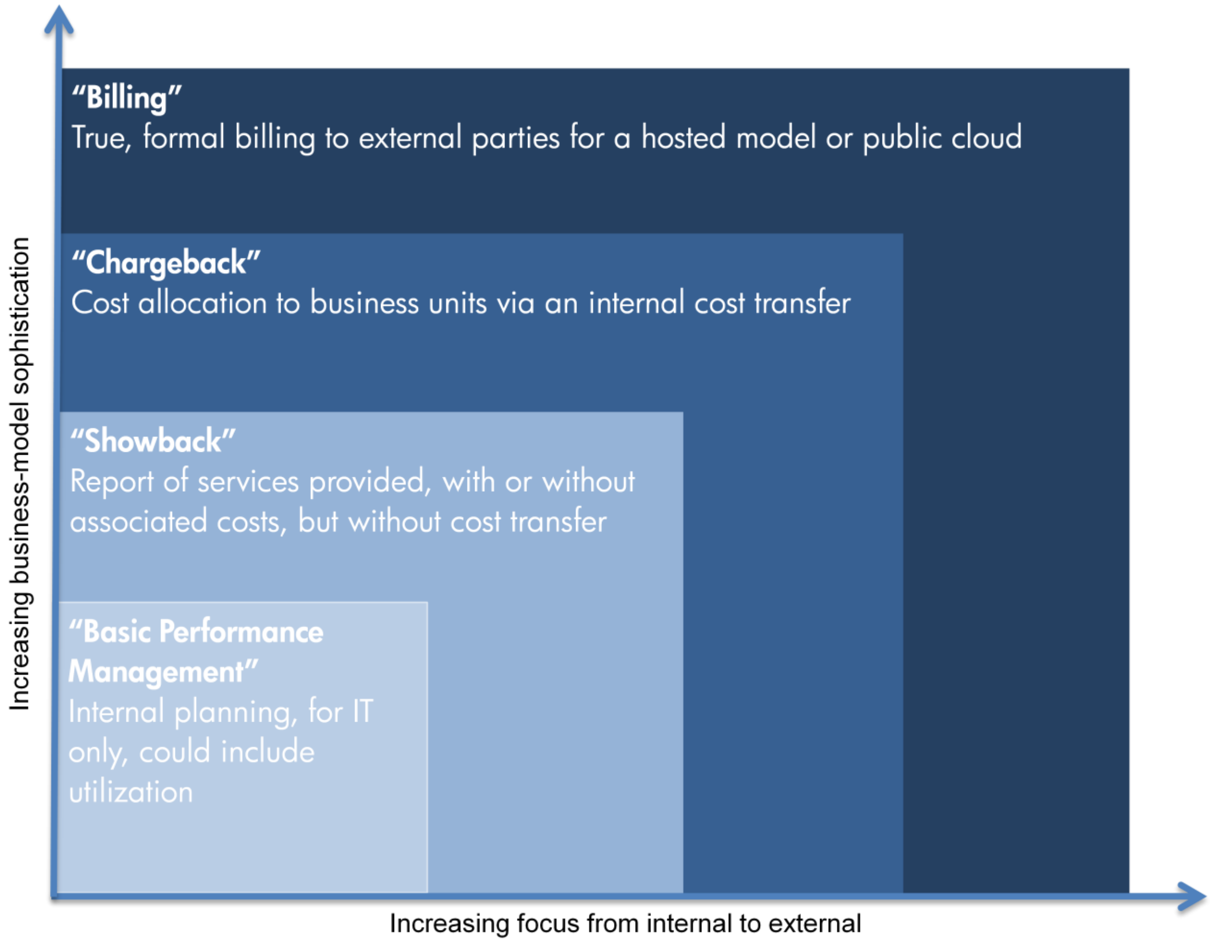
HP CloudSystem Matrix is the ideal platform for private cloud and infrastructure-as-a-service (IaaS). It enables the provisioning of infrastructure and applications in minutes rather than months, resulting in greater agility and speed, as well as TCO savings. CloudSystem Matrix offers a self-service infrastructure portal for auto-provisioning and built-in lifecycle management to optimize infrastructure, monitor applications, and ensure uptime for cloud and traditional IT. A common requirement for a cloud solution is to have a clear understanding of usage data for show-back, charge-back, or billing purposes. The charge-back process that allows IT to allocate costs to internal users or cost locations for their services, while the billing process allows creation of invoices to external parties. This paper describes some tools and techniques that can be used to extract usage data in a Matrix environment for show-back, charge-back, and/or billing purposes. It describes the services model in Matrix, the attributes that would be of interest in a cloud environment, and the APIs and CLIs to extract the related usage data. It also describes two different architectural approaches in gathering this data into a database for report generation. This paper concentrates on the *extraction* of data for show-back, charge-back, and billing purposes, but it does *not* describe the details needed for the actual charge-back and billing process itself.

Overview

HP CloudSystem Matrix is the ideal platform for private cloud and infrastructure-as-a-service (IaaS). It enables the provisioning of infrastructure and applications in minutes rather than months, resulting in greater agility and speed, as well as TCO savings. CloudSystem Matrix offers a self-service infrastructure portal for auto-provisioning and built-in lifecycle management to optimize infrastructure, monitor applications, and ensure uptime for cloud and traditional IT. The underlying physical servers in CloudSystem Matrix are HP BladeSystem servers, which can include physical and virtual resource pools. In addition, Matrix can also manage Matrix HP rack-mount servers and non HP servers as virtual resource pools. Supported Hypervisors include VMware ESX, Microsoft Hyper-V and HP Integrity Virtual Machines. For more information about CloudSystem Matrix see www.hp.com/go/matrix .

Charge-back is the practice of charging business units for the IT services they use. Taken a step further, service providers need to actual bill for services used -- literally presenting a bill to the end customer. In many other situations, IT only needs to simply provide a periodic report of services provided. This later practice is often referred to as 'show-back'. The following table illustrates the various terminology used in the spectrum of 'charge-back' sophistication.

Figure 1: Levels of 'Charge-back' sophistication



This paper will focus exclusively on the aspect of service usage and utilization data along with its user association. Additional details in authorization, approach to cost centers etc. and other general charge-back and billing processes are not covered.

The most straight forward model for usage is to report or charge based on the service with a fixed cost which was provided to the end customer/user. An example would be a customer/user that is charged for a virtual server with 2 cores and 8 GB of memory and associated network and storage resources. The charge would be independent of how actively the virtual server was used. Alternatively, IT organizations may charge based on the amount of utilization. For example, charging a customer based on the average CPU utilization of a server over the billing period. One or both of these approaches can be used with Matrix and will be covered in this paper.

CloudSystem Matrix itself does not include a charge-back and/or billing mechanism. On the other hand it does include features to facilitate integration with external charge-back and/or billing solutions.

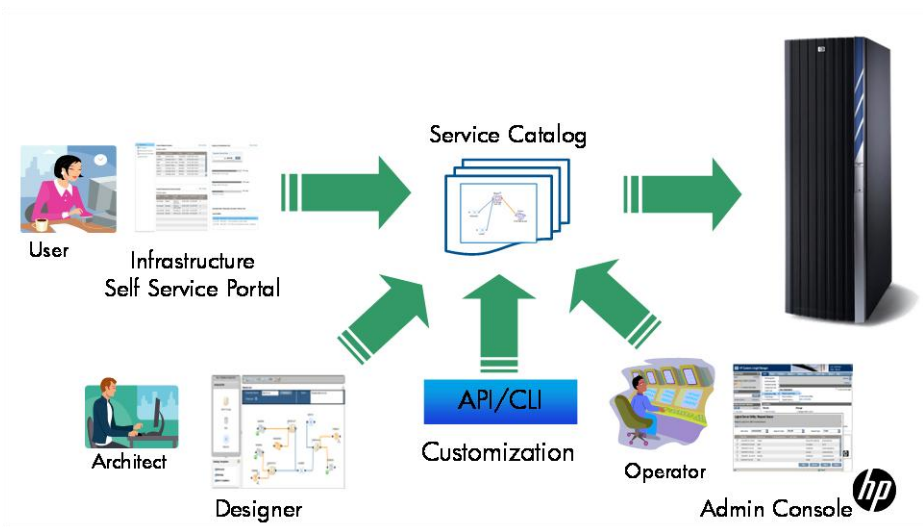
- APIs, CLIs and customizable workflows provide a means for external programs to capture and store charge-back/billing information
- Charge-back/billing related customer selectable fields are provided in the service catalog and user self service portal. These can be used to set and track cost and billing codes of services

CloudSystem Matrix

The CloudSystem Matrix Service Catalog and Charge-back Data

CloudSystem Matrix has a service catalog model which is based on the concept of a CloudSystem Matrix *template*. Each template describes a unit of infrastructure containing servers, storage, and network. IT Architects create templates and publish them in the Service Catalog. IT users browse the catalog of templates, select the one which they need and request an instance of it through a Self Service Portal. The now running infrastructure based on the template is a CloudSystem Matrix *service*. Although you can define the cost on a component basis, for charge-back in a CloudSystem Matrix environment, the basic unit of granularity is a service (once the costs are rolled-up to the entire service).

Figure 2: HP CloudSystem Matrix service catalog interfaces



When the IT Architect creates the template, they specify basic computing properties such as physical or virtual servers or a combination of both, the compute properties of the servers such as number of processors and memory, the OS image to be loaded, network connections, storage and SAN connections. In addition, they also can set optional attributes for a charge-back solution.

- cost values of individual components – servers, storage and network connections
- a cost value associated with the top level template itself

Figure 3 shows a simple service template of a single Virtual Machine. The component details are expanded to show the cost fields available to the IT Architect.

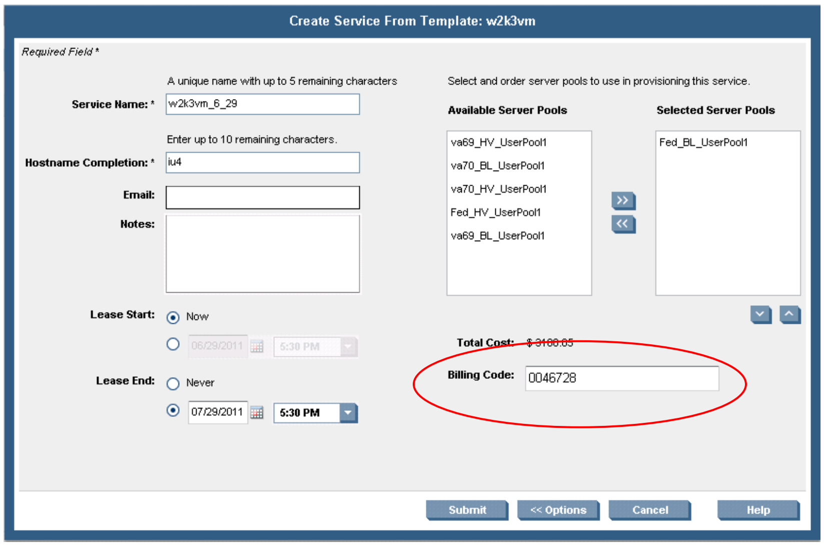
Figure 1: Cost fields of a service catalog entry

In addition, when a user requests a service through the portal, as of Matrix version 6.3, they have the option of entering a value which can be used for charge-back or billing:

- billing code- representing a department, cost center or customer for billing purposes

The following figure shows the billing code field in the Self Service Portal

Figure 2: Self Service Portal Request with Billing Code



Once the service has been created as the result of a request, it will have the attributes of the template plus those of the user and their billing code.

What happens if the user does not enter a billing code? The CloudSystem Matrix authorization processing by default requires that an IT administrator approves the request. At that time the existence of a billing code can be checked. For automated approvals the CloudSystem Matrix approval workflow can be customized to check and require a billing code.

At creation time a service can also have an optional lease period defined. The service will continue to exist and run until either the user explicitly requests that it should be deleted, or the lease expires. When the lease expires, the service is automatically placed in an offline state until either the lease is extended or the user requests that it is deleted.

Operations on a Service and Charge-back

Users can request a number of operations on a service, the implications for charge-back are listed:

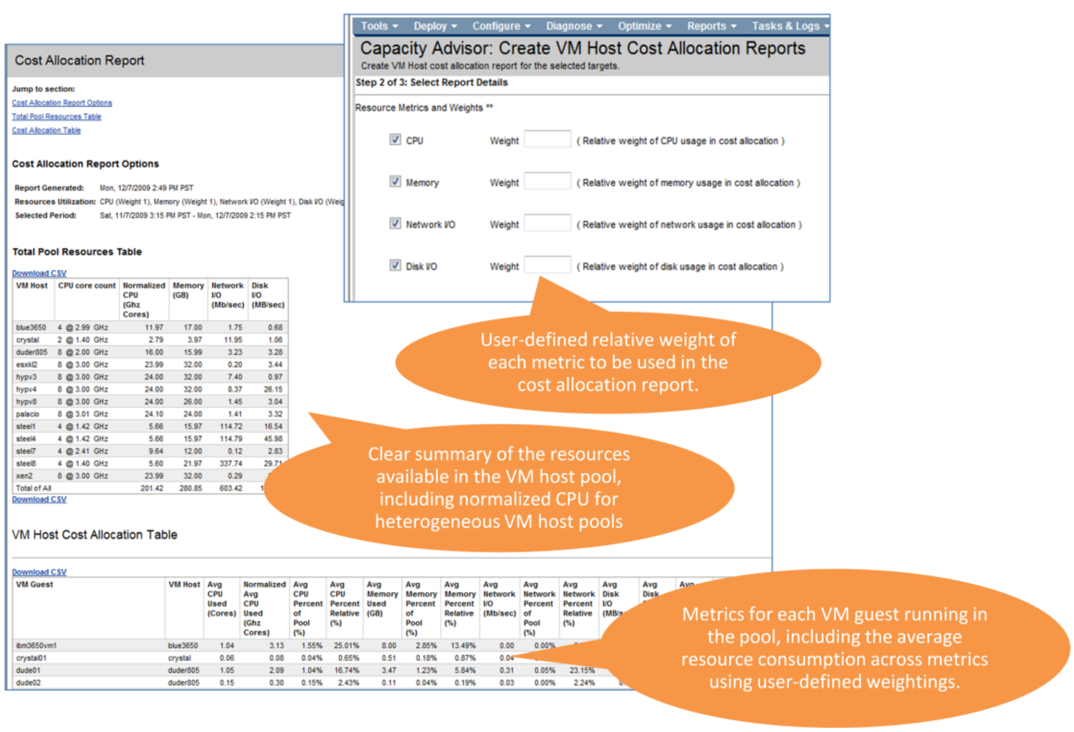
- Service creation: natural starting point for accruing charges, also for one time startup costs if desired
- Service deletion: natural end point for accruing charges, also for one time teardown costs if desired
- Add server(s) to a service: as a result of this operation, the resources of the service are increased making it an optional point to increase the rate of accruing charges
- Add data disk(s) to server(s) of a service: As with adding servers, the resources of the service are increased making this an optional point to increase the rate of accruing charges

- Suspend server(s) of a service: the resources consumed by the service decreases with suspension, this is an optional point to decrease the rate of accruing charges. Note, that when suspended, a service can still be holding on to resource allocations. So in many situations actual resource utilization may be a better indicator of when to decrease charges.
- Resume server(s) of a service: the resources of the service are increased, this is an optional point to increase the rate of accruing charges
- Change lease: The user requests that the lease changes for a service. This would not directly impact the resources used by the service.

As a minimum, the service creation and service deletion events are the basis of accruing charges.

In addition, Matrix can track the utilization of the resources and can present it in a standard "Cost Allocation" report whereby each of the parameters of the various servers or VMs can be presented in rank order according to percentage of resource utilized over a specific period. For many users this can give a clear idea of how much of a given resource is being used by which workload. This could form the basis of a basic cost allocation mechanism, although the data would have to be further -manually- matched-up with known cost information. An example of this report is shown below:

Figure 5: Capacity Planning Cost Allocation Report



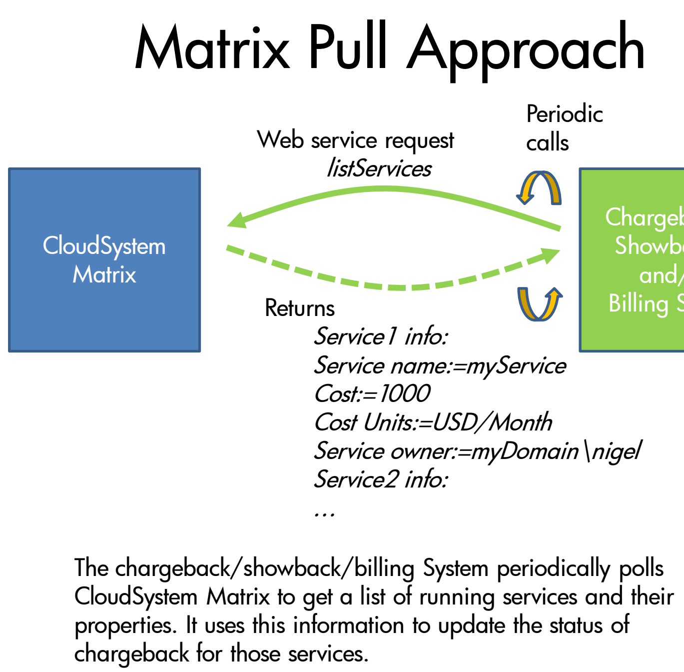
The parameters and interfaces of CloudSystem Matrix are flexible and broad allowing charge-back solutions from simple to more advanced. A charge-back solution built on CloudSystem Matrix can use one or a combination of the following high-level approaches:

- 'pull': an external program or script periodically polls Matrix using the API or CLI to retrieve necessary information about services and enters the information into an external database
- 'push': custom workflow(s) installed in the CloudSystem Matrix call out and update external programs or database at major events in a service's life cycle

In either case, an external database is used to collect information and form the basis for reports.

The pull approach using the CloudSystem Matrix API/CLI

Figure 6: Pull solution using the API/CLI



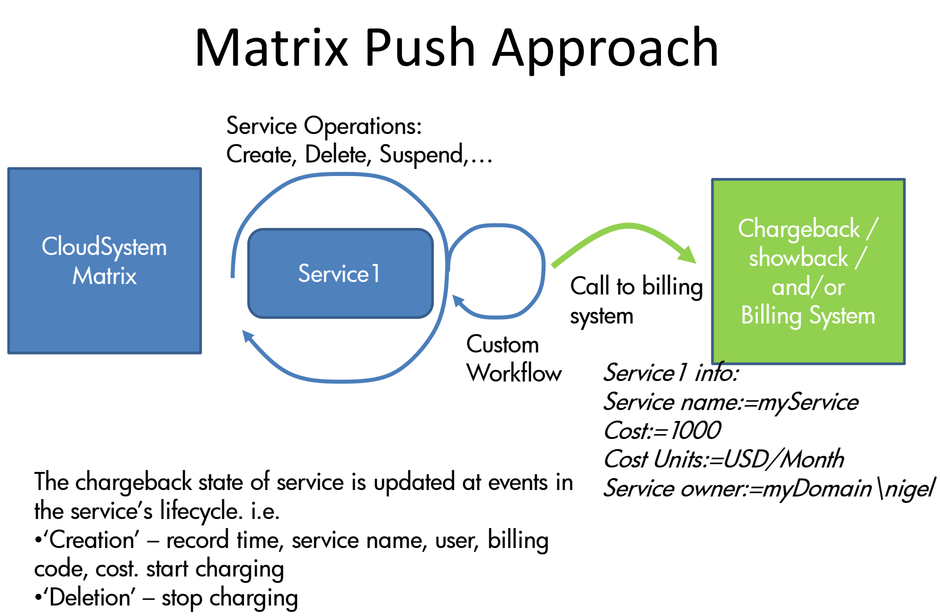
A script or program is written to call CloudSystem Matrix and is setup to run on a regular interval such as every 15 minutes. Each time the program runs, it obtains a list of all current user services and their attributes. One of these attributes is a cost value set for the service by the IT Architect that created and published the entry for the service in the

Service Catalog. With this information, the program updates the billing system. If a service is not already in the billing system's database, it is added to it. If a service no longer appears in the list returned from CloudSystem Matrix, it is considered deleted. At any given time a report can be run against the billing service's database. This approach is used by the "Usage Metering and Reporting" functionality available as a service from HP services. The "Usage Metering and Reporting" functionality is part of the HP CloudStart solution offering or as part of the HP CloudSystem Matrix. Built on an HP Converged Infrastructure, HP CloudStart Solution simplifies and speeds private cloud deployments. Consisting of hardware, software, and services, it empowers businesses to deliver pay-per-use services reliably and securely from a common portal, while offering the ability to scale and deploy new services automatically. Real-time access to consumption and charge-back reports allow you to operate a private cloud in the same way as a public cloud. For additional details click:

>> CloudStart

The push approach using CloudSystem Matrix Workflows

Figure 7: A push approach using workflows



A custom workflow is added to the service template in the service catalog of Matrix. When a user performs an operation on a service from the self-service portal , i.e. create, delete, suspend, etc., the custom workflow would run. As with the pull approach, the workflow has access to the cost value the IT architect assigned to the service in the service catalog. The workflow makes a call to the billing system to update the status of the service from a charge-back perspective in the billing system's database. At any time a report can be run against the billing service's database to print out services over a period of time and their charges.

This approach utilizes the Workflow programming environment of CloudSystem Matrix and requires a suitable remote script or program on the billing program which the workflow can call.

Optional Enhancements

Accessing and processing the top level cost assigned to a service by the IT architect in the service catalog is enough to create a charge-back solution. But it is possible to augment this with the following if desired:

- Per-component cost data assigned by the IT Architect and a property of the service: Per-server, per-storage, per-network connection cost values
- Configuration information about the service: number and type of servers, server CPU and memory attributes etc.
- Utilization information of CPU, memory, disk, network or power of the service

Pull versus Push?

There are trade-offs for the two approaches. One or the other may be better for a particular implementation.

- Push can be less computationally demanding on the Central Management Server (CMS) server and will scale better in larger environments
- Push is a natural fit for charging based on change requests rather than service resources delivered
- Pull will be more natural fit if utilization data is also being integrated
- Pull does not require modifying Matrix's workflow environment and workflow development skills

Pulling charge-back data from CloudSystem Matrix using the CLI/API

Matrix provides CLI and API interfaces so that customers can optionally customize the solution and integrate it with their own environment. The same set of operations are offered through a remote WebService API or a CLI.

A list of all running services and their properties can be obtained using these interfaces. This capability can be the basis of a 'pull' approach where an external utility periodically polls the interfaces to gather current charge-back properties of the services in a remote database. Then as needed, queries are run against the external database to generate a charge-back or show-back report.

Polling is necessary because Matrix will only return the state of currently defined services. There is no way to use the interface to determine services which no longer exist.

Accessing service information through APIs and CLIs

The API and CLI interfaces are described in detail in the paper:

- "Matrix Integration Interfaces" http://h20195.www2.hp.com/V2/GetPDF.aspx/4AA0-9219ENW.pdf

The key calls of the interface for charge-back are:

- listServices:
  - This call returns a list of <ServiceInfo> entries, one for each service
- getService:
  - This call returns a <ServiceDetails> entry for the service named as an input argument

Working with the listServices call

If an IT shop chooses to use only the top level cost field of a service, the listServices call will return all the required data:

- Service name
- Status
- Service cost
- Owner name
- Billing code

In this example the CLI is used to invoke the call.

```
C:\Users\hpice>ioexec list services
Enter password for user VA70CMS\hpice:

Service:
  Name: w2k8ph_6_24_1_u3
  Status: UP
  Based on Template: w2k8phys
  Created: 6/24/11 8:57 AM
  Last Modified: 6/24/11 8:57 AM
  Cost: 8,500 $
  Lease Start: 6/24/11 7:48 AM
  Lease End: 7/24/11 7:47 AM
  Owner Name: VA70CMS\iouser3
  Billing Code: 0046728
  Notes:

Service:
  Name: rhelvm_6_23_1_u1
  Status: UP
  Based on Template: rhelvm
  Created: 6/23/11 3:33 PM
  Last Modified: 6/23/11 3:33 PM
  Cost: 3,100.05 $
  Lease Start: 6/23/11 3:20 PM
  Lease End: 7/23/11 3:20 PM
  Owner Name: VA70CMS\iouser
  Billing Code: 0034568
  Notes:

Service:
  Name: w2k3vm_6_23_1_u2
  Status: UP
  Based on Template: w2k3vm
  Created: 6/23/11 3:34 PM
  Last Modified: 6/23/11 3:34 PM
  Cost: 3,100.05 $
```

```
  Lease Start: 6/23/11 3:22 PM
  Lease End: 7/23/11 3:21 PM
  Owner Name: VA70CMS\iouser2
  Billing Code: 0091237
  Notes:
```

An equivalent call into the API using a C# client looks like;

```
…
//call the IO API and write the returned service names

Console.WriteLine("The Services:");

IOService.ServiceInfo[] service = wcfClient.listService();

for (int i = 0; i < Service.Length; i++)
            {
            Console.WriteLine(service[i].name);
            }
…
```

Or from java;

```
…
System.out.println("Invoking listServices...");
try {
            java.util.List<ServiceInfo> serviceList = port.listServices();

            System.out.println(
            "Operation completed with these services returned:");

            for (ServiceInfo service : serviceList) {
                    System.out.println(service.getName());
            }
}…
```

Working with the getService call

For more advanced charge-back reporting, the getService call would be used as well. This call additionally provides the following about a service:

- The sub-cost fields for components of the service:
    - service overhead sub-cost
    - server sub-cost
    - network sub-cost
    - storage sub-cost
- The configuration fields describing all the servers, network connections, and storage of the service
    - number of servers, processors per server, memory of server, etc

By working down into the hierarchy and walking through the lists, the properties of individual servers, storage connections, and SAN connections can be retrieved. As an example, the processor count of a server in the service is located as follows:

- ServiceDetails->LogicalServerGroupDetails->LogicalServerDetails->LogicalServerInfo->processorCount

Complete details of the LogicalServerGroup are described in the Integration Interfaces paper. The following example is a CLI invocation of the getService call.

```
C:\Users\hpice>ioexec get service -s w2k8ph_6_24_1_u3 -v
Enter password for user VA70CMS\hpice:
```

```
Name: w2k8ph_6_24_1_u3
Status: UP
Based on Template: w2k8phys
Created: 6/24/11 8:57 AM
Last Modified: 6/24/11 8:57 AM
Cost: 8,500 $
Lease Start: 6/24/11 7:48 AM
Lease End: 7/24/11 7:47 AM
Owner Name: VA70CMS\iouser3
Billing Code: 0046728
Notes:

Logical Server Group:
  Name: WinServer
  Service Name: w2k8ph_6_24_1_u3
  Minimum Server Count: 1
  Maximum Server Count: 1
  Active Server Count: 1
  Boot Sequence Ordinal: 0
  Status: UP
  Notes:
  Created: 6/24/11 8:57 AM
  Last Modified: 6/24/11 8:57 AM

  Logical Server:
    Name: WinServer-1
    Logical Server Group Name: WinServer
    Service Name: w2k8ph_6_24_1_u3
    Host Name: wp1u3
    Resource Binding ID: rd://va69cms.va70mgmt.hp.local/37333036-3831-5355-4531-30393739314E
    Parent Resource Binding ID:
    Is Virtual: No
    Is Recoverable:
    Is Base: Yes
    Memory Size: 512 MB
    Processor Count: 1
    Processor Speed: 1,024 MHz
    Processor Architecture: x86_64
    Status: UP
    Cost: 5,000 $
    Created: 6/24/11 8:57 AM
    Last Modified: 6/24/11 8:57 AM

    Logical Disk:
      Name: Disk1-1
      Stereotype Name: Disk1
      Size: 40 GB
      RAID Level: RAID5
      Storage Type: FC-SAN
      Is Bootable: Yes
      Is Shareable: No
      Is Single Path: No
      Cost: 50 $
      Notes:
      Created: 6/24/11 8:57 AM
      Last Modified: 6/24/11 8:57 AM

    Logical Software:
      Name: IO W2k8 R2 SP1 x64
      Type: IO W2k8 R2 SP1 x64
      OS Type: IO W2k8 R2 SP1 x64
      Deployment Service Type: RDP
      OS Customization File Name:
      Virtualization Type: UNKNOWN
      Ordinal: 1

    Logical Network Interface:
      Name: NIC0
      Is Primary: Yes
      Created: 6/24/11 8:57 AM
      Last Modified: 6/24/11 8:57 AM
```

```
      Logical IP Address:
        Address: 192.168.120.203
        Assignment Type: DHCP
        Cost: 500 $
        Created: 6/24/11 8:57 AM
        Last Modified: 6/24/11 8:57 AM
      Logical Subnet:
        Name: va70deploy
        Is Public: Yes
        Is Shareable: Yes
        IP Type: IPV4
        Resource Binding ID: va70deploy
        Notes:
        Created: 6/24/11 8:57 AM
        Last Modified: 6/24/11 8:57 AM
Fixed Cost: 1,000 $

C:\Users\hpice>
```

Pushing charge-back data from Matrix

Another mechanism that can be used to customize Matrix's behavior is via workflows. Workflows can be thought of as script extensions that can either be specific to a service or generic to all services (global). Using workflows a site can include additional custom automated operations to be performed at events in a service's life cycle.

An alternative to the 'pull' approach mentioned in the last section is to utilize the workflows to perform charge-back related processing at points in service's life cycle and 'push' charge-back data out to an external program or tools. For example a workflow which runs at the end of a service creation could be used to update an external database that the new service exists and charges should be accrued.

Customizing service deployment behavior with workflows

Matrix uses the same programming language and development environment for workflows as HP SW Operations Orchestration (OO). Matrix includes an embedded, limited version of OO, which enables you to initiate from Matrix workflows. The following document describes OO workflows and their development environment in detail.

- HP Operations Orchestration Software Studio
- Software Version: 9.00
- Guide to Authoring Operations Orchestrations Flows
- Available at http://support.openview.hp.com

CloudSystem Matrix utilizes OO workflows in a specific way:

- Matrix workflows are only invoked at specific points in service life cycles in an automated manner without user interaction
- Matrix workflows are always invoked by the Matrix run time in the same way with the same predefined input arguments
- For more details on the CloudSystem Matrix's unique workflows environment see:

<TBD reference to CI lab / Didier's new paper on programming in a Matrix environment >

There are two kinds of Matrix workflows: service specific and global. The service specific workflow mechanism is provided so that the services created from a specific template can have custom processing unique to them. Global workflows on the other hand are used so that the same common custom processing can be done for all templates or services consistently.

Service-specific workflows

Service specific workflows are registered in advance with a template. Then whenever that template is invoked to create a service, or the created service has operations performed on it, the service-specific workflow runs. For every operation where a workflow can be added, there are two attachment points:

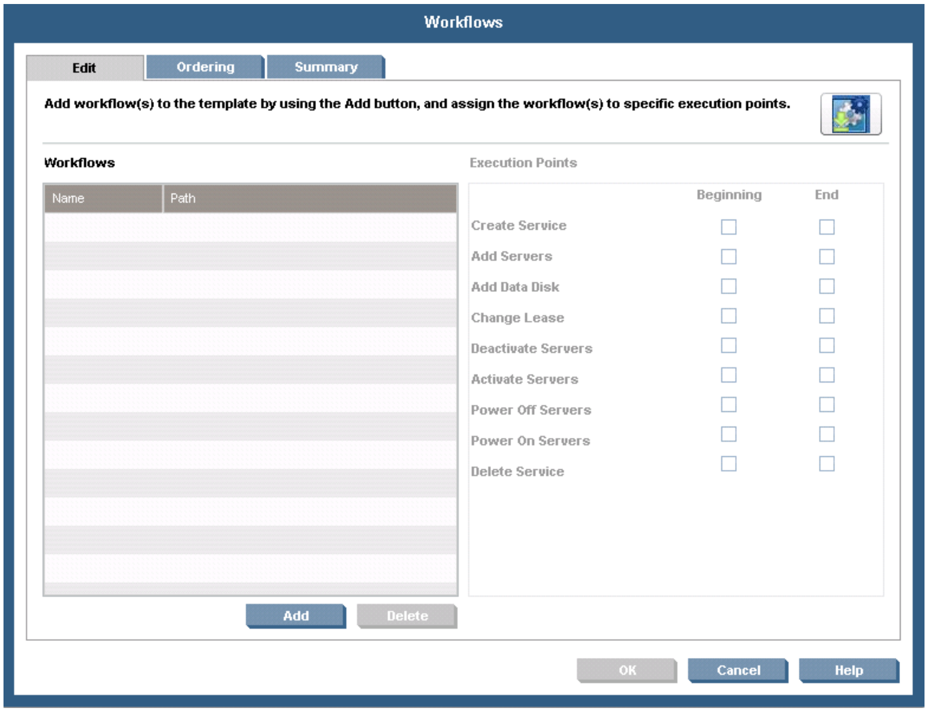
- The operation beginning execution point
- The operation end execution point

When using workflows for charge-back the end execution point is most useful. When it is invoked the operation on the service has completed.

The service operations where a service-specific workflow can be attached are as follows:

- Create Service
- Delete Service
- Add Servers
- Add Data Disk
- Standby Servers or Service
- Resume Servers or Service
- Change Lease

For each, a workflow could be created and attached to the end execution point. In the case of charge-back the workflow could do the appropriate processing to update a remote database to the new state of the service. To use this approach, the workflow needs be attached individually to all templates in the service catalog where charge-back is desired.



Global workflows

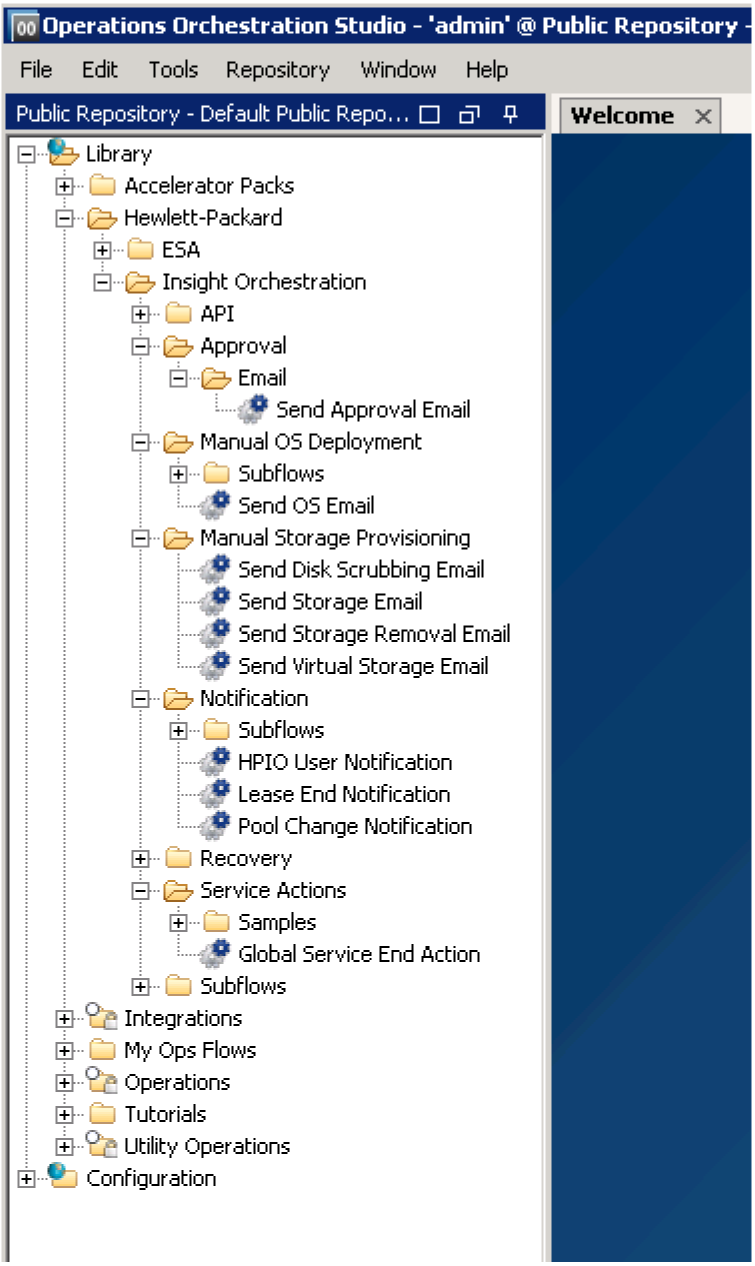
Global workflows do not appear in the service catalog and are not properties of templates. Instead they are 'installed' by placing the workflow in a particular directory in repository of workflows known to CloudSystem Matrix.

There are a number of global workflow attachment points.

- Approval
- Global End Action
- Manual Storage Provisioning
- Manual Disk Scrubbing
- Manual Storage Removal
- Manual OS Provisioning
- User Notification
- Lease Expiring Notification
- Pool Change Notification

Of these, the Global End Action workflow does not have a predefined use and is called at the end of all operations on a service. The default flows and their location in the Library of flows is shown in the following figure from the OO Studio workflow development tool.

Figure 8: OO Studio Workflow Tool



All global workflows have a default implementation which is installed and provided by HP. In the case of the Global End Action the provided workflow sends email to the administrator notifying of the results of the operation. To install a new custom workflow to replace the default provided one, the old workflow is copied off and a new one copied into the reserved path in the hierarchy where Matrix looks to find the Global End Action flow. To combine the default processing of sending email notifications with any new processing, the two pieces of processing need to be combined into a single workflow.

As an alternative to using a service specific workflow(s) for charge-back, a Global End Action workflow could be used. It would parse its input arguments to determine the operation which just ended and branch accordingly.

Service specific or global workflows for charge-back?

Advantages of using service specific workflow

- The workflow can be added selectively to only those templates in the service catalog for which charge-back is desired.
- Because of the attachment point for each service operation, a unique simpler workflow can be created for each service operation - as opposed to a single workflow which must branch.

Advantages of using a global workflow

- The workflow will be applied to all templates/services without having to be explicitly attached to each in the service catalog.
- The workflow is transparent to architects working on template specific workflows. If additional template specific processing is required, the workflow for charge-back does not complicate adding it.

Extracting utilization data for a service from Matrix

In many cases, in addition to the allocation metrics, it may be beneficial to view the actual utilization of the various systems under control. Allocation refers to whether or not a service has been assigned to a given user whereas utilization refers to how much of those resources are actually used during the time window under consideration. CloudSystem Matrix stores utilization data of physical and virtual servers and makes the data available to operators and external programs via a CLI. The data it collects includes CPU, memory and disk IO utilization and is referred to as a server profile. Profiles are identified by the hostname or IP address of the server. The CLI interface for listing and dumping the contents of profiles is capprofile.

A complete reference on the latest Capacity planning and utilization data in Matrix is in the document:

- HP Insight Capacity Advisor User Guide
- http://h18004.www1.hp.com/products/solutions/insightdynamics/info-library.html

A profile is identified by the name of the server in SIM. Example output from a call to list all available profiles is as follows:

```
C:\Users\hpice>capprofile
Name                                   Available History
wviou1                                 06/08/11 07:00 pm EST - 06/16/11 12:00 am EST
wp1u3                                  06/08/11 07:00 pm EST - 06/16/11 12:00 am EST
rv1u1                                  06/15/11 07:00 pm EST - 06/16/11 12:00 am EST
…
C:\Users\hpice>
```

Each server is listed with the date range of the available utilization data.

The data in the profile can be dumped in the following example:

```
C:\Users\hpice>capprofile  -x wviou1
#Profile:wviou1
#Host:wviou1
#CPU:1 @ 2.666GHz
#Memory:512MB
#OS:<UNKNOWN>
#Model:VMware Virtual Platform
#OSName:Microsoft Windows Server 2003, Enterprise Edition (32-bit)
#Version:A.03.00.00
YYYYMMDDhhmm,UTIS,CPU_UTIL,MEM_UTIL,NET_UTIL,DISK_UTIL,CPU_ALLOC,MEM_ALLOC,PHYS_CPUS,PHYS_MEM
201106081900,1307574000,0.00615,0.01000,0.00000,0.00273,2.05060,0.50000,2.05060,0.50000
201106081905,1307574300,0.00533,0.01100,0.00000,0.00273,2.05060,0.50000,2.05060,0.50000
201106081910,1307574600,0.00615,0.01633,0.00000,0.00273,2.05060,0.50000,2.05060,0.50000
201106081915,1307574900,0.00574,0.01400,0.00000,0.00273,2.05060,0.50000,2.05060,0.50000
201106081920,1307575200,0.00656,0.01533,0.00000,0.00273,2.05060,0.50000,2.05060,0.50000
201106081925,1307575500,0.00595,0.01666,0.00000,0.00273,2.05060,0.50000,2.05060,0.50000
201106081930,1307575800,0.00513,0.02166,0.00000,0.00273,2.05060,0.50000,2.05060,0.50000
201106081935,1307576100,0.00656,0.01466,0.00000,0.00280,2.05060,0.50000,2.05060,0.50000
…
```

After the header describing the server and the table of data, the remaining rows contain a sample of data for a given time stamp. Additional arguments to capprofile can restrict the output to a specific date range.

Augmenting service information with utilization information

The capacity planning utilization data is maintained for up to four years in the CloudSystem Matrix database, even for services which no longer exist. So there is some flexibility on deciding on a schedule to retrieve and process the information into the billing system. The decision is most driven by how often reports are anticipated to be run.

To extract the utilization data for the servers of a service, the server's hostname or IP addresses needs to be available. If these are not already in the billing system because of processing described earlier, the CloudSystem Matrix API/CLI needs to be called to get this mapping.

Once the server names are available the CapAd capprofile command can be run via a remote shell mechanism to dump the utilization information for the date range of interest. The raw data is impractical and unnecessary to load into the billing system. Instead, a script/filter should be run on the data to get a single average utilization number for the period to record in the billing system.

Summary

HP CloudSystem Matrix provides the necessary attributes of services and the APIs so that a charge-back/show-back/billing solution can be implemented. This paper has described two approaches and the interfaces to extracting the raw charge-back data so that an external database could be populated.

HP CloudSystem Matrix enables the provisioning of infrastructure and applications in minutes rather than months, resulting in greater agility and speed, as well as TCO savings. CloudSystem Matrix offers a self-service infrastructure portal for auto-provisioning and built-in lifecycle management to optimize infrastructure, monitor applications, and ensure uptime for cloud and traditional IT.

This paper described some tools and techniques that can be used to extract usage data in a Matrix environment for show-back, charge-back, and/or billing purposes. It described the services model in Matrix, the attributes that would be of interest in a cloud environment, and the APIs and CLIs to extract the related usage data. It also described two different architectural approaches in gathering this data into a database for report generation.

For more information

To read more about Matrix, go to www.hp.com/go/matrix

4AA3-8068ENW, Created November, 2011