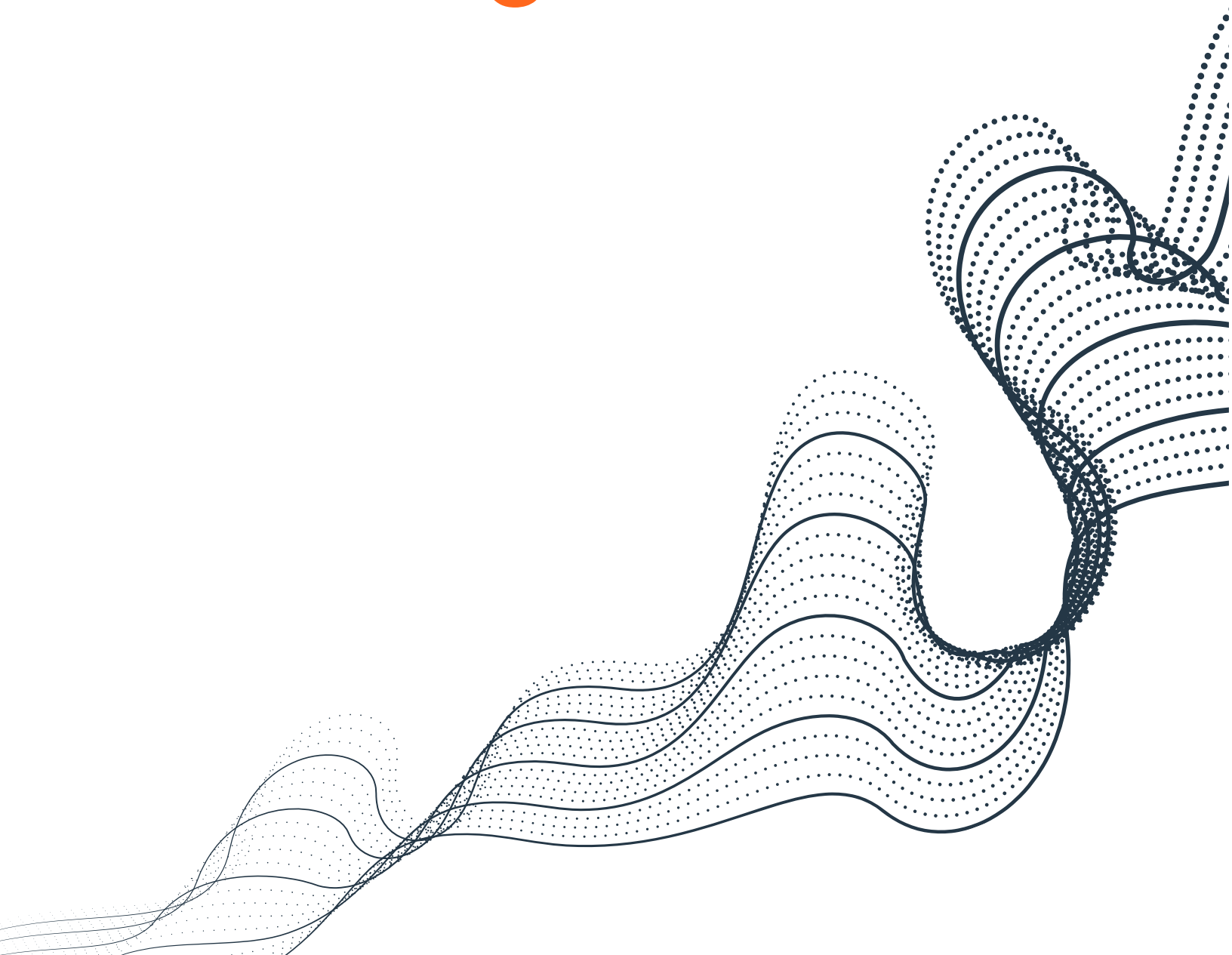
EMBOTICS

vCommander
Version 7.5.0

Cloud Automation Planning Guide



Contents

Cloud Automation Planning Guide

Introduction

Your goal is faster fulfillment of service requests through self-service and automated provisioning. Our goal at Embotics is to help you set up cloud automation as quickly, efficiently and painlessly as possible.

To that end, this guide:

- presents questions for you to consider, to accelerate the planning and discovery process
- provides best practices
- explains the Embotics vCommander ecosystem, so that you can understand how vCommander will fit into your systems and processes

Think of this guide as a stage in our conversation between signing the paperwork and beginning your implementation.

The Big Picture: People and Process

One of the benefits of self-service automation is the ability to limit access to the underlying private or public cloud virtual infrastructure, which we call managed systems (for example, VMware vCenter or an Amazon Web Services account). vCommander's role-based access control means that your administrative users no longer require direct access to the managed system – they can do everything they need to in vCommander instead.

Self-service automation enables users to view and manage VMs, including the ability to request new VMs or request changes to existing VMs. Using the web-based Service Portal, cloud infrastructure administrators provide stakeholders with an elegant but information-rich view of their assets without the need for direct access to the underlying private or public cloud infrastructure. End-user access to the Service Portal can be customized to your needs. vCommander also provides powerful workflow capabilities, such as quota management, which ensures predefined resource limits for organizations and end users.

vCommander provides:

- an unlimited number of customizable Service Portal roles, providing delegated administration for each organization, as well as fine-grained control over permissions
- four administrative roles, controlling login to vCommander and the administrative tasks that each user can perform
- six levels of infrastructure access rights, controlling visibility of the managed system and the tasks each administrative user can perform on your infrastructure

Design considerations

Who needs administrative access to vCommander? What vCommander role and managed system access rights do these users require?

Who needs direct access to the managed systems themselves?

Do you need to delegate administrative tasks to one or more organization managers, enabling you to lighten the load on the vCommander administrators? These users need a Service Portal role with delegated admin and/or management permissions.

Which of the following tasks would it make sense to delegate?

- adding and removing members
- modifying members' roles
- assigning the primary contact for an organization
- managing the media library
- assigning quotas to members
- approving members' service requests
- monitoring quota usage

Best practices

Once you've chosen the small number of users who require access to the managed system and to vCommander, all other users should have access through the Service Portal only.

Typically the person responsible for a business unit, the organization manager has extended permissions for managing an organization's members, quota and assets. Tailor these permissions to the technical abilities of your organization managers.

Examples

Examples of Administrative User Account Configuration

Learn more

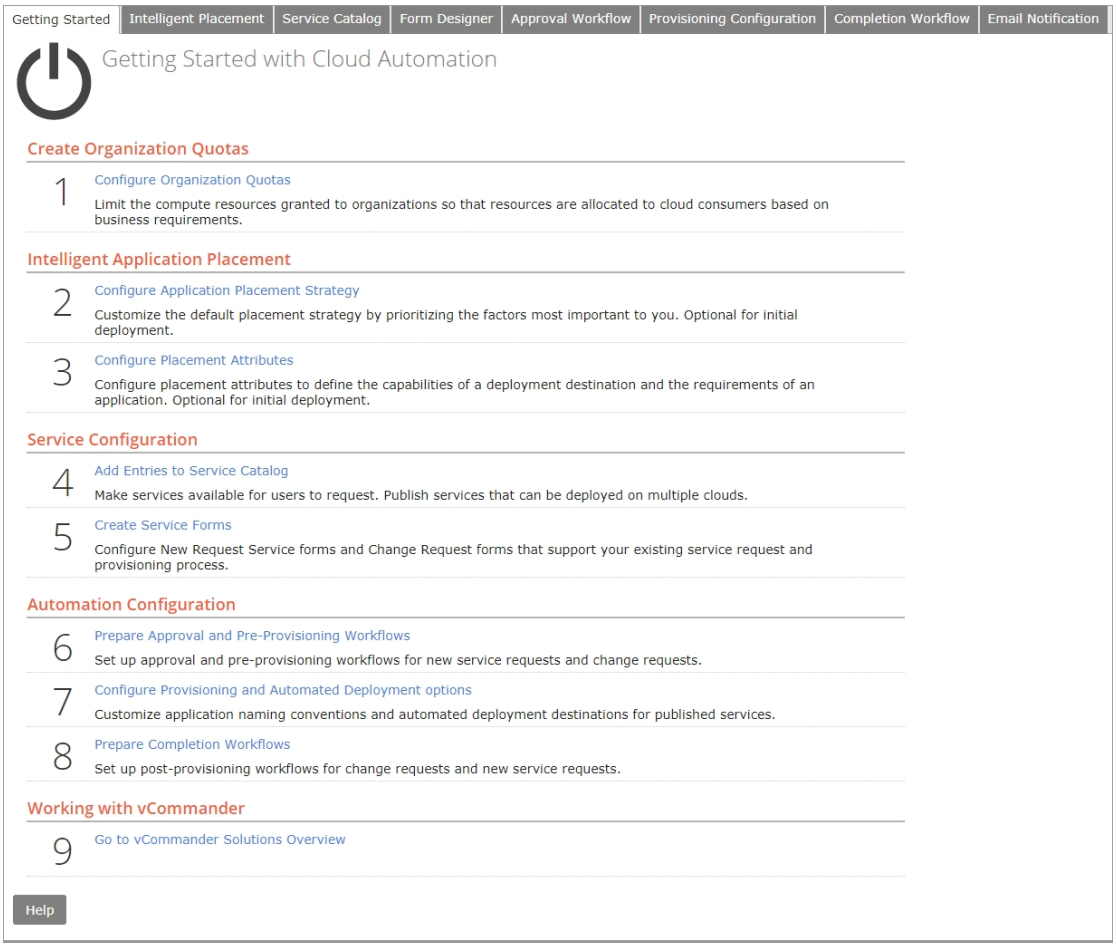
Understanding vCommander Administrative Roles and Permissions

Understanding vCommander Access Control

Customizing Service Portal Roles for End Users

Steps to New Service Request Automation

vCommander's straightforward user interface makes it easy to automate the service request process. Just to go to **Help > Getting Started with Cloud Automation**. This section of the guide provides detail on each of the steps.



Step 1: Configure Organization Quotas

An organization is a group of consumers with a common business purpose - for example, your development group, or one of your customers. Organizations, the fundamental building blocks of vCommander multi-tenancy, enable you to:

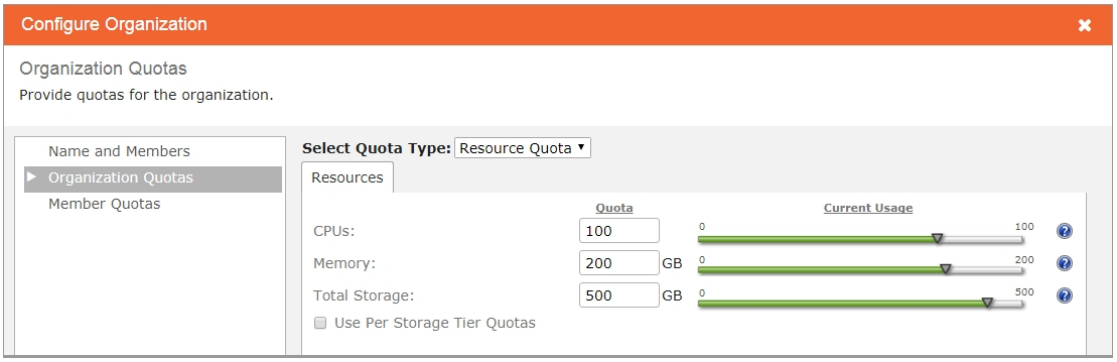
- segregate and secure data for your consumer groups
- delegate administrative tasks to consumers
- set up distinct cloud automation configurations for your consumer groups
- generate reports for each organization

Whether you're a service provider or an enterprise with multiple stakeholders, you need to segregate data for the various groups of users that consume your IT services. Data segregation provides enhanced security, easier management and cost savings. In the multi-tenant vCommander model, the entire service request process is unique to each organization.

Quotas allow you to limit the compute resources or limit the daily costs for an organization so that you can assign available resources to your consumer groups based on their business requirements.

Quota enables you to limit the resources granted to an organization based on their business requirements and budget. You can configure one of the following types of quota for each organization:

- resource-based quota, which enables you to limit the number of vCPUs, the amount of RAM and the amount of storage. You can set both global and per-tier storage quota.
- cost-based quota, which enables you to limit the total daily cost of VMs
- per-destination quota, which enables you to configure distinct quota limits for each deployment destination available to a user or organization



Design considerations

How many organizations do you need? Think about your requirements for data segregation, reporting by organization, controlling the visibility of the service catalog and request forms, and delegating administrative tasks.

Before you configure quotas, you have a few decisions to make:

- Will you set a global quota for the entire organization, or will you set limits for each destination where an organization can deploy services? You can configure a different type of quota for each destination. This is useful when you have different costs and resource constraints for different workload types, such as development compared with production.
- Will you set a resource quota or a cost quota? Resource-based quotas allow you to limit the number of vCPUs, the amount of RAM and the amount of storage. Cost-based quotas allow you to limit the total daily cost of VMs. When you choose per-destination quota, you can configure a different type of quota for each destination.
- If you're setting a resource quota, will you set quota limits for individual storage tiers?
- Will you set quota limits for the organization as a whole, individual members, or both?
- Do you need a quota-based approval process?

Best practices

Your team structure and the layout of your cloud infrastructure will determine how assets such as service catalog entries and deployment destinations are exposed to users.

If you're a service provider, you likely sell a block of resources to each customer; you can configure quota limits to alert your sales team that an organization requires additional resources. For enterprise customers, enable your development team to achieve rapid provisioning without an approval process, while ensuring that they stay within their quota limit.

Examples

Let's say you want to limit the resources for an organization to 20 CPUs, 80 GB of memory and 200 GB of storage. If your organization currently has four members, you can:

- set identical quotas for each member. Each member is limited to 5 CPUs, 20 GB of memory and 50 GB of storage.
- set individual quotas for each member. One member is allowed up to 8 CPUs, and the other 3 are limited to 4 CPUs each.
- allow each member to consume up to the maximum available quota for the organization, by not setting member quotas. If two members have used up all 20 CPUs, the other members won't have available quota.

As another example, if you use one storage tier for swap space, you can exclude this tier from storage quota calculations, while setting per-tier quota limits for your other storage tiers.

If you're using per-destination quota, for a single organization, you can set distinct resource quota limits for each destination. Or, you can set a cost quota for the Production destination, a resource quota for Staging and Development, and no quota limit for Lab.

Per-destination quotas can be useful for such scenarios as cloud-bursting: if an organization exceeds its quota on its private cloud destinations, then vCommander can deploy to a public cloud destination instead. See Configuring Attributes for Intelligent Placement to learn how to configure this type of scenario.

Automate Customer Onboarding

Walk-through: Configuring Organizations

Learn more

Creating Organizations for Multi-Tenancy

Setting Quotas for Multi-Tenancy

Configuring Per-Destination Quotas for an Organization

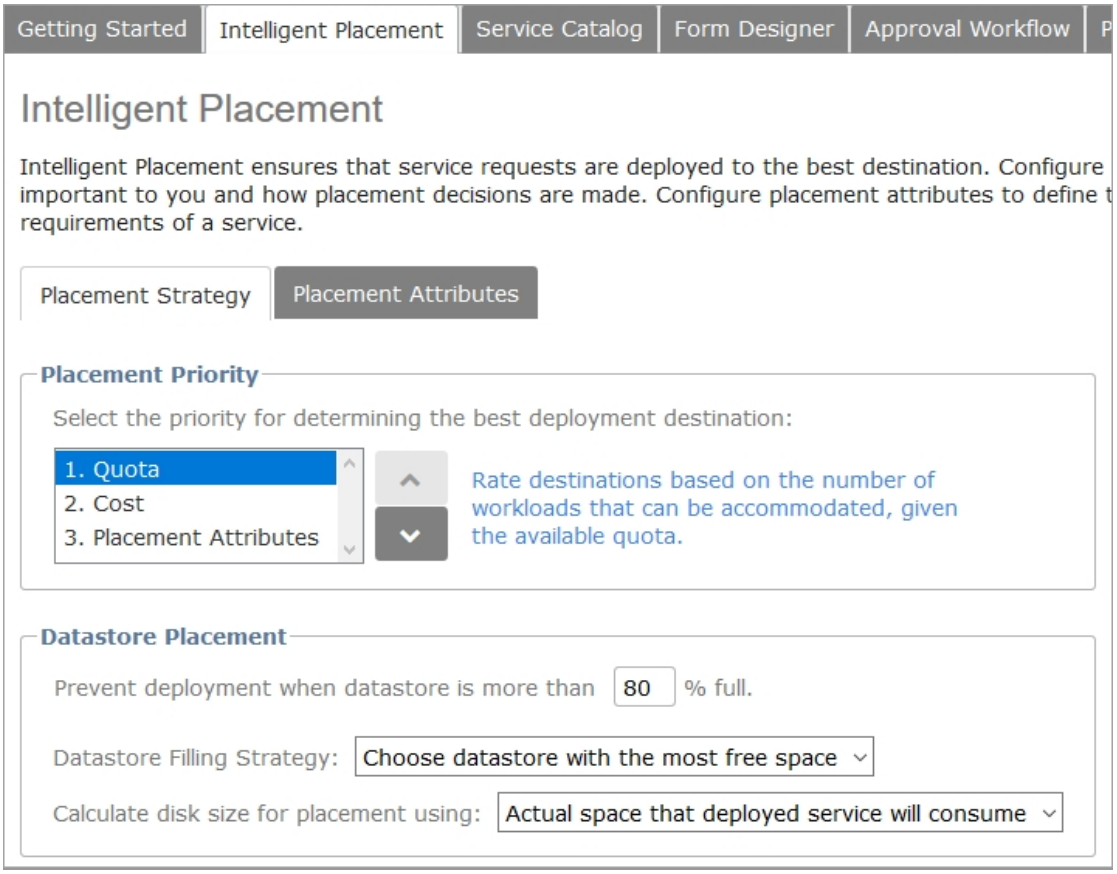
Configuring a Global Resource Quota for an Organization

Configuring a Global Cost Quota for an Organization

Step 2: Configure Application Placement Strategy

Notice that once you've completed Step 1: Configure Organization Quotas in the Getting Started with Cloud Automation wizard, subsequent steps in automating the new service request process follow the tabs shown in the image below, from left to right. Let's continue to the Intelligent Placement tab.

Intelligent Placement ensures that service requests are deployed to the best location. In this step, you create a customized global placement strategy for new services.



Placement priority

When vCommander predicts and selects the best deployment destination for a new service request, it first filters out invalid destinations. Each valid destination is then rated, based on the following criteria:

- **Placement attributes:** Does this destination provide the capabilities preferred for this service?
- **Quota:** How much quota is available on this destination for the requester and the requester's organization, when per-destination quota is configured?
- **Cost:** How much will this service cost on this destination?

By default, these three factors are prioritized in the order they appear above, but if cost is more important to you than quota, for example, you can customize the priority.

Datastore placement

Datastore placement settings apply to vCenter and SCVMM managed systems. You can customize how full a datastore must be before vCommander no longer deploys to it, how datastores are chosen for deployment, and how disk size is calculated.

How much of a datastore's storage must be consumed before you stop automatically deploying VMs to it? Should vCommander choose the datastore with the most free space, or leave the largest block of space unused?

It's important to understand the concept of "reserved storage". vCommander reserves storage as soon as deployment of a new service request or fulfillment of a change request starts. This is true for manual deployment and fulfillment, automated deployment and fulfillment, scheduled fulfillment, vCenter migration, and the manual Reconfigure Resources command. This ensures that new and expanded disks are placed

on a datastore with sufficient storage space. Once deployment or fulfillment has succeeded (or failed), the storage reservation is released.

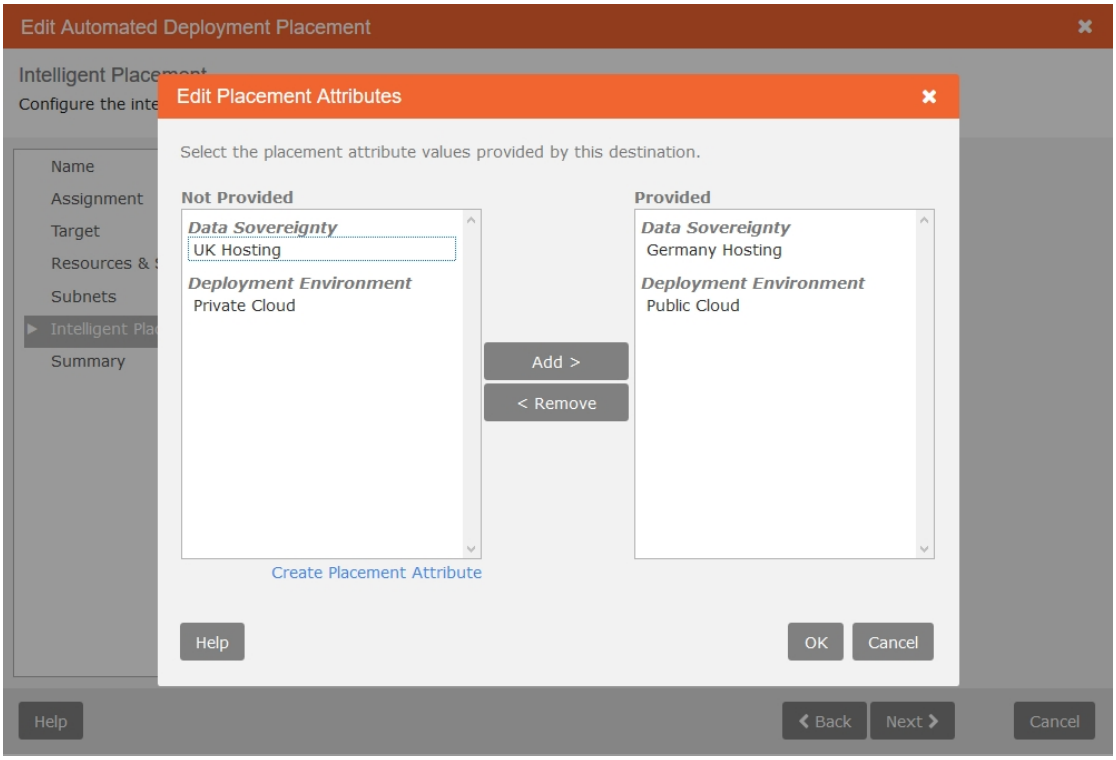
Learn more

How Intelligent Placement Works

Configuring a Global Placement Strategy

Step 3: Configure Placement Attributes

By identifying the capabilities of each deployment destination and the requirements of a service, placement attributes help ensure that service requests are deployed to the best destination. For example, placement attributes can help vCommander decide whether to deploy to public or private cloud, or which datacenter or geographic region is best suited to a service.



You can configure two types of placement attribute:

- **Fixed Requirement**: If a destination doesn't provide this capability, it's filtered out of the list of valid destinations.
- **Selectable Values**: A prioritized list of capabilities. Destinations providing one or more of these capabilities are given a placement rating.

Design considerations

Once you've configured placement attributes, you assign placement attribute values to deployment destinations and published services. When you assign a placement attribute value to a deployment destination, you're identifying the capabilities of that destination, to help ensure that services are deployed to the best destination. When you assign a placement attribute value to a published service, you're identifying the requirements of that published service, to help ensure that services are deployed to the best destination.

You can enable requesters to specify placement requirements for new services. Consider the following:

- You can also enable users to select a destination by adding the Destination form element to the request form.
- Placement attributes are service-level attributes and must be added to service-level forms. Placement attributes can't be added to blueprint forms. Because service-level forms are assigned to users, adding a placement attribute to a form means that it is displayed whenever a user requests any service.

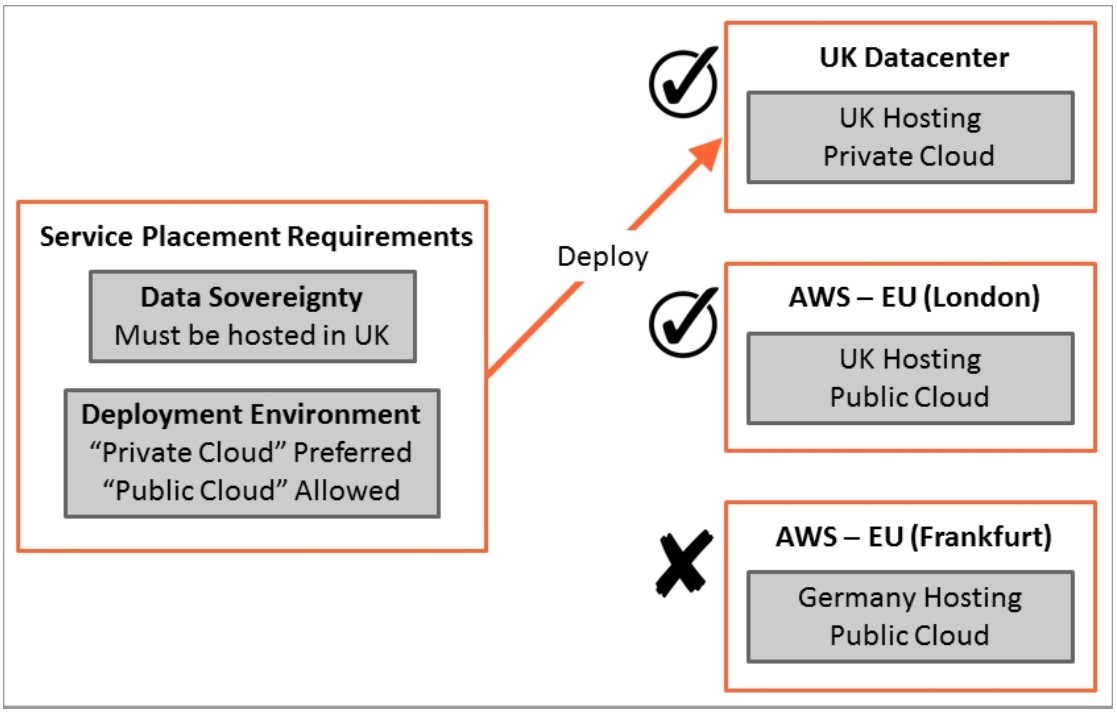
Best practices

The vCommander REST API v2 enables you to create placement attributes and apply them to deployment destinations and published services. This can help with adding placement attributes to existing destinations and services, as well as easing the onboarding process. For examples, see the built-in help for the vCommander REST API v2 PowerShell client.

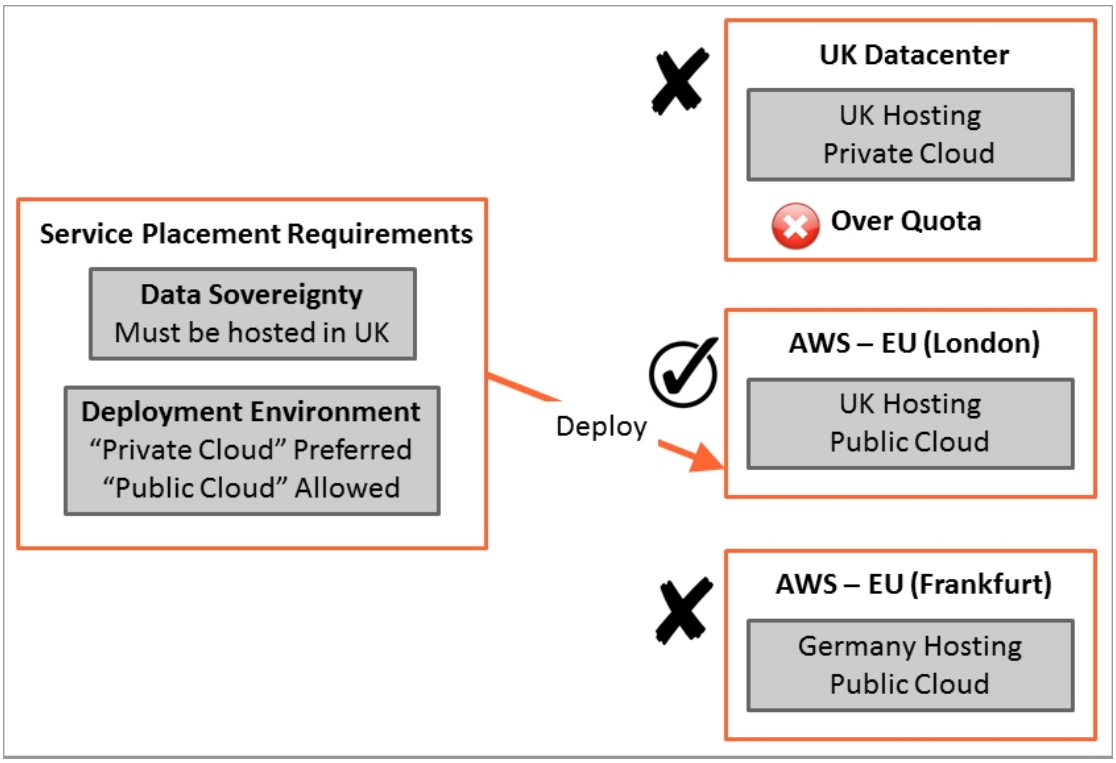
Examples

Let's say a service must be hosted in the UK, and while it can be deployed either on-premise or in a public cloud, an on-premise destination is preferred.

Three destinations are available to users requesting this service. The following diagram shows that the UK datacenter is the best destination, because it provides both data sovereignty and the preferred deployment environment.



However, per-destination quota limits have been configured for the requester's organization. Because this service would cause the requester to exceed quota limits on the UK datacenter, the UK datacenter is no longer the best destination. The AWS - EU (London) destination is now chosen, because it satisfies the data sovereignty requirement, and the requester has sufficient quota.



See Configuring Attributes for Intelligent Placement to learn how to set up this and other examples, such as:

- All Oracle database instances must be deployed on a specific cluster that's licensed for Oracle
- All services requiring backup must be deployed to a destination that supports backup
- Requester selects the service level, which affects where the service can be deployed

Learn more

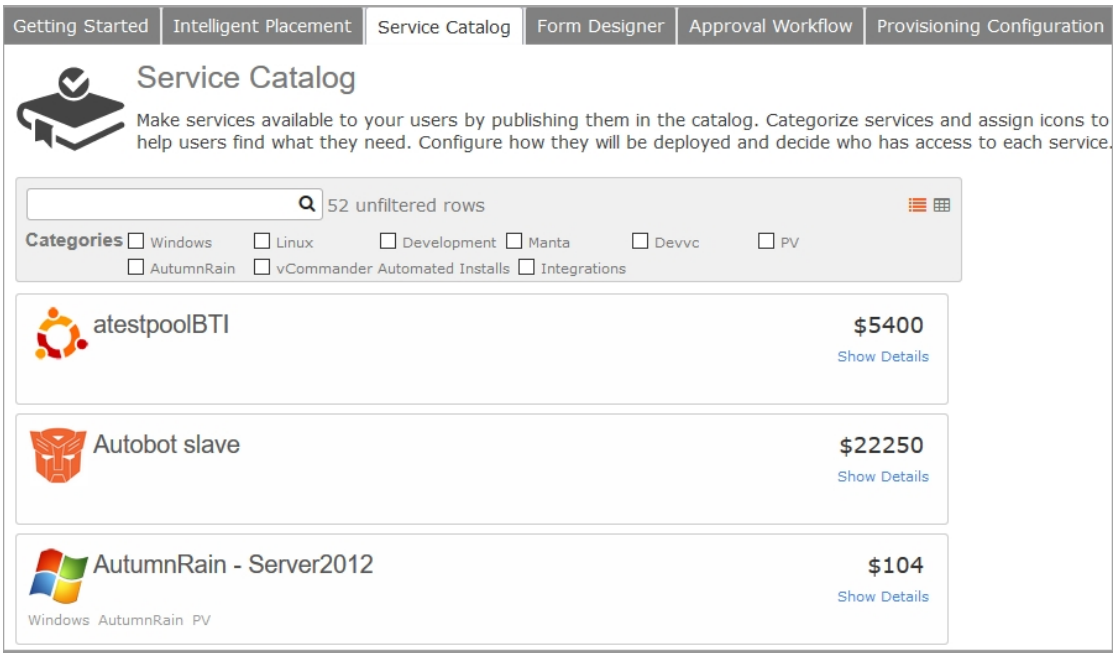
Configuring Attributes for Intelligent Placement

Step 4: Add Entries to Service Catalog

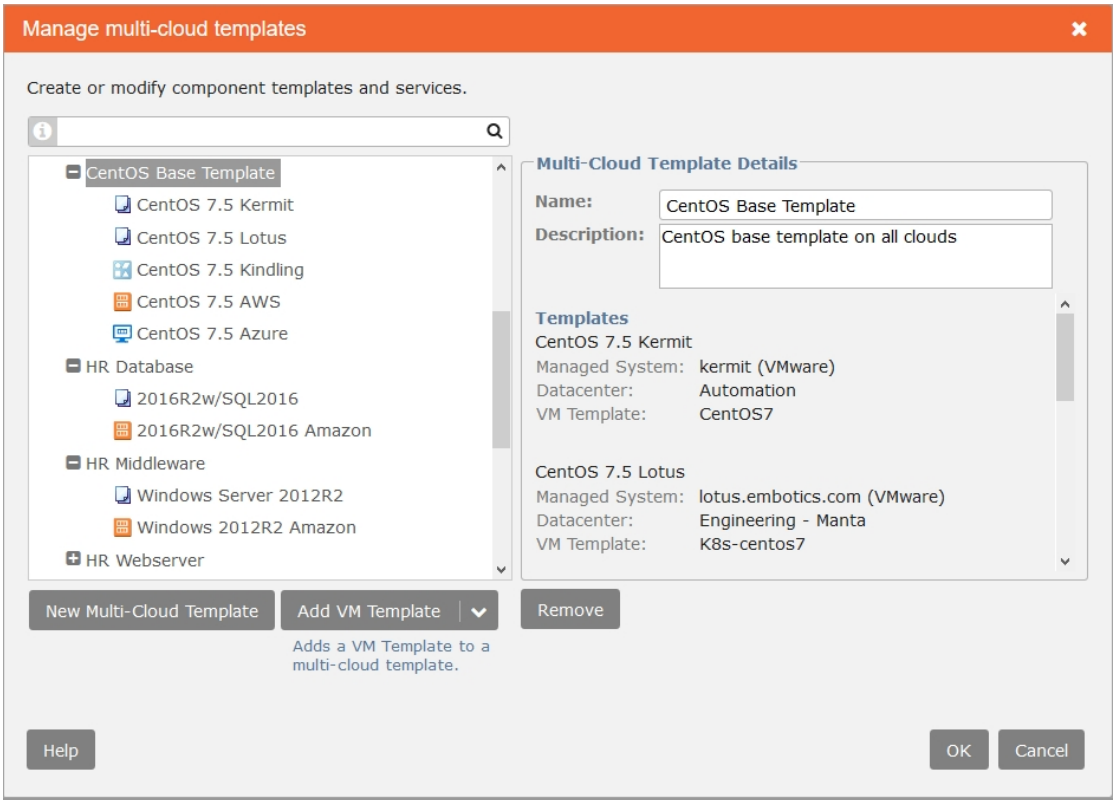
A service can consist of anything from a single VM to a combination of service components, such as:

- multi-cloud templates
- VM templates
- Amazon Marketplace AMIs
- virtual service templates
- cloud templates (CloudFormation and ARM templates)
- OVA/OVF templates
- custom components – used to represent both non-virtual assets (such as a phone) as well as tasks that modify existing assets (such as the installation of a database instance on an existing server)

A service can be predefined (for example, as a vApp in vCenter) or built from individual components in vCommander.

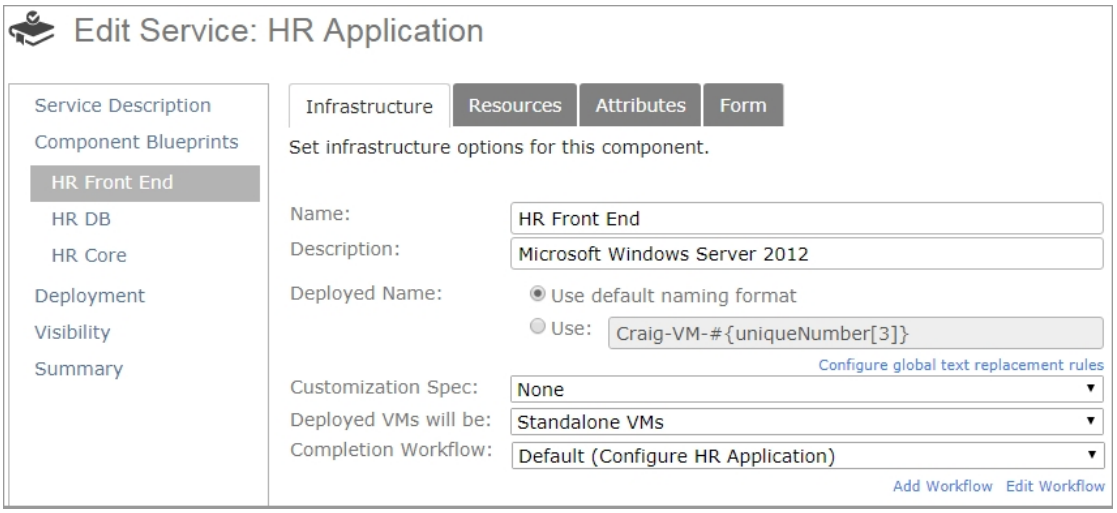


Multi-cloud templates are the recommended service catalog building block. With multi-cloud templates, you can create a single service that can be deployed on multiple datacenters, as well as multiple private, public or hybrid clouds. Multi-cloud templates also make it easier to keep the service catalog up to date.



Blueprints enable you to configure default settings for each component in your service catalog. You can configure options related to:

- infrastructure–such as a name, description, and customization specification
- post-provisioning completion workflows, which allow you to install and configure applications
- resources–such as CPU count, memory, instance type, storage and networking
- custom attributes–metadata such as project code and cost center
- groups–such as expiry, rightsizing, guest OS scanning, power schedule and maintenance groups
- third-party integrations–such as Puppet classes and groups



Blueprints also enable you to create component-specific request forms.

Design considerations

Will all services be globally available to all organizations, or will you create some custom services for each organization (or group)?

Will services be predefined or customizable? Will your catalog contain exactly what your users want, or more generic services that your users can customize at request time? A smaller service catalog that allows user customization of resources, integrations and applications requires less maintenance but can offer the same level of flexibility.

Do services need to be fenced (network isolation)?

Do deployed services require a defined naming convention? For example, you may want to name all SQL services requested by Development with a pattern such as DEV-SQL-PRD-001, DEV-SQL-PRD-002, and so on.

Will users be allowed to change the resource defaults, like adding extra disks, memory or CPUs?

If you integrate with Puppet or Chef, do you want to allow users to configure this information on the request form?

What custom metadata do you want to attach to services?

Best practices

Use multi-cloud templates as the building block for all services, even for single-cloud services, to create an easier-to-maintain service catalog.

Use the 80/20 rule for your service catalog: include the services that are most often requested.

Offer only those services relevant to each organization and each user by limiting visibility.

Name services appropriately.

Use meaningful icons to represent services.

Categorize services to make them easy to find.

Keep templates up-to-date in the service catalog with vCommander's template replacement feature.

Examples

Create a service that includes several instances of a database, and install the same base OS for each database by assigning the same completion workflow to each component in the service.

Create a service that contains everything a new hire needs, such as a desktop machine and installed software applications.

Create a lab service– everything required to set up a software testing lab environment.

Automatically Tag Deployed VMs with Their Server Roles

Specifying VM Names Using Request Form Values

Specifying the VM Network Using Request Form Values

Learn more

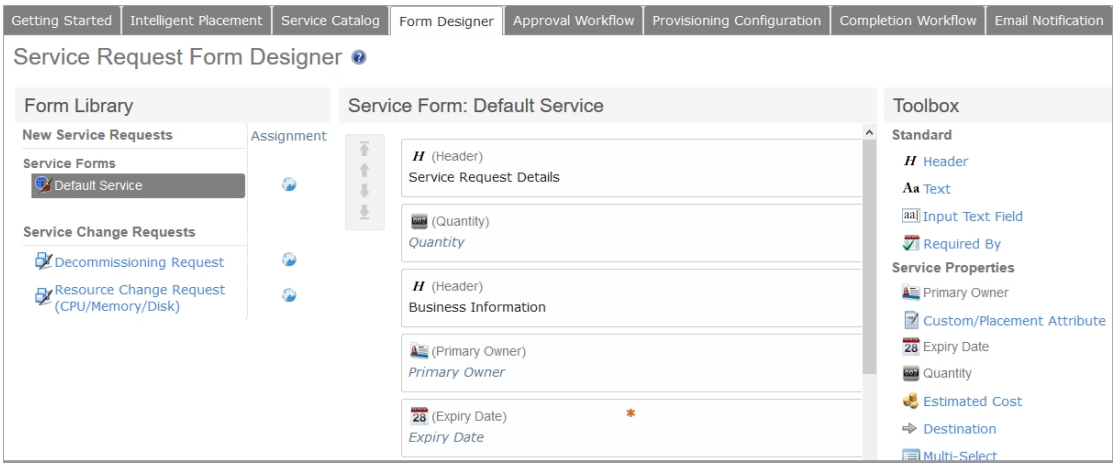
Managing the Service Catalog

Adding a Multi-Cloud Service to the Catalog

Creating an Efficient Service Catalog

Step 5: Create Service Forms

You can build any number of service request forms to support your existing service request and provisioning process. New service request forms are built in two parts: one for the service as a whole, which includes metadata that applies to all components, and a blueprint specific to each component in the service.



Design considerations

Do you need to present different organizations (or groups) with unique information? For example:

- Will users be allowed to choose where the service is deployed (the destination)?
- Does any "business process" data need to be tracked, such as project codes or ticket numbers?
- Will users be able to set expiry dates, or specify that a service will never expire?

- Will users be able to specify placement requirements for requested services? If so, you need to add placement attributes to service-level forms.

Best practices

Tailor request forms according to how knowledgeable your users are, and how much freedom you want to grant them in customizing requested services. If you don't want to allow any customization, configure read-only forms.

Include an expiry date on the form to enable lifecycle management.

Examples

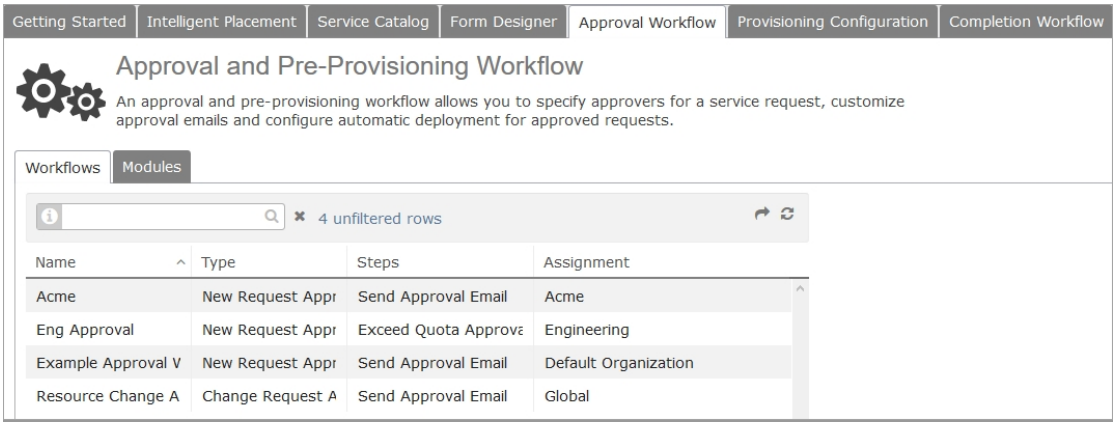
Example: New Service Request Forms

Learn more

Customizing Service Request Forms

Step 6: Prepare Approval and Pre-Provisioning Workflows

Approval and pre-provisioning workflows enable you to specify approvers for a service request, customize approval emails and configure automated deployment for approved requests. You can configure conditional workflow steps and use variables to access information captured during the request process.



Design considerations

Do you require multi-stage approval?

What needs to happen before a VM is deployed?

Who or what controls whether a new service request is approved one or more individuals, quota calculations, or some other factor?

If you configure quota limits, will you automatically reject requests that exceed quota, or will you allow approvers to make this decision?

Does anyone need to be notified, without requiring their approval?

Once a request is approved, should automatic deployment commence, or will an administrator manually deploy the service?

Best practices

Create an approval process that supports your business requirements – for example, to control costs, to control service deployments, or to justify service requests.

For software development teams, as long as their request is within quota, allow for automatic request approval. This allows IT to quickly meet the needs of the business.

For service providers, consider how your services are sold. If sold as blocks of resources, you will use quota and notify sales teams. If sold as on-demand, all-you-can-eat, automatically approve requests to be billed at a later date.

Examples

Set up a two-stage approval process: line of business approval, followed by IT approval.

Add a second level of approval for requests that will exceed quota.

One of our customers has configured a daily cost quota for each of their 20 organizations. If a service request is within the organization's daily cost quota, the Approval and Pre-Provisioning workflow sends an approval email to the organization manager. Service requests that will exceed that quota display a warning, notifying the requester that the request may not be approved. The request is then automatically rejected, and a notification email is sent to the requester.

Another customer sends an approval email to one email address if the request will exceed an organization's CPU quota, and to a different email address if the request will exceed the organization's storage quota.

Approval workflow examples on our Knowledge Base

Learn more

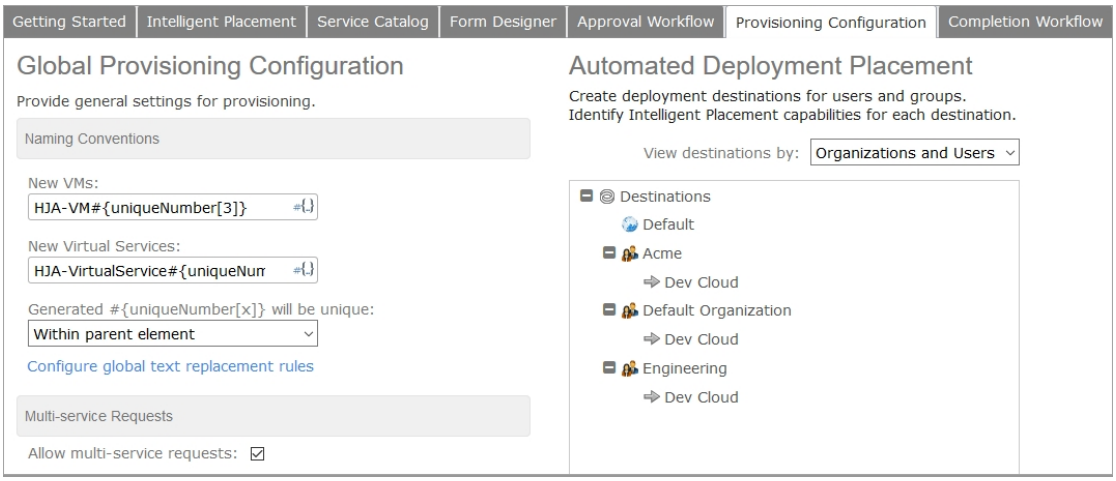
Walk-through: Creating a Workflow

Creating an Approval and Pre-Provisioning Workflow

Configuring a Quota-based Service Request Approval Process

Step 7: Configure Provisioning and Automated Deployment Options

vCommander can automatically deploy new service requests once they're approved. When automated deployment is configured, new service requests can move from approval to completion with no intervention. Automated deployment means that administrators are freed from decisions related to capacity, storage and networking. vCommander emails you when automated deployment failures occur, and when changes in your infrastructure cause deployment destinations to become invalid.



Design considerations

vCommander enables you to configure placement rules that are as simple or as sophisticated as you require. Here are just a few considerations in this area:

- What will determine where VMs are deployed? For example, do all SQL servers need to go to one destination? Perhaps the web team has its own cluster?
- Do you need to take advantage of vCommander IP pools or our integration with BlueCat™ IPAM? vCommander IP pools work well when you need you to reserve resources for particular users or groups or to make sure that certain ranges are used for particular purposes. When you integrate with BlueCat IPAM, you can create host records or DHCP reservations from BlueCat™ IPAM during provisioning with vCommander.
- How are naming conventions defined? You can configure a global naming convention, or naming can be determined by the service catalog. You can use variables for unique numbering; you can also include information such as the requester or service type. Requesters can also define service names, in full or in part.
- Will your service catalog contain multi-homed services, or will organizations have the option to deploy to multiple networks? If so, you need to assign zones to networks, rather than simply assigning individual networks to the deployment destination.
- Consider assigning placement attributes to deployment destinations, as well as to the service request form.

Best practices

Configure automated placement and networking for deployed services based on the types of services you will offer (for example, fenced or multi-homed services), as well as how knowledgeable your users are and how much control you want to allow them. If you don't want to allow any end-user control, don't add the Destination form element to the new request form.

Examples

If you're running a Dev/Test shop, you'd likely want to allow your users to choose the network zone and add adapters as required when requesting a service. You might also want to allow Dev/Test users to reconfigure networking settings for existing VMs.

If you're a service provider, on the other hand, you may want to allow requesters to indicate their service requirements by selecting from a drop-down list (such as Backup, Monitoring or None), and then set up completion workflows to handle these requirements.

Automating VM Customization through Workflows: Examples

Learn more

Provisioning Configuration

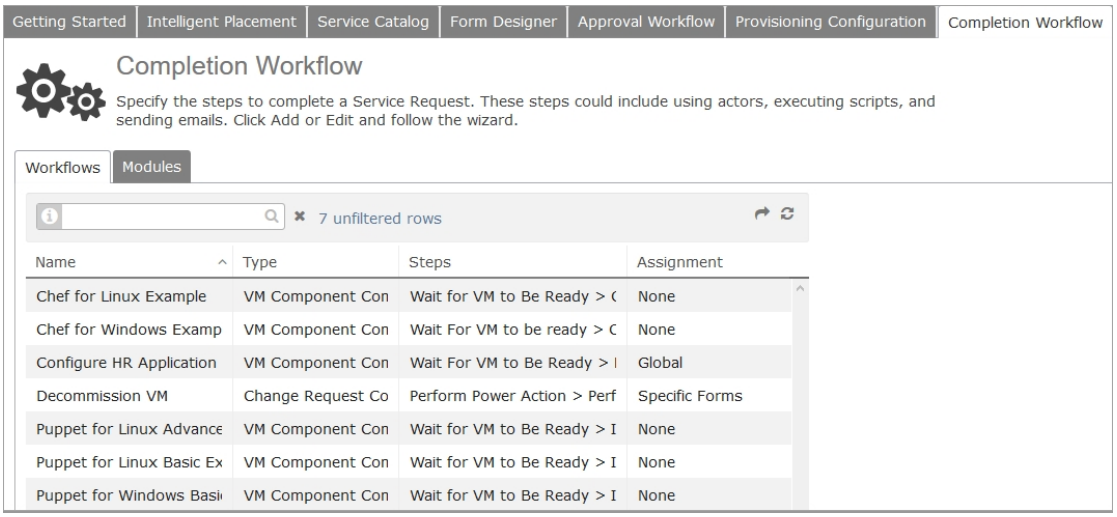
Networking and IP Management

Configuring and Managing IP Pools

Integrating BlueCat™ IPAM IP Address Management with vCommander

Step 8: Prepare Completion Workflows

Completion workflows support post-provisioning tasks for new service requests, such as installing applications, integrating or updating third-party systems, and informing key stakeholders. You can configure conditional workflow steps and use variables to access information captured during the request process. You can configure completion workflows for an entire service, or for individual components.



Design considerations

Does software need to be installed after deployment? Do you need to run other scripts? If so, is there a preferred method of doing so today?

Should additional ownership beyond the person who requested the VM be set at this time?

Are additional notifications desirable?

Are there specific IPAM requirements that can be implemented using completion workflows?

Best practices

Use completion workflows to automate the most error-prone and time-consuming tasks.

Examples

A completion workflow can:

- Wire multiple components together from a networking perspective: If you have a service containing a Tomcat server, a SQL server database, and an nginx web server, a service-level completion workflow can configure and link the three components by updating configuration files with proper networking information (such as IP addresses).

- Configure a load balancer or firewall for a group of components: Once all deployed components in a service are configured, a service-level completion workflow can communicate with a load balancer to include the service into its cluster.
- Advertise that a service is available: A service-level completion workflow can specify that whenever an asset, VM or service is created or deleted, your IT management and CMDB software are updated.
- Assign an IP address
- Install an operating system, patches and applications

Walk-Through: Creating a Workflow

Examples: Automating VM Customization through Workflows

Completion Workflow Examples on our Knowledge Base

Learn more

Creating a Completion Workflow

Configuring Virtual Networking through a Workflow Step

Configuring OS Networking through a Workflow Step

Migrating a VM through a Workflow Step

Steps to Change Request Automation

Changes to services can be managed through service change requests, through policy, directly by users, or through a combination of these methods. The benefit of change requests is that you can set up an approval process. Requiring a change request also reduces the chance of accidental decommissioning.

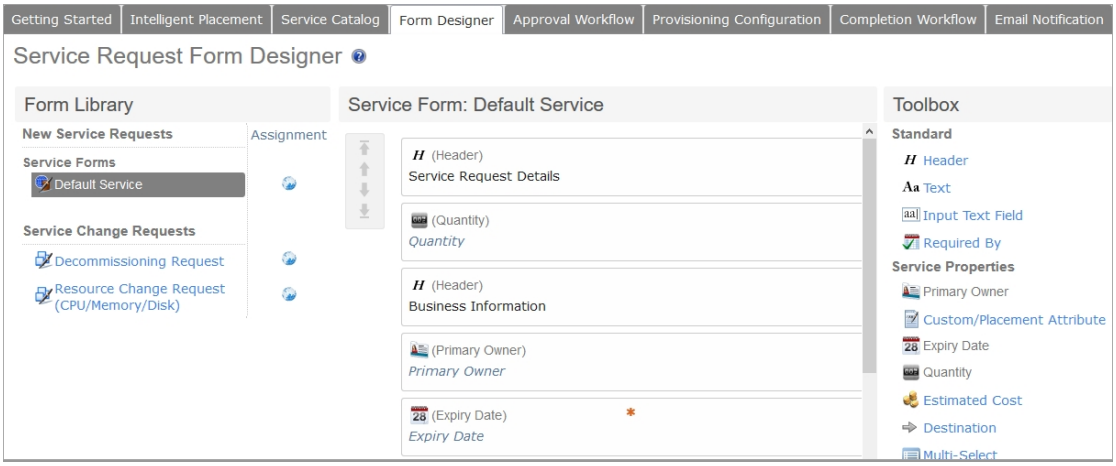
Decide which types of service change request it makes most sense to automate:

- resource changes
- rightsizing recommendations
- ownership (including migration as necessary)
- decommissioning

In this section

Step 1: Change Request Forms

You can customize any number of forms for different users, groups and organizations, as well as for different types of change request. vCommander provides default forms for decommissioning requests and resource change requests.



Design considerations

Do you need to present different organizations or groups with unique information? For example:

- Do you need to track "business process" data, such as project codes or ticket numbers?
- Will users be allowed to change CPU, memory or storage resources, or change the instance type?
- Will users be able to change expiry dates, or specify that a service will never expire?
- Do you need maintenance windows for automated fulfillment of changes requiring power-down?
- Should some users be granted permission to specify when a change request will be fulfilled?

- If you're managing Google Cloud Platform, do you want to allow users to request custom instance types?

Best practices

Tailor your change request process to how knowledgeable your users are, and how much freedom you want to allow them. For example, requiring users to reconfigure resources through a change request ensures that the change can be reviewed and approved. However, you may want to grant power users permission to modify resources directly, as well as the ability to schedule change requests.

The same holds true for expiry extension. While you can also provide expiry extension through policy, the benefit of change requests is that you can configure an approval process.

Examples

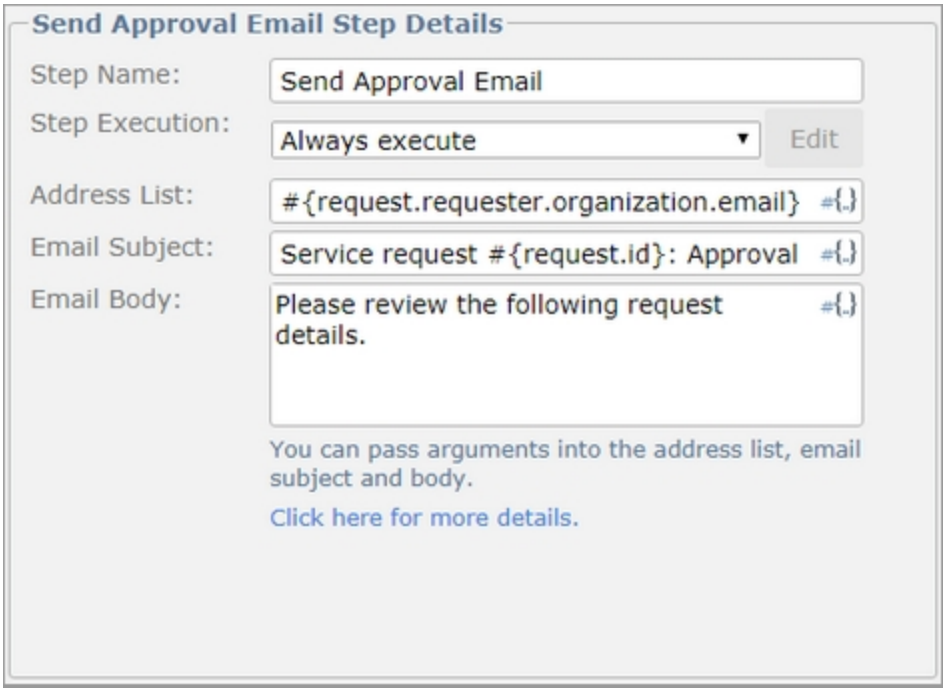
Automating Self-Service Decommissioning

Learn more

Customizing Service Request Forms

Step 2: Change Request Approval

Using an approval workflow, you can specify approvers and customize approval emails for change requests. Conditional workflow steps and variables provide considerable flexibility. Variables also permit access to information captured during the request process.



Design considerations

For resource change requests, do you want to make approval dependent on quota availability?

When a Service Portal user has permission to manage an organization, they can request changes to organizational services that they don't own – including automatic decommissioning. Do you want to require primary owner approval before any changes are made?

Does your IT organization adhere to ITIL?

When can requested changes be fulfilled?

Best practices

Your business policies and procedures should dictate the formality of the approval process for change requests.

Examples

Requiring Primary Owner Approval for Changes Requested by Non-Owners

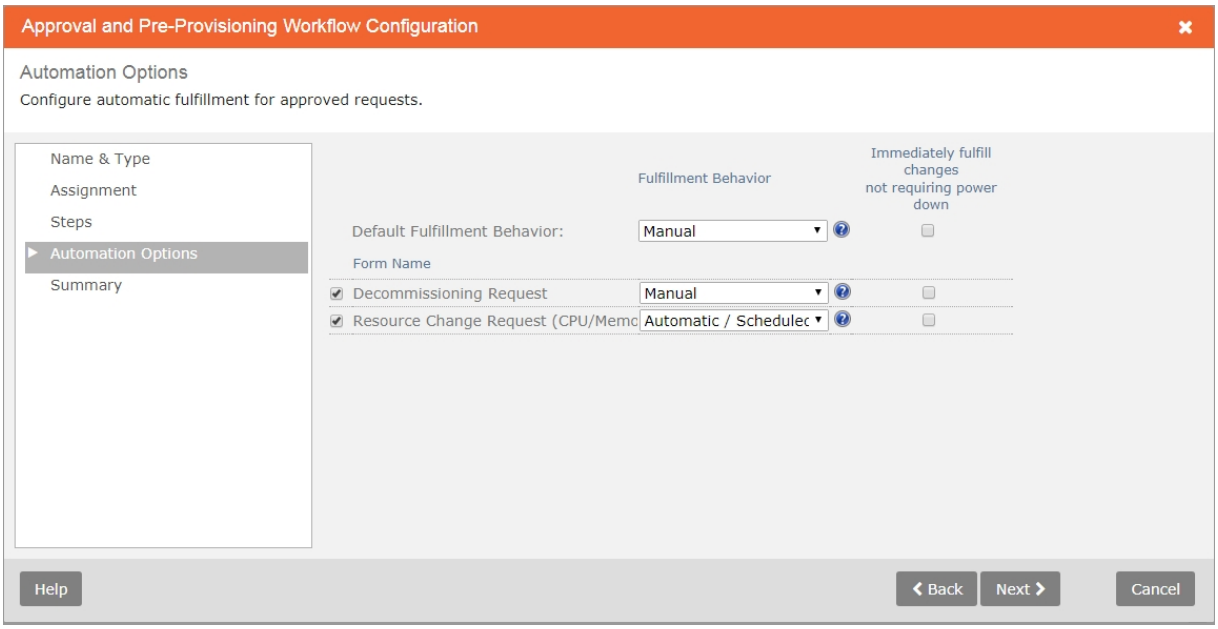
Choose Approvers Based on Request Form Used

Learn more

Creating an Approval and Pre-Provisioning Workflow

Step 3: Automatic Fulfillment of Change Requests

vCommander provides flexible options for the automation of change request fulfillment. You can specify distinct options for each of your change request forms. As with new service requests, change requests can move from approval to completion with no intervention.



Design considerations

For each type of change request, decide whether fulfillment should be:

- manual or automatic
- immediate, scheduled, or during the maintenance window

Do you need to configure different behavior for changes requiring power-down? You can create an approval workflow that causes change requests requiring a power-down to be automatically fulfilled during the next maintenance window. Or, you can add conditional workflow steps that are executed only when the change requires a power-down, providing fine-grained control over change request approval and fulfillment.

Best practices

Automate change request fulfillment whenever possible, to remove unnecessary burden from your administrators.

Examples

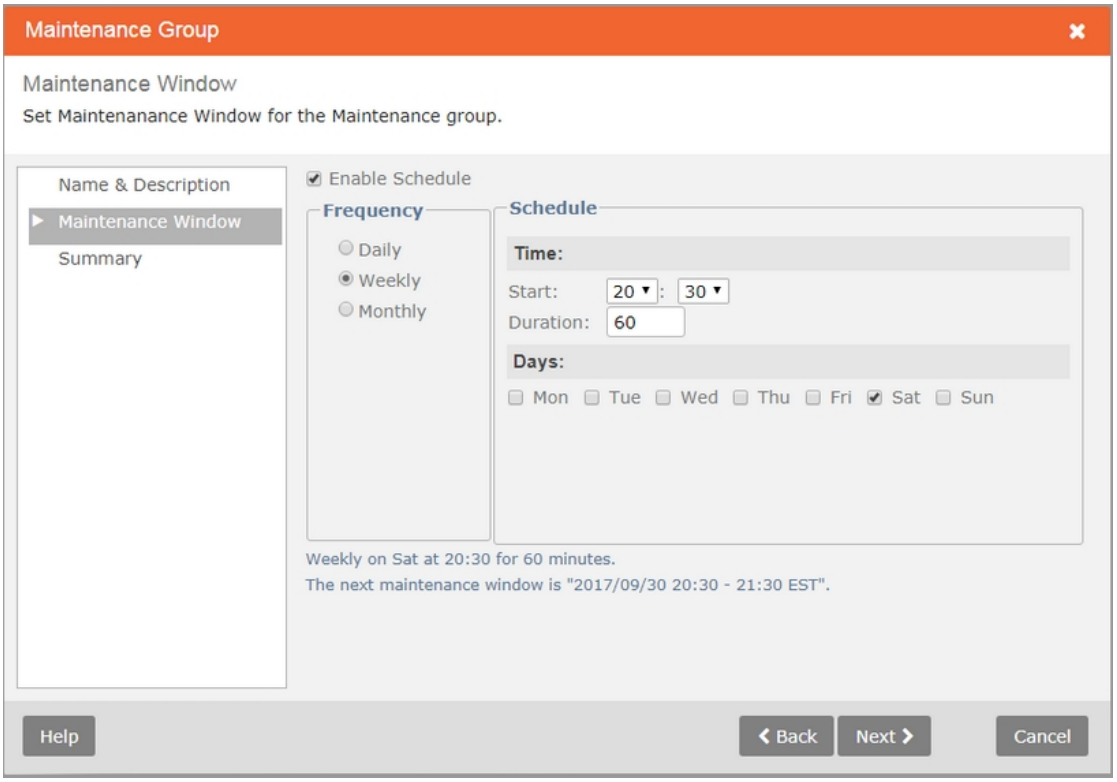
Automating Self-Service Decommissioning for Specific Users

Learn more

Creating an Approval and Pre-Fulfillment Workflow for Change Requests

Step 4: Maintenance Windows for Fulfilling Change Requests

Maintenance windows make it easy to automatically fulfill change requests and VM rightsizing recommendations at expected, well-known times. While the fulfillment time for a change request can be manually scheduled, maintenance windows enable you to remove the burden from the administrator while still avoiding any impact on customers' use of their services.



Design considerations

Do you need distinct windows for different organizations, or for production and development services?

Do you want to allow Service Portal users to schedule their own change request fulfillment? What about allowing them to schedule rightsizing recommendations during the maintenance window?

Best practices

Configure the Default Attributes policy to automatically assign the correct maintenance group to new services.

Examples

VM resource changes: A user submits a resource change request that requires powering down the VM. The approval workflow for this change request form is configured to fulfill disruptive requests during the maintenance window.

Virtual service decommissioning: A user submits a request to decommission a virtual service. The requester uses the Schedule form element to specify that the virtual service should be decommissioned during the maintenance window.

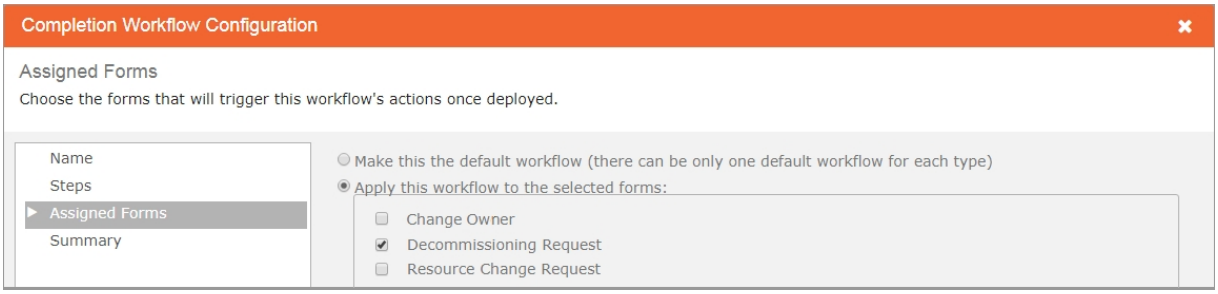
Database backup: A user submits a change request to run a database backup. The approval workflow for this change request form is configured to fulfill requests during the maintenance window. A completion workflow with an Execute Script step to run the backup is also assigned to this change request form.

Learn more

Configuring Maintenance Windows

Step 5: Completion Workflows for Change Requests

Completion workflows control what happens after a change request is fulfilled. You can configure a distinct completion workflow for each of your change request forms. Conditional workflow steps and variables provide considerable flexibility. Variables also permit access to information captured during the request process.



Design considerations

Are some of your user groups experienced enough to manage their own self-service? For example, in a lab environment, administrator approval likely isn't necessary for actions by a testing group that regularly decommissions its own services.

Best practices

Use completion workflows to automate the most error-prone and time-consuming tasks.

Examples

Use a decommissioning workflow to power off a VM and delete it from disk.

Automating Self-Service Decommissioning for Specific Users

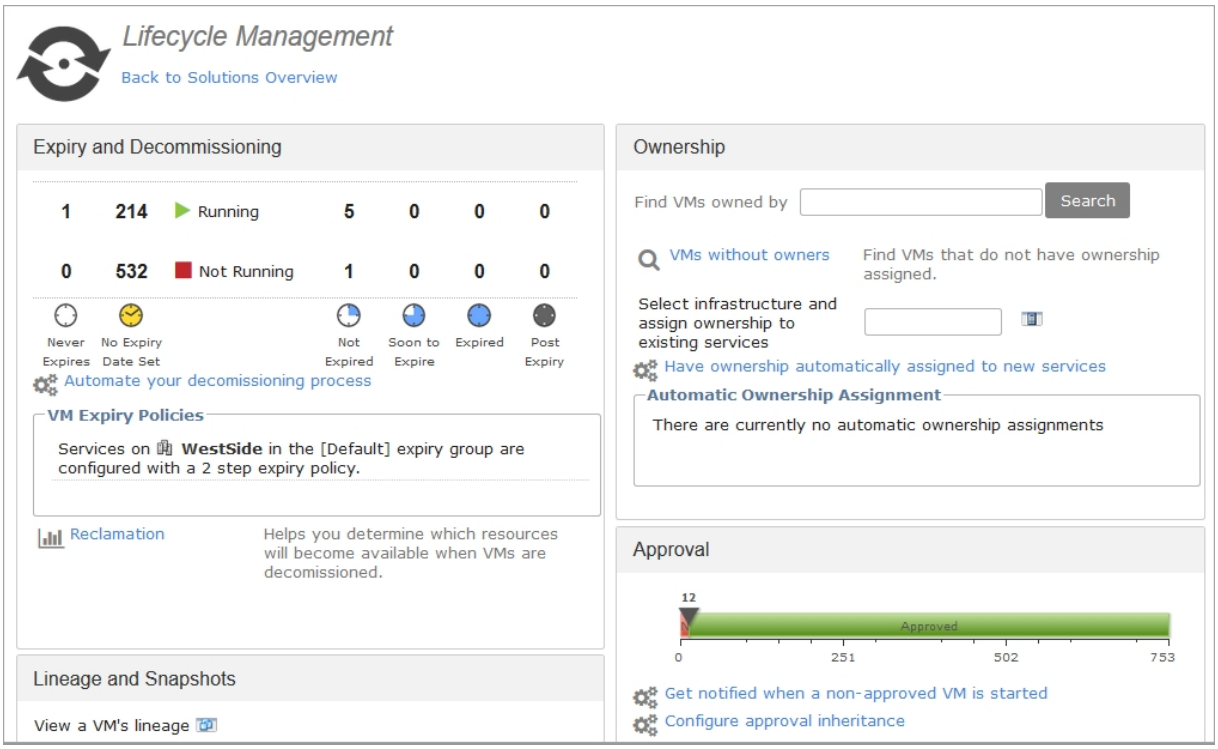
Automatically Expanding Partitions on Resized Disks

Learn more

Creating a Completion Workflow

Service Lifecycle Management

With the outdated service ownership model, the same workload is used, re-used and repurposed for months or years. The rental model, by contrast, provides just-in-time provisioning based on standardized workloads. vCommander automates the management of the VM lifecycle, ensuring that you're not paying for unused services.



Using a combination of ownership assignment, expiry information and custom metadata, you can automate service lifecycle management. Flexible notification and expiry extension options ensure that users are well informed and able to keep services active for as long as they're needed, without introducing sprawl.

Assigning ownership has several benefits. Remember that organizational ownership, the key multi-tenant concept, enables configuration of quotas for your consumer groups. And because an organization can have one or more managers with permissions to manage an organization's VMs, organizational ownership allows you to delegate administrative tasks.

Custom attributes enable you to assign an unlimited amount of metadata to services and cloud infrastructure. Once assigned, this metadata persists throughout a service's lifecycle, enabling administrators to know exactly what a workload is being used for. You can use custom attributes as filters for vCommander's custom searches and built-in reports.

Design considerations

Would you benefit from delegating responsibility for VM management to multiple individuals, for example, help desk, IT contact and primary owner?

Who needs to access a particular service through the Service Portal?

Who needs to receive email notification about expiry and vCommander policy alerts?

Will you allow users to specify that a service should never expire?

Will you allow users to extend the lifetime of their services? If so, will you limit the number of times that a user can extend service expiry dates?

Will you allow Service Portal users to edit custom attribute values?

Do you need particular custom attribute values to persist to deployed VMs, or are they needed only during the service request process?

Do you need to archive certain services rather than decommission them immediately?

Is managerial approval required before decommissioning can occur?

Best practices

Use policy to ensure that all services, whether created by vCommander or created in the managed system and managed by vCommander, have assigned owners.

Ensure that all services have an expiry date or are set to never expire. Implement lifecycle management, including the enforcement of expiry dates on request forms and using decommissioning policies for short-lived workloads.

Automate the decommissioning of expired services based on data retention policies.

Examples

If you're a service provider, you could configure policy to assign service ownership at the folder level, to ensure that organizations are charged for all services they're using.

Design your custom attributes so that budget or profit center code information for a VM or an asset number for a cloud infrastructure element can be applied.

Configure a completion workflow for a decommissioning request to power off and delete a VM.

Requiring Primary Owner Approval for Changes Requested by Non-Owners

Send Expiry Emails to All VM Owners, But Allow the Primary Owner to Extend the Expiry Date

Allow Requesters to Specify the Machine Role, Application and Version

Synchronizing vCenter Notes, Tags and Annotations to vCommander Custom Attributes

Add a Custom Attribute to the Request Form for Specifying the Network

Automate Self-Service Decommissioning for Specific Users

Learn more

Assigning Ownership to Services

Assigning Service Ownership with the Default Ownership Policy

Requiring Primary Owner Approval for Changes Requested by Non-Owners

Managing Service Expiry

Controlling Expired Services with the Expiry Policy

Managing Expiry Extension Emails

Tracking Expiry Extensions Using vCommander

Using Custom Attributes to Add Infrastructure Metadata

Deleting a Deployed Service

What's Next?

You can find more information on vCommander here:

- Knowledge Base
- Training Videos
- Online vCommander Documentation
- Blog

Integration with external systems

Embotics Corporation provides a REST API with a comprehensive set of features to provide automation and integration capabilities. A few examples of how our customers benefit from our REST API:

- Dev/test lab automation for enterprise
- Multi-tenant customer on-boarding for service providers
- Synchronization of asset inventory with third-party systems

Training and professional services

Embotics is happy to provide training and professional services – just ask.

Support

If you need additional help and support, send an email to support@embotics.com or telephone 877-599-0494 (toll-free in Canada and the US) or +1 613-599-0494.

Version 7.5.0

Document issue 1.0