vCloud APIs - Architecture, Integration

VMware Coffee Talk Webinar - vCloud API

- Prasad Pimplaskar

Ecosystem Engineering, VMware

Disclaimer

This session may contain product features that are currently under development.

This session/overview of the new technology represents no commitment from VMware to deliver these features in any generally available product.

Features are subject to change, and must not be included in contracts, purchase orders, or sales agreements of any kind.

Technical feasibility and market demand will affect final delivery.

Pricing and packaging for any new technologies or features discussed or presented have not been determined.

“These features are representative of feature areas under development. Feature commitments are subject to change, and must not be included in contracts, purchase orders, or sales agreements of any kind. Technical feasibility and market demand will affect final delivery.”

About the speaker

Prasad works as Sr. Member of Technical Staff in VMware’s EcoSystem Engineering Group, mainly focusing on the vCloud and vSphere Web Services API. Currently he is working with vCloud Service Providers and ISV partners for smooth adoption of VMware’s vCloud vision.

Prasad has more than 15 years of experience in Enterprise Software technologies in various technical and management capacity.

Prasad holds B.S. in Computer Engineering and M.S. in Software Engineering with emphasis on Enterprise Technologies. He also teaches graduate classes as a visiting faculty in San Jose State University.

Takeaways

How vCloud Ecosystem is placed

Know about the vCloud API

Understand the flow of API using the vCloud resources

Know about upcoming Java Library for vCloud API

Agenda

vCloud Ecosystem

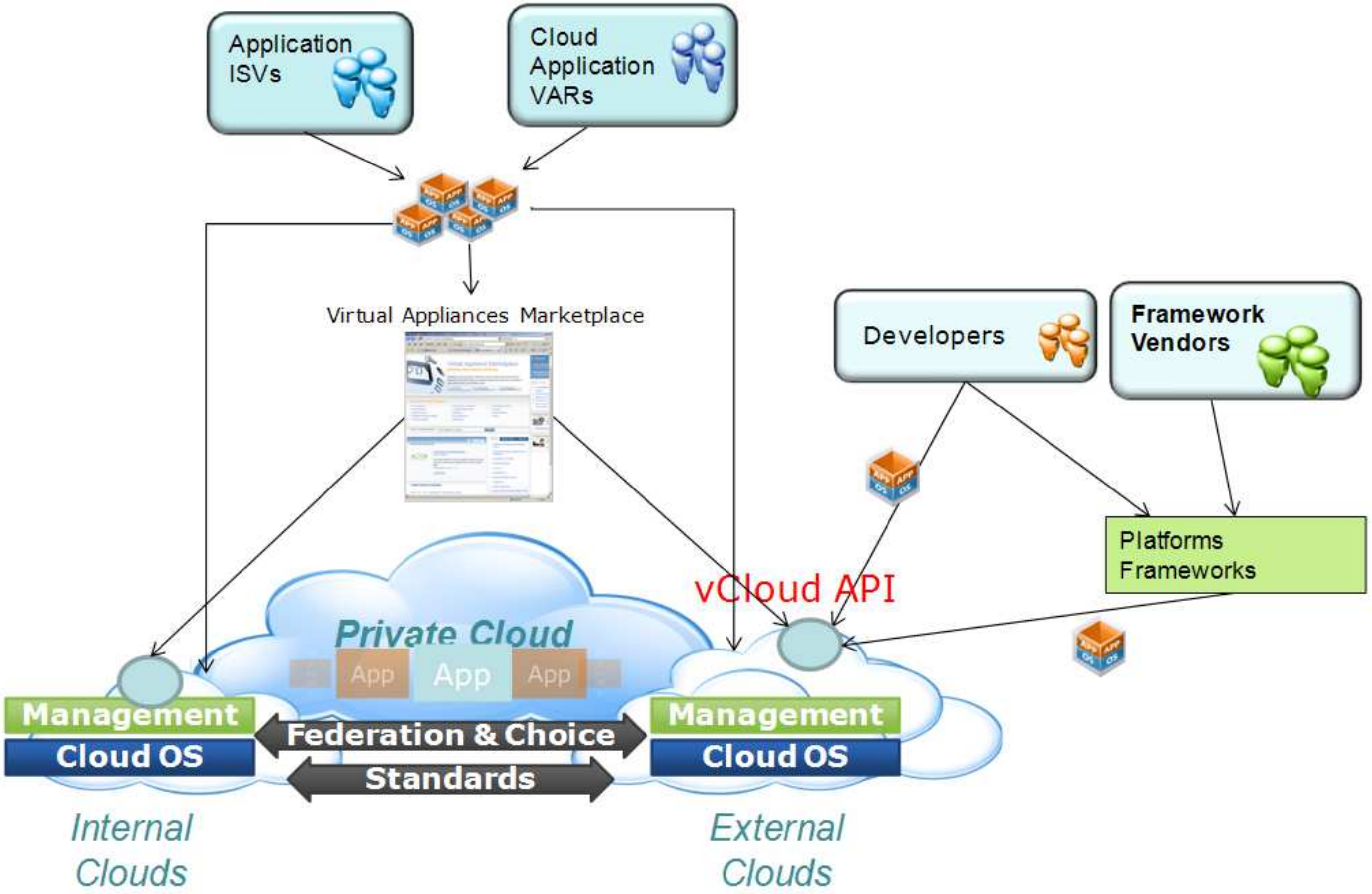
vCloud API concepts

vCloud API deep-dive

vCloud API and vCloud Express

vCloud API - Java Library

vCloud Ecosystem



vCloud Ecosystem

Content Providers

- ISVs, VARs, IT Admins, Enthusiasts, Developers

Content

- Virtualized Software Solutions (vApps)
- Existing legacy solutions, or new apps written for the Cloud

Content Respositories

- Object Stores, Virtual Appliance Marketplace

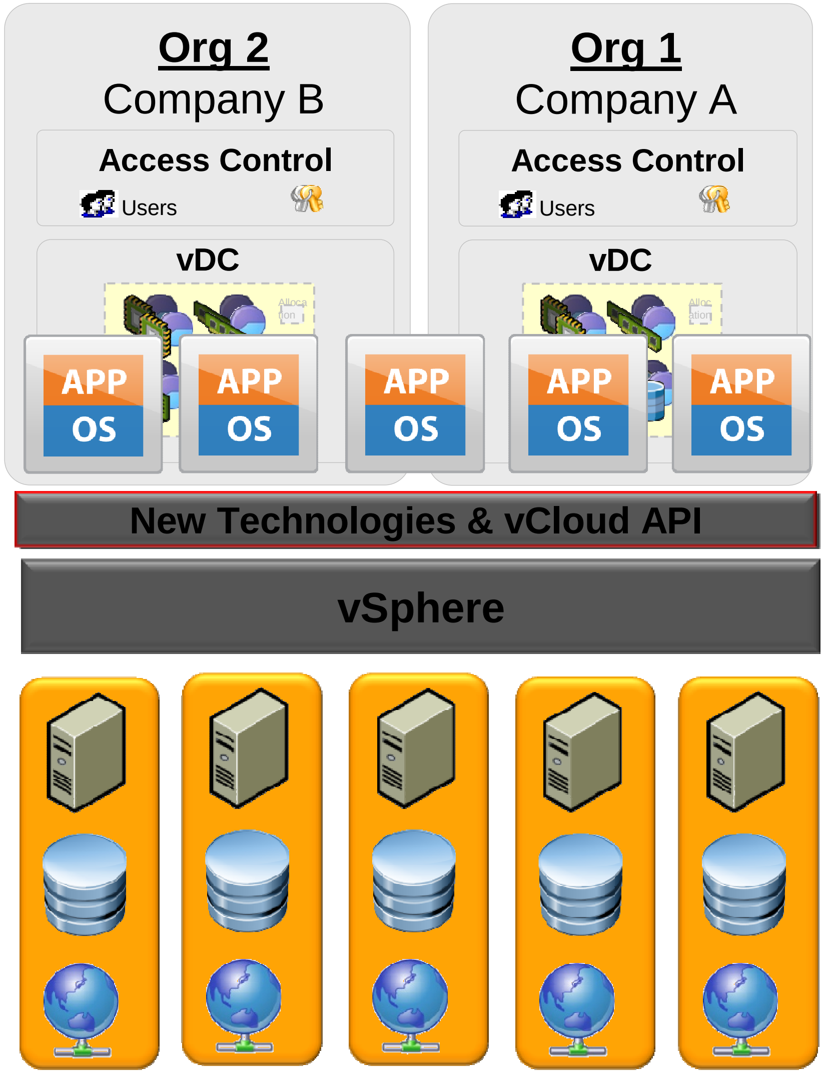
Cloud Service Providers

Enterprises

Federation

vCloud High Level View

- Provide SW so that anyone can easily create cloud
- Supports all applications
- Can scale up to large deployments
- Secure multi-tenancy
- Controlled programmatically through standard interfaces



Org 2
Company B
Access Control
Users
vDC

Org 1
Company A
Access Control
Users
vDC

New Technologies & vCloud API

vSphere

vCloud Added Value

Pure Virtual

- Physical Infrastructure is Abstracted Away

Scalable

- Management Layer
- Compute Capacity

Self-Service User Interface

Standard API

- Admin
- User

vApp: The Next Generation VM

An uplifting of a virtualized workload

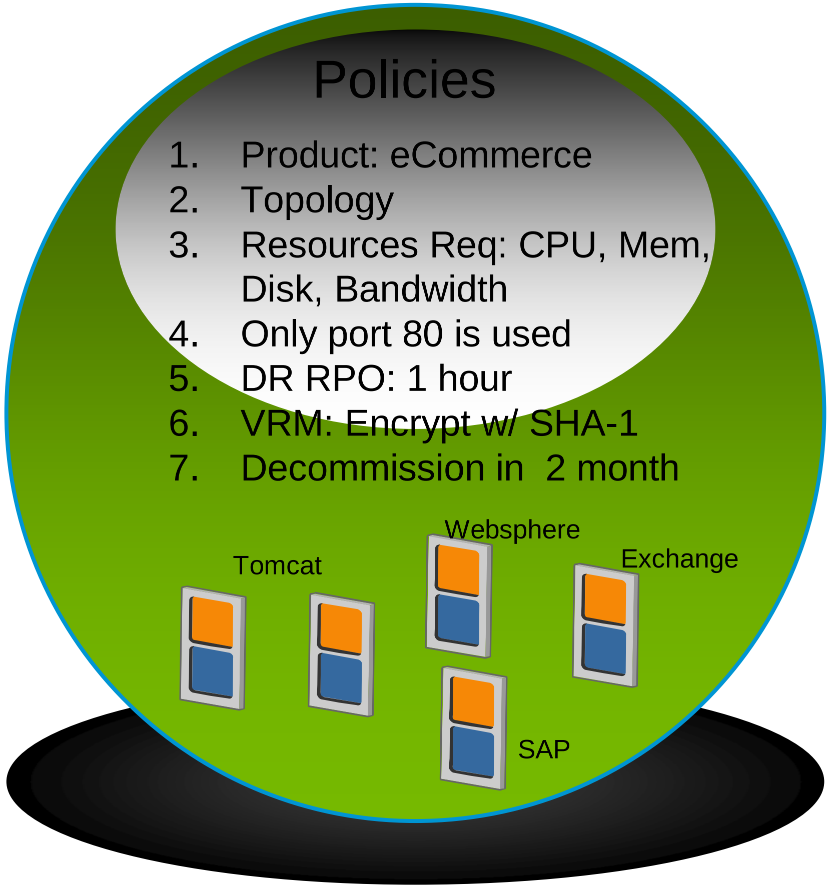
- VM = Virtualized Hardware Box
- App = Virtualized Software Solution
- Takes the benefits of virtualization: encapsulation, isolation and mobility higher up the stack

Properties:

- Comprised of one or more VMs (may be multi-tier applications)
- Encapsulates requirements on the deployment environment
- Distributed as an OVF package

Built by:

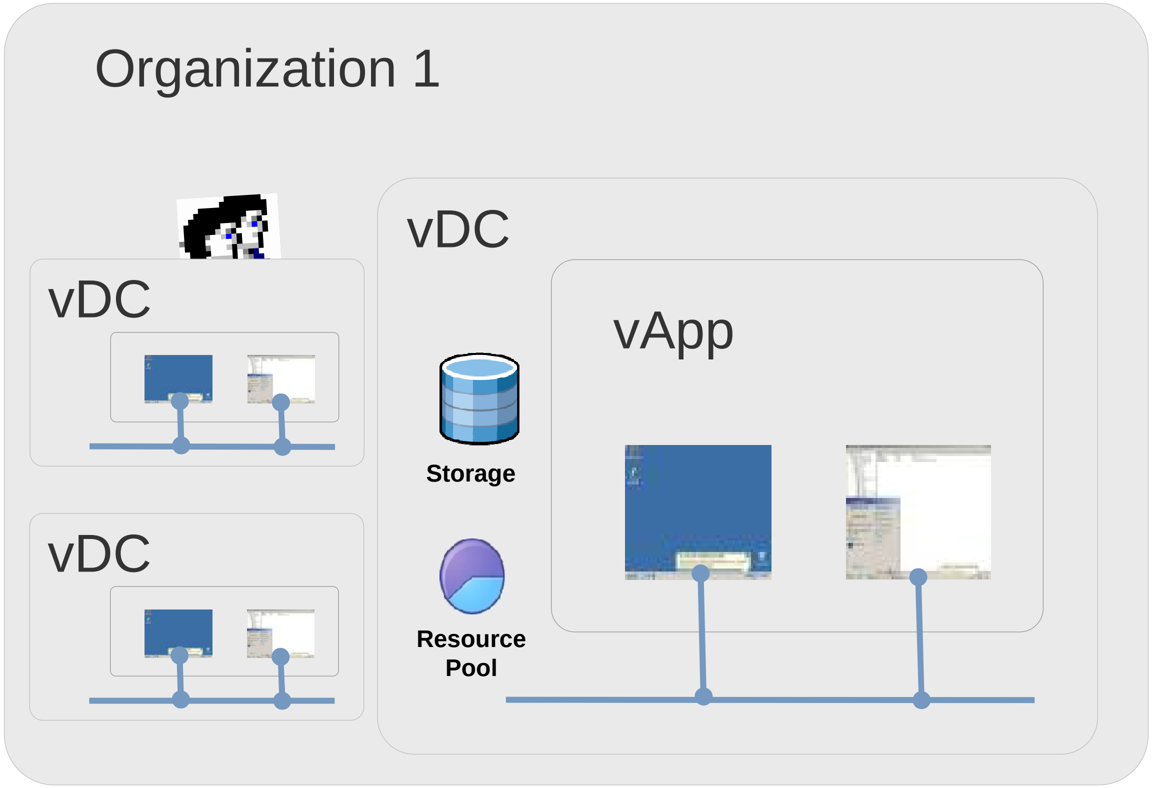
- ISVs / Virtual Appliance Vendors
- IT administrators
- SI/VARs



Policies

1. Product: eCommerce
2. Topology
3. Resources Req: CPU, Mem, Disk, Bandwidth
4. Only port 80 is used
5. DR RPO: 1 hour
6. VRM: Encrypt w/ SHA-1
7. Decommission in 2 month

Abstractions



Agenda

vCloud Ecosystem

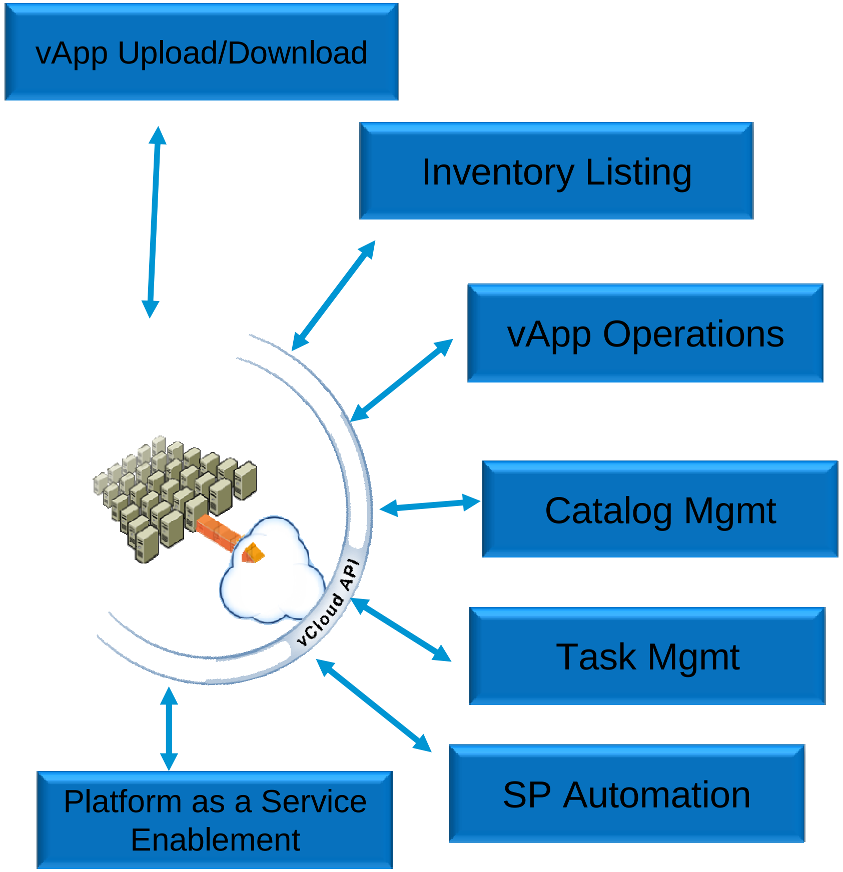
vCloud API concepts

vCloud API deep-dive

vCloud API and vCloud Express

vCloud API - Java Library

vCloud API Functions and Details



Spans vCenter Instances, Internal and External Clouds

Subset implemented by vCloud Express Partners

Submitted to DMTF Cloud Standards Incubation Group

Detailed spec and programming guide available at

http://www.vmware.com/go/vcloudapi

vCloud API Salient Points

An Ecosystem API

- Platform independent
  - Can be used from virtually any platform
  - Can be implemented on a variety of platforms
- Straightforward to use and implement
- Powerful and extensible
- Standards-based

Pure-virtual API

- Resource semantics described independent of physical semantics
  - Physical infrastructure is hidden
- Simpler semantics focused on end-use and not on virtualizing

vCloud API Features

Release in ‘Open’ form

- Submitted to DMTF

Interface for:

- Providing and consuming resources in the Cloud
- Deploying and managing virtualized workloads in the Cloud
- Migrating virtualized workloads between Clouds.

REST based

Multi-tenant

Self-service

Service provider automation

vCloud API

vCloud API collectively is:

- Way to enable clients to interact with clouds using established, standard based protocols and data formats
- Result of combined efforts by VMware and its partners to create the programming framework and enable the Ecosystem.
- Conceptual models of the system; Entities and relationships between them
- REST interfaces that exposes the functionality
- Mapping of the entities to REST resources

vCloud design goals:

- Extensibility
  - Start with small set of functionality using limited entities and increase the scope subsequently
- Modularity
  - Share API components (resources, entities) between different functionality sets
- Usability
  - Simple to use and easy to adopt considering different types of client needs

Representational State Transfer

Style of architecture based on the design that uses resources and focuses on their states

Not a standard but a design pattern that recommends use of standards (such as HTTP, URL)

Application state and functionality are abstracted into resources

Every resource is located by URL and represented using XML, JSON etc.

The resource state is manipulated or inquired using GET, POST, DELETE and PUT

- Sounds like HTTP and web operations(?)
- Best suited for web development and web oriented technologies
- World Wide Web is collection of resources accessed using URLs.

This essentially provides us CRUD methods hence can be applied to any resource

Real life models translated to resources

REST directive:

- NO side effects of GET operation
- PUT is idempotent
- …

Why REST

REST is stateless and thus improves scalability

Hyperlinks in the resources avoids separate discovery mechanism for resources

Web friendly HTTP CRUD operations works well with proxies and firewalls

Loose coupling of service implementation and access

No enforcement of OO so that resource implementation can change without much affecting the client logic

No enforcement of representation protocol

- Payload can be XML, JSON and different content types as defined in the HTTP specification.

Widely adopted and easy to use

Extensible - we can use different content types and add new resource representations without breaking existing implementation or client code

Why REST

REST does not enforce the content type.

Resource entities commonly use XML/JSON so that consumption in terms of object is easy

The XML payload can be constrained by schema (which defines the object model) so that the client and server are free to use OO and modeling techniques

Given resource can be accessed by multiple URLs

- E.g. <vm-uri>/nic/{id} and <network-uri>/nic/{id} may refer to same resource

Certain operations are seamlessly represented by RESTful (HTTP CRUD) way

- Create resource (create network) – POST
- Retrieve resource (Get organization details) – GET
- Update resource (change network connection of VM) – PUT
- Delete resource (delete vApp) – DELETE

REST in Real Life Which is *Virtual*

Representing actions

- In real life everything can not represented using the CRUD methods such as the operations on resources. E.g. power on/off VM

REST purist – RESTful vs. REST-like

- CRUD operations in practice limits the API and its usability
- Limited web vocabulary limits the functionality made available by API or needs lots of operation overloading.
- Very few commercially available Pure REST API. We take the REST-like REST-RPC hybrid approach where in operations are overloaded e.g. POST in certain conditions means execute the overloaded action.

Operations on virtual resources

- CRUD operations on virtual resources
- VM power on/off

Long running tasks are also resources

- Any POST, PUT, DELETE operation resulting in Task creation will return a Task resource back
- The Task resource provides more information and can be queried to get the result (success or error)

vCloud API and vSphere API

vSphere API: “Under the Hood” API

- Used to create virtual resources
- Virtualization API
  - Exposes physical<->virtual mapping
- Targeted at sys-admins
- Product specific
  - Exposes vSphere’s capabilities
  - Tied to a specific implementation
- Rich and powerful
- VMware’s Cloud OS Platform API

vCloud API: “Driver Seat” API

- Uses virtual resources
- Pure-virtual API
  - Hides underlying physical resources
- Targeted at cloud tenants
- Product agnostic
  - Standardizable
  - Variety of implementations
- Simple yet powerful
- Implementable on the vSphere API
- Not a replacement for the vSphere API

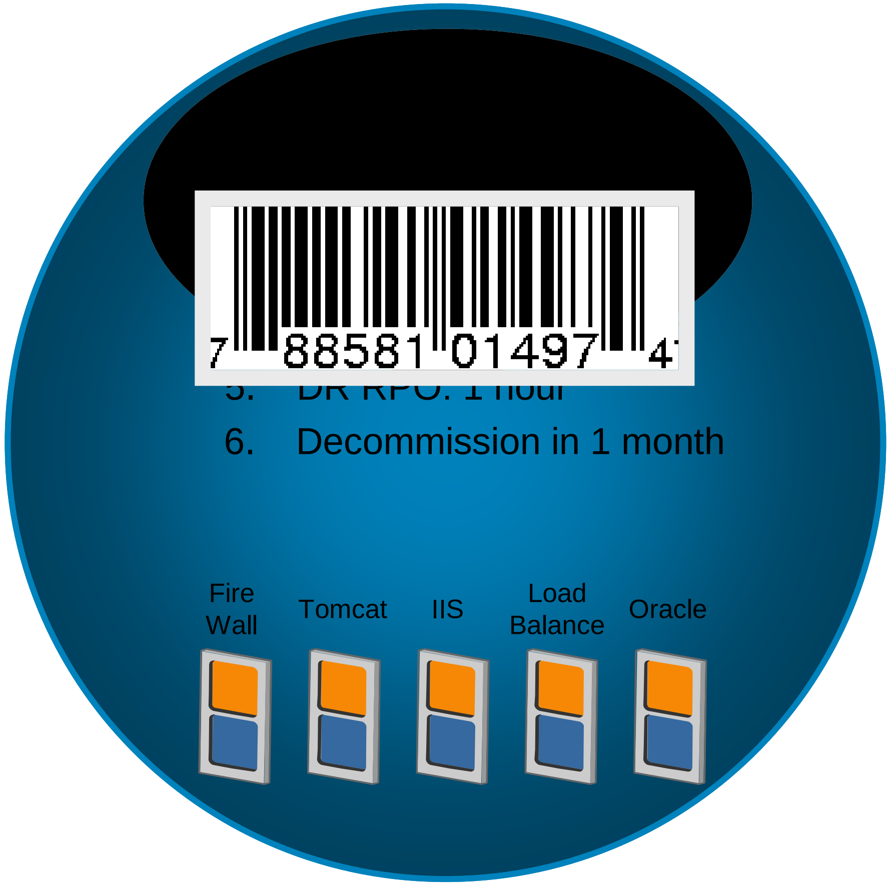
vApp – Next Generation VM Concept

An uplifting of a virtualized workload

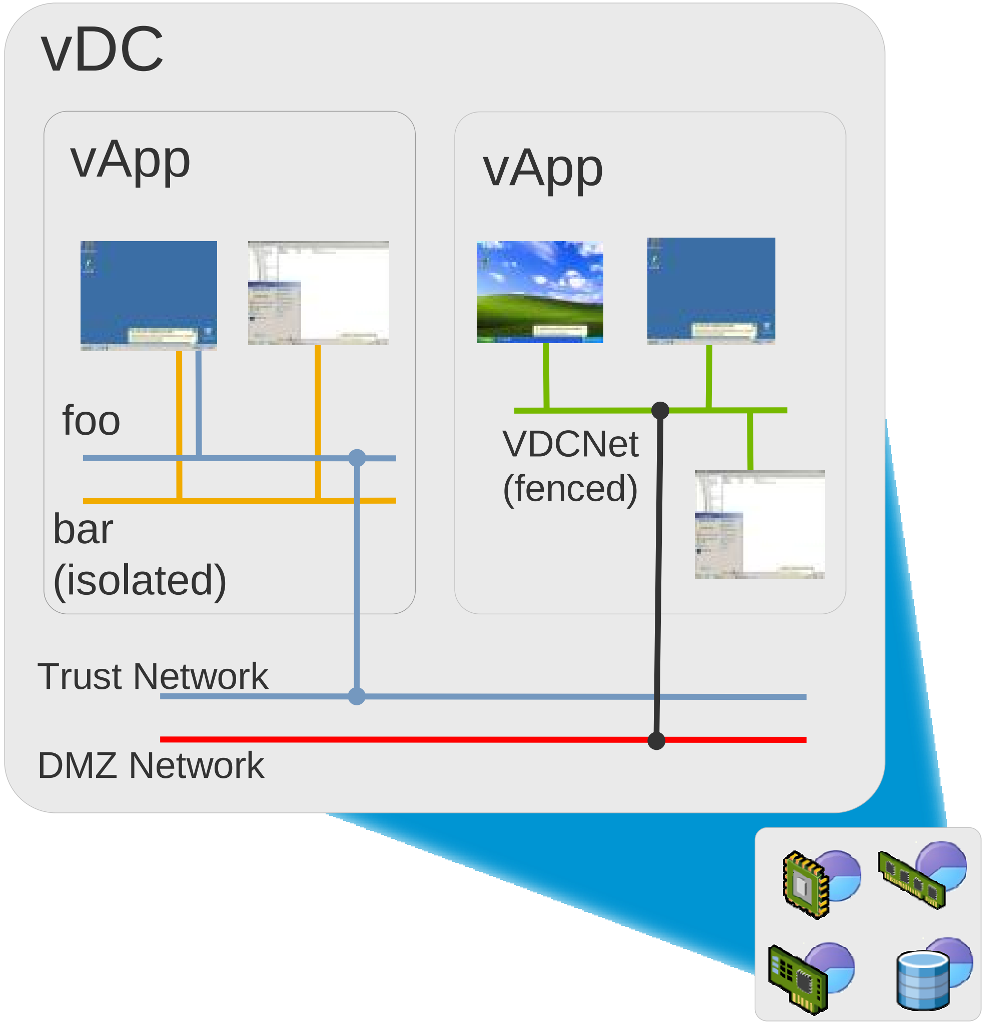
- VM = Virtualized Hardware Box
- vApp = Virtualized Software Solution
- Encapsulation, isolation and mobility higher up the stack

Properties:

- Comprised of one or more VMs
- Encapsulates requirements on the deployment environment
- Distributed as an OVF package



vDC – vApp Deployment Environment



Type of Commodity

- Compute, Storage and Network SLAs

Quantity

- MB of RAM, MHz of CPU, GB of Storage

Scope for Over-provisioning

Other features

- L2 Networks
- Persistent vApps

Agenda

vCloud Ecosystem

vCloud API concepts

vCloud API deep-dive

vCloud API and vCloud Express

vCloud API - Java Library

REST Resource Type - Entities

Entity resources

- Corresponds to things that can be persistent and are independent of API technology, regardless of REST model, data model, OO model
- Helps define objects from the data/model.
- <vm-uri> is an entity representing VM but <vm-uri>/action/poweroff is not since its not a thing that can be persisted

Entity properties

- Properties as in OO sense which comprise the state of the entity
- Accessed by GETting the entity resource
- Also accessed using the subordinate URI of the entity URI e.g. <vm-uri>/name
- Does not include reference to other entities. No entity relationship using the properties

Entity links

- Created by the Server and Read only for the client
- Can be treated as simple property whose value is a URI to some other resources
- Every entity also provides a URL to itself.

REST Resource Type - Entities

Entity resources are defined in XML schema

Entity can have zero or more attributes and zero or more sub-components

Attributes are presented using XML data types

Entities can have mutable sub elements accessed by HREF attribute which is a URI

Behavior

- **GET** returns the representation of the entity.
- **PUT** if applicable updates the representation of the entity. The body of the PUT must include the entire representation of the entity that will replace the original representation.
- **DELETE** destroys the entity
- **POST** is N/A

REST Resource Type - Facets

Represents self contained portion of resource functionality

- Examples: <vapp-uri>/power, <vapp-uri>/snapshot

Can have action and properties but not Links

Facet actions are described in entity functionality

- Example

```
<VApp ...> :..
     <Link rel="power:powerOn"
       href="https://vcloud.example.com/v1/vapp/1983/power/action/powerOn" />
:..
</VApp>
```

API Versioning

vCloud API schema versions are reflected in the URL that are used to access the resources. The version is also reflected in the namespace for the vCloud Resources definition in the supplied XML schema.

Xsi:schemaLocation=http://www.vmware.com/vcloud/api/v1 Catalog.xsd
xmlsns:=http://www.vmware.com/vcloud/api/v1

API Version Request / Response

```
GET http://vcloud.example.com/api/versions
```

```
<SupportedVersions xmlns="http://www.vmware.com/vcloud/versions"
   xsi:schemaLocation="http://www.vmware.com/vcloud/versions
   http://vcloud.example.com/api/versions/schema/versions.xsd"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <VersionInfo>
      <Version>1</Version>
      <LoginUrl>http://vcloud.example.com/api/v1/login</LoginUrl>
      <MediaTypeMapping> … </MediaTypeMapping>
   </VersionInfo>
</SupportedVersions>
```

Example: login

Request

```
POST http://vcloud.example.com/api/v0.9/login?username=”example-
user”:password=”Hell0”
```

Response

```
Date: <request-date>

Expires: <expiration-date>

Set-Cookie: vcloud-token=<token>;Path=/
Content-Type: application/vnd.vmware.vcloud.orgslist+xml

<?xml version="1.0" encoding="UTF-8"?>
<OrgList xmlns="http://www.vmware.com/vcloud/v0.9" ... >
   <Org type="application/vnd.vmware.vcloud.org+xml"
    name="Example Corp.“
    href="http://vcloud.example.com/api/v0.9/org/1"/>
   <Org> … </Org>
   <Org> … </Org>
   <Org> … </Org>
</OrgList>
```

Simple Example: Power On

Request

```
POST https://vcloud.example.com/api/v0.9/vapp/vapp-
413/power/action/powerOn
```

Response

```
202 Accepted

<?xml version="1.0" encoding="UTF-8"?>
<Task href="https://vcloud.example.com/api/v0.8/task/389"
    type="application/vnd.vmware.vcloud.task+xml"
    startTime="2009-7-31T09:30:47Z"
    status="running" ...>

    <Link rel="task:cancel”
      href="htt.../task/389/action/cancel"/>

    <Owner href="https://vcloud.example.com/api/v0.9/vapp/vapp-413"
      type="application/vnd.vmware.vcloud.vapp+xml“
      name="My vApp"/>
</Task>
```

Simple Example: Look at a vApp

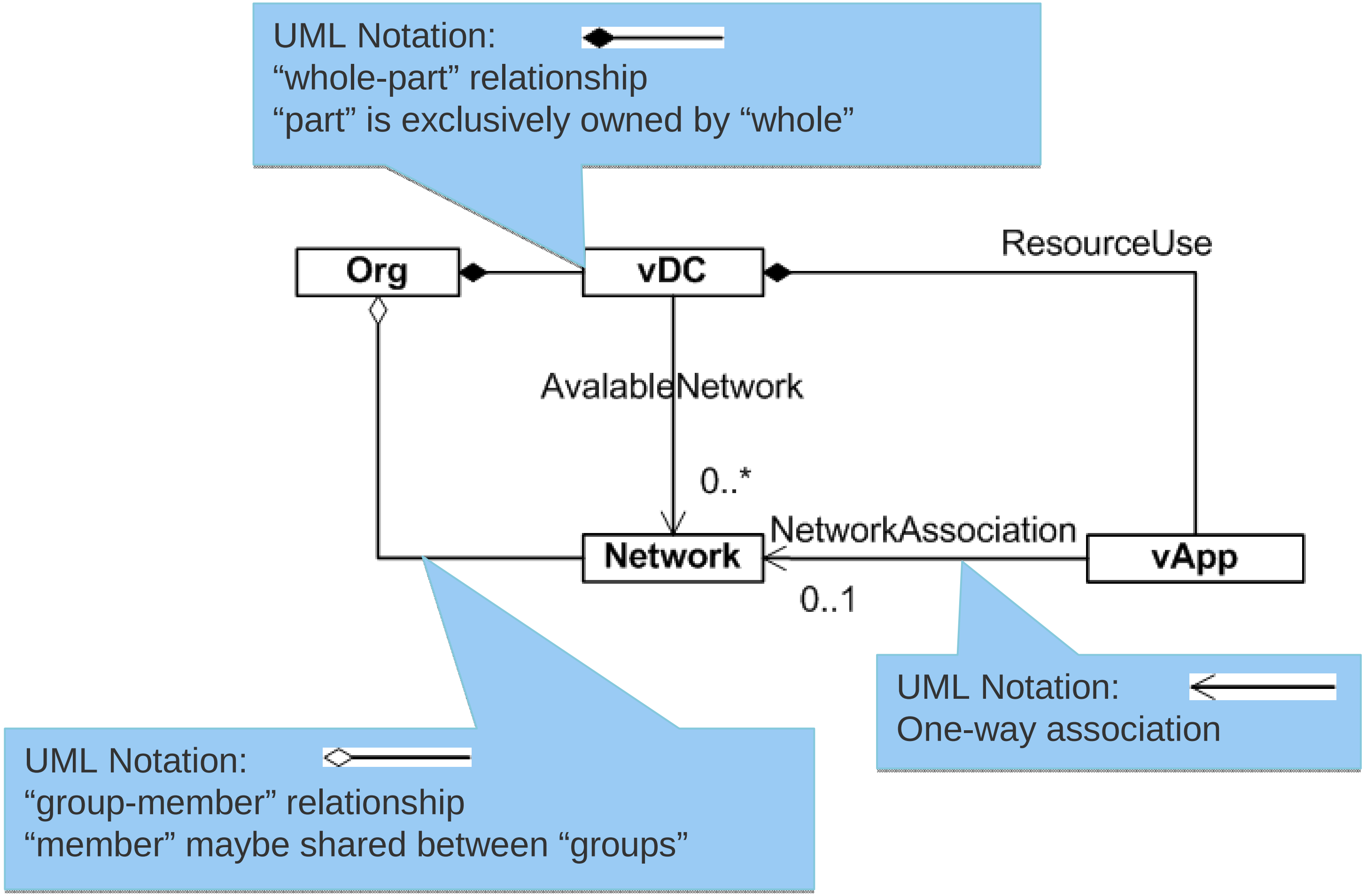
Request

```
GET https://vcloud.example.com/api/v0.9/vapp/413

Content-type: vnd.vmware.cloud.vapp+xml
```

Response

```
<VApp name="My vApp"
    status="1"
    href="https://vcloud.example.com/vapp/413" ...>

  <Link rel="up" href= "https://vcloud.example.com/vdc/128"/>
  <NetworkSection>...</NetworkSection>
  <ovf:OperatingSystemSection ...>
     <Link rel="edit" href="http..." ... />
     <Description>Microsoft Windows Server 2003</Description>
  </ovf:OperatingSystemSection>
  <ovf:VirtualHardwareSection ovf:transport="iso">
     <Link rel="edit" href="http..." ... />
     <Item>…</Item>
     …
  </ovf:VirtualHardwareSection>
</VApp>
```

Entity Model

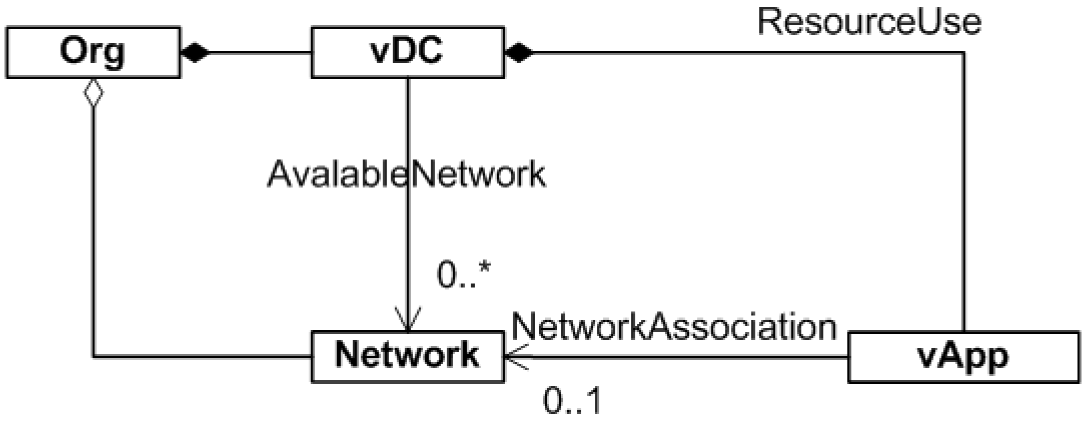


UML Notation: ◆———
"whole-part" relationship
"part" is exclusively owned by "whole"

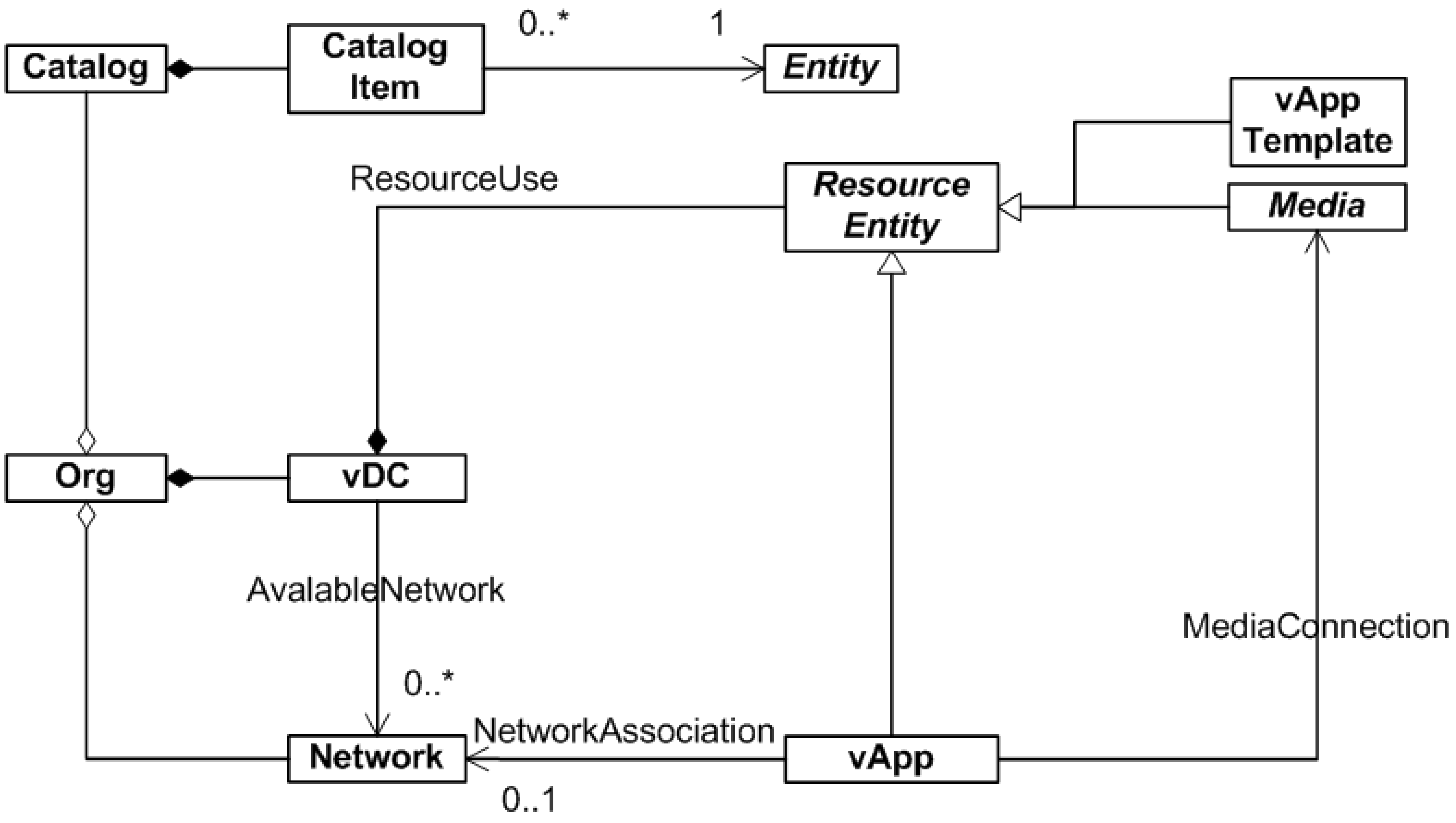
UML Notation: <———
One-way association

UML Notation: ◇———
"group-member" relationship
"member" maybe shared between "groups"

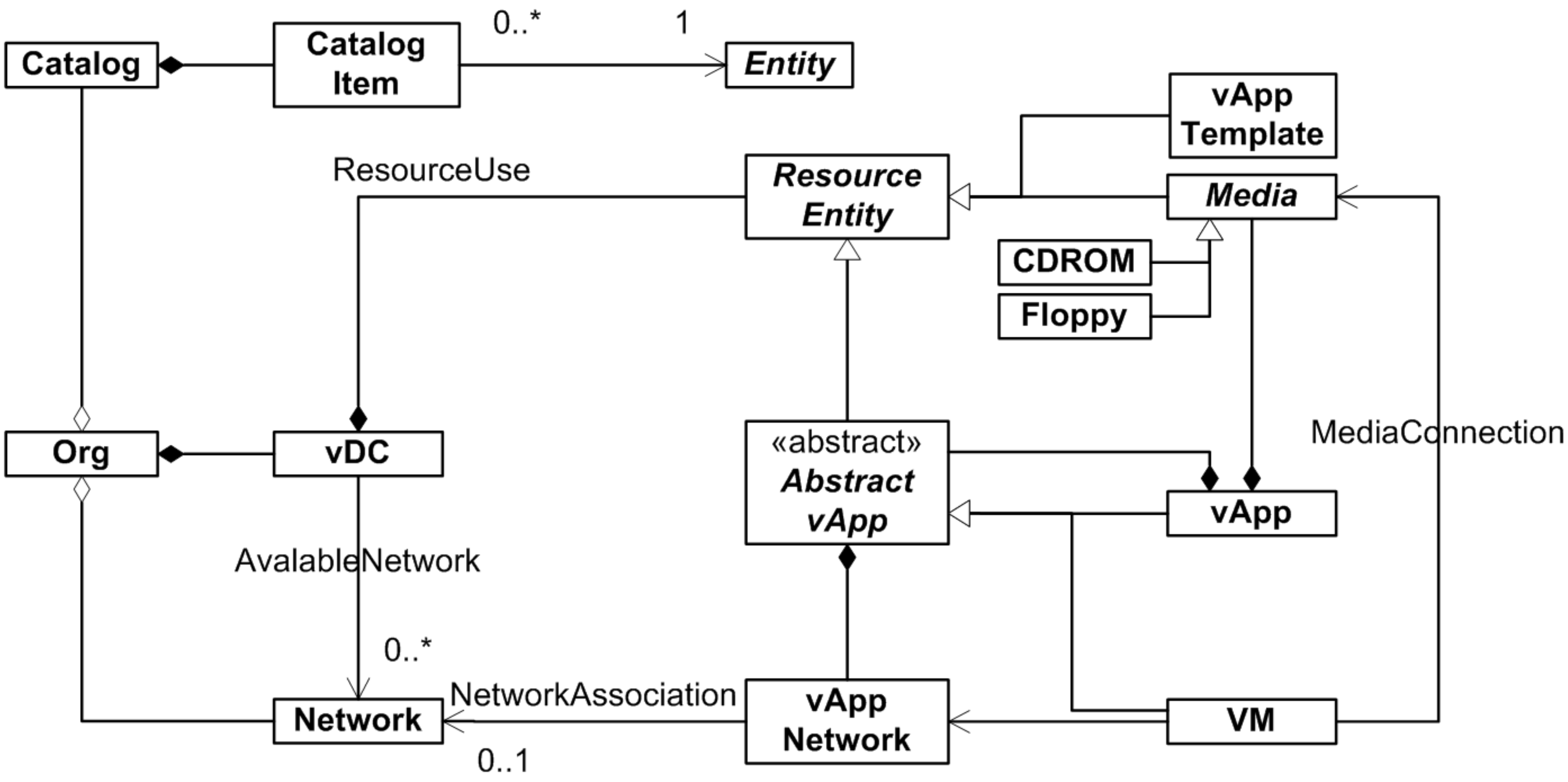
Entity Model



Entity Model



Entity Model



Get Information about an Organization

```
<Org href="https://vcloud.example.com/api/v0.9/org/25" name=“engineering-org"

  xmlns="http://www.vmware.com/vcloud/v1"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema">

      <Link rel="down" href="https://vcloud.example.com/api/v0.9/vdc/9"

        type="application/vnd.vmware.vcloud.vdc+xml" name="Miami Env1"/>

      <Link rel="down" href="https://vcloud.example.com/api/v0.9/vdc/9/catalog"

        type="application/vnd.vmware.vcloud.catalog+xml" name="Miami Env1 Catalog"/>

      <Link rel="down" href="https://vcloud.example.com/api/v0.9/tasksList/9"

        type="application/vnd.vmware.vcloud.tasksList+xml" name="Miami Env1 Tasks
 List"/>

</Org>
```

Supported Link Relationships

Rel Attribute	Description	Applicable HTTP Verb
Down	An item in this container	GET
Add	Add to this container	POST
Up	The container resource of ‘*this*’	GET
Remove	Remove *‘this’*	DELETE
Edit	Edit/Update *‘this’*	PUT
Copy	Copy *‘this’* entity to a destination specified in the request body	POST
Move	Move *‘this’* entity to a destination specified in request body	POST
Upload:default	Upload *‘this’* item	PUT
Download:default	Download *‘this’* item	GET

Example Workflow: Instantiating a vApp Template

Finding a vApp Template

Request

```
GET https://vcloud.example.com/api/v0.9/vdc/128
```

Response

```
<Vdc href="https://vcloud.example.com/api/v0.9/vdc/128"
    name="Main Vdc" ...>

    <ResourceEntities>
       <ResourceEntity
          href="https://vcloud.example.com/api/v0.9/vAppTemplate/111"
          type="application/vnd.vmware.vcloud.vAppTemplate+xml"
          name="Ubuntu Template with vsftpd"/>
       <ResourceEntity href="https://vcloud.example.com/api/v0.9/media/112"
          type="application/vnd.vmware.vcloud.media+xml"
          name="Ubuntu Boot Floppy"/>
       <ResourceEntity href="https://vcloud.example.com/api/v0.9/media/113"
          type="application/vnd.vmware.vcloud.media+xml"
          name="Ubuntu ISO Image"/>
    </ResourceEntities>
   ...
</Vdc>
```

Get Information about a vApp Template

Request

```
GET https://vcloud.example.com/api/v0.8/vAppTemplate/111
```

Response

```
<VAppTemplate
href="https://vcloud.example.com/api/v0.8/vAppTemplate/111"
   name="Ubuntu Template with vsftpd"
   status="1"
   ...>
   <Description>Description of Ubuntu Template with
vsftpd</Description>

</VAppTemplate>
```

Instantiating a vApp Template

Request

```
POST
https://vcloud.example.com/api/v0.8/vdc/128/action/instantiateVAppTemplate

<InstantiateVAppTemplateParams
   name="Linux FTP server" ...>
   <VAppTemplate href="https://.../vAppTemplate/111" />
   <InstantiationParams ...>
      <NetworkConfigSection>
         <NetworkConfig name="My vApp Net">
            <Features>
               <vmw:FenceMode>allowInOut</vmw:FenceMode>
               <vmw:Dhcp>true<vmw:Dhcp>
            </Features>
            <NetworkAssociation href="https://.../network/14">
         </NetworkConfig>
      </NetworkConfigSection>
   </InstantiationParams>
</InstantiateVAppTemplateParams>
```

Creating vApp Template Using OVF Package

A vCloud API client can create a vApp template when it has access to the OVF package using following 3 easy steps.

- Initiate Upload :
  This steps identifies the target vDC and uses the ‘uploadVAppTemplate’ action, which results in creation of vApp template entity.

- Upload OVF descriptor:

The vApp Entity created above provides a link to load the OVF descriptor. The client is expected to use simple PUT operation to upload the contents.

- Upload the Disk Files:

As a result of uploading the OVF contents the vApp now lists the VMDK files with the corresponding URL and attribute ‘ovfDescriptorUploaded’ = True.

This final step uploads the disk file contents using series of HTTP PUT requests for every File in the list provided by vApp entity.

Initiate upload of vApp Template

Request

```
POST
https://vcloud.example.com/api/v0.9/vdc/128/action/uploadVAppTemplate

Content-Type:
application/vnd.vmware.vcloud.uploadVAppTemplateParams+xml


<UploadVAppTemplateParams name="Ubuntu Template"
 transferFormat="application/ovf+xml">
      <Description>My Ubuntu vApp Template</Description>
</UploadVAppTemplateParams>
```

Initiate upload of vApp Template

```
200 OK
Content-Type: application/vnd.vmware.vcloud.vAppTemplate+xml
<VAppTemplate name="Ubuntu Template“
   href=http://vcloud.example.com/api/v0.9/vAppTemplate/268
   status="0" ovfDescriptorUploaded="false”
   type="application/vnd.vmware.vcloud.vAppTemplate+xml" ... >
   <Link type="application/vnd.vmware.vcloud.vdc+xml" rel="up”
    href="http://vcloud.example.com/api/v0.9/vdc/128" />
   <Description>My Ubuntu vApp Template</Description>
   <Files>
      <File name="descriptor.ovf" bytesTransferred="0“>
         <Link rel="upload:default“
          href="http://vcloud.example.com/transfer/.../descriptor.ovf"/>
      </File>
   </Files>
</VAppTemplate>
```

upload ovf

```
PUT /local_pathname_to_ovf_descriptor_file
http://vcloud.example.com/transfer/.../descriptor.ovf>
```

```
200 OK
Content-Type: application/vnd.vmware.vcloud.vAppTemplate+xml
…
<VAppTemplate ovfDescriptorUploaded="true" status="0“  name="Ubuntu Template"
  href=http://vcloud.example.com/api/v0.9/vAppTemplate/268
  type=”application/vnd.vmware.vcloud.vAppTemplate+xml” .>
     <Link type="application/vnd.vmware.vcloud.vdc+xml" rel="up“
       href="http://vcloud.example.com/api/v0.9/vdc/128" />
         <Description>My Ubuntu vApp Template</Description>
         <Files>
               <File size="3940" bytesTransferred="3940" name="descriptor.ovf”
checksum="...“>
                     <Link rel="download:default“
href="http://vcloud.example.com/transfer/.../descriptor.ovf" />
               </File>
         <File size="1950489088" bytesTransferred="0“  name="example-disk0.vmdk”
           checksum=“ fabbad334523432444989bbbffeea5561 ">
                 <Link rel="upload:default“
href="http://vcloud.example.com/transfer/.../example-disk0.vmdk" />
         </File>
          <File . name="example-disk1.vmdk" .>
.           ..
          </File>
       </Files>
</VAppTemplate>
```

Complete List of Operations

vApp Operations

- POST <vapp-uri>/action/{deploy, undeploy}
- POST <vapp-uri>/power/action/{powerOn, powerOff}
- POST <vapp-uri>/power/action/{reset, suspend}
- POST <vapp-uri>/power/action/{shutdown, reboot}
- GET <vapp-uri>/screen
- POST <vapp-uri>/screen/action/acquireTicket

vApp Configuration Operations

- POST <vapp-parent-element-uri>
- DELETE <vapp-element-uri>
- PUT <vapp-element-uri>

Inventory Listing

- GET <vapp-uri>
- GET <vdc-uri>
- GET <vAppTemplate-uri>
- GET <media-uri>
- GET <network-uri>

Catalog Management

- GET <catalog-uri>
- POST <catalog-uri>/catalogItems

Upload/Download/Provisioning Operations

- POST <vdc-uri>/action/composeVApp
- POST <vdc-uri>/action/instantiateVAppTemplate
- POST <vdc-uri>/action/uploadVAppTemplate
- POST <vdc-uri>/media
- PUT <upload-uri>
- GET <download-uri>
- DELETE <resourceEntity-uri>

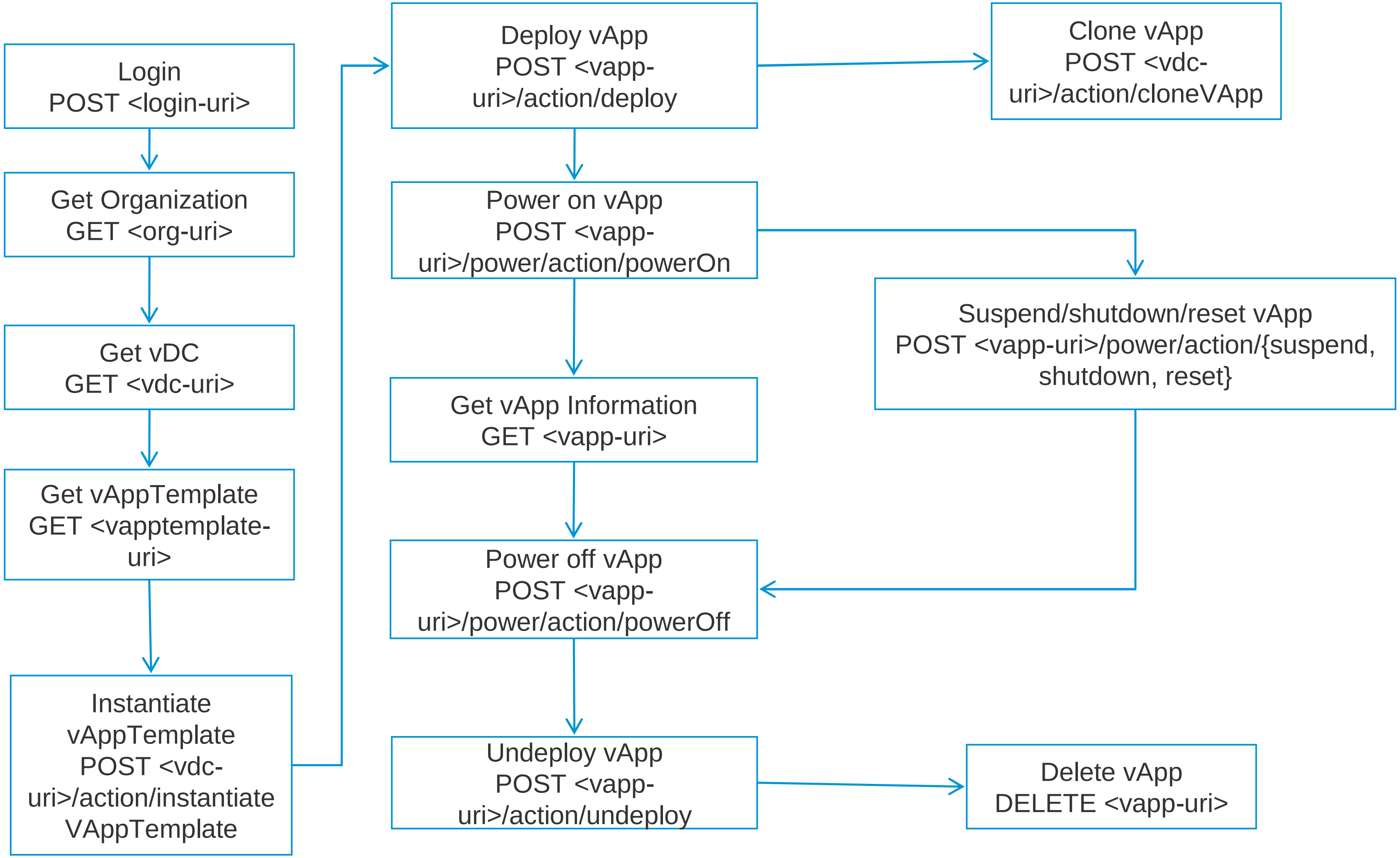
Task Management

- GET <tasks-list-uri>
- GET <task-uri>
- POST <task-uri>/action/cancel

Service Provider Automation

- Create/Delete/Update Org
- Create/Delete/Update vDC for an Org
- Create/Delete/Update an Org Network
- Create/Delete/Update Org Catalogs
- Create/Delete/Update Users, Groups, Roles

API Workflow


Login
POST <login-uri>
Get Organization
GET <org-uri>
Get vDC
GET <vdc-uri>
Get vAppTemplate
GET <vapptemplate-uri>
Instantiate vAppTemplate
POST <vdc-uri>/action/instantiateVAppTemplate
Deploy vApp
POST <vapp-uri>/action/deploy
Clone vApp
POST <vdc-uri>/action/cloneVApp
Power on vApp
POST <vapp-uri>/power/action/powerOn
Suspend/shutdown/reset vApp
POST <vapp-uri>/power/action/{suspend, shutdown, reset}
Get vApp Information
GET <vapp-uri>
Power off vApp
POST <vapp-uri>/power/action/powerOff
Undeploy vApp
POST <vapp-uri>/action/undeploy
Delete vApp
DELETE <vapp-uri>

Administrative API

- Administrative extensions to the vCloud API
- Administrative Operations need administrative credentials of vCloud administrator
- Access to Administrative entities such as User, Group, Role, Provider vDC
- Access to administrative view of entities e.g. AdminOrg, AdminVdc
- Special URL to access the entities http://vcloud.example.com/api/v0.9/admin
- The above URL gives list of Top Level Administrative Entities in a vCloud: OrganizationReferences, ProviderVdcReferences, RightRefrences, RoleReferences, Networks

Organization Administration

Task	Request	Request Body Type	Response Type
Create an Organization	POST vCloud-URL/admin/orgs	AdminOrg	AdminOrg
Get Administrative View of an Organization	GET vCloudURL/admin/org/org-id	None	AdminOrg
Modify an Organization	PUT vCloudURL/admin/org/org-id	AdminOrg	AdminOrg
Remove an Organization	DELETE vCloudURL/admin/org/org-id	none	

vDC Administration

- Provider vDC
  - Created by vCloud Service Provider using tools specific to platform (e.g. vSphere)
  - Entities are read only to the API client
- AdminVdc
  - Created by vCloud Administrator
  - Created to allocate subset of Provider vDC resources and assigned to a vDC in Organization
- Administrative view of vDc can be obtained by using vDC's admin URL which then returns AdminVdc entity

vDC Administration Requests

Task	Request	Request Body Type	Response Type
Examine the contents of a Provider vDC	GET vCloudURL/admin/providervdc/id	None	providerVdc
Allocate a vDC to an Organization	POST vCLoudURL/admin/vdcs/id	AdminVdc	Task
Get an Administrative View of a vDC	GET vCloudURL/admin/id	None	AdminVdc
Modify a vDC	PUT vCloudURL/admin/id	AdminVdc	Task
Remove a vDC	DELETE vCloudURL/admin/id	none	

User, Group and Role Administration

vCloud Administrator is like a ‘root’
Every user exists within the context of an Organization.
vCloud Administrator adds users to an Organization by POSTing User Body
vCloud Administrator can also import users from an LDAP directory service

vCloud Administrator can also import groups from an LDAP directory service

Roles associates names with set of rights.
Role names must be unique in a vCloud instance.
vCloud administrator aggregates a set of rights in a Role Body

User Administration

Task	Request	Request Body Type	Response Type
Create or Import a User	POST vCloudUrl/admin/org/id/users	User	Task
Create an Administrative View of a User	GET vCloudUrl/admin/user/user-id	None	User
Modify User metadata	PUT vCloudUrl/admin/user/user-id	User	Task
Remove User	DELETE vCloudUrl/admin/user/user-id	None	

Group Administration

Task	Request	Request Body Type	Response Type
Import a Group	POST vCloudUrl/admin/org/org-id/groups	Group	Task
View Group Metadata	GET vCloudUrl/admin/group/group-id	None	Group
Modify User metadata	PUT vCloudUrl/admin/group/group-id	Group	Task
Remove User	DELETE vCloudUrl/admin/group/group-id	None	

Roles Administration

Task	Request	Request Body Type	Response Type
Create a Role	POST vCloudUrl/admin/roles	Role	Task
View Roles metadat	GET vCloudUrl/admin/role/role-id	None	Role
Modify Role	PUT vCloudUrl/admin/role/role-id	Role	Task
Remove a Role	DELETE vCloudUrl/admin/role/role-id	None	

Controlling Access

Access control operations allows Administrator to Control access to Catalogs and vApps

Task	Request	Request Body Type	Response Type
Controlling Access to Catalogs	PUT vCloudUrl/org/org-id/catalog/cat-id/controlAccess	ControlAccessParams	
Controlling access to vApps	PUT vCloudUrl/org/org-id/vapp/vapp-id/controlAccess	ControlAccessParams	

List of Administration API

Organization Administration

POST vCloudUrl/api/v0.9/admin/orgs

GET vCloudUrl /api/v0.9/admin/org/<id>

PUT vCloudUrl /api/v0.9/admin/org/<id>

DELETE vCloudUrl /api/v0.9/admin/org/<id>

Role Administration

POST vCloudUrl /api/v0.9/admin/roles

GET vCloudUrl /api/v0.9/admin/role/<id>

PUT vCloudUrl /api/v0.9/admin/role/<id>

DELETE vCloudUrl /api/v0.9/admin/role/<id>

User Administration

POST vCloudUrl /api/v0.9/admin/org/<id>/users

GET vCloudUrl /api/v0.9/admin/user/<id>

PUT vCloudUrl /api/v0.9/admin/user/<id>

DELETE vCloudUrl /api/v0.9/admin/user/<id>

Vdc Administration

POST vCloudUrl/api/v0.9/admin/org/<id>/vdcs

GET vCloudUrl/api/v0.9/admin/vdc/<id>

PUT vCloudUrl/api/v0.9/admin/vdc/<id>

DELETE vCloudUrl/api/v0.9/admin/vdc/<id>

Groups Administration

POST vCloudUrl/api/v0.9/admin/org/<id>/groups

GET vlLoudUrl/api/v0.9/admin/groups/<id>

PUT vCloudUrl/api/v0.9/admin/groups/<id>

DELETE vCloudUrl/api/v0.9/admin/groups/<id>

Agenda

vCloud Ecosystem

vCloud API concepts

vCloud API deep-dive

vCloud API and vCloud Express

vCloud API - Java Library

VMware vCloud™ Express

NEW

A new class of cloud compute services offered by vCloud ecosystem partners

Infrastructure as a Service

- On Demand
- Pay-as-you-go

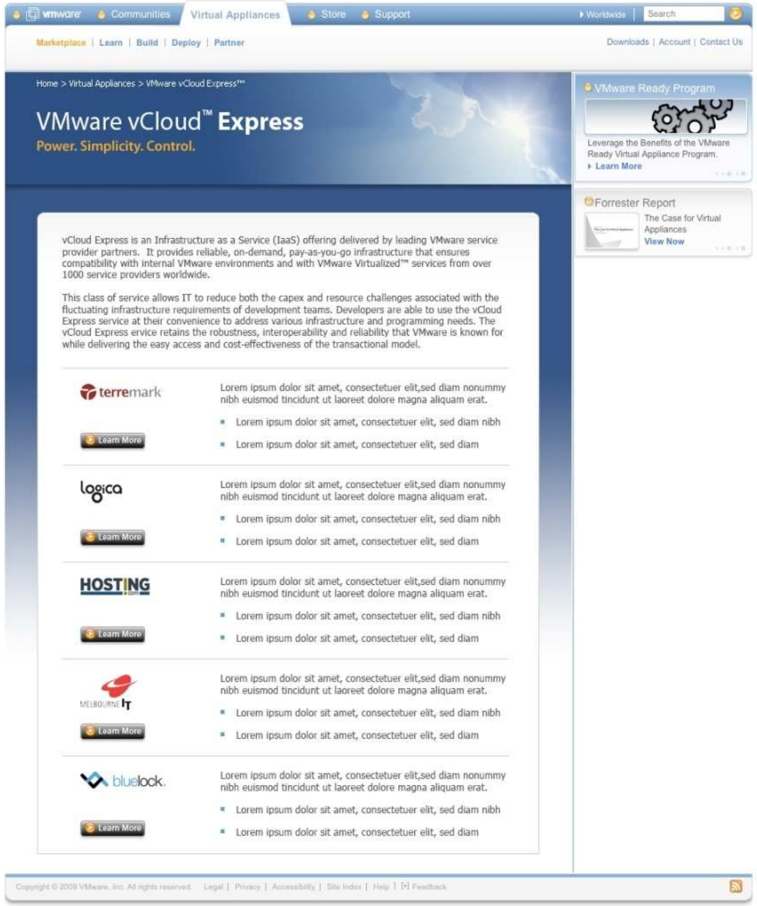
logica
MELBOURNE IT
bluelock
HOSTING.com
terremark

- Self-Service Portal
- vCloud API
- Web-based Signup/Activation
- Utility Pricing
- Credit Card Billing
- Interoperability Across Service Providers

http://www.vmware.com/vcloudexpress

http://www.vmware.com/vcloud-api

VMware vCloud™ Express


vmware
Communities
Virtual Appliances
Store
Support
Worldwide
Search
Marketplace | Learn | Build | Deploy | Partner
Downloads | Account | Contact Us
Home > Virtual Appliances > VMware vCloud Express™
VMware Ready Program
Leverage the Benefits of the VMware Ready Virtual Appliance Program.
Learn More
Forrester Report
The Case for Virtual Appliances
View Now
terremark
logica
HOSTING .com
MELBOURNE IT
bluelock.
Learn More
Copyright © 2009 VMware, Inc. All rights reserved. Legal | Privacy | Accessibility | Site Index | Help | [+] Feedback

VMware vCloud™ Express

Power. Simplicity. Control.

vCloud Express is an Infrastructure as a Service (IaaS) offering delivered by leading VMware service provider partners. It provides reliable, on-demand, pay-as-you-go infrastructure that ensures compatibility with internal VMware environments and with VMware Virtualized™ services from over 1000 service providers worldwide.

This class of service allows IT to reduce both the capex and resource challenges associated with the fluctuating infrastructure requirements of development teams. Developers are able to use the vCloud Express service at their convenience to address various infrastructure and programming needs. The vCloud Express ervice retains the robustness, interoperability and reliability that VMware is known for while delivering the easy access and cost-effectiveness of the transactional model.

terremark

Lorem ipsum dolor sit amet, consectetuer elit,sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat.

- Lorem ipsum dolor sit amet, consectetuer elit, sed diam nibh
- Lorem ipsum dolor sit amet, consectetuer elit, sed diam

logica

Lorem ipsum dolor sit amet, consectetuer elit,sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat.

- Lorem ipsum dolor sit amet, consectetuer elit, sed diam nibh
- Lorem ipsum dolor sit amet, consectetuer elit, sed diam

HOSTING.com

Lorem ipsum dolor sit amet, consectetuer elit,sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat.

- Lorem ipsum dolor sit amet, consectetuer elit, sed diam nibh
- Lorem ipsum dolor sit amet, consectetuer elit, sed diam

MELBOURNE IT

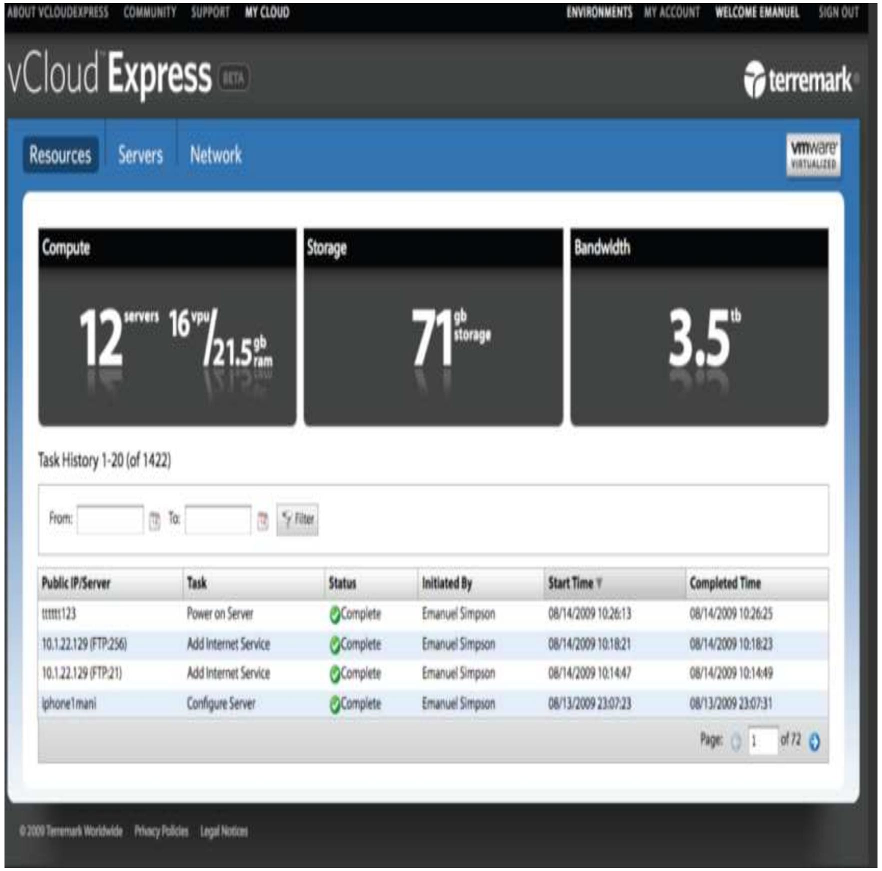
Lorem ipsum dolor sit amet, consectetuer elit,sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat.

- Lorem ipsum dolor sit amet, consectetuer elit, sed diam nibh
- Lorem ipsum dolor sit amet, consectetuer elit, sed diam

bluelock.

Lorem ipsum dolor sit amet, consectetuer elit,sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat.

- Lorem ipsum dolor sit amet, consectetuer elit, sed diam nibh
- Lorem ipsum dolor sit amet, consectetuer elit, sed diam



Compute: 12 servers 16 vpu / 21.5 gb ram

Storage: 71 gb storage

Bandwidth: 3.5 tb

Task History 1-20 (of 1422)

From: To:

Public IP/Server	Task	Status	Initiated By	Start Time	Completed Time
tttttt123	Power on Server	✓Complete	Emanuel Simpson	08/14/2009 10:26:13	08/14/2009 10:26:25
10.1.22.129 (FTP:256)	Add Internet Service	✓Complete	Emanuel Simpson	08/14/2009 10:18:21	08/14/2009 10:18:23
10.1.22.129 (FTP:21)	Add Internet Service	✓Complete	Emanuel Simpson	08/14/2009 10:14:47	08/14/2009 10:14:49
iphone1mani	Configure Server	✓Complete	Emanuel Simpson	08/13/2009 23:07:23	08/13/2009 23:07:31

Page: 1 of 72

Agenda

vCloud Ecosystem

vCloud API concepts

vCloud API deep-dive

vCloud API and vCloud Express

vCloud API - Java Library

Why Java Library?

Get Java developers on board

API is not just interfaces and protocols

Clients expect solid Object Model in line with their use cases and programming language of choice

Make the REST Resources available in Java

Helper and utility classes to address the client use cases

REST API modularity reflected into different packages

Packages that connect to different back end services (e.g. REST API, transfer service, chargeback …)

Java Library Design Principles

High fidelity to REST API entity/resource models

Simple and clean design to help Predictability

No object life cycle management at the client side

Use design patterns

API Commandments

- Less is more.
- REST model types should be accessible to the clients as is; do not hide them
- Use composition to handle the use cases and not inheritance.
- Hide the URL semantics
- Expose the Object Oriented nature of REST resources
- Associate the resources with their operations

Code Sample

```
VcloudClient client =
new VcloudClient("https://example.vcoud.vmware.com");

client.setProxy("proxy.vmware.com", 3128);

// Named references to the Organization that user can access
HashMap<String, ReferenceType> orgs =
client.login(“UserName", “password");

// Now get org for given name
ReferenceType  orgRef = orgs.get(name);
```

Code Sample : Get Catalog Items

```
import com.vmware.vcloud.api.rest.schema.ReferenceType;

import com.vmware.vcloud.sdk.Organization;
import com.vmware.vcloud.sdk.Catalog;

try {
   org = Organization.getOrganizationByReference(client, orgRef);

   for(LinkType ln: org.getCatalogLinks()) {
      Catalog cat = Catalog.getCatalog(client, ln);

      for(ReferenceType ref: cat.getItemReferences()) {
         System.out.println(“Item Name: “  + ref.getName());
      }

   }

} catch (VCloudException e) {
   // Handle the exception
   e.printStackTrace();
}
```

Code Sample : Get and ‘Power On’ vApp

```
try {
   org = Organization.getOrganizationByReference(client, orgRef);
   vdcLink = org.getVdcLinkById(<org Id>);

   Vdc vdc = Vdc.getVdc(client, vdcLink);

   // Named collection of vApp
   HashMap<String, ReferenceType> vapps = vdc.getVappRefsByName();
   ReferenceType vappRef  = vapps.get(“AppServer1”);

   // Here we get the vApp
  Vapp vapp = Vapp.getVappByReference(client, vappRef);

   // Now perform an operation
   Task task = vapp.powerOn();
   …

} catch (VCloudException e) {
   // Handle the exception
   e.printStackTrace();
}
```

Code Sample : Upload vApp template and VMDK Files

```
Vdc vdc = Vdc.getVdcById(client, vdcId);

try {
   VappTemplate vappTempl = vdc.uploadVappTemplate(
      getUploadvAppTemplateParams("SDK-Sample-Test", "Test Template - PP"));

   File f = new File(“ovf File Path”);

   FileInputStream fis = new FileInputStream(f);

   vappTempl.uploadOVFFile(fis);

   System.out.println("Done Upload ..." + vappTemplId);

    // Get vAppTemplate and check  ovfUploaded to be True
   VappTemplate newVappTempl = ….
   // Now Upload the VMDK/Disk Files
   File f1 = new File(vmdk);
   FileInputStream fis1 = new FileInputStream(f1);

   // The file name below should match the one in above file name list.
   newVappTempl.uploadVappFile("dsl-with-tools-disk1.vmdk", fis1, f1.length());

} catch (VCloudException e) {
   e.printStackTrace();
} catch (FileNotFoundException e) {
   e.printStackTrace();
}
```

Call to Action

- Save the date for VMworld Developer Event
  - Dedicated event for software developers
  - vSphere APIs, vCloud APIs, Applications
  - Stay tuned for more details http://blogs.vmware.com/developer

Participate in vCloud API community

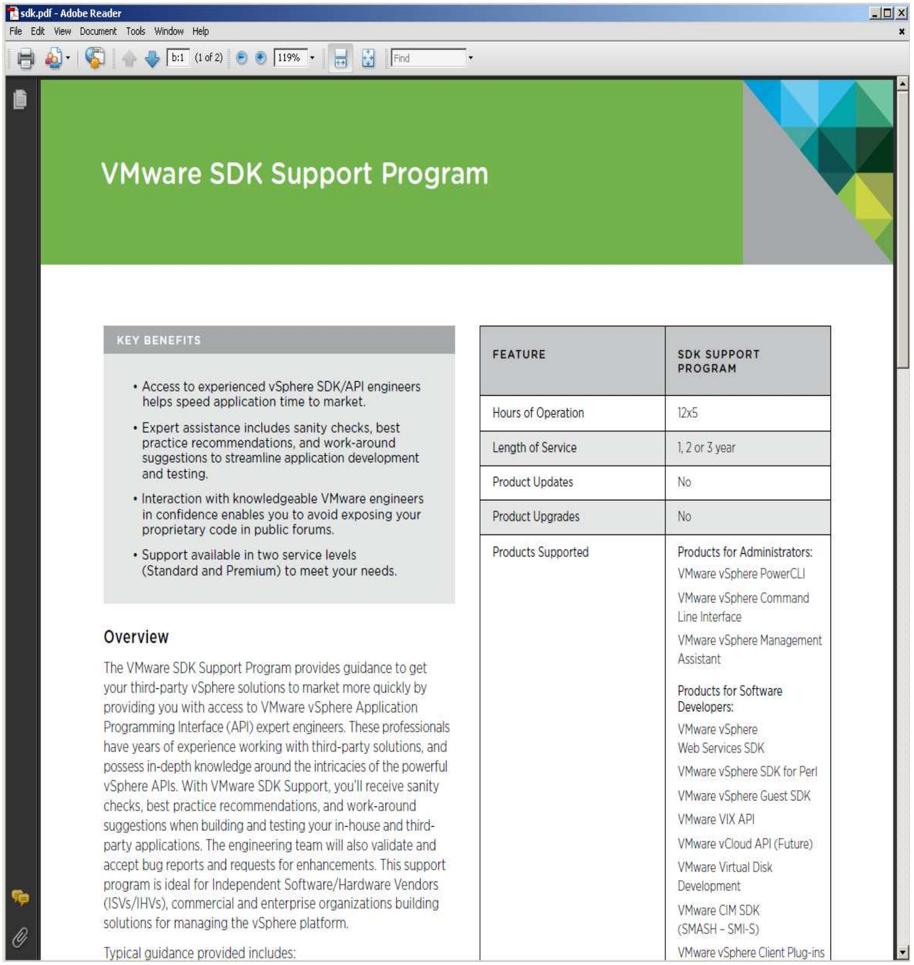
http://vmware.com/go/vcloudapi

Online Resources

http://www.vmware.com/solutions/cloud-computing/

http://www.vmware.com/solutions/cloud-computing/vcloud-api.html

New ! SDK Developer Support for your organization



VMware SDK Support Program

KEY BENEFITS

- Access to experienced vSphere SDK/API engineers helps speed application time to market.
- Expert assistance includes sanity checks, best practice recommendations, and work-around suggestions to streamline application development and testing.
- Interaction with knowledgeable VMware engineers in confidence enables you to avoid exposing your proprietary code in public forums.
- Support available in two service levels (Standard and Premium) to meet your needs.

Overview

The VMware SDK Support Program provides guidance to get your third-party vSphere solutions to market more quickly by providing you with access to VMware vSphere Application Programming Interface (API) expert engineers. These professionals have years of experience working with third-party solutions, and possess in-depth knowledge around the intricacies of the powerful vSphere APIs. With VMware SDK Support, you'll receive sanity checks, best practice recommendations, and work-around suggestions when building and testing your in-house and third-party applications. The engineering team will also validate and accept bug reports and requests for enhancements. This support program is ideal for Independent Software/Hardware Vendors (ISVs/IHVs), commercial and enterprise organizations building solutions for managing the vSphere platform.

Typical guidance provided includes:

FEATURE	SDK SUPPORT PROGRAM
Hours of Operation	12x5
Length of Service	1, 2 or 3 year
Product Updates	No
Product Upgrades	No
Products Supported	Products for Administrators: VMware vSphere PowerCLI VMware vSphere Command Line Interface VMware vSphere Management Assistant Products for Software Developers: VMware vSphere Web Services SDK VMware vSphere SDK for Perl VMware vSphere Guest SDK VMware VIX API VMware vCloud API (Future) VMware Virtual Disk Development VMware CIM SDK (SMASH – SMI-S) VMware vSphere Client Plug-ins

- Dedicated Support
- Developer to Developer
- Flexible 1,2,3 year terms
- Support for vCloud API, SDKs when GA
- Contact your VMW / Partner representative

http://vmware.com/go/sdksupport