Open Cloud Exchange Interface

by Logan Bernard and Shiran Sukumar

Proposal Content

High-Level Goals

I. Provide a user interface to allow multiple, potentially distrusting providers to offer their services

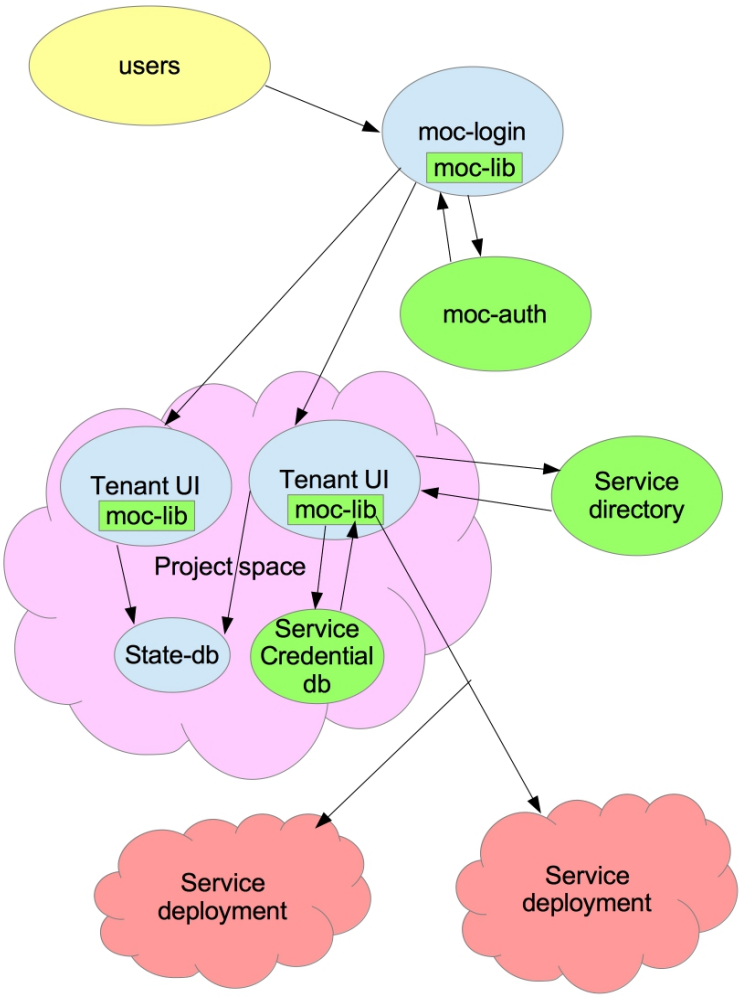
II. Enable a marketplace to show the high (Hadoop Service, PaaS, Appliances) and low (VMs, Storage, Computer, Networking, etc.) level customizable offerings of multiple providers

III. Develop a simple, straightforward user experience for non-expert users

Requirements

I. Security - Must secure user credentials, service and project data
II. Scalability - Must scale to large number of users, projects, and services
III. Extensibility - Must be able to extend interface and allow third-party service interaction
IV. Functionality - Must create, manage, and deploy projects
V. Performance - Must perform efficiently for production use

Architecture



OCxi architecture

Components:

- OCx Library: library used to authenticate user credentials, create project space, accesses state and credential db, accesses service directory
- OCxi : user interface for OCx
- moc-login: interface for users to log into ocx and view project list
- moc-auth: authentication server for user login
- service-directory: maintains all endpoints and information associated with resources (ie availability, image size, SSH, CPU, price, version, name, description, etc.)
- service deployment: low and high level services could be a VM, appliance, or service (ie Hadoop)
- state-db: stores endpoints for services, users associated and permissions
- SC-DB (Service Credential Database): holds credentials for all project services
- tenant-ui: project and marketplace interface
- project space: virtual space for tenants (tenant ui, state db, sc-db)

Architecture Diagram

User credentials are authenticated by OCx-auth, an authorization server. Once authenticated the Project Space is created for the user's Tenant UI. The Project Space will contain databases to keep track of the project state and service credentials. The Tenant UI uses the OCx Library to acquire information on the Project Space's services. The OCx Library calls the Service Directory to gather information on available services, endpoints, and credentials. The OCx Library manages the credentials of the associated services in the SC-DB and stores service endpoint information in the State DB. Using the stored endpoints, the Tenant UI uses its API to connect with services. It must authenticate with the central authority to interact with its services. After this process, the Tenant UI provides a marketplace of available services and a project management interface or an extensible, third party interface (e.g. Hadoop) for interaction with services.

Design Implications

- **User login authorized by OCx authority, not required on a Project Space level**
    - Pro
        - One login to access all projects and associated service credentials
        - Each Project Space specific to users; reduces complexity, benefits namespacing
    - Con
        - OCx Login Interface must scale to manage all user credentials

- **Central Keystone Authority**
    - Pro
        - Manages group project credentials; benefits security, streamlines authority by authenticating on a per project basis
        - One place for all Service Providers provides simplicity
        - Source of truth for all Service Providers; benefits security
    - Con
        - If compromised, creates large scale vulnerability
        - Must be scalable for all service providers

- **Tenant UI accesses DBs and services using the OCx Library**
    - Pro
        - Layer of abstraction prevents users from doing inadvertent damage
        - Modularizes components (State DB, etc), which benefits security and extensibility
        - Performance benefits from scaling on a per project basis (e.g. less users, less shared resources and reduced library and DB queries)
    - Con
        - Adds extra layer of developers complexity

- **Project Space spawns a Tenant UI for each user**
    - Pro
        - Marketplace Interface does not have to scale for all users, reduces

load to interface
        - Each Tenant has a copy of the library, improves request speeds
    - Con
        - Service Directory must respond to each Tenant UI; scaling requirement
        - Individual VM per user; large number of resources; time spent spawning VM

- **Tenant UI calls Service Directory**
    - Pro
        - Service state info stored in one central location; benefits security and scaling
        - Service Providers need to only respond to Service Directory; benefits scaling and simplicity
    - Con
        - Service Directory must respond to each Tenant UI; scaling requirement

- **Project Space uses SC-DB**
    - Pro
        - Service credentials abstracted from the user; benefits security, simplicity
        - e.g. rogue user can't have direct control over underlying services
        - establishes a permissions hierarchy (e.g. admin controls the Project Space services and users)
    - Con
        - Only project users can access and manipulate services; limits project-project interaction

- **Project Space uses State-DB**
    - Pro
        - project's service endpoint info and permissions hierarchy maintained without a persisting Project Space instance
        - OCx workload reduced when projects are shut down
        - One state-DB per project provides security to each project's database

and simplifies update for each user change

- Con
  - Larger storage space for each project

- **Extensible UI**
  - Pro
    - Allows third party to build extensible interfaces
  - Con
    - OCxLibrary must be capable of accounting for potential third party service
    - Requires uniform UI styling

Related Work

Through determining how OCxi will compete in the already existing market, research was required to evaluate the design. The evaluation criteria examined the aesthetic, experience, and the practicality.

AWS

Examining the Amazon Web Service, it is a basic interface that did not seem very user friendly. The AWS is the most widely used interface on the market and does provide a baseline for what one can expect to accomplish. The design was very simple and did not offer any components that could be incorporated into the design. Although AWS is not intuitive, it can be used to build example projects quickly.

JUJU

The most advanced, customizable, and intuitive user experience that provides the OCxi with a great model for presenting a unique user interface for setting up services. JuJu uses a GUI that allows users to drag and drop nodes, select their configuration and deploy. The ease of setting up OpenStack in their demo made a great case for adopting a similar interface. The model of being able to drag-and-drop or click and change components of a cloud infrastructure is very practical and user friendly. Ultimately, to provide the most customizable experience and be able to set up any configuration on the OCx, OCxi needs to be open to possibilities. Rather than limit users to ready-to-deploy packages, we aim to present choices and rather than preventing deployments if there are configuration conflicts. JuJu is a model that OCxi incorporates into its design, because in the future more people will likely want a visualization tool to see their deployments.

OCx-UI

The OCx-UI represents a standard for what the OCx needs. While examining OCx-UI the aim was to find what made the OCx-UI desirable and useful for the OCx. The ease of explanation and intuitive experience stood out as major factors. While JuJu is extremely powerful, it may be hard for some users to quickly use. OCx-UI has a universal design that almost anyone should be able to find exactly what they are looking for and purchase. Most of the design requirements were derived from understanding the OCx-UI.

Horizon

OpenStack's UI, Horizon, is a barebones interface to deploy projects. The UI is simple, and clean, but is very bland. Too bland for the OCx's needs. However, the usefulness with Horizon is the wide familiarity with the system. While OCxi will not look like Horizon, the pop up options will try to remain consistent. For example, there is a horizontal bar that represents a VM and the edit buttons should contain the same information that Horizon has in its implementation. The goal with evolving Horizon is to provide a richer experience without over complicating the pre existing user interface.

User Stories

1. New user creates a project, purchases and deploys a VM
2. User Logout
3. Returning user destroys an active VM and project.
4. Returning user alters an active VM and project.
5. Returning user moves an active VM to another project.
6. Returning user builds custom VM
7. User Adds Service (extensibility)
8. User Renames Network
9. User Renames VM
10. Admin adds User
11. User Views Billing (note not in scope)

1. New user creates a project, purchases and deploys a VM

- OCX-Login
    - On the OCX-Login page, the user selects 'Register' and is sent to a registration page.
- New User Registration
    - After submitting basic information (e.g. email, password, name), the user is registered with the OCX-Authority and the user is sent to the Projects Page.
- Projects Page
    - The user's available projects (names and descriptions) are listed for selection. Only one project may be selected and entered at a time, sending the user into the specified Project Space.
    - For a new user, only the 'Create New Project' option is available. After submitting a project name and description, the project is deployed and the user is sent to the new Project Space as the project's admin.

Project Space

- OCX-I (marketplace)
    - For a new project, the user lands in the marketplace.
    - The user selects 'Prebuilt' VM
    - VM Details (Overlay)
        - The user purchases the 'Prebuilt' VM from the Recommended section (first row of items); name of the VM is specified
        - (small/med/large) OS Image, CPU's, Storage Size, SSH info (requires private key), Network (hidden management network, private network)
        - *(stretch goal) application deployment options; can choose things such as apache, wireshark, etc to be loaded with the image*
- Purchase Page
    - Payment Details (Overlay)
        - user is prompted to add a credit card/payment type, billing information
        - clicks save information and enters an associated name
    - select payment type (the one just registered)
- Confirmation Page
    - overall charge of the purchase; finalize purchase

    - sent to Project Management on confirmation
- Project Management
    - User selects their purchased ‘Prebuilt-Small’ VM
    - VM Details (Overlay)
        - the user is able to see details of the VM once again, and chooses to power it on.
        - Use SSH: User clicks details and finds the IP and SSH Public Key
        - options to attach public network, add private network
        - *(stretch goal) Open Console: VNC*
- At this point, a new user, project, and VM have been created. The user can access their current VM using SSH, or continue purchasing and deploying items from the marketplace via the project management page.

2. User Logs Out

- OCX-Login
    - The user enters their credentials and selects ‘Login’
- Projects Page
    - The user views their list of project names and descriptions, and enters (is sent to the Project Space of) ‘Project 1’.

Project Space

- User does some actions within the Project Space
- User clicks logout
- User taken to the login page

3. Returning user destroys an active VM and project

- OCX-Login
    - The user enters their credentials and selects ‘Login’
- Projects Page
    - The user views their list of project names and descriptions, and enters (is sent to the Project Space of) ‘Project 1’.

Project Space

- Project Management
    - The returning user lands on the project management page upon entering their project’s space.
    - the user selects the ‘Project Admin’ button
- Project Admin (user / project description management)
    - the user selects ‘Destroy Project’
    - the user enters their credentials as a confirmation of their intention and authorization to destroy the project (must also be clean)
    - error message indicates the user must delete all VMs first
    - user returns to the project management page
- Project Management
    - They see the details of their project. There is one VM, which is active; this VM is selected
    - VM Details (Overlay)
        - the user shuts down the VM.
        - the user selects ‘destroy VM’, and does a second confirmation of their action
    - user selects the ‘Projects Admin’ button
- Project Admin (user / project description management)
    - the user selects ‘Destroy Project’
    - the user enters their credentials as a confirmation of their intention and authorization to destroy the project
    - deletion confirms
    - takes user to projects page

4. Returning user alters an active VM and project

- OCX-Login
    - The user enters their credentials and selects ‘Login’
- Projects Page
    - The user views their list of project names and descriptions, and enters (is sent to the Project Space of) ‘Project 1’.

Project Space

- Project Management
    - The returning user lands on the project management page upon entering their project’s space.
    - They see the details of their project. There are several active VMs, the user wants to connect two VMs over the same network
    - User selects VM-1 to view details
    - VM Details (Overlay)
        - clicks Network tab and chooses to create a new network
        - User chooses to add a private network and adds a name “net1”
        - user associates “net1” with VM
    - user selects another VM-2
    - VM Details (Overlay)
        - views the network tab; user associates “net1” with VM-2
        - user also clicks network dropdown (lists available networks) and associates the public network

5. Returning user moves an active VM to another project

- OCX-Login
    - The user enters their credentials and selects ‘Login’
- Projects Page
    - The user views their list of project names and descriptions, and enters (is sent to the Project Space of) ‘Project 1’.

Project Space

- Project Management
    - The returning user lands on the project management page upon entering their project’s space.
    - User selects VM-1 to view details
    - VM Details (Overlay)
        - clicks ‘Transfer’
        - selects from a list of projects which they have admin access to
        - receive confirmation to stay on current page or go to project management page with the moved VM

6. Returning user builds custom VM

- OCX-Login
    - The user enters their credentials and selects ‘Login’
- Projects Page
    - The user views their list of project names and descriptions, and enters (is sent to the Project Space of) ‘Project 1’.

Project Space

- Project Management
    - The returning user lands on the project management page upon entering their project’s space.
    - The user decides to make a custom VM
    - User selects ‘Marketplace’ tab
    - The user begins choosing components by selecting the ‘Compute’, ’Network’, and ‘Storage’ categories
        - Compute
            - Number of CPUs
            - Speed (slow, medium, fast)
            - Location
        - Network
            - selects choice of hardware
            - defines private/public/gateway/router
        - Storage
            - Size (small, medium, large)
            - Location
            - Speed (slow, medium, fast)
        - Software Packages
            - Images (Ubuntu, Windows, etc.)
            - Software Bundles (Wireshark, SSH, etc.)
    - When the user finishes choosing all the components the user can press the ‘Purchase’ button
- Purchase Page
    - select payment type or register card

- Confirmation Page
    - view charge; finalize purchase
    - sent to Project Management on confirmation

7. User Adds Service (extensibility)

- OCX-Login
    - The user enters their credentials and selects ‘Login’
- Projects Page
    - The user views their list of project names and descriptions, and enters (is sent to the Project Space of) ‘Project 1’.

Project Space

- Project Management
    - The returning user lands on the project management page upon entering their project’s space.
    - The user decides to add a Service/Appliance
    - User selects ‘Marketplace’ tab
    - The user selects a Service or Appliance (HPC, Hadoop, Protein Modeling, etc.)
        - User chooses from a list of possible configuration options (Size, Speed, Network, Image, CPU)
    - User selects “Purchase’ button
- Purchase Page
    - select payment type or register card
- Confirmation Page
    - view charge; finalize purchase
    - sent to Project Management on confirmation
- Project Management
    - User selects Service
    - Service Details (Overlay)
        - selects manage which opens Service Interface (e.g. Hadoop’s native interface)
- Service Interface
    - User interacts with Service Interface (3rd party)
    - Any changes are reflected upon their service, and can be seen via the Project Management page

8. Returning user renames a Network

- OCX-Login
    - The user enters their credentials and selects ‘Login’
- Projects Page
    - The user views their list of project names and descriptions, and enters (is sent to the Project Space of) ‘Project 1’.

Project Space

- Project Management
    - The returning user lands on the project management page upon entering their project’s space.
    - User selects VM-1 to view details
    - VM Details (Overlay)
        - clicks Network tab
        - clicks edit button next to ‘net1’
        - renames ‘net1’ to ‘new-net1’
            - warning thrown about other VMs attached to ‘net1’
            - these VMs will now be connected to ‘new-net1’

9. Returning user renames a VM

- OCX-Login
    - The user enters their credentials and selects ‘Login’
- Projects Page
    - The user views their list of project names and descriptions, and enters (is sent to the Project Space of) ‘Project 1’.

Project Space

- Project Management
    - The returning user lands on the project management page upon entering their project’s space.
    - User selects VM-1 to view details
    - VM Details (Overlay)
        - clicks name
        - renames VM-1 to ‘VM-new’

10. Admin adds a user

- OCX-Login
    - The user enters their credentials and selects ‘Login’
- Projects Page
    - The user views their list of project names and descriptions, and enters (is sent to the Project Space of) ‘Project 1’.

Project Space

- Project Management
    - The returning user lands on the project management page upon entering their project
    - User clicks ‘Project Admin’ button
    - Views ‘People’
    - Clicks ‘add user’
    - Fills out username and permissions (user must already be registered with OCx)
    - Saves new user information

11. User Views Billing (note not in scope)

- OCX-Login
    - The user enters their credentials and selects ‘Login’
- Projects Page
    - The user views their list of project names and descriptions, and enters (is sent to the Project Space of) ‘Project 1’.

Project Space

- Project Management
    - The returning user lands on the project management page upon entering their project
    - User decides to view billing and clicks ‘Billings’ tab
- Billing
    - User views an overview of billing
    - User selects ‘Project 1’
        - ‘Project 1’ expands to show details of charges
        - User clicks ‘VM-1’ to view details of charges
    - User decides to pay bill
    - User click ‘Pay’
    - Users selects payment method
    - User goes to confirmation page
- Confirmation Page
    - User views a simplified overview
    - User Presses ‘Charge’
    - User receives confirmation and returned to ‘BIlling’ page

Tasklist

The OCx Library, Service Directory, and SC-DB (service credentials database) are out of scope, although much of the Project Space's functionality is dependent upon these components. We will emulate interaction with these components. For our project's purposes, the Login Page and Project Space will be hosted on the same web server.

Login Page

- create django page (1 day)
  - make fields for : username, password
  - make button for : login, register, forgot password
- create moc-auth demo database (3 days)
  - create apache server
  - create demo table of credentials
  - create example request function call
  - create example return value

Projects Page

- create django page (1 day)
  - list user's associated projects (name+description of each)
  - select project and go to management page (in project space)
- create demo projects-db (1 day)
  - create project's table to store project name and description and associated users with permissions

Project Space

- create demo state-db to hold project's service endpoints and project info
  - create table for endpoints (2 days)
- Pseudo-API to emulate interaction with OCx Library (4 days)

example request for service to credentials db
example return for service from credentials db
function call to Service Directory (out of scope) for information about services

Project Management

create django page
request/function call for VMs and Services of project (2 days)
display VMs and Services (1 week)
get information from service directory
connect to services via library
display service interface
display list of VMs and Services
display VM/Service details (cpu, image, network, ssh, etc)
add buttons to edit VMs
add button to call service interface (third party) *(stretch goal)*
news feed of user changes *(stretch goal)*

Billing *(stretch goal)*

create django page (1 week)
get billing information via library
display overview
display individual component charges
add pay button
add confirmation button
add charge button

MarketPlace

create django page (1 week)
- get available services via library
- display categories sidebar
- display services by category
- add appliances
- display configuration pane
- add purchase
- search option *(stretch goal)*
- filter remaining choices *(stretch goal)*
- drag and drop overview diagram *(stretch goal)*

Milestones

Mid-to-End October

Goals: Final Proposal; Final Design; UI Implementation; Demo

Final Proposal and Visualization

Oct 16 - Oct 23

- We will create a final proposal and visualization of our design.
- The finalized proposal
- The finalized design with focus on a market-first approach

Basic Django Implementation of Design

Oct 24 - Oct 31

- Begin the implementation using Django.
- Build out wireframe design
- Third party user will be able to maneuver through the interface

Mid November

Goals: UI Implementation; OCx Library Integration with OCxi; Demo

OCx Library Integration

Nov 3 - Nov 14

- Focus on the underpinnings that will be necessary to make the OCxi function.
- Lay out the API calls necessary for: user authentication, retrieval of available resources, and service manipulation (attachment to User Cloud, billing, etc).
- Integrate a model of the OCx Library (which will be developed by the OCx)

End of November

Goals: Enhanced and Full User Experience

Visual Enhancement and Finalized User Stories

Nov 17 - Nov 25

- Click through the OCxi and understand the intended functionality
- Expose the OCx Library interaction
- Visual elements will be refined

December

Goals: Final Project Report and Presentation

Project Fixes: Odds and Ends

Dec 1 - Dec 9

- Buffer time to pick up any slack

Final Presentation

Dec 9

- Able to give a full walk-through of the OCxi