**Software Design Specification**

**SLYP Project**



**March 20, 2016**

**Team 1**

**Kieran Dunbar**

**Michael Carbone**

**Kevin Reynolds**

**Meagan Johnson**

**Kevin Caulfield**

**Table of Contents**

1. Introduction
   1. Purpose of this Document
   2. Scope of the Development Project
   3. Definitions, Acronyms, and Abbreviations
      1. Definitions
      2. Acronyms and Abbreviations
   4. References
   5. Major Software Requirements
   6. Design Constraints, Limitations
      1. Constraints
      2. Limitations
   7. Changes to Requirements
   8. Overview of Document
2. Data Design
   1. Data Objects and Resultant Data Structures
   2. DS and Global Variables
   3. File and Database Structures
3. System Architecture Description
   1. Overview of Modules / Components
   2. Structure and relationships
4. Detailed Description of Components
   1. Component Template Description

4.x. Description of Component x

1. Interface Design

5.x. Description of Component x

1. Appendix
2. **Introduction** 
   1. *Purpose of this Document*

The purpose of this document is to provide a detailed description of the problem Slyp solves, the components of the software and the implementation of said software. This document is meant to be used by developers/coders working with Slyp as well as testers verifying its functionality.

* 1. *Scope of the Development Project*   
       
     The product under development is called Slyp. Slyp is designed to enable the migration of virtual machines between vSphere, Hyper-V and AWS-EC2. Slyps primary functions are: exporting virtual machines, converting virtual disks between .vmdk and .vhd formats, and deploying virtual machines. Slyps most distinct feature is the ability to execute these functions in sequence, migrating a vm from one virtualization platform to another. This will allow the user to relocate vms in the event of a hypervisor specific vulnerability or replicating a vm both on a public cloud and a private datacenter. Slyp is limited to working with vSphere, Hyper-V and AWS-EC2.
  2. *Definitions, Acronyms, and Abbreviations*
     1. Definitions

|  |  |
| --- | --- |
| **Term** | **Definitions** |
| Amazon Web Services | The cloud computing platform developed by Amazon |
| Bucket | An instance of the AWS storage service S3. |
| Microsoft Hyper-V | An instance of the type 1 hypervisor developed by Microsoft. When used in the document, Hyper-V refers to a deployment of one or more Hyper-V hosts. Slyp will interface directly with Hyper-V hosts. |
| Active Directory | A directory service developed by Microsoft for Windows domain networks. |
| Powershell | Shell scripting language developed by Microsoft for ease of local and remote management. |
| VMware vSphere | An instance of the virtualization software suite developed by VMware. In this document, vSphere refers to a deployment of VMware ESXi hosts being managed by a vCenter server. Slyp will interface directly with the vCenter server. |
| VMware ESXi | The type 1 hypervisor developed by VMware used to virtualize operating systems |
| VMware vCenter | A modified version of Suse Linux used to manage multiple ESXi hosts and enable advanced functionality of the vSphere suite. |
| Virtual Machine | Refers to a virtualized operating system. Includes the configuration files and virtual hard disks. |

* + 1. Acronyms and Abbreviations

|  |  |
| --- | --- |
| **Acronym** | **Definition** |
| 2012 R2 | Windows Server 2012 R2 |
| AWS | Amazon Web Services |
| AD | Active Directory |
| CLI | Command-line interface |
| ESXi | Elastic Sky X |
| EC2 | Elastic Compute Cloud |
| IP | Internet Protocol |
| HV | Hyper-V |
| OVF | Open Virtualization Format |
| OVA | Open Virtualization Archive |
| PS | Powershell |
| S3 | [Amazon Simple Storage Service](https://aws.amazon.com/s3/) |
| SDK | Software Development Kit |
| VHD | Virtual Hard Disk. 2TB capacity |
| VHDX | Virtual Hard Disk. 64TB capacity + error correcting |
| VM | Virtual Machine |
| VMS | Virtual Machines |
| VMDK | Virtual Machine Disk |
| VS | vSphere |
| XML | Extensible Markup Language |

* 1. *References*

1. Amazon Web Services, Inc. "Amazon Web Services (AWS) - Cloud Computing Services." Amazon Web Services, Inc. Amazon Web Services, Inc., n.d. Web. 23 Mar. 2016.
2. AWS. (n.d.). Python. Retrieved March 24, 2016, from https://aws.amazon.com/developers/getting-started/python/
3. Boto. (n.d.). Boto/boto. Retrieved March 24, 2016, from https://github.com/boto/boto
4. Boto3 Reference. (n.d.). Retrieved February 12, 2016, from https://boto3.readthedocs.org/en/latest/reference/core/boto3.html
5. Daems, S. (2014, August 28). VMware: Nesting Hyper-V 2012 R2 on vSphere ESXi 5.5. Retrieved March 20, 2016, from http://blog.vmpros.nl/2014/08/28/vmware-nesting-hyper-v-2012-r2-on-vsphere-esxi-5-5/
6. Frommel, O. (n.d.). Controlling Amazon Cloud with Boto » ADMIN Magazine. Retrieved March 24, 2016, from http://www.admin-magazine.com/Articles/Integrating-AWS-Cloud-Services-with-Your-Custom-Apps
7. Garnaat, M. (n.d.). Boto: A Python interface to Amazon Web Services. Retrieved March 24, 2016, from http://boto.cloudhackers.com/en/latest/
8. Garnaat, M. (n.d.). Boto 2.39.0 : Python Package Index. Retrieved March 24, 2016, from https://pypi.python.org/pypi/boto
9. Hyper-V Cmdlets. (n.d.). Retrieved February 12, 2016, from https://technet.microsoft.com/en-us/library/hh848559.aspx
10. Jha, V. (n.d.). Getting Started with Boto (python Interface for AWS ) TO THE NEW Blog | TO THE NEW Blog. Retrieved March 24, 2016, from http://www.tothenew.com/blog/getting-started-with-boto-python-interface-for-aws/
11. Manage Hyper-V Server Remotely. (2012, September 4). Retrieved March 20, 2016, from https://technet.microsoft.com/en-us/library/jj647785.aspx
12. Omta, T. (2015, June 11). How to Convert a VMWare VMDK to Hyper-V VHD. Retrieved March 20, 2016, from https://blogs.msdn.microsoft.com/timomta/2015/06/11/how-to-convert-a-vmware-vmdk-to-hyper-v-vhd/
13. Osborne, R. (2013, March 10). Windows Server 2012 Hyper-V Best Practices (In Easy Checklist Form). Retrieved March 20, 2016, from <https://blogs.technet.microsoft.com/askpfeplat/2013/03/10/windows-server-2012-hyper-v-best-practices-in-easy-checklist-form/>
14. Poojary, N. (2015). Boto: Using Python to automate AWS services. Retrieved March 24, 2016, from http://cloudacademy.com/blog/boto-python-automate-aws-services/
15. Siron, E. (2013, October 16). Storage and Hyper-V Part 4: Formatting and File Systems. Retrieved March 20, 2016, from http://www.altaro.com/hyper-v/storage-and-hyper-v-part-4-file-systems/
16. System Requirements and Installation Information for Windows Server 2012 R2. (2015, October 20). Retrieved March 20, 2016, from <https://technet.microsoft.com/en-us/library/dn303418.aspx>
17. Vanover, R. (2011, September 14). Nesting Hyper-V with VMware Workstation 8 and ESXi 5. Retrieved March 20, 2016, from https://www.veeam.com/blog/nesting-hyper-v-with-vmware-workstation-8-and-esxi-5.html
18. VSphere 6.0 Documentation Center. (n.d.). Retrieved February 12, 2016, from http://pubs.vmware.com/vsphere-60/index.jsp?topic=/com.vmware.wssdk.apiref.doc/right-pane.html
19. VMware, Inc. "Cloud Computing." With VMware Cloud Technology. VMware, Inc., n.d. Web. 23 Mar. 2016.
20. Vmware/pyvmomi. (n.d.). Retrieved February 12, 2016, from https://github.com/vmware/pyvmomi
21. Dunbar, K., Johnson, M., Reynolds, K., Caulfield, K., & Carbone, M. (n.d.). Software Requirement Specification SLYP Project.  
      
    1. *Major Software Requirements*
       1. Python 2.7
          1. Boto3 SDK
          2. PyVmomi SDK
          3. Python Libraries

* Ssl
* Sys
* Os
* Time
* Datetime
* Tarfile
* Re
* ConfigParser
* Threading
* Subprocess
  + 1. Powershell 4.0
       1. Hyper-V CMDLETs
  1. *Design Constraints, Limitations*
     1. Constraints
        1. Slyp will only run on Windows Server 2012 R2
        2. One instance of Slyp is meant to be run at a time
        3. One VM operation is meant to be done at a time. Performance cannot be guaranteed beyond one concurrent VM operation.
     2. Limitations
        1. Slyp is limited to working with vSphere, Hyper-V and AWS-EC2.
  2. *Changes to Requirements*   
       
     N/A
  3. *Overview of Document*The remaining sections of this document outline the data design, system architecture descriptions, detailed descriptions of those components, and the interface design, followed by the appendix. Section 2 discusses the data design, showing the data objects and structures, as well as the file and database structures. The system architecture descriptions are given in section 3, explaining how the system is organized. Section 4 contains a detailed description of all the components. Section 5 shows the possible clui options.

1. **Data Design** 
   1. *Data Objects and Resultant Data Structures*

See appendix 6.1

* 1. *DS and Global Variables*

N/A

* 1. *File and Database Structures*

N/A

1. **System Architecture Description**  
   1. *Overview of Modules / Components*
2. Hyper-V Slyp - uses Python’s Subprocess call to run dynamic Hyper-V CmdLets to Power On, Power Off, Import VM, Export VM
3. vCenter Slyp - uses PyVmomi and Multiple Python libraries to connect to VCenter and handle Importing VMs, Exporting VMs, Power On VMs, and Power Off VMs dynamically.
4. AWS Slyp - uses AWS and Python’s SubProcess to connect to AWS EC2 and handle Importing VMs, Exporting VMs, Power On VMs, and Power Off VMs dynamically.
5. Configuration - Interactive CLI tool to gather target hypervisor hostnames, IP addresses, usernames and passwords.

**Data Flow Diagram**

* 1. *Structure and Relationships*   
       
     See Appendix 6.1

1. **Detailed description of components**
   1. *Component Template Description*   
        
      **SLYP SDS component template**   
        
      **Identification:  
        
      Type:**   
        
      **Purpose:  
        
      Function:  
        
      Subordinates/ Modules used:**   
        
      **Dependencies:**   
        
      **Resources:**   
        
      **Processing:**   
        
      **Data:**
   2. **Identification:**VMware Slyp  
        
      **Type:**   
      Module  
        
      **Purpose:**   
      To facilitate the interaction between the Slyp server and a VMware host.  
        
      **Function:**VMware Slyp takes input from the configuration file at C:/Slyp/config.ini and allows Slyp to issue commands to a target VMware host. The operations that the commands allow include: getting the status of VMs, turning VMs on, turning VMs off, exporting a VM, and importing a VM.   
        
      **Subordinates/ Modules used:**   
      Export VM - utilizes the Export-VM commandlet followed by the appropriate parameters as described in the configuration file initiating the export of the VM.

Import VM - utilizes the Import-VM commandlet followed by the appropriate parameters as described in the configuration file initiating the import of the VM.

Get VM Information - utilizes the Get-VM commandlet followed by the target Hyper-V system defined in the configuration file returning the running status and uptime of all VMs on the target system.

Power on VM - utilizes the Start-VM commandlet followed by the name of the VM to be powered on. If the given VM exists it will power up.

Power off VM - utilizes the Stop-VM commandlet followed by the name of the VM to be powered off. If the given VM exists it will power off.  
  
**Dependencies:**   
Powershell 4.0 provides the commandlets that drive the module, allowing for secure reliable communication between windows machines.

Python 2.7 provides the logic.

**Resources:**   
N/A  
  
**Processing:**   
Get status of VM - The python script parses the configuration file into useable variables and uses a subprocess call. This command then returns the status of the specified VM on the VMware host and also writes the output to a text file in the C:/Slyp/ directory.

Power on VM - The python script parses the configuration file into useable variables and uses a subprocess call consisting of a VM name. This command turns the given VM on.

Power off VM - The python script parses the configuration file into useable variables and uses a subprocess call consisting of a VM name. This command turns the given VM off.

Export VM - The python script parses the configuration file into useable variables and uses a subprocess call containing a VM name, export format, and target AWS S3 bucket location. This command then initiates the export of the VM to be exported to the export destination.

Import VM - The python script parses the configuration file into useable variables and uses a subprocess call containing a VM name, import format, and source AWS S3 bucket location. This command then initiates the import of the VM to be imported to the import destination.  
  
**Data:**   
All of the data used by VMwareSlyp is located in the configuration file at C:/Slyp/config.ini and is parsed into relevant variables at run time. The variables include hostname, VM name, import path, and export path.

* 1. **Identification:**

Hyper-V Slyp  
  
**Type:**

Module  
  
**Purpose:**   
To facilitate the interaction between the Slyp server and a Hyper-V host.

**Function:**

Hyper-V Slyp takes input from the configuration file at C:/Slyp/config.ini and allows Slyp to issue commands to a target Hyper-V host. The operations that the commands allow include: getting the status of VMs, turning VMs on, turning VMs off, exporting a VM, and importing a VM.   
  
**Subordinates/ Modules used:**   
Export VM - utilizes the Export-VM commandlet followed by the appropriate parameters as described in the configuration file initiating the export of the VM.

Import VM - utilizes the Import-VM commandlet followed by the appropriate parameters as described in the configuration file initiating the import of the VM.

Get VM Information - utilizes the Get-VM commandlet followed by the target Hyper-V system defined in the configuration file returning the running status and uptime of all VMs on the target system.

Power on VM - utilizes the Start-VM commandlet followed by the name of the VM to be powered on. If the given VM exists it will power up.

Power off VM - utilizes the Stop-VM commandlet followed by the name of the VM to be powered off. If the given VM exists it will power off.  
  
**Dependencies:**   
Windows Server 2012R2 is used by this module as an OS for the application runtime.

Active Directory is used for authentication while communicating with Hyper-V and DNS hostname resolution.

Powershell 4.0 provides the commandlets that drive the module, allowing for secure reliable communication between windows machines.

Python 2.7 provides the logic and executes powershell commandlets.

**Resources:**   
N/A

**Processing:**   
Get status of VM - The python script parses the configuration file into useable variables and uses a subprocess call to powershell consisting of all the variables strung together into a valid powershell command. This command then returns the status of all running VMs on the Hyper-V host and also writes the output to a text file in the C:/Slyp/ directory.

Power on VM - The python script parses the configuration file into useable variables and uses a subprocess call to powershell consisting of the commandlet and a VM name strung together into a valid powershell command. This command turns the given VM on.

Power off VM - The python script parses the configuration file into useable variables and uses a subprocess call to powershell consisting of the commandlet and a VM name strung together into a valid powershell command. This command then turns the given VM off.

Export VM - The python script parses the configuration file into useable variables and uses a subprocess call to powershell consisting of the commandlet, target Hyper-V system, VM to be exported, and the export destination path strung together into a valid powershell command. This command then initiates the export of the VM to be exported to the export destination.

Import VM - The python script parses the configuration file into useable variables and uses a subprocess call to powershell consisting of the commandlet, target Hyper-V system, VM to be imported, and the path where the VM to be imported is located strung together into a valid powershell command. This command then initiates the import of the VM from its existing location.

**Data:**

All of the data used by Hyper-V Slyp is located in the configuration file at C:/Slyp/config.ini and is parsed into relevant variables at run time. The variables include hostname, VM name, import path, and export path.

* 1. **Identification:**AWS Slyp  
       
     **Type:**   
     Module  
       
     **Purpose:**   
     To facilitate the interaction between the Slyp server and a Hyper-V host.

**Function:**

AWS Slyp takes input from the configuration file at C:/Slyp/config.ini and allows Slyp to issue commands to a target AWS host. The operations that the commands allow include: getting the status of VMs, turning VMs on, turning VMs off, exporting a VM, and importing a VM.   
  
**Subordinates/ Modules used:**

Export VM - utilizes the Export-VM commandlet followed by the appropriate parameters as described in the configuration file initiating the export of the VM.

Import VM - utilizes the Import-VM commandlet followed by the appropriate parameters as described in the configuration file initiating the import of the VM.

Get VM Information - utilizes the Get-VM commandlet followed by the target AWS system defined in the configuration file returning the running status and uptime of the particular VM on the target system.

Power on VM - utilizes the Start-VM commandlet followed by the name of the VM to be powered on. If the given VM exists it will power up.

Power off VM - utilizes the Stop-VM commandlet followed by the name of the VM to be powered off. If the given VM exists it will power off.  
  
**Dependencies:**

AWS is used as a cloud provider to host VMs

AWS EC2 servers are used to contain an imported or exported VM

AWS S3 file storage provides temporary storage of a backed up VM

Boto3 is a library used to communicate with AWS

Python2.7 provides the logic and executes Boto3 commands  
  
**Resources:**   
N/A  
  
**Processing:**   
Get status of VM - The python script parses the configuration file into useable variables and uses a subprocess call to Boto3. This command then returns the status of the specified VM on the AWS host and also writes the output to a text file in the C:/Slyp/ directory.

Power on VM - The python script parses the configuration file into useable variables and uses a subprocess call to Boto3 consisting of a VM name. This command turns the given VM on.

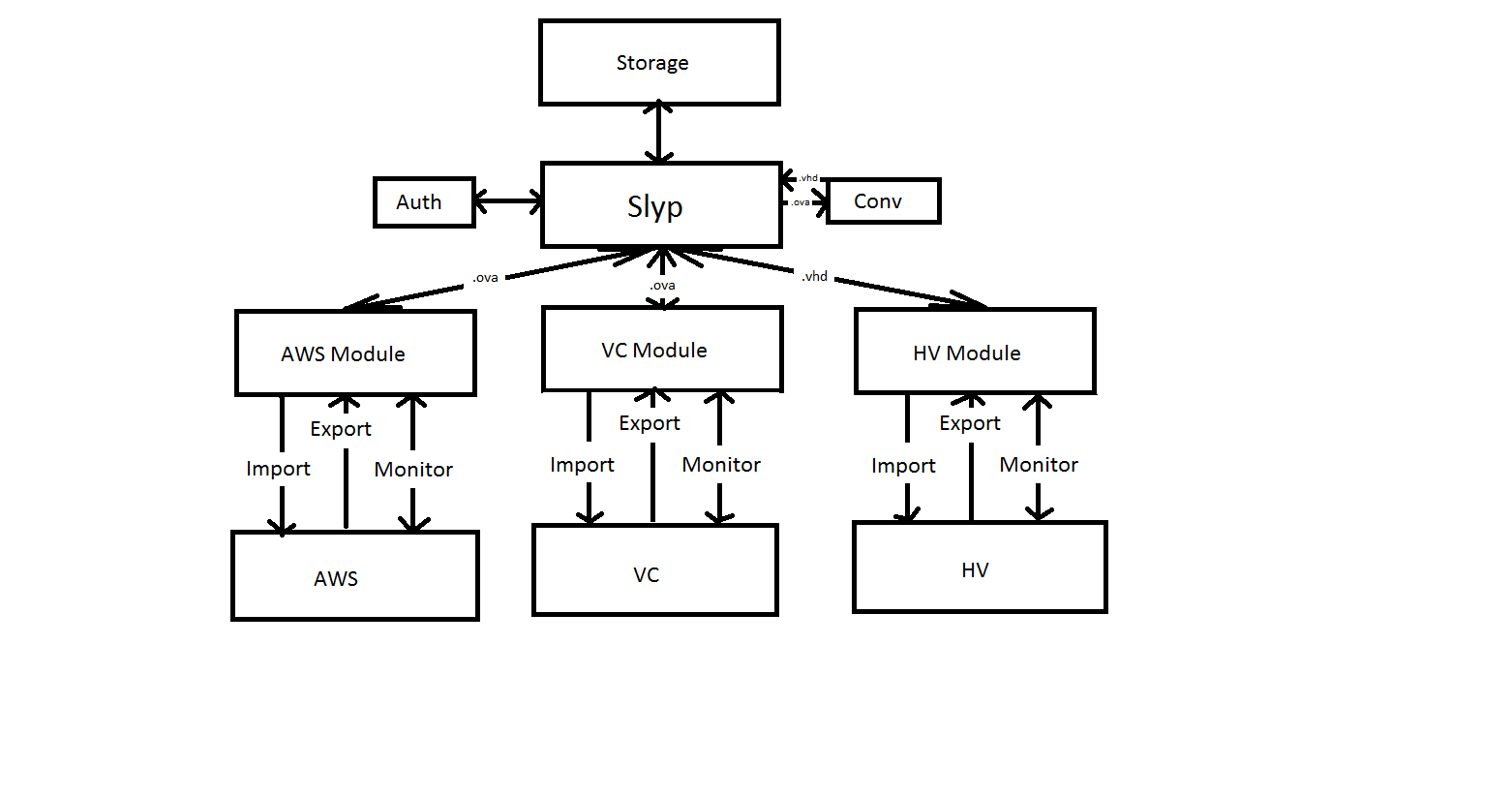
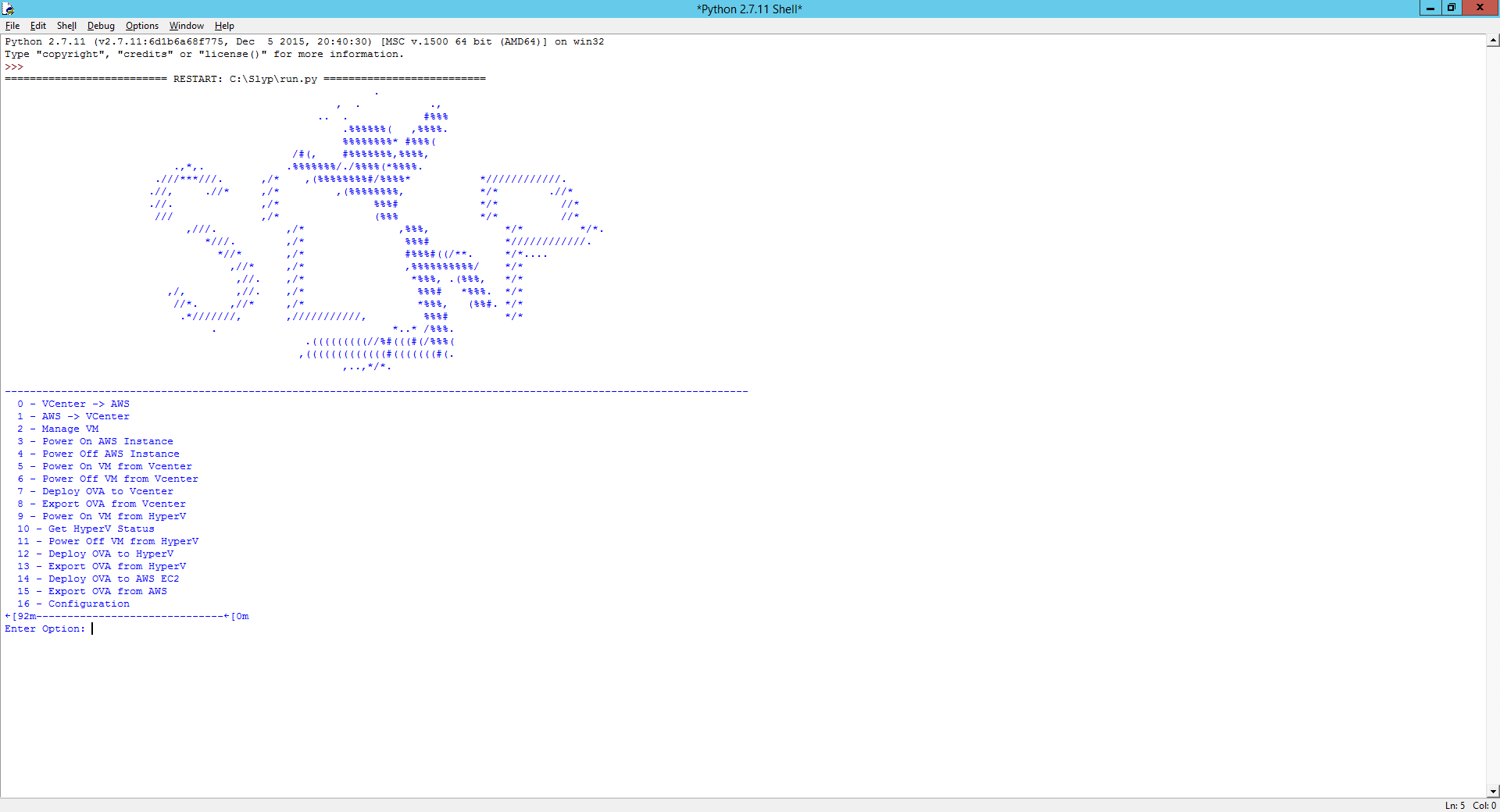
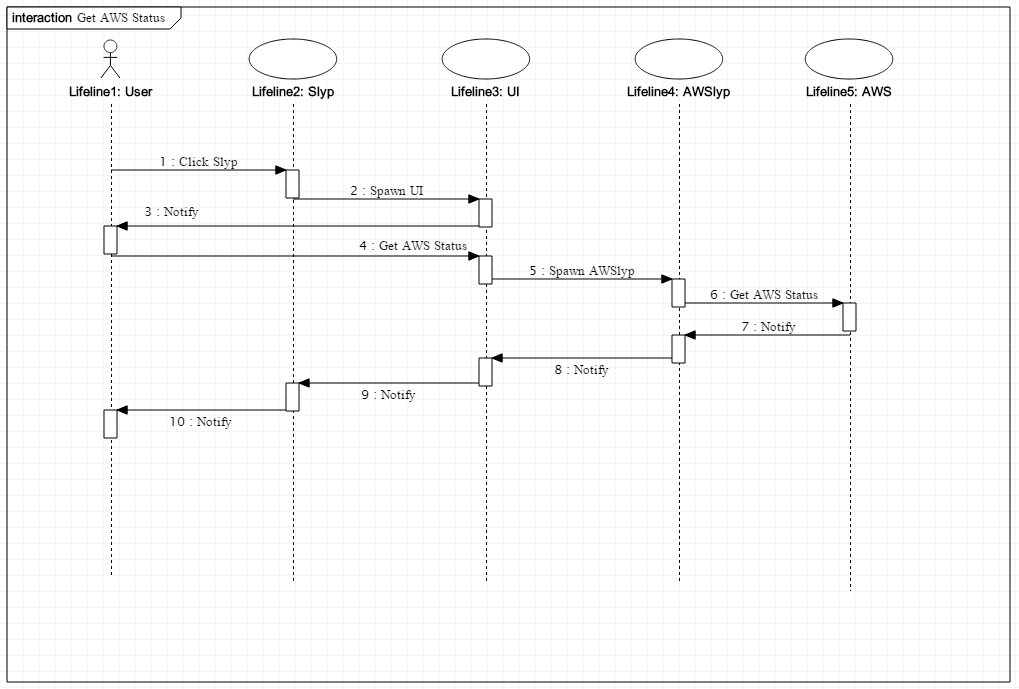
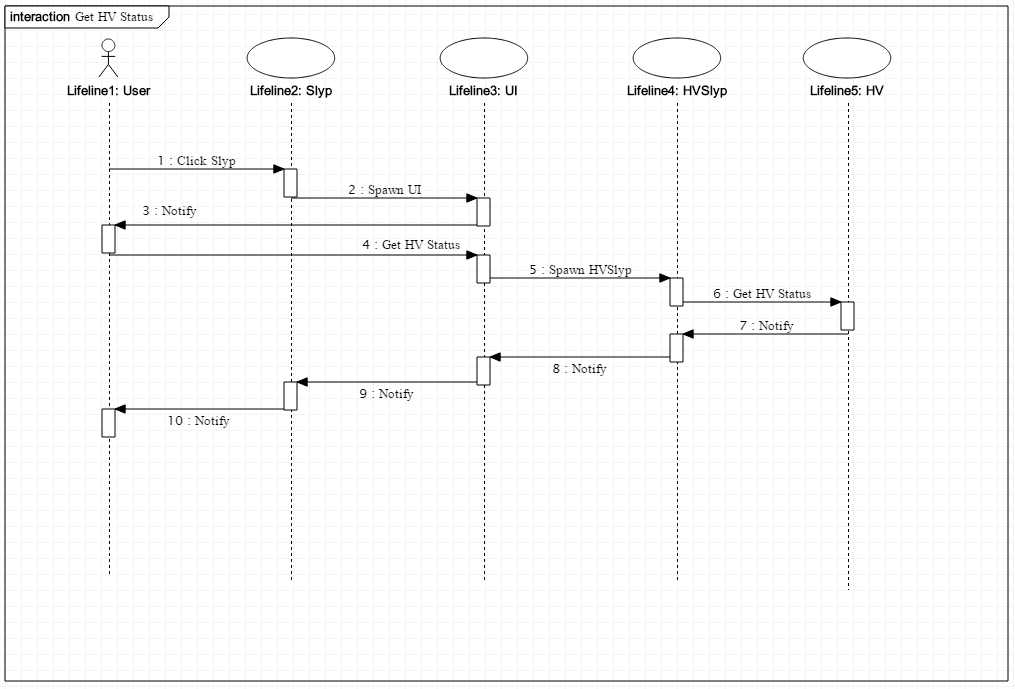
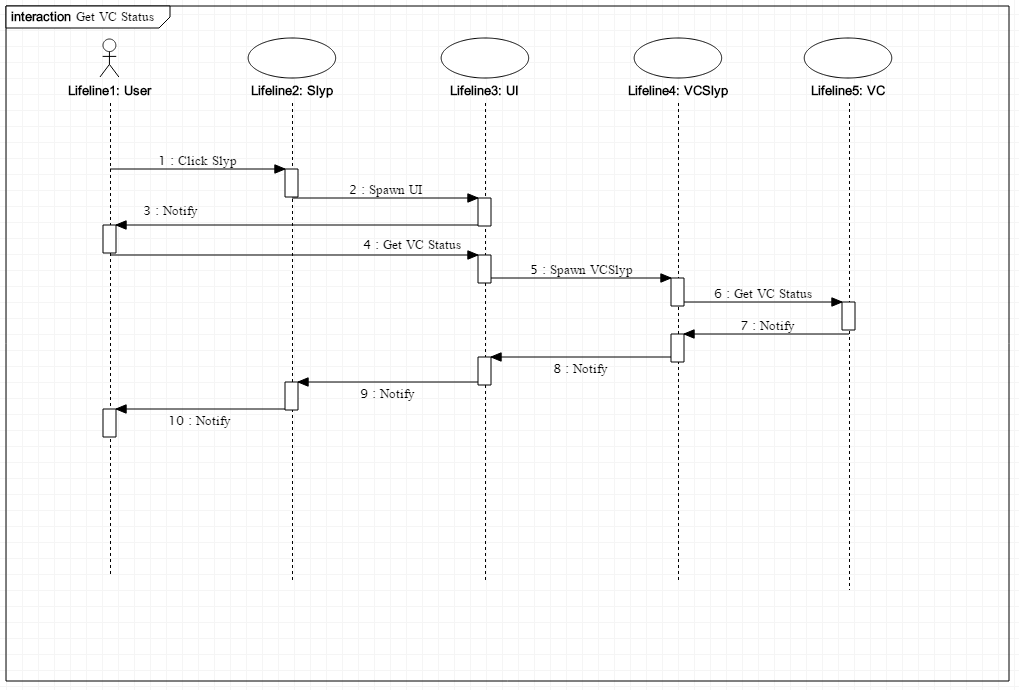
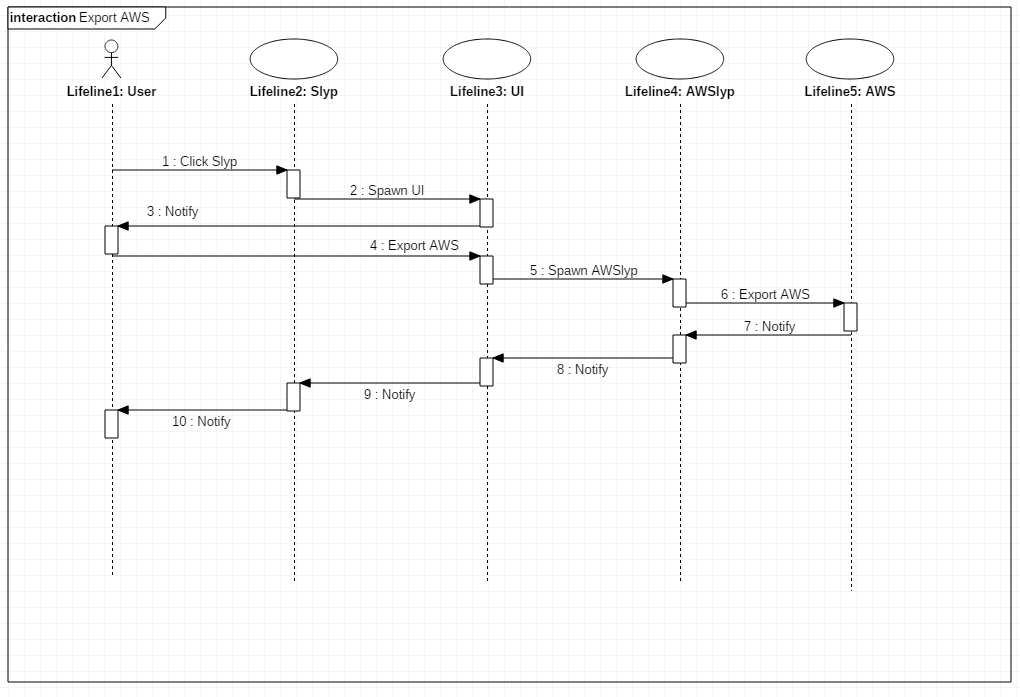
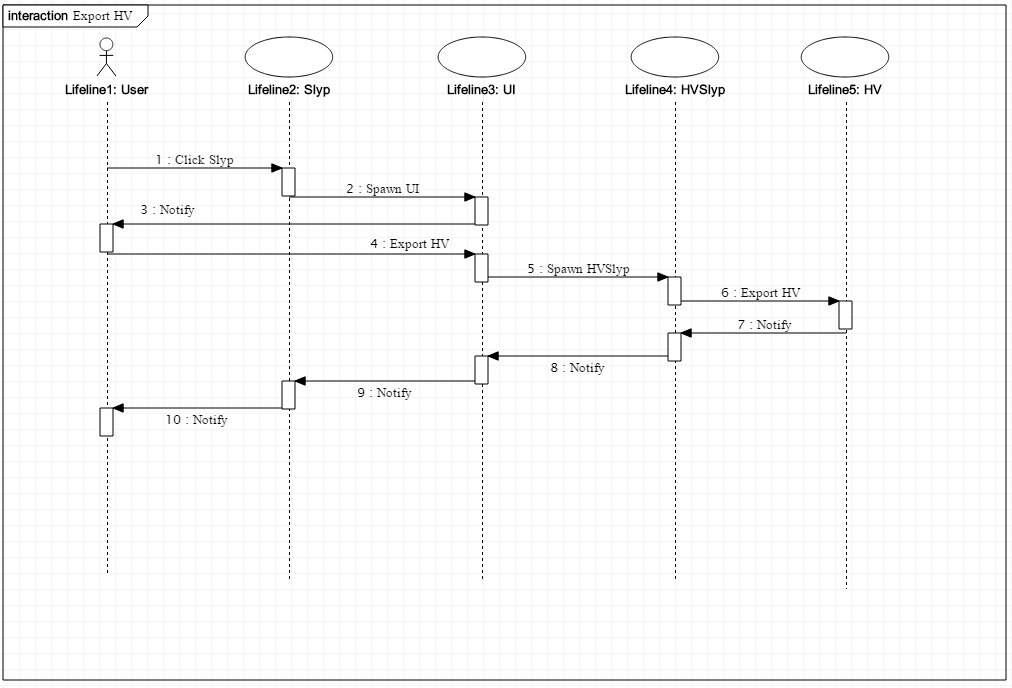
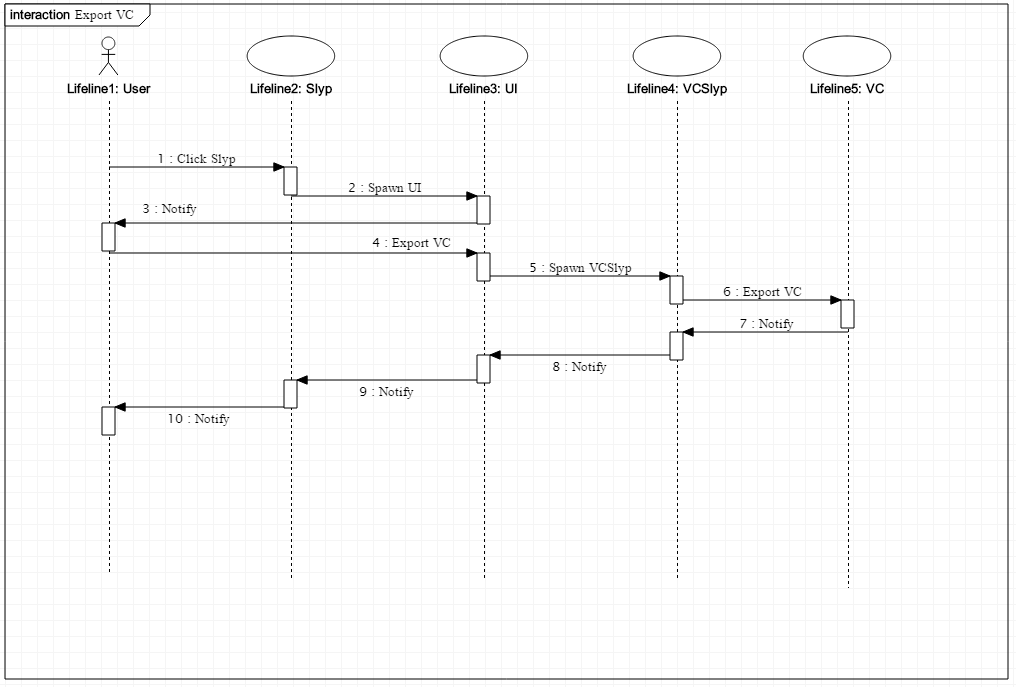
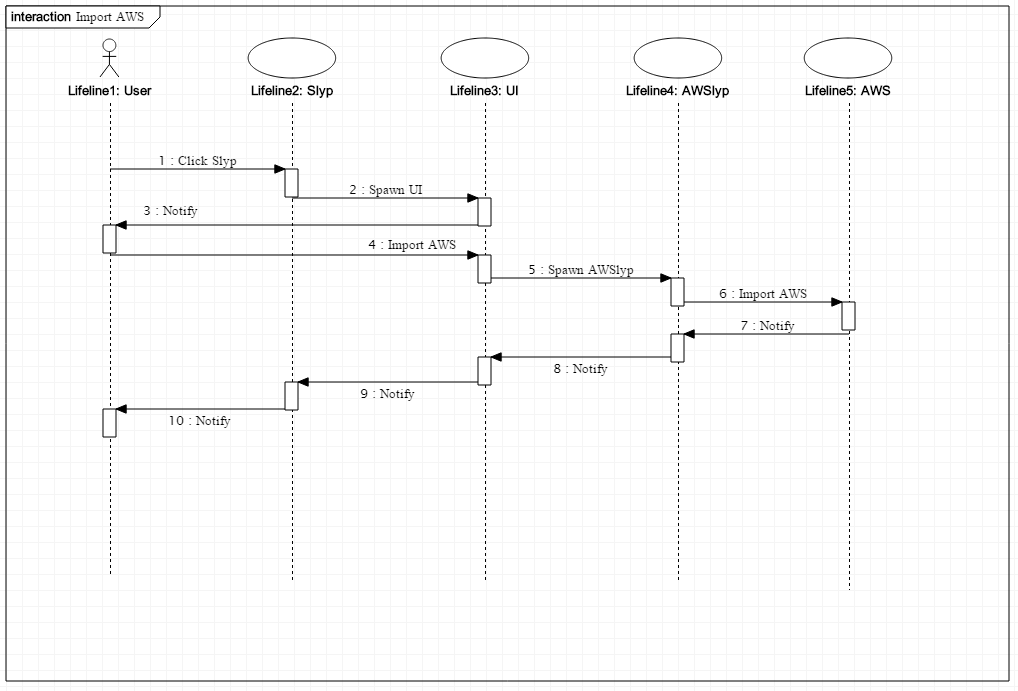
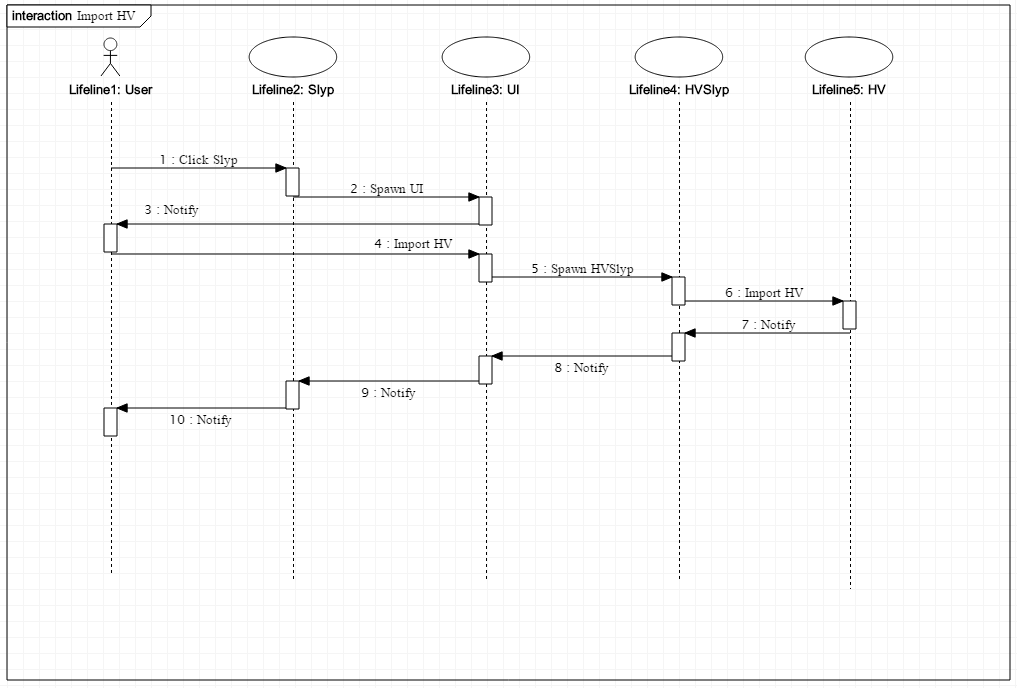
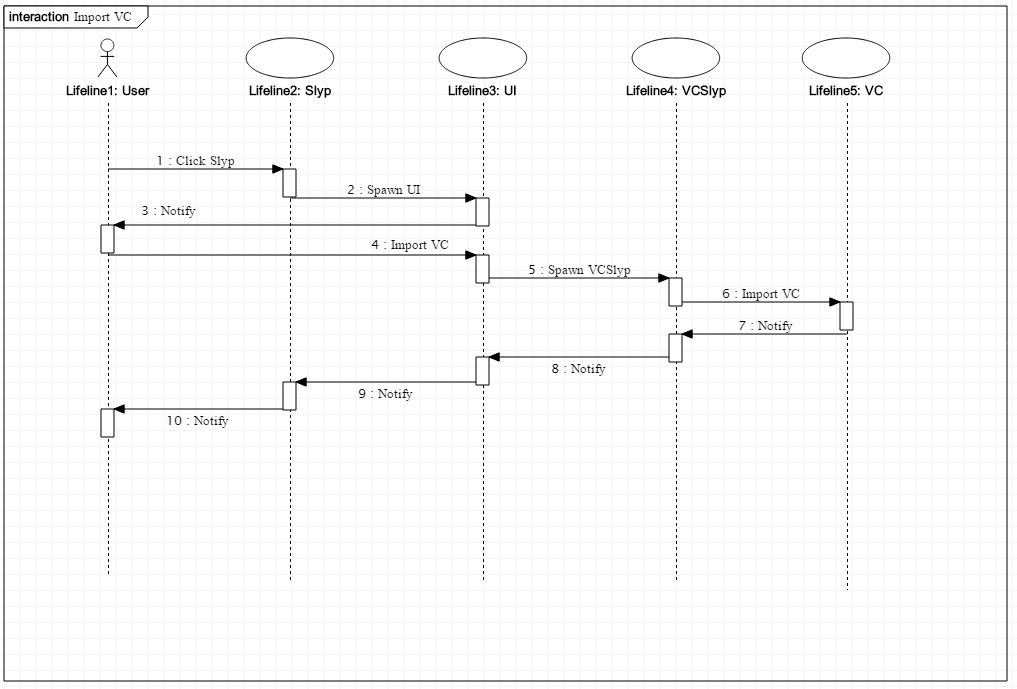
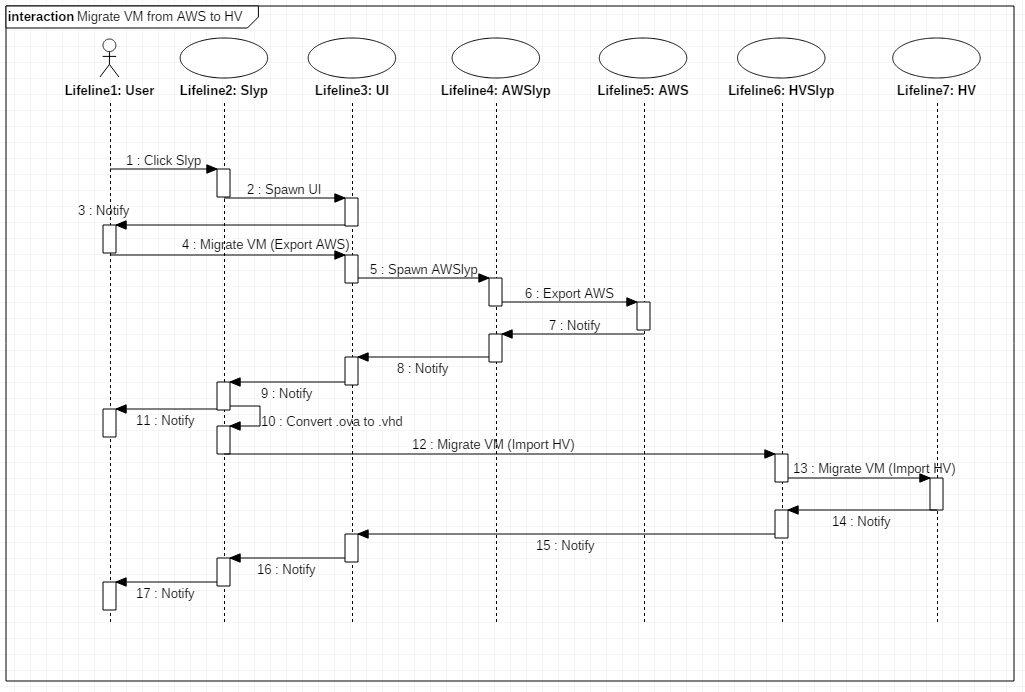
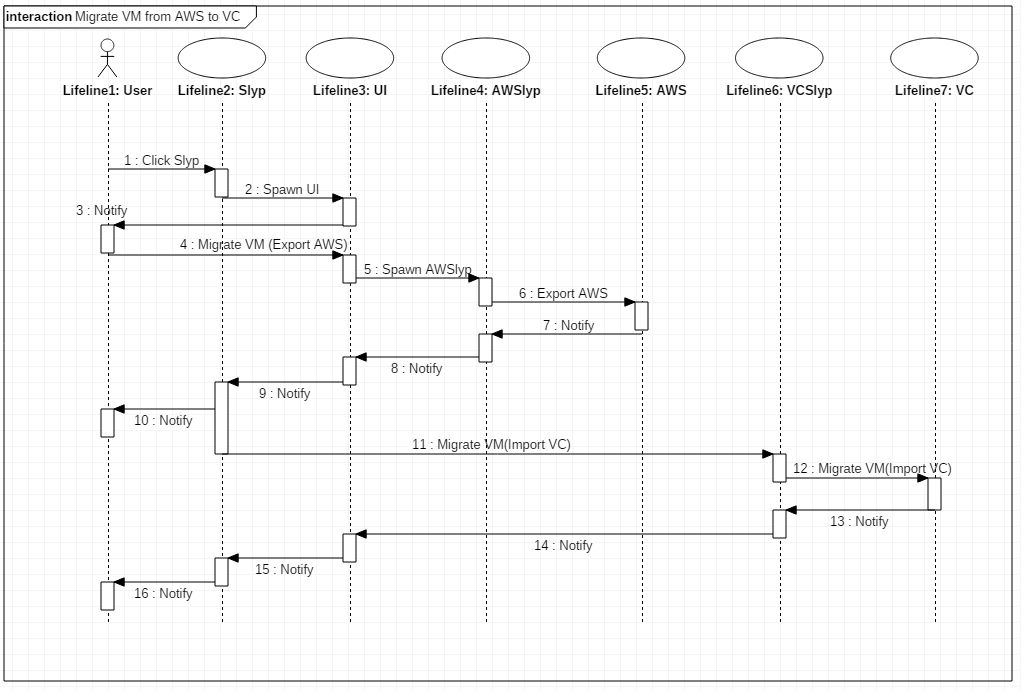
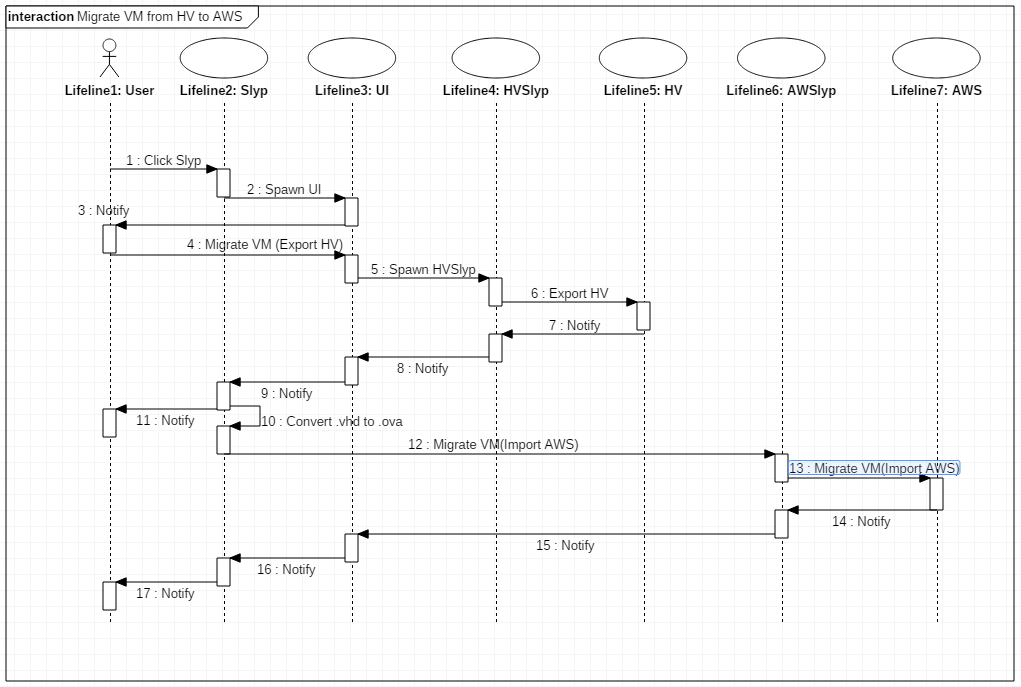
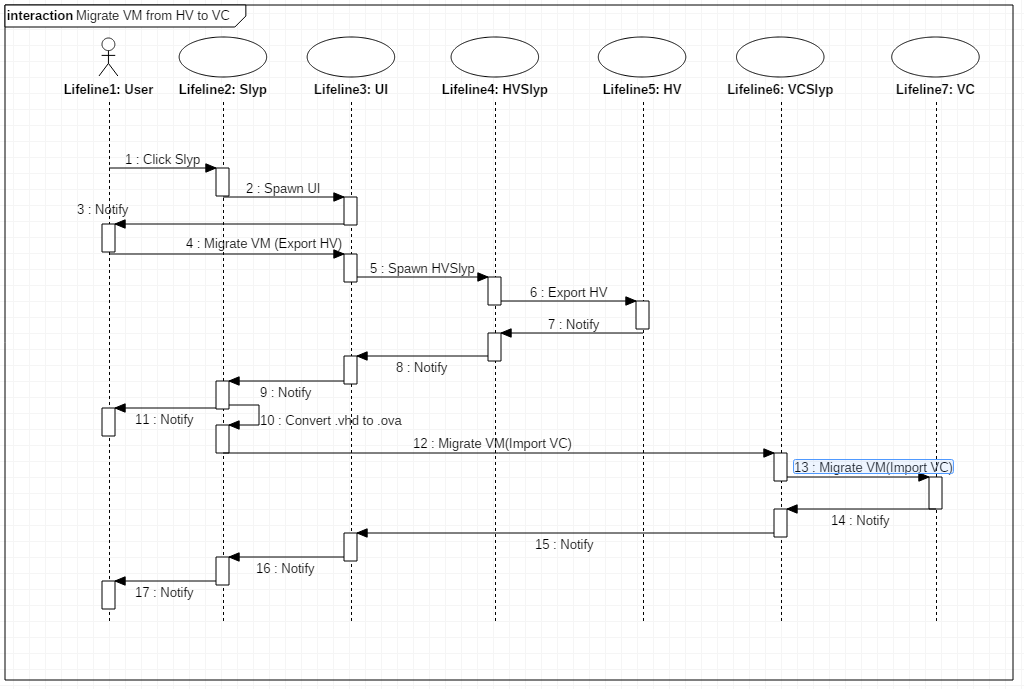
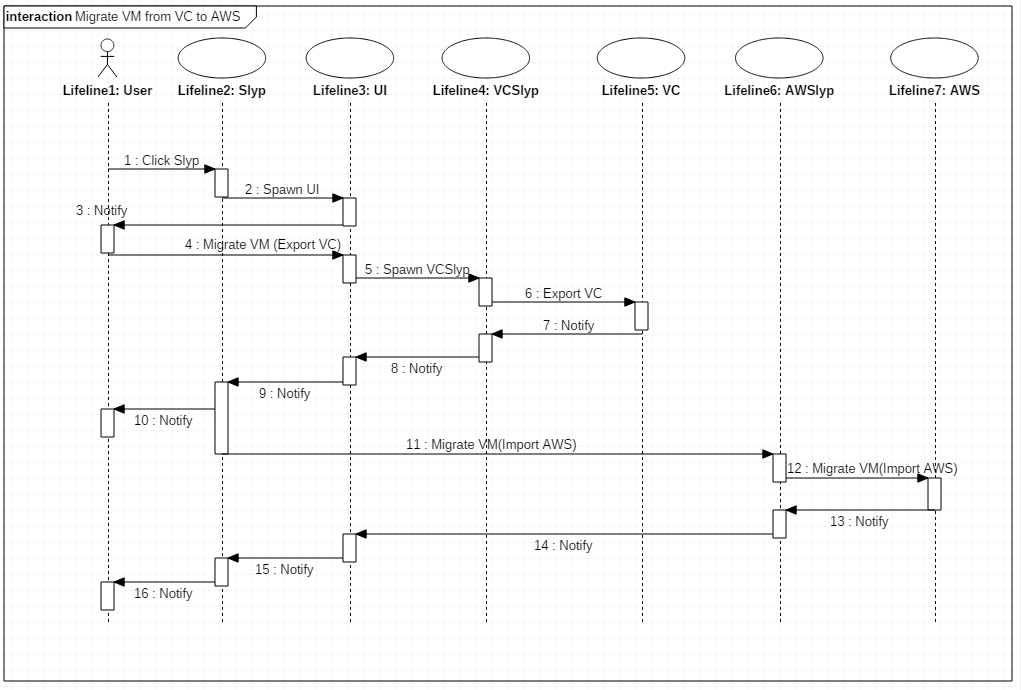
Power off VM - The python script parses the configuration file into useable variables and uses a subprocess call to Boto3 consisting of a VM name. This command turns the given VM off.

Export VM - The python script parses the configuration file into useable variables and uses a subprocess call to Boto3 containing a VM name, export format, and target AWS S3 bucket location. This command then initiates the export of the VM to be exported to the export destination.

Import VM - The python script parses the configuration file into useable variables and uses a subprocess call to Boto3 containing a VM name, import format, and source AWS S3 bucket location. This command then initiates the import of the VM to be imported to the import destination.

**Data:**

All of the data used by Hyper-V Slyp is located in the configuration file at C:/Slyp/config.ini and is parsed into relevant variables at run time. The variables include hostname, VM name, import AWS S3 bucket, and export AWS S3 bucket.

1. **Interface Design**   
     
   See Appendix 6.2  
     
   Slyp uses a command line user interface that consists of ascii art for the borders and logo and lists all possible operations. Selecting a menu item executes the given operation.
2. **Appendix**
   1. Data Flow Diagram  
      
   2. 
   3. 
   4. 
   5. 
   6. 
   7. 
   8. 
   9. 
   10. 
   11. 
   12. 
   13. 
   14. 
   15. 
   16. 
   17. 