
Open Automation Documentation

Release 0.1

Rob Edwards, Matt Day, Russ Whitear

April 06, 2016

1	Introduction	3
1.1	What is UCS Director?	3
1.2	What is Open Automation?	5
2	Getting Started	7
2.1	Setting up Environment	7
2.2	Build Project	10
2.3	Import to UCS Director	10
2.4	Vaidation	10
3	Architecture	11
4	Cookbook	13
5	Account	15
6	Tasks	17
6.1	Config.java	17
6.2	Data.java	18
6.3	jdo.files	18
6.4	Main Module	18
7	Reports	21
8	Example Plugins	23
9	Why and who	25

This guide was created to help with developing a custom Open Automation plugin for UCS Director. It is not an official Cisco guide however deep knowledge of the products, using official documentation and utilising resources available to the authors this has been put together. Hopfully it will help on your journey and if you have feedback please get in touch or even make updates yourself and make a pull request.

The code is open source, and available on [github](#).

The documentation for the site is organized into the following sections:

Introduction

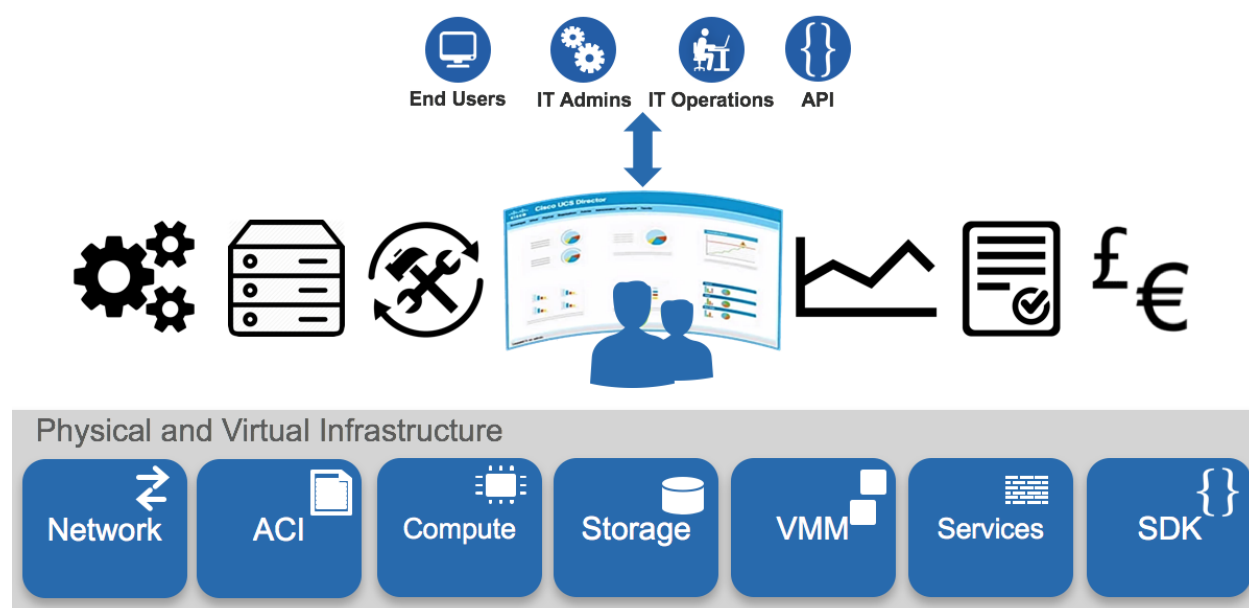
Note: These pages are being built up at the moment and not complete, please treat them as a work in progress!

1.1 What is UCS Director?

UCS Director is a tool to help you automate your data centre infrastructure ([Cisco Website](#)). It abstracts hardware and software into programmable tasks that are assembled together to provision infrastructure across computing, networking and storage resources that reside on multiple hypervisor. The business value is;

- Simplified infrastructure provisioning and management that takes minutes rather than weeks
- Physical, virtual and multi-vendor management from a single place to accommodate heterogeneous data centres
- Increased IT agility allowing IT to have greater impact on the effectiveness of the business.

The following image summarises the functionality of UCS Director;



The main call out feature are;

- Tasks and workflows - more than 2200 out of the box tasks (about 2400 person days of effort), these can be used to configure workflows that combine useful tasks into a single goal.
- Bare Metal - Assist with the provision and configuration of bare metal (PXE)
- Lifecycle Management - manage day to day tasks of the entire infrastructure from a single place
- Metrics - Collects some performance related metrics (NOT a performance management tool)
- Reports - Generate reports on the infrastructure
- Showback/chargeback - Tracks the usage of infrastructure resources to showback usage for groups (or extracted for billing engines)

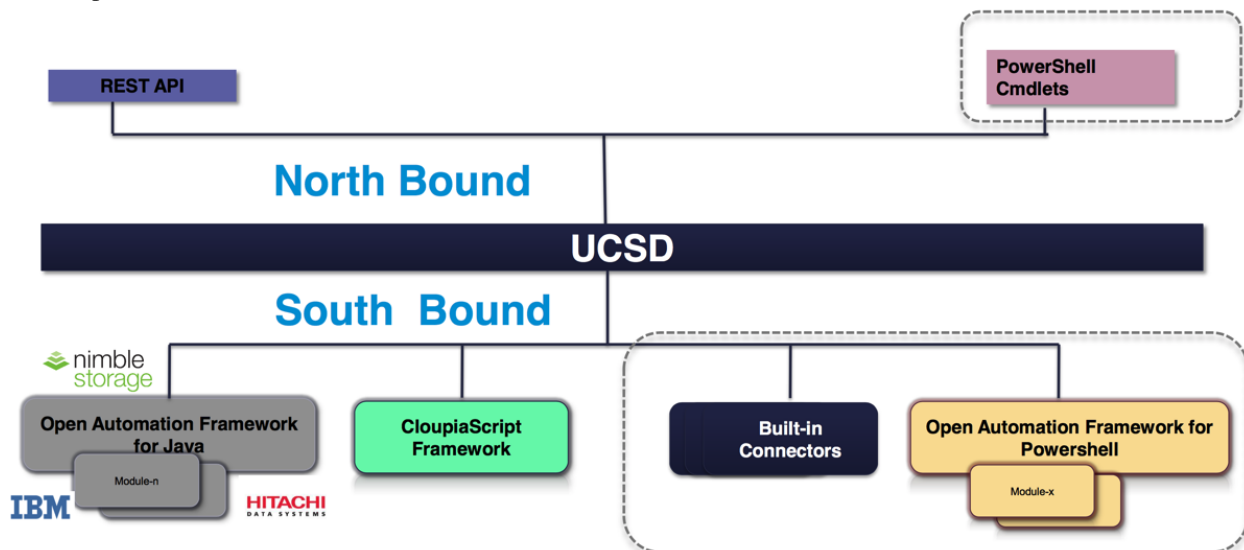
It is possible to interact with UCS Director in multiple ways;

- Southbound
- Northbound

Access Northbound, access to the UCSD functionality, can be via the web GUI, RestAPI or Powershell Cmdlets (uses the RestAPI).

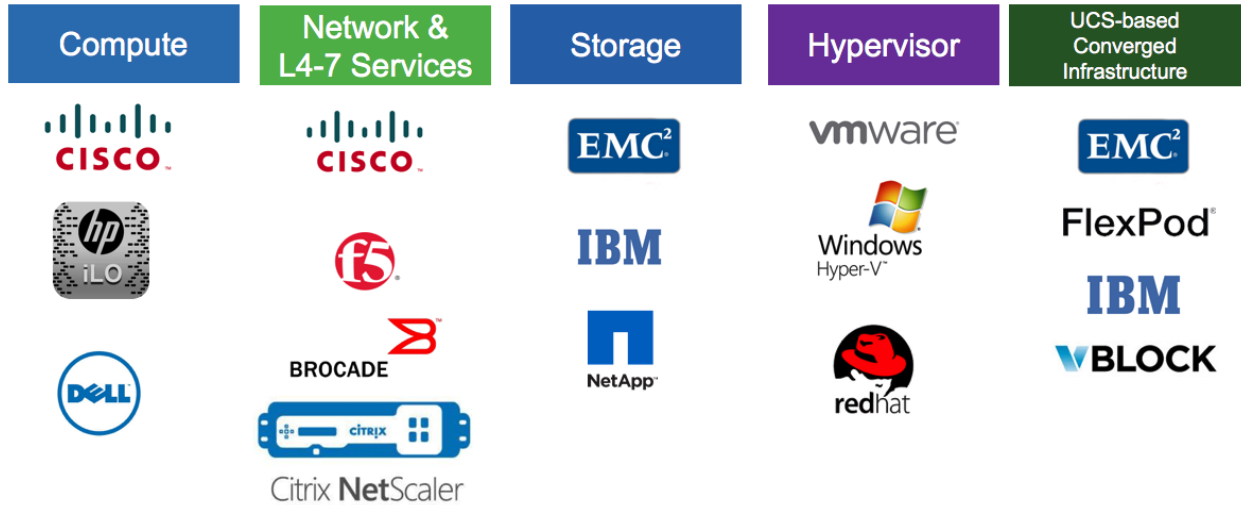
The Southbound access, communication to infrastructure, can be achieved by one of these mechanisms;

- Built-in Connectors
- Powershell
- Cloupiascript
- Open Automation



The built-in connectors are the devices that are supported out of the box. These supported devices have automation tasks prebuilt and tested by Cisco. The official list of infrastructure supported by UCS Director is [here](#).

Supported vendors;



Powershell support is via a powershell agent and allows interaction with the underlying infrastructure using powershell functionality.

The CloudpaaS framework allows you to create custom tasks (based around JavaScript) to interact with managed devices or it could be expanded to interact with things that are not managed in UCS Director. These tasks are quick ways to expand the capabilities of UCS Director and many examples can be found on the [Cisco Communities site](#).

1.2 What is Open Automation?

While the use of CloudpaaS and the Powershell agent makes UCS Director a big flexible tool its southbound capabilities don't end there, it also has the concept of 'Open Automation'. This can be used to make 3rd party devices look like they are natively supported by UCSD by allowing you to;

- develop reports and report actions
- inventory devices
- track changes made to the system through the module
- develop tasks that can be used for workflows
- develop and schedule repeatable tasks
- set up new resource limits

Open Automation is in essence a Java SDK that can be used to create a connector/plugin that can be imported into UCSD.

The official Cisco documentation (always improving) for this can be found [here](#).

Getting Started

To start we will run through how to setup your development environment and build the example dummy plugin that is on the Cisco download site for UCS Director.

Note: Make sure that you pull the dummy example for the version of UCS Director you are running!

The tasks that we will run through with you are;

- Getting started and setting up Eclipse
- Importing dummy SDK into Eclipse
- Build Dummy SDK
- Import Dummy SDK & Enable
- Check Dummy SDK loaded (including logging info)

If you are already using a Java IDE and happy with the process to build and import plugins skip ahead.

2.1 Setting up Environment

2.1.1 Downloads

The first thing to do is [download Eclipse](#) for the operating system that you use. The one we used was 'Eclipse IDE for Java Developers'.

The screenshot shows the Eclipse website's Downloads page. The header includes the Eclipse logo, navigation links (GETTING STARTED, MEMBERS, PROJECTS, MORE), and a search bar. The main content area is titled 'HOME / DOWNLOADS' and includes a 'Packages' link. A large banner promotes the Eclipse Installer, stating it's the easiest way to install and update the development environment, with 799,548 downloads. Below this, there are links to download Eclipse packages for Mac OS X (Cocoa). A sidebar on the left features a 'FOSS4G NORTH AMERICA 2016' banner. The right sidebar contains 'RELATED LINKS' (Compare & Combine Packages, New and Noteworthy, Install Guide, Documentation, Updating Eclipse, Forums) and 'MORE DOWNLOADS' (Other builds, Eclipse Mars (4.5), Eclipse Luna (4.4), Eclipse Kepler (4.3)). The 'Eclipse IDE for Java Developers' package is highlighted with a red box.

Once downloaded follow the installation instruction, we won't go through these here.

Now you should download the UCS Director Dummy Plugin for your version from [Cisco CCO](#). The file that you are looking for is *cucsd-open-automation-sdk-bundle-<ver>.zip*.



Download Software

Download Cart (0 items) [Feedback](#) [Help](#)

[Downloads Home](#) > [Products](#) > [Servers - Unified Computing](#) > [UCS Director](#) > [UCS Director 5.4](#) > [UCS Director Virtual Appliance Software-5](#)

UCS Director 5.4

[Expand All](#) | [Collapse All](#)

▼ Latest

5

▼ All Releases

► 5

Release 5

[Add Device](#)
[Add Notification](#)

cucsd_patch_5_4_0_2 upgrade patch

File Information	Release Date ▼	Size	
Cisco UCS Director 5.4.0.0 Cloupia Script Bundle cucsd-cloupia-script-bundle-5.4.0.0.zip	05-NOV-2015	1.30 MB	Download Add to cart Publish
Cisco UCS Director 5.4.0.0 Cloupia Script Code Samples cucsd-cloupiascript-code-samples-5.4.0.0.zip	05-NOV-2015	0.43 MB	Download Add to cart Publish
Cisco UCS Director 5.4.0.0 Open Automation MD5 Checksum - 7c92b4d8093d86ed4bb60f166e240feb cucsd-open-auto-sdk-bundle-5.4.0.0.zip	05-NOV-2015	9.38 MB	Download Add to cart Publish
Cisco UCS Director 5.4.0.0 REST API (MD5 Checksum - 2b38390ffe486fef7e8e1855b324e05f) cucsd-rest-api-sdk-5.4.0.0.zip	05-NOV-2015	15.88 MB	Download Add to cart Publish

The download will contain two zip's

cucsd-open-auto-sdk-bundle-5.4.0.0 2			
Name	Date Modified	Size ▼	Kind
cucsd-open-auto-sdk-sample-5.4.0.0.zip	4 Nov 2015, 07:43	9.4 MB	ZIP archive
cucsd-open-auto-sdk-javadocs-5.4.0.0.zip	18 Oct 2015, 23:18	543 KB	ZIP archive

The first is the sample, or dummy, open automation code while the second is the Javadocs that we will use to enhance Eclipse.

If you unzip and browse the 'sample' zip you should see something similar to this;

cucsd-open-auto-sdk-sample-5.4.0.0			
Name	Date Modified	Size ▼	Kind
▼ Open_Automation	Today, 13:12	10.7 MB	Folder
▶ lib	4 Nov 2015, 07:41	10 MB	Folder
▶ bin	4 Nov 2015, 07:41	398 KB	Folder
▶ src	4 Nov 2015, 07:41	263 KB	Folder
▶ resources	4 Nov 2015, 07:41	9 KB	Folder
build.xml	27 Oct 2015, 21:13	3 KB	XML
▶ cloudsense	4 Nov 2015, 07:41	1 KB	Folder
ReadMe.txt	27 Oct 2015, 21:13	649 bytes	Plain Text
▶ poddefinition	4 Nov 2015, 07:41	604 bytes	Folder
foo.feature	27 Oct 2015, 21:13	146 bytes	Document

2.1.2 Eclipse

As we are using GitHub to share and collaborate on the examples we create the [Eclipse EGit plugin](#) was added to our environment. This is optional.

2.1.3 Setup Project

Now that we have an IDE and the sample open automation code downloaded we need to import it into an Eclipse workspace.

import project java versions - JRE 1.8 Set Java compiler to JRE 1.8 java docs

2.2 Build Project

2.3 Import to UCS Director

2.3.1 Import plugin

2.3.2 Enable plugin

2.3.3 Debug import

Check Dummy SDK loaded (including logging info)

2.4 Validation

Check it works

Architecture

Explain the architecture of the plugin

- Account
- Tasks
- Reports
- Cloudsense
- Pod Definitions
- Resources
- etc.

Cookbook

description of code and example snippets

Account

Tasks

To create tasks and make them available you will need to create two classes;

- <name>Config.java
- <name>Task.java

The jdo.files will also need to be created/updated.

You will also need to add the task classes to the main module

6.1 Config.java

```
@PersistenceCapable(detachable = "true", table = "spark_dummy")
public class SparkDummyConfig implements TaskConfigIf {

    public static final String displayLabel = "Spark Get Inventory";
    @Persistent
    private long configEntryId;
    @Persistent
    private long actionId;

    @FormField(label = "Spark Array IP", help = "spark Array IP Address", mandatory = true)
    @UserInputField(type = WorkflowInputFieldTypeDeclaration.IPADDRESS)
    @Persistent
    private String ipAddress = "";

    @FormField(label = "Spark Username", help = "spark username", mandatory = true, type = FormFieldDefinition.TEXT)
    @UserInputField(type = SparkConstants.GENERIC_TEXT_INPUT)
    @Persistent
    private String username;

    @FormField(label = "Password", help = "Password", mandatory = true, type = FormFieldDefinition.TEXT)
    @UserInputField(type = SparkConstants.PASSWORD)
    @Persistent
    private String password;
```

6.2 Data.java

```
public class SparkDummyTask extends AbstractTask {

    private static Logger logger = Logger.getLogger( SparkDummyTask.class );

    @Override
    public void executeCustomAction(CustomActionTriggerContext context,
        CustomActionLogger actionLogger) throws Exception {
        SparkDummyConfig config = (SparkDummyConfig) context.loadConfigObject();

        String username = config.getUsername();
        String password = config.getPassword();
        String ipAddress = config.getIpAddress();

        String arrayName = "";

        /* Code */
    }

    @Override
    public TaskConfigIf getTaskConfigImplementation() {
        return new SparkDummyConfig();
    }

    @Override
    public String getTaskName() {
        return SparkDummyConfig.displayLabel;
    }

    @Override
    public TaskOutputDefinition[] getTaskOutputDefinitions() {
        return null;
    }
}
```

6.3 jdo.files

```
//Each package with config classes should have a file called jdo.files
//Each config class should be listed in the format below
//This file informs the build file which classes need to go through JDO enhancement
+SparkDummyConfig
```

6.4 Main Module

```
@Override
public AbstractTask[] getTasks() {
    AbstractTask task1 = new SparkDummyTask();
    AbstractTask task2 = new SparkMessageCreateTask();

    AbstractTask[] tasks = new AbstractTask[2];
    tasks[0] = task1;
```

```
tasks[1] = task2;  
  
return tasks;  
}
```

Reports

Example Plugins

List of examples that have been created by the community (or other vendors). These are great ways to learn how others have build plugins (best way to learn in my eyes)

Plugin	Description	Creator	Location
Nimble		Community	https://github.com/rwhitear42/nimble
Nimble		Community	https://github.com/rwhitear42/slimcea
Infoblox		Community	https://github.com/rwhitear42/infoblox
3Par		Community	https://github.com/CiscoUKIDCDev/HP3ParPlugin
Spark		Community	https://github.com/CiscoUKIDCDev/UCSD-OA-Spark
Pure		Pure Storage	https://github.com/purestorage/ucs-director-plugin

Other comercial plugins are availbe however these will not be tracked here.

Why and who

Quick description of why and who has been building up these pages