Chapter 5 :

STACKALYTICS

- Introduction
- Quick revisiting of all accounts – openstack foundation
- What is it?
- Significance of Stackalytics
- How to set up Stackalytics

5. STACKALYTICS

In this chapter we will cover:

1: Quick run through on setting the accounts like Git, Gerrit, Jenkins & Launchpad etc.
2: Talking about Stackalytics in depth and understanding it's structure/ process.
3: Significant of the Stackalytics.
4: Understanding the setup of the Stackalytics.

Introduction

Stackalytics is a data visualization tool that collects data from GitHub and presents it in an array of useful forms. Not only can Stackalytics break down the data but it also lets you track commits and overall lines of code.

The aim of the Stackalytics is to provide transparent and meaningful statistics regarding contributions to both OpenStack itself and projects related to OpenStack. Transparency is important so that the community can have confidence that all calculations are correct and fair. So "transparent" means that anyone can double check the methods of calculation Stackalytics uses. Meanwhile, results must be meaningful to be useful. "Meaningful" means that anyone may submit a correction that will adjust the influence of appropriate statistical data. For example, auto-generated code, mass renaming, automatic refactoring, auto-generated config files, and so on can artificially inflate various statistics. Stackalytics makes it possible to avoid these problems as they're discovered.

1. Quick revisiting of all accounts – openstack foundation

Contributing to OpenStack can seem especially daunting to a newcomer as it follows a slightly more complex process. We will summarize this process of quick view of all the required accounts for this.

Git, Gerrit, Jenkins & Launchpad

If you're not familiar, here's a quick overview of the tools used by OpenStack to manage development of both code and documentation. If you're already familiar you can skip this part.

Git is a distributed version control system very commonly used in open source projects. GitHub is a commercial hosted git repository for collaborating using Git. OpenStack's actively developed code and documentation is hosted there.

Gerrit is a code review tool developed by Google that integrates with Git. It allows testing and review of code before it's committed to a project.

Jenkins is a continuous integration (CI) tool. What Jenkins does is automatically build and test code with the proposed changes to make sure it didn't break anything.

Launchpad is a website by Canonical/Ubuntu for collaborating on open source projects. The main use for it in OpenStack is bug tracking.

Accounts

Now that you know what the tools are, you'll need a couple of accounts. You'll need to use the same email address for all of them.

- The OpenStack Foundation- You'll want an account anyway, but this is a requirement to contribute. You'll also have to digitally sign the individual contributor agreement.
- Launchpad- Your account here is used as a single-sign-on source for Gerrit as well. Launchpad will generate a username for you, but if you wish to change it, now is the time.
- Gerrit- While it uses your launchpad id to sign you in, you'll need to sign in once to create an account and select a username.

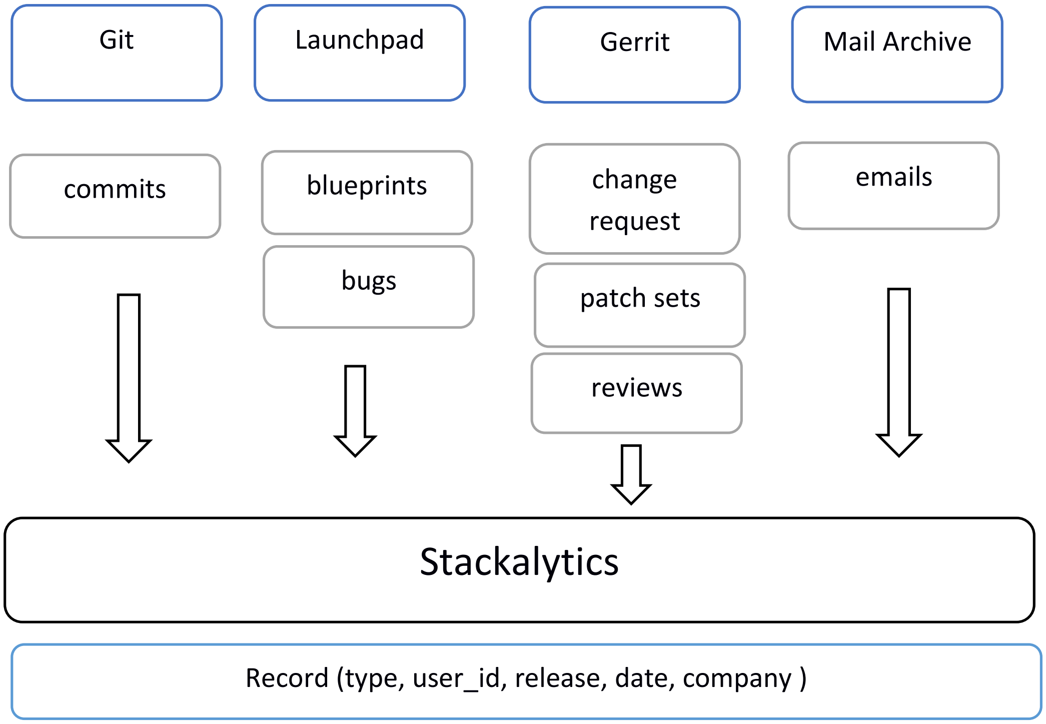


Figure 1: Basic idea for account synch with Stackalytics

2. What is it?

Stackalytics is a service that collects and processes development activity data such as commits, lines of code changed, and code reviews, blueprints and makes it possible to visualize it in a convenient web dashboard. The Stackalytics dashboard makes it possible to view data by project, company, contributor, and other factors..

Some of the features of stackalytics are listed below:

- Extraction of author information from git log, store it in the database;
- Calculate metrics on number of lines changed (LOC) and commits;
- Mapping authors to companies and launchpad ids;
- Filter statistics by time, modules, companies, authors;
- Extract blueprint and bug ids from commit messages;
- Auto-update of database

3. Significance of Stackalytics

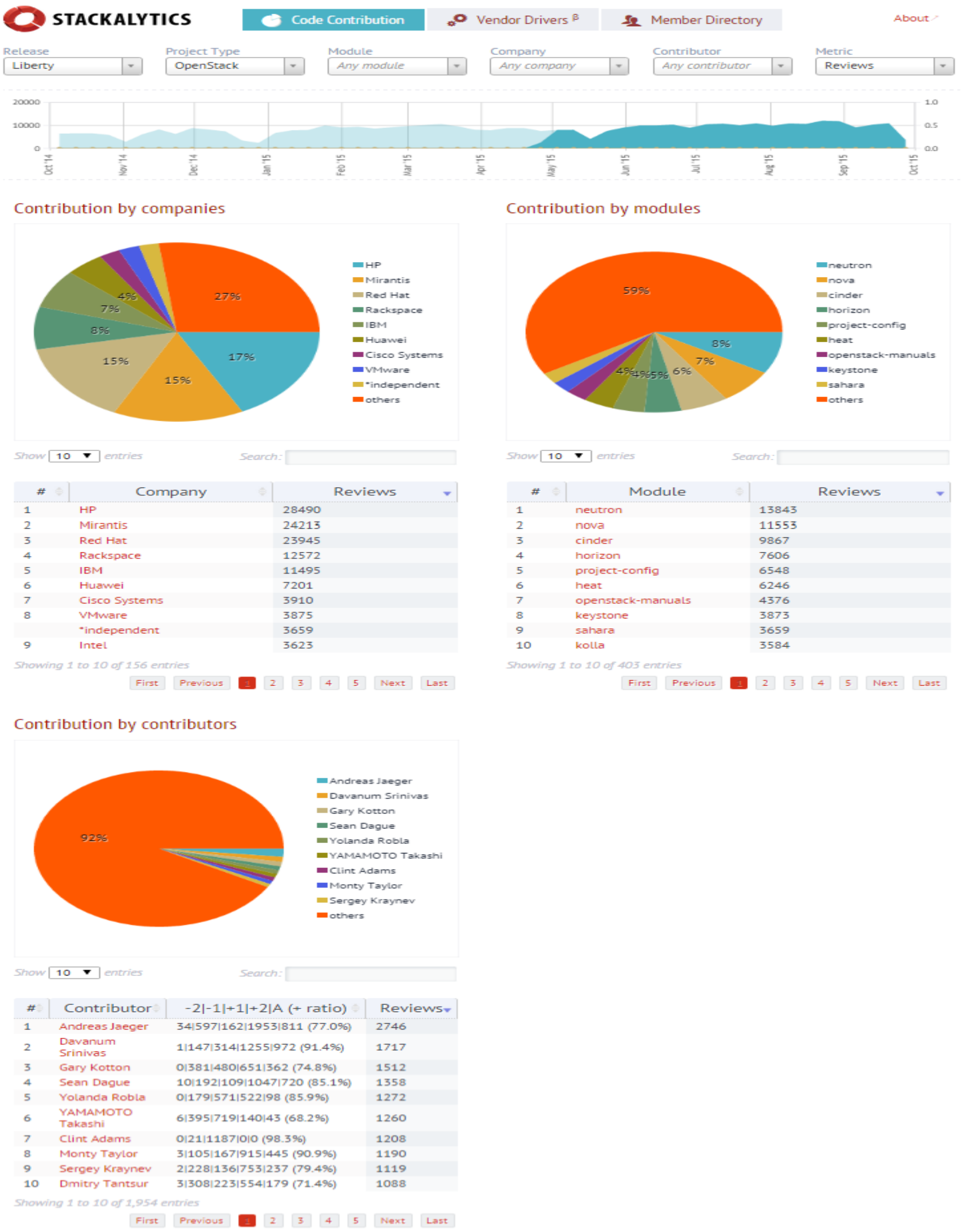
In the cloud space where platforms are rapidly iterating and infrastructure vendors struggle to keep up, it is difficult to build an infrastructure that would work seamlessly. Keeping this requirement in mind, Stackalytics, a dashboard to give customers clarity about which infrastructure solutions are interoperable with OpenStack, was built by an OpenStack contributor, Mirantis.

Stackalytics contains three main characteristic base view:

- Code Contribution
- Vendor Drivers
- Member Directory

Code Contribution

The home page of stackalytics is a pretty interesting page. It lists the top contributions by **companies, modules and contributors** with some statistics that helps us to have an idea about the overall contribution. Additionally we can also see the metrics on the page for different categories, also we can go by individual company or module to have a detail look. The figure below depicts the front page.



Release: Liberty
Project Type: OpenStack
Module: Any module
Company: Any company
Contributor: Any contributor
Metric: Reviews

Contribution by companies

#	Company	Reviews
1	HP	28490
2	Mirantis	24213
3	Red Hat	23945
4	Rackspace	12572
5	IBM	11495
6	Huawei	7201
7	Cisco Systems	3910
8	VMware	3875
	*independent	3659
9	Intel	3623

Showing 1 to 10 of 156 entries

Contribution by modules

#	Module	Reviews
1	neutron	13843
2	nova	11553
3	cinder	9867
4	horizon	7606
5	project-config	6548
6	heat	6246
7	openstack-manuals	4376
8	keystone	3873
9	sahara	3659
10	kolla	3584

Showing 1 to 10 of 403 entries

Contribution by contributors

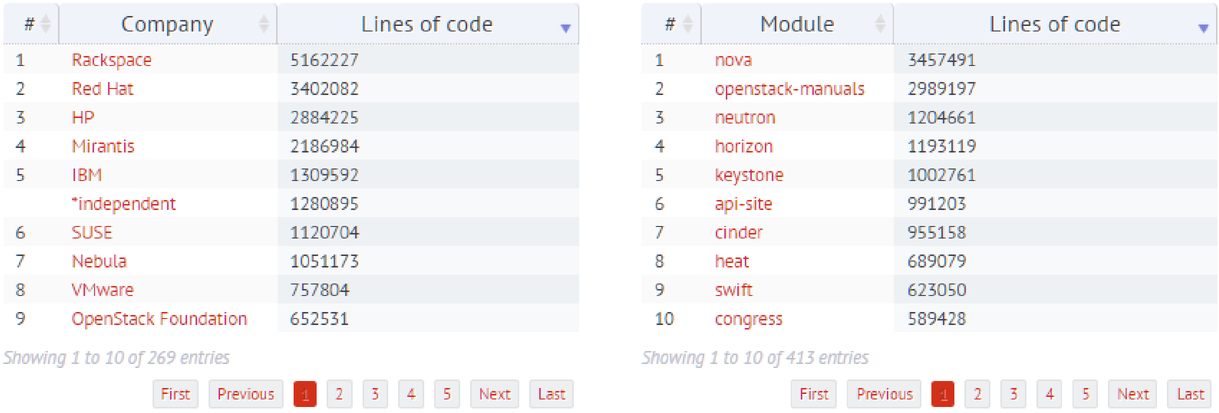
#	Contributor	-2|-1|+1|+2|A (+ ratio)	Reviews
1	Andreas Jaeger	34|597|162|1953|811 (77.0%)	2746
2	Davanum Srinivas	1|147|314|1255|972 (91.4%)	1717
3	Gary Kotton	0|381|480|651|362 (74.8%)	1512
4	Sean Dague	10|192|109|1047|720 (85.1%)	1358
5	Yolanda Robla	0|179|571|522|98 (85.9%)	1272
6	YAMAMOTO Takashi	6|395|719|140|43 (68.2%)	1260
7	Clint Adams	0|21|1187|0|0 (98.3%)	1208
8	Monty Taylor	3|105|167|915|445 (90.9%)	1190
9	Sergey Kraynev	2|228|136|753|237 (79.4%)	1119
10	Dmitry Tantsur	3|308|223|554|179 (71.4%)	1088

Showing 1 to 10 of 1,954 entries

Figure 2: Stackalytics Homepage

Lines and Commits

One of the more important metrics that illustrates work hours and productivity is the number of commits made against the lines of code produced by developers. A commit is a unit of work that is performed by a developer when they create, fix, or delete some code in a particular module. This code is then processed by Gerrit, Jenkins, and/or SmokeStack, reviewed by at least two core developers, and is finally merged into the Master Branch. Lines of Code (or LoC) is the number of actual lines of code that are created, fixed, or deleted.



#	Company	Lines of code
1	Rackspace	5162227
2	Red Hat	3402082
3	HP	2884225
4	Mirantis	2186984
5	IBM	1309592
	*independent	1280895
6	SUSE	1120704
7	Nebula	1051173
8	VMware	757804
9	OpenStack Foundation	652531

Showing 1 to 10 of 269 entries

First Previous 1 2 3 4 5 Next Last

#	Module	Lines of code
1	nova	3457491
2	openstack-manuals	2989197
3	neutron	1204661
4	horizon	1193119
5	keystone	1002761
6	api-site	991203
7	cinder	955158
8	heat	689079
9	swift	623050
10	congress	589428

Showing 1 to 10 of 413 entries

First Previous 1 2 3 4 5 Next Last

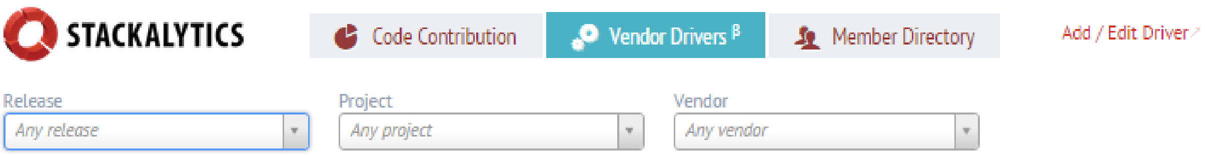
Figure 3: statistical view for the matric "Line of Code"

The difference between these two is critical because the values can show project managers where the team's productivity is yielding results.

Consider a developer who produces 5,000 lines of code and yet only has a handful of small commits. Alternately, a developer who produces 400 lines of code might have dozens of commits. This kind of information can help project managers better guide their projects and developers. Additionally, you can compare one company's progress with your own or others. For greater detail into the development tracks being taken on various projects.

Vendor Drivers

This shows a central list of all drivers for OpenStack. Vendors can add information about their drivers. The drivers can be searched by **Release** (Liberty, Juno etc), **Project**(Cinder, Neutron, Nova etc) and **Vendor**(AMD, Cisco, Ceph etc). The figure 2 below shows this feature.



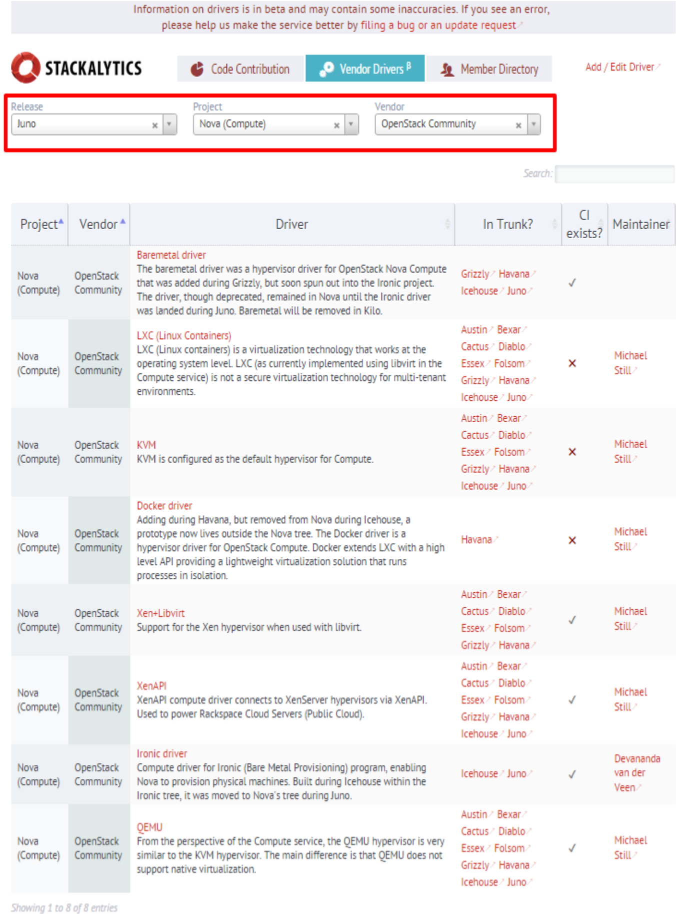
Release: *Any release*

Project: *Any project*

Vendor: *Any vendor*

Figure 4: Stackalytics > Vendor Driver page

Apart from that, for a novice user, if he wants to know the information about the driver, then there is a documentation link provided in the driver information which can be accessed by users to view the detailed information about the driver. For instance, we searched for drivers in **Juno** version in the project **Nova** by the vendor **OpenStack Community** and we got the results shown in the figure 3 below:



Information on drivers is in beta and may contain some inaccuracies. If you see an error, please help us make the service better by filing a bug or an update request

Release: Juno
Project: Nova (Compute)
Vendor: OpenStack Community

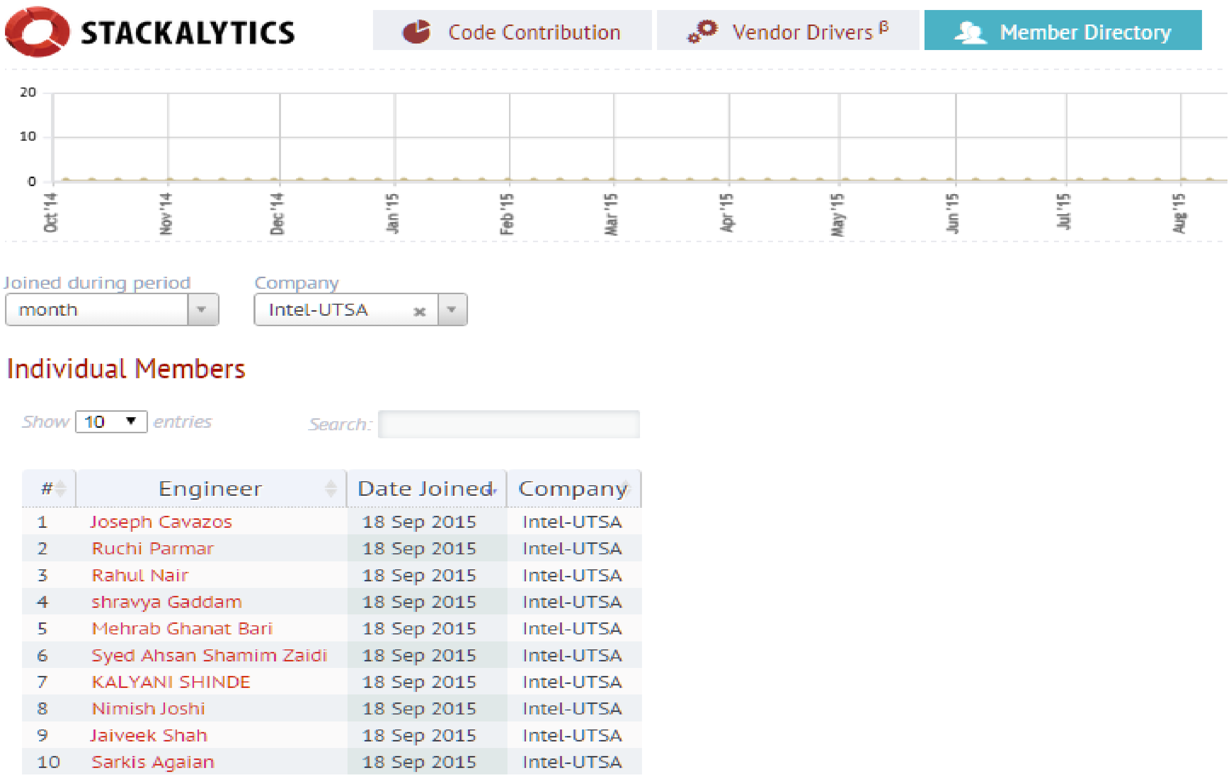
Project	Vendor	Driver	In Trunk?	CI exists?	Maintainer
Nova (Compute)	OpenStack Community	Baremetal driver The baremetal driver was a hypervisor driver for OpenStack Nova Compute that was added during Grizzly, but soon spun out into the Ironic project. The driver, though deprecated, remained in Nova until the Ironic driver was landed during Juno. Baremetal will be removed in Kilo.	Grizzly Havana Icehouse Juno	✓	
Nova (Compute)	OpenStack Community	LXC (Linux Containers) LXC (Linux containers) is a virtualization technology that works at the operating system level. LXC (as currently implemented using libvirt in the Compute service) is not a secure virtualization technology for multi-tenant environments.	Austin Bexar Cactus Diablo Essex Folsom Grizzly Havana Icehouse Juno	✕	Michael Still
Nova (Compute)	OpenStack Community	KVM KVM is configured as the default hypervisor for Compute.	Austin Bexar Cactus Diablo Essex Folsom Grizzly Havana Icehouse Juno	✕	Michael Still
Nova (Compute)	OpenStack Community	Docker driver Adding during Havana, but removed from Nova during Icehouse, a prototype now lives outside the Nova tree. The Docker driver is a hypervisor driver for OpenStack Compute. Docker extends LXC with a high level API providing a lightweight virtualization solution that runs processes in isolation.	Havana	✕	Michael Still
Nova (Compute)	OpenStack Community	Xen+Libvirt Support for the Xen hypervisor when used with libvirt.	Austin Bexar Cactus Diablo Essex Folsom Grizzly Havana	✓	Michael Still
Nova (Compute)	OpenStack Community	XenAPI XenAPI compute driver connects to XenServer hypervisors via XenAPI. Used to power Rackspace Cloud Servers (Public Cloud).	Austin Bexar Cactus Diablo Essex Folsom Grizzly Havana Icehouse Juno	✓	Michael Still
Nova (Compute)	OpenStack Community	Ironic driver Compute driver for Ironic (Bare Metal Provisioning) program, enabling Nova to provision physical machines. Built during Icehouse within the Ironic tree, it was moved to Nova's tree during Juno.	Icehouse Juno	✓	Devananda van der Veen
Nova (Compute)	OpenStack Community	QEMU From the perspective of the Compute service, the QEMU hypervisor is very similar to the KVM hypervisor. The main difference is that QEMU does not support native virtualization.	Austin Bexar Cactus Diablo Essex Folsom Grizzly Havana Icehouse Juno	✓	Michael Still

Showing 1 to 8 of 8 entries

Figure 5: Vender driver page details

Search and Drilldown

In addition to the view selectors, you can also dynamically search for information. Each column of data has a search field at the top. Enter search terms and watch as the results change, which helps you find what you’re looking for. Every list has a search field, making it faster and easier to find something specific.



Joined during period: month

Company: Intel-UTSA

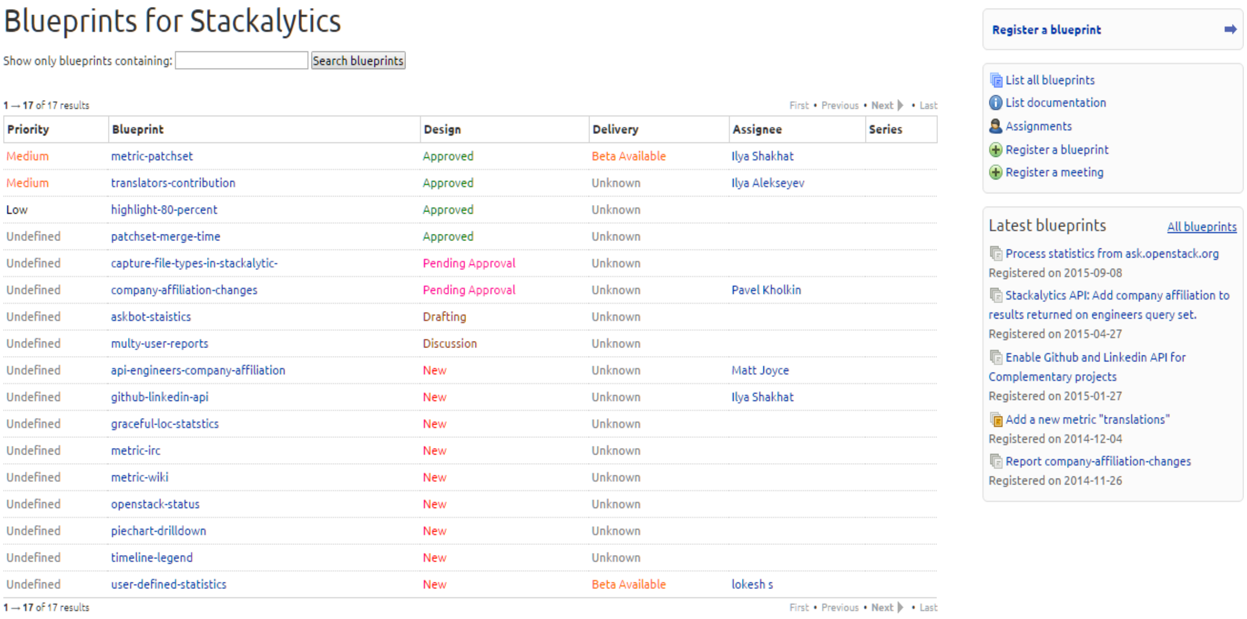
Individual Members

Show 10 entries Search:

#	Engineer	Date Joined	Company
1	Joseph Cavazos	18 Sep 2015	Intel-UTSA
2	Ruchi Parmar	18 Sep 2015	Intel-UTSA
3	Rahul Nair	18 Sep 2015	Intel-UTSA
4	shravya Gaddam	18 Sep 2015	Intel-UTSA
5	Mehrab Ghanat Bari	18 Sep 2015	Intel-UTSA
6	Syed Ahsan Shamim Zaidi	18 Sep 2015	Intel-UTSA
7	KALYANI SHINDE	18 Sep 2015	Intel-UTSA
8	Nimish Joshi	18 Sep 2015	Intel-UTSA
9	Jaiveek Shah	18 Sep 2015	Intel-UTSA
10	Sarkis Agaian	18 Sep 2015	Intel-UTSA

Figure 6: Member Directory View

You can also examine the project blueprints used by the developers to manage the planned changes and improvements.



Blueprints for Stackalytics

Show only blueprints containing: Search blueprints

1 → 17 of 17 results First • Previous • Next • Last

Priority	Blueprint	Design	Delivery	Assignee	Series
Medium	metric-patchset	Approved	Beta Available	Ilya Shakhat	
Medium	translators-contribution	Approved	Unknown	Ilya Alekseyev	
Low	highlight-80-percent	Approved	Unknown		
Undefined	patchset-merge-time	Approved	Unknown		
Undefined	capture-file-types-in-stackalytic-	Pending Approval	Unknown		
Undefined	company-affiliation-changes	Pending Approval	Unknown	Pavel Kholkin	
Undefined	askbot-staistics	Drafting	Unknown		
Undefined	multy-user-reports	Discussion	Unknown		
Undefined	api-engineers-company-affiliation	New	Unknown	Matt Joyce	
Undefined	github-linkedin-api	New	Unknown	Ilya Shakhat	
Undefined	graceful-loc-statstics	New	Unknown		
Undefined	metric-irc	New	Unknown		
Undefined	metric-wiki	New	Unknown		
Undefined	openstack-status	New	Unknown		
Undefined	piechart-drilldown	New	Unknown		
Undefined	timeline-legend	New	Unknown		
Undefined	user-defined-statistics	New	Beta Available	lokesh s	

1 → 17 of 17 results First • Previous • Next • Last

Register a blueprint

- List all blueprints
- List documentation
- Assignments
- Register a blueprint
- Register a meeting

Latest blueprints All blueprints

Process statistics from ask.openstack.org
Registered on 2015-09-08

Stackalytics API: Add company affiliation to results returned on engineers query set.
Registered on 2015-04-27

Enable Github and Linkedin API for Complementary projects
Registered on 2015-01-27

Add a new metric "translations"
Registered on 2014-12-04

Report company-affiliation-changes
Registered on 2014-11-26

4. How to set up the Stackalytics

- **Step 1:** First login into your SSH and then we will Clone the Stackalytics repo.

```
$ git clone https://git.openstack.org/stackforge/stackalytics.git
```

- Step 2: Setting the local repository for the Git review: If using **SSH** for Git Review, you will be prompted to allow the connection the first time.

```
$ cd stackalytics
$ git review -s
```

- Step 3: Creating a New Branch for the Proposed Change:

```
$ git checkout -b update_user_info
```

- Step 4: Editing the default_data.json file:

```
$ <vi,nano,etc> etc/default_data.json
```

Place your entry in alphabetical order of the ***launchpad_id*** (For an example)

```
{
    "launchpad_id": "lisasmith",
     "gerrit_id": "lisa-smith",
     "companies": [
         {
             "company_name": "Intel",
             "end_date": null
         }
     ],
     "user_name": "Lisa Smith",
     "emails": ["lisa.smith@intel.com", "lisasmith@example.com"]
 },
```

- Step 5: Check the Git status and stage the file for committing:

```
$ git status
$ git add .
```

- Step 6: Check the Git status again.

```
$ git status
```

- Step 7: Commit the File and write the Commit message.

```
$ git commit -a
// You only need to provide a topic line similar to the following:
$ Update info for Lisa Smith
```

- Step 8: Git review to send the changes:

```
$ git review
```

A link to the change in Gerrit should be displayed in the console.

References:

- https://github.com/stackforge/stackalytics/blob/master/etc/default_data.json
- http://stackalytics.com/
- https://wiki.openstack.org/wiki/Stackalytics
- https://stackalytics.readthedocs.org/en/latest/userdoc/api_v1.0.html
- https://wiki.openstack.org/wiki/Stackalytics/HowToRun
- https://pypi.python.org/pypi/stackalytics/0.8.2
- http://www.slideshare.net/shakhat/stackalytics
- https://www.mirantis.com/blog/stackalytics-com-whos-growing-the-openstack-pie/
- https://blueprints.launchpad.net/stackalytics