Summer Fellowship Report

On

Development of Ubuntu installer, Debian Package and PPA

Submitted by

Himanshu Singh

Under the guidance of

Prof. Siddhartha Ghosh
Civil Engineering Department
IIT Bombay
&
Mentors - Danish Ansari, Ajmal Babu MS
FOSSEE, Osdag
IIT Bombay

July 11, 2019

Acknowledgment

I would like to thank the FOSSEE project from IIT Bombay for giving me an opportunity to do internship with Osdag. The internship opportunity was a great chance for me to learn and develop myself professionally. It helped me to enhance my knowledge in creating Ubuntu installer, Debian packages, PPA and in general software development. I feel grateful to have met so many wonderful people and professionals who guided me through this internship period.

I would like to specially acknowledge Prof. Siddhartha Ghosh with my deepest gratitude who in spite of being busy with his duties, took time out to hear, guide and keep me on the correct path and allowing me to carry out my project at their esteemed research lab (SSRR lab).

I would also like to acknowledge the members of the Osdag team; Danish Ansari (Project Research Assistant), Ajmal Babu MS (Project Research Engineer) and Sourabh Das (Project Research Associate) for their careful and precious guidance which was extremely valuable for my both theoretical and practical study.

I consider this opportunity as a big milestone in my career development. I shall strive to use the acquired skills and knowledge in the best possible way, and I will continue to work on its improvement, in order to attain desired career objectives.

Contents

Chapter 1

Introduction to Osdag

Osdag internship is provided under the FOSSEE project. FOSSEE project promotes the use of FOSS (Free and Open Source Software) tools to improve quality of education in our country. FOSSEE encourages the use of FOSS tools through various activities to provide an alternative to proprietry softwares. It encourages people from industry and academia to use FOSS and help in development either by providing their valuable feedbacks or directly contributing to the projects.

The FOSSEE project is a part of the National Mission on Education through Infrastructure and Communication Technology(ICT), Ministry of Human Resources and Development, Government of India.

Osdag is one such open source software which comes under the FOSSEE project. Osdag internship is provided through FOSSEE project. Any UG/PG/PhD holder can apply for this internship. And the selection will be based on a screening task.

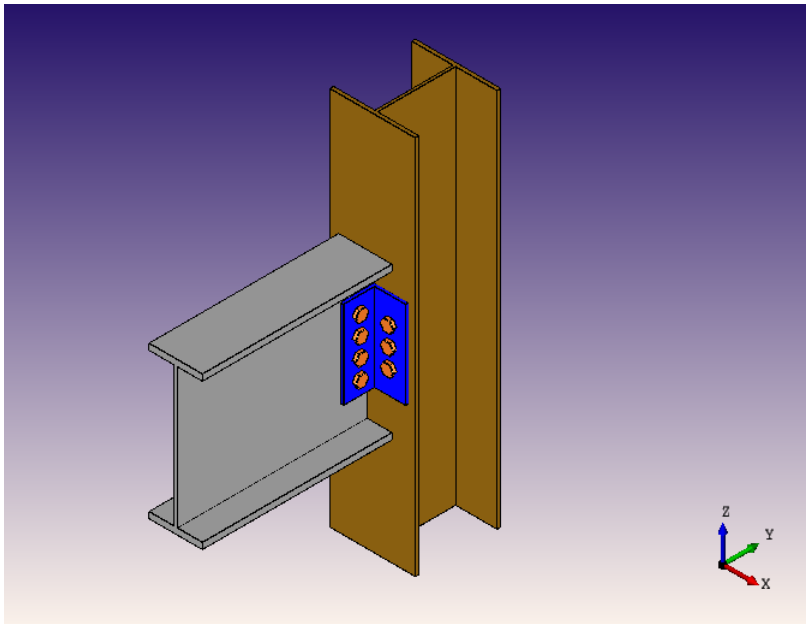
1.1 What is Osdag ?

Osdag is a cross-platform free and open-source software for the design of steel structures, following the Indian standard IS 800:2007. It allows the users to design steel connections, members and systems using a graphical user interface. The interactive GUI provides a 3D visualisation of the designed component and creates images for construction/fabrication drawings.

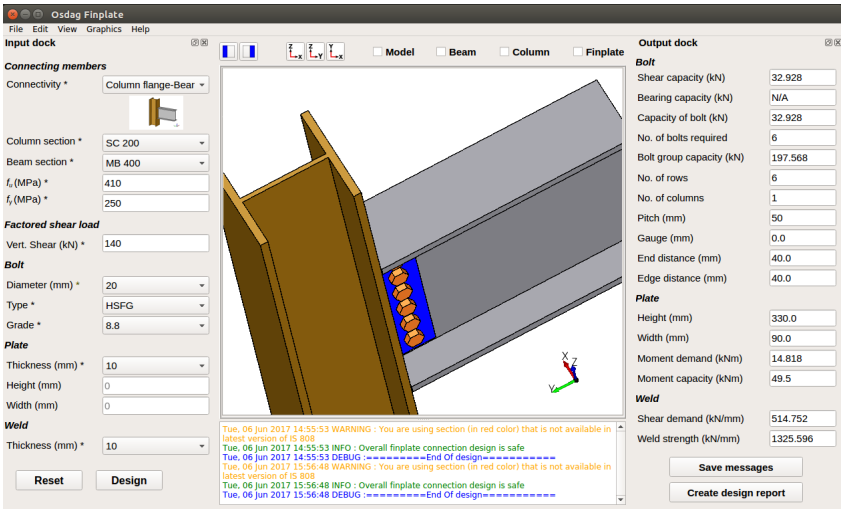
It is used for solving steel structures problems and to see how the connection will look after practical implementation. There are different modules available in Osdag with various connectivities.

Osdag provides various features such as:

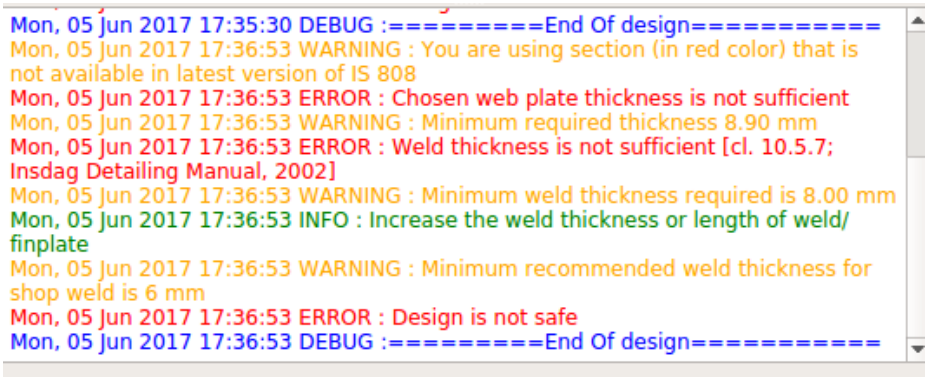
- An interactive window displaying a 3D CAD model, which provides a clear visualisation of the designed component.
- Creation of 3D CAD models that can be imported to generic CAD softwares.



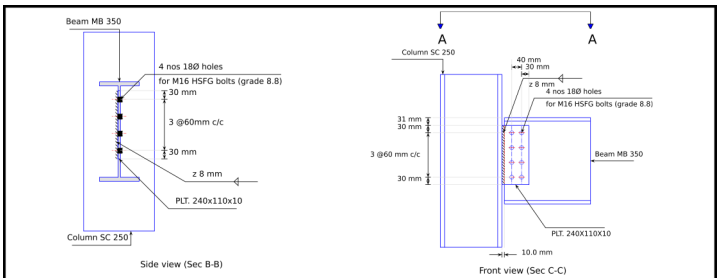
- User-friendly input and output docs, with text-validated fields grouped according to the design flow.



- A text window for message display, that also suggests necessary changes if a trial design is found unsafe.

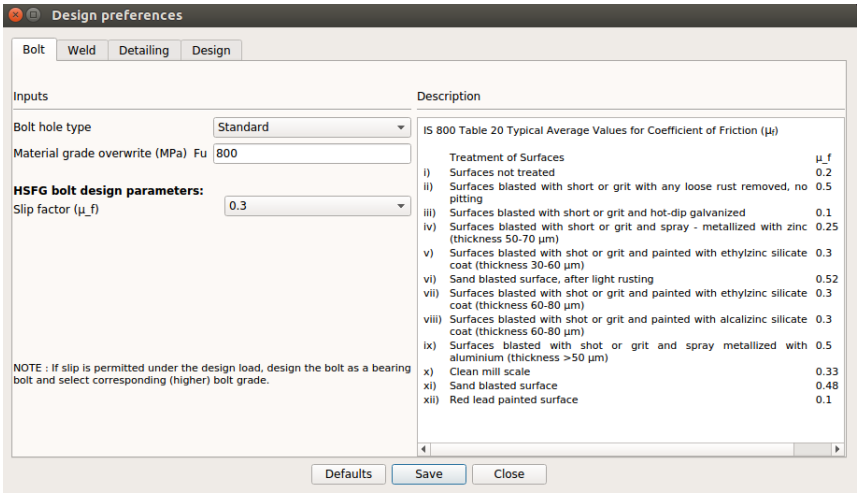


- Creation of a professional design report showing all necessary checks, design calculations as per IS 800:2007, and standard views of the designed component.



Design Check			
Check	**Required**	**Provided**	**Remark**
Bolt shear capacity(kN)		Vdsf = ((0.3*1*1.0*87.92)/(1.25)) = 21.1008 [cl. 10.4.3]	
Bolt bearing capacity(kN)		NA	
Bolt capacity (kN)		21.1008	Pass
No. of bolts	120/21.1008 = 5.7	7	Pass
No. of column(s)	≤ 2	2	
No. bolts per column		4	

- Creation of 2D vector (and raster) images that can be used in a design report or class assignment.
- Selection of design preferences, considering different construction and detailing aspects, using a design preference toolbox.



1.2 Who can use ?

Osdag is generally created for industry professionals but it also keep students in mind. As Osdag is funded by MHRD, Osdag team tries to manipulate software in such a way that it can be used by the students during their academics and to give them a better insight look in the subject.

Basically Osdag can be used by anyone starting from novice to professionals. It's simple and sober user interface makes it flexible and attractive than the other softwares. Also there are video tutorials to get started. The video tutorials of Osdag can be accessed here.

Chapter 2

Development work on Linux Ubuntu

As Osdag is a Free and Open Source Software it is necessary to have it available for the linux platform. On our mission to promote FOSS among the general people it is a good idea to create an application for Ubuntu Linux as it is one of the most used linux distribution.

Another important reason to develop an application for Ubuntu is **Great and Supportive Community -** The Ubuntu Community is a community that welcomes new Ubuntu users and developers. They know that sharing apps with them requires hard work, passion and dedication so they are there to support you. We dont need marketing to promote our apps, the community will do it for us. If the app is special to Ubuntu users and helps them to complete their tasks they will give love back to us by reviewing your app and rating it on the Ubuntu Software Center.

We already had Linux installer with bash scripts, it just needed to be updated (updating package versions and adding the newer ones). The task was to create a Debian package to Osdag so as to submit to Debian repository, and can be ultimately made available to ubuntu store so that it can be installed using the command

```
sudo apt-get install osdag
```

To provide users with the latest feature before the completion of Ubuntu six month update cycle we need to create PPA for Osdag so a user can add that PPA and install it from there.

2.1 Ubuntu installer for Osdag

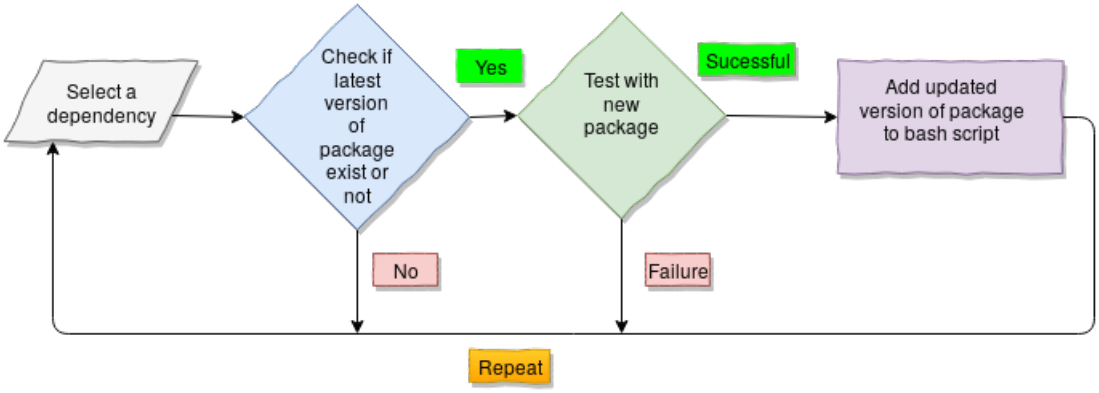
Ubuntu installer for Osdag contains two bash scripts.

1. 1-install-Miniconda2-latest-Linux-x86_64.sh
2. 2-install-osdag.sh

The first script is to install the Miniconda2, on the system if it does not have it already installed. This bash script is available on the anaconda website to install the latest version of Miniconda.

All the dependencies are installed from the dependencies folder using the command 'conda install'. conda is the python package manager for Anaconda python, becomes available when Miniconda is installed.

The second bash script creates a configuration file for Osadg, desktop launcher and installs the dependencies from dependencies folder so as to avoid any conflict with a change in the version of packages.



Ubuntu Installer Workflow

The main task was to update the packages to the latest version (downloading and storing them in dependencies folder), updating bash script accordingly. Several new packages were added namely pycairo and openpyxl.

Some of the packages dropped support for the python2 and the latest version of some packages was not working properly with our source code, those were not updated.

2.2 Creating Debian package

Debian is divided into three distributions: stable, testing and unstable. Whenever a new package is added, or an existing package is updated, it goes into unstable. Once it has been in unstable for ten days without revealing serious bugs, it automatically moves into testing. When the release manager decides it's time for a new release, he declares the testing distribution as frozen. This means that no new packages can be added, and no existing ones may be updated.

2.2.1 Advantages of having Debian packages

1. Debian is one of the primary Linux distributors along with fedora, RedHat, Opensuse. Ubuntu operating system is

a free and open-source Linux distribution based on the Debian.

2. Creating a Debian package means a very wide reach for the software used and reviewed by a large number of people, makes it better.

3. It is easier to create a package for other Linux distributions such as for Redhat RPM with making minimal changes.

4. Debian have a huge community of dedicated open source contributors and have more than 3000 permanent developers, which provide a chance to attract contributor to our project.

5. Adding you package to Debian repository is one of the best ways to get our package added to Ubuntu repository and It updates its repository regularly from Debian repository. Once the package is added to the Ubuntu apt repository it will provide Easier installation with sudo apt-get command.

2.2.2 Programs needed for the development

1. **debhelper -** A collection of programs that can be used in a debian/rules file to automate common tasks related to building Debian packages. Programs are included to install various files into the package, compress files, fix file permissions, integrate your package with the Debian menu system, debconf, doc-base, etc. Most Debian packages use debhelper as part of their build process.

2. **dh-make -** dh-make is necessary to create the skeleton of our package, and it will use some of the debhelper tools

for creating packages. They are not essential for this purpose, but are highly recommended for new maintainers. It makes the whole process very much easier to start and to control afterwards.

3. **fakeroot -** this utility lets you emulate being root, which is necessary for some parts of the build process.

4. **devscripts -** The devscripts package provides a collection of scripts which may be of use to Debian developers and others wishing to build Debian packages.

5. **gnupg -** a tool that enables you to digitally sign packages. This is especially important if you want to distribute packages to other people, and you will certainly be doing that when your work gets included in the Debian distribution.

6. **python setuptools -** Setuptools is a package development process library designed to facilitate packaging Python projects by enhancing the Python standard library distutils. It includes: Python package and module definitions Distribution package metadata Test hooks Project installation Platform-specific details

   Other required tools are already installed with Ubuntu OS so no need to install them separately.

2.2.3 Required files under the debian directory

1. **control -** This file contains various values which dpkg, dselect, apt-get, apt-cache, aptitude, and other package management tools will use to manage the package.

   Architecture: any - The generated binary package is an architecture-dependent one usually in a compiled language.

Architecture: all - The generated binary package is an architecture-independent one usually consisting of text, images, or scripts in an interpreted language.

2. **Build-Depends -** One very important thing to get right is the Build-Depends line in the source package stanza. dh_python2 will correctly fill in the installation dependencies (via python:Depends.

   Add following build dependencies: debhelper (>= 11)

   dh-python

   python-all

   python-setuptools

   If source code supports Python 2, we need to add this line in the source package stanza:

   X-Python-Version: >= 2.7

3. **copyright -** This file contains the name of files, License type (such as MPL-1.1 or GPL-2 or LGPL-2.1), developer name etc. example:

```
Files: src/js/editline/*
Copyright: 1993, John Doe
           1993, Joe Average
License: MPL-1.1
```

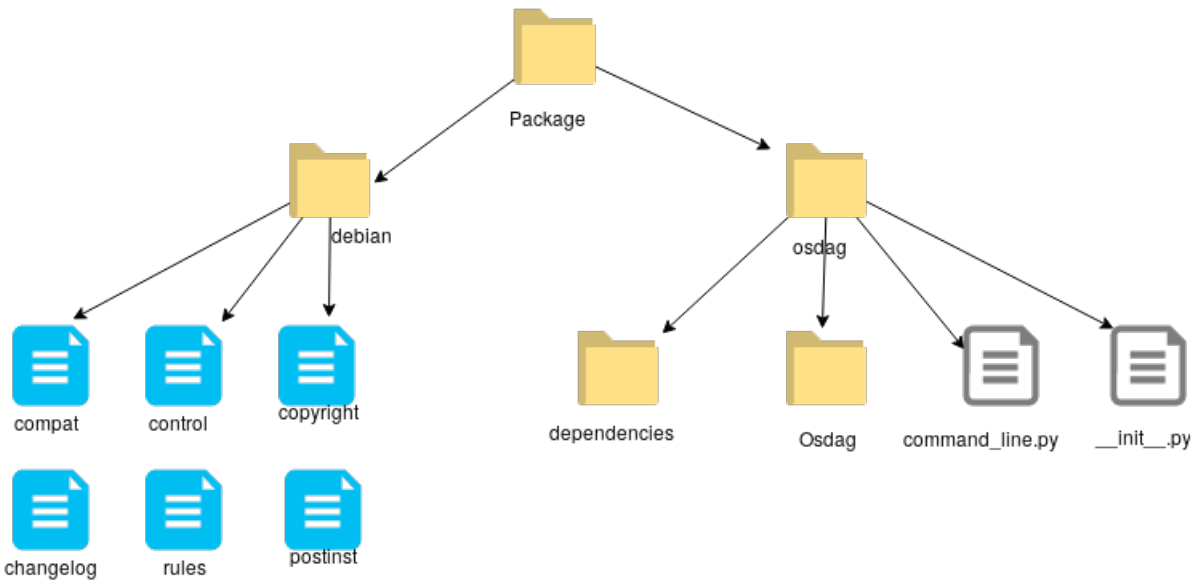
4. **changelog -** This is a required file, which has a special format described in Debian Policy Manual "debian/changelog". This format is used by dpkg and other programs to obtain the version number, revision, distribution, and urgency of the package. If any bug has been fixed then bug fix number and developer name should also be added into it.

5. **rules -** The contents of this file depend on whatever build system one chooses, here we choose python2.

6. **preinst -** It is a script containing a list of tasks to be performed before installation of a package such as installing debian packages, displaying messages, interaction with the user (messages like 'this installation requires 200MB disk space, do you install yes/no?').

7. **postinst -** This script shows messages after installation such as successful installation of failure with possible reasons etc.

8. **compat** The compat file defines the debhelper compatibility level. Currently, we should set it to the debhelper v10. We may use compat level v9 in certain circumstances for compatibility with older systems.



Directory Structure for Debian Package

2.2.4 Building the package

After creating the above files following steps needs to be followed

- Issue the following command in the source directory: This will do everything to make full binary and source packages for us. It will:

  clean the source tree (debian/rules clean)
  build the source package (dpkg-source -b)
  build the program (debian/rules build)
  build binary packages (fakeroot debian/rules binary)
  make the .dsc file
  make the .changes file, using dpkg-genchanges

- Above command will create osdag_0.1.tar.gz, osdag_0.1_all.deb, osdag_0.1.dsc and osdag_0.1_amd64.changes files in top level directory.

  If the build result is satisfactory, sign the .dsc and .changes files with our private GPG key using the debsign command. We need to enter the secret pass phrase, twice.

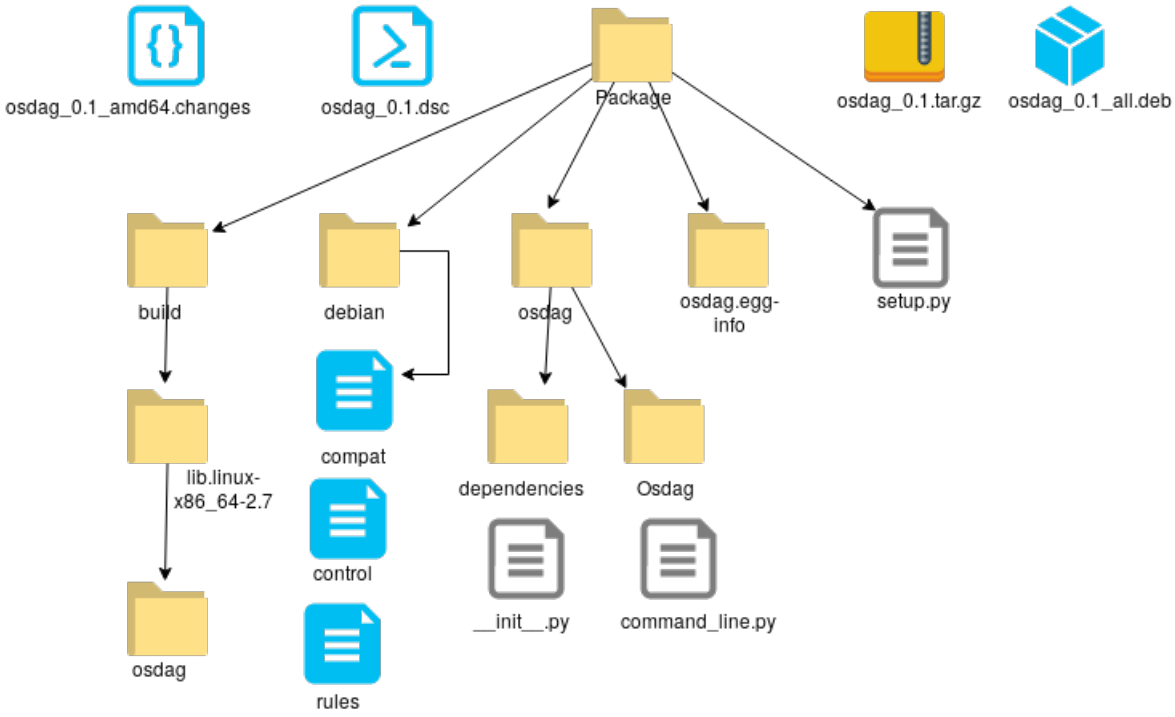
- debuild- We can automate the build activity around executing the dpkg-buildpackage command package further with the debuild command.

- The debuild command executes the lintian command to make a static check after building the Debian package. The lintian command can be customized with the following in the /.devscripts file:

```
DEBUILD\_DPKG\_BUILDPACKAGE\_OPTS="-us -uc -I -i"

DEBUILD\_LINTIAN\_OPTS="-i -I --show-overrides"
```

- debuild -us -uc This command will build the package and other files without signing the package with GPG key.



Folder structure after creating Debian Package

2.2.5 After building package

After building the Debian package it can be directly distributed by distribute.deb file. Alternatively, it can be added to Debian, few steps need to be taken for this.

1. **The DFSG -** An important part of Debian is the DFSG, short for Debian Free Software Guidelines. It is a set of guidelines about the license of a package, and a package goes into different sections of Debian depending on whether its license is DFSG-compatible or not. If it is, it

goes into the Debian main section; if it isn't, it goes into the non-free section; if it is DFSG-compatible but depends on other packages in the non-free section, it goes into contrib.
A package is basically DFSG-compatible if its source code is available, and if everybody is allowed to modify and distribute it; examples of compatible licenses are the GNU General Public License, the BSD License and the Artistic License.

2. **Filing an ITP -** Before starting packaging, we should file an ITP (Intent To Package). That is a special bug report saying that we want to package some product. So the other developers know that we're working on it and will not start working on it, too. One can easily file an ITP using the reportbug tool and specifying wnpp as the package.

3. **Getting a Sponsor and Uploading -** Without being an accepted Debian Developer one cannot upload to the Debian archive directly. We have to package our product and to ask for a sponsor on the debian-mentors mailing list. A sponsor is a (generally) seasoned Debian developer who will check our packages and give hints and support until he thinks they are ready to be accepted into Debian. Then he will upload them and they will be added to Debian.

2.3 Creating PPA for Ubuntu

A Personal Package Archive (PPA) is a special software repository for uploading source packages to be built and published as an APT repository by Launchpad.

Part of the appeal of Ubuntu is its six-month release cycle. Every six months a new version of the free operating system is released into the wild, complete with updates for all of the software. This is great but can be a trifle disappointing from time to time. For example, if a new version of software comes out but users have to wait until the next version of Ubuntu comes out to try it. PPA provides a way to use the latest version of the software as soon as they are released by the developers without waiting for it to be included with Ubuntu six-month update by adding the PPA repository.

```
$ sudo add-apt-repository ppa:package_name
$ sudo apt-get update
$ sudo apt-get install package_name
```

Creating gpg key

GPG is the Gnu Privacy Guard and it is an implementation of OpenPGP (Open Pretty Good Privacy). It is an encryption technique that was originally developed for use in e-mail exchanges that is now used in a number of different applications such as code signing for Linux code repositories and source code repositories like github. OpenPGP is a hybrid of the two-key cryptography approach where the message to be exchanged (called plaintext) is first compressed and then a session key is created as a one-time use secret key. The compressed plaintext is then encrypted with the session key.

We need gpg key to sign the Ubuntu code of conduct to activate our PPA on Launchpad.

1. Open Passwords and Encryption Keys. You can search it by the name seahorse in Ubuntu Applications.

2. Select the My Personal Keys tab, select your key and open the property window by pressing Space Bar or double-clicking with your pointer. Select the Details tab of the property window.

3. Select the Fingerprint text (the ten blocks of numbers and letter). Copy the text by pressing the Ctrl+c keys together. Command line tool can be installed by running the command: sudo apt-get install gnupg

4. Open a terminal and enter:

   ```
   gpg --fingerprint
   ```

   GPG will display a message similar to:

   ```
   pub   rsa2048 2019-06-25 [SC]
   E538 D358 C4FC 4928 E55E  701F 70C5 C579 B0B5 AC17
   uid           [ultimate] Himanshu Singh
   <himanshu16mr13@gmail.com>
   sub   rsa2048 2019-06-25 [E]
   ```

Registering gpg key

1. Highlight and copy public part of fingerprint ie. B0B5AC17 In terminal run following command:

   ```
   gpg --keyserver keyserver.ubuntu.com --send-keys B0B5AC17
   ```

2. Check whether your keys are available on Ubuntu Key server

   ```
   gpg --keyserver hkp://keyserver.ubuntu.com --search-key

   himanshu16mr13@gmail.com
   ```

It will return a key having last 8 characters same as our public key.

3. Register your key at Ubuntu Launchpad website. Next, after matching key with Ubuntu keyserver Launchpad will send email to you at registered mail id with instructions on finishing the process. Hence a PPA will created on Launcpad.

Uploading a package to a PPA

From terminal issue the following command from debian packaging directory

```
dput ppa:your-lp-id/ppa <source.changes>
```

If the package is built successfully, an email notification will be sent.

Chapter 3

Conclusion

On the whole, this internship was a useful experience. I have gained new knowledge, skills and met many new people. I achieved several learning goals, and have moved a step further in achieving other. I got insight into professional practice. Internship has proved to be satisfactory and it has allowed as an opportunity to get an exposure of the practical implementation of theoretical fundamentals.

Here during the internship period I developed my skills in following softwares/tools :

1. Osdag
2. Basics of Python
3. Latex
4. Git and Git hub
5. Tools for communication

Throughout the internship, I found that several things are important:

- **Critical and Analytical Thinking**

To organize our tasks and assignment, we need to analyse our problems and assignment, and to formulate a good solution to the problem. We would have to set contingency plan for the solution, so that we are well prepared for the unforeseeable situations.

- **Time Management**

  As overall project staff and programmer are always racing against tight time line and packed schedule, a proper time management will minimize facing overdue deadlines. An effective time management allows us to do our assignment efficiently and meet our schedules. Scheduling avoids time wastage and allows us to plan ahead, and gaining more as a result.

- **Goal Management**

  It is better to sub-divide the goals to a few achievable tasks, so that we will be gaining more confidence by accomplishing those tasks.

- **Colleague Interactions**

  In working environment, teamwork plays a vital role in contributing to a strong organization. Teamwork is also essential in reaching the goals of the organization as an entity. Thus, communicating and sharing is much needed in the working environment. Therefore, we should be respecting each other in work, and working together as a team, instead of working alone. This is because working together as a team is easier in reaching our targets, rather than operating individually.

- **Tools for communication**

Most of the open-source community uses the following tools for communicating, asking questions and announcements etc. While seeking solutions to some of the problems I learned to use these tools.

IRC - Internet Relay Chat (IRC) is an application layer protocol that facilitates communication in the form of text. The chat process works on a client/server networking model. IRC clients are computer programs that users can install on their system or web-based applications running either locally in the browser or on a 3rd party server. It is like instant messaging but is much older than the platforms we currently use. I used Freenode which is an IRC network used to discuss peer-directed projects.

An IRC client is the vehicle that connects you to the global network of IRC servers. A variety of applications are available, so, whether you are on Windows, Linux, macOS. I used Hexchat for this purpose.

Mailing-list - A mailing list is a collection of names and addresses used by an individual or an organization to send material to multiple recipients. The term is often extended to include the people subscribed to such a list, so the group of subscribers is referred to as "the mailing list", or simply "the list". Mailing lists preceded web forums and can provide similar functionalities.

I learned the difference between top-posting and bottom posting, and why the latter is preferred in the open-source community.

Forums - An Internet forum, or message board, is an online discussion site where people can hold conversations in the form of posted messages. They differ from chat

rooms in that messages are often longer than one line of text, and are at least temporarily archived. I used to ask ubuntu, Launchpad forum, Unix stack exchange, learned about asking questions, providing detailed response etc.

I would like to once again appreciate everyone who has made my internship training a superb experience.