Decision Point Analysis on Learning Process Models in FLOSS mailing Archives

Patrick Mukala, Ph.D

Department of Computer Science
University of Pisa, Italy
{patrick.mukala}@di.unipi.it

Abstract. Numerous studies continue to explore the potential of social interactions between people in Free/Libre Open Source Software (FLOSS) environments. While the dynamics of interactions in these environments can be understood from different perspectives, we put a particular focus on any interactions resulting in knowledge transfer and acquisition. As learning platforms, FLOSS communities provide immense opportunities for improving software engineering skills. People who engage in FLOSS activities both acquire and improve their software development skills. For this reason, it is very helpful to understand how these learning interactions occur. In this paper, we make use of the decision miner in process mining to conduct our analysis. The purpose of such an endeavour is twofold. Firstly, we provide empirical insights into how people learn while exchanging emails in FLOSS mailing archives. Lastly, we go a step further by providing insights behind the motivation into learning participants' decisions on their learning paths.

Keywords: Process-Flow Analysis, FLOSS Data, Educational Data Mining, Learning in FLOSS, Decision Mining, Learning Analytics

1 Introduction

Free\Libre Open Source Software (FLOSS) environments are increasingly dubbed as learning environments where practical software engineering skills can be acquired. Numerous studies have extensively investigated the collaborative model within these environments [3–6,9,11,15,26,27]. Such a collaborative model entails knowledge exchange following a specific learning process [17,18,21].

Most of initial results are produced either through interviews or questionnaires. Given that this information is taken into consideration for the design of hybrid software engineering courses in institutions of higher learning [4,13,24], we believe that it is critical that more investigations be conducted in order to provide a more solid framework that explains how students in FLOSS environments acquire and exchange knowledge. Therefore, it calls for a more empirical approach through analyzing the data generated in FLOSS environments. However, to our knowledge, there has been limited or no attempt in specifically analyzing the data generated in FLOSS environments to this end. In an attempt

to contribute in this direction, [19,20,22] conducted a study by proposing an approach based on process mining. At the heart of this study, it has been observed that the learning process in FLOSS environments occurs in 3 phases: Initiation, Progression and Maturation [19,20,23]. A description of these phases is provided through modeling and process mapping using process mining [19,22].

This description supports that learning participants follow different paths during the learning process. We believe that an analysis of choices of activities could foster the definition of learning models in FLOSS environments by providing critical additional insights related to the decision making. In order to understand why a learning participant, with a specific role (Novice or Expert) chooses one set of activities instead of another, we make use of the decision miner in process mining [25] [1]. The idea is that in a process instance, we identify those parts of the model where the process is split into alternative branches, also called decision points [25]. Based on data attributes associated to the cases in the event log we subsequently want to find rules defining the choice for any of the existing routes.

The primary objective of this endeavour is to provide some additional insights into the choice of learning paths in FLOSS environments. Our understanding of the learning dynamics can be enhanced with a complete description of factors motivating the choices of different learning paths in separate repositories.

The reminder of the paper is structured as follows: Section 2 gives a brief overview of learning processes as well as the FLOSS repositories chosen for our experiment. In Section 3, we briefly set the scope of our analysis and describe the related repositories. In Section 4 we briefly describe the results and discuss these results before concluding in Section 5.

2 Learning Processes in FLOSS Environments

The bulk of reports on FLOSS members' profiling have found that FLOSS members in these communities hold different roles that define their responsibilities and participation in the community activities [7, 8, 12]. These include testers, debuggers, project managers, co-developers and the core developers that make up the core development team. Among these roles, project initiators and the core development team remain at the heart of any development project in the community. This is made up of a small number of developers while the rest of contributors, referred to as the enhanced team, perform additional tasks such as feature suggestions, testing and query handling [12]. Apart from FLOSS participants who play roles with direct impact on FLOSS project, we can also distinguish between passive and active users of FLOSS products. Passive users are observers whose only active role is the mere use of the products. Active users are members of the community who do not necessarily contribute to the project in terms of coding, but whose support is made through testing and bug reporting [7, 8, 12].

As highlighted by Aberdour [2], participants increase their involvement in the project through a process of role meritocracy. This implies that passive users

could move from their state of passiveness to active users, from bug reporters until they possibly become part of the core team [22]. All these roles represent crucial contributions required for the overall project quality. However, in FLOSS environments, moving to a higher state is regarded as a reward and recognition of members' abilities and contributions [2]. Addionally, such role migration is also seen as moving to a higher skill level [11] exemplifying how new skills are developed in these environments.

Hence, it has been found that a typical learning process in FLOSS occurs in three main phases: Initiation, Progression and Maturation [19, 23]. In every phase, a number of activities are executed between Novices and Experts. A Novice is considered as any participant in quest of knowledge while the knowledge provider is referred to as the Expert [16]. Due to constraints related to space in this paper, we illustrate only the Initiation phase as depicted in Figures 1 and 2.

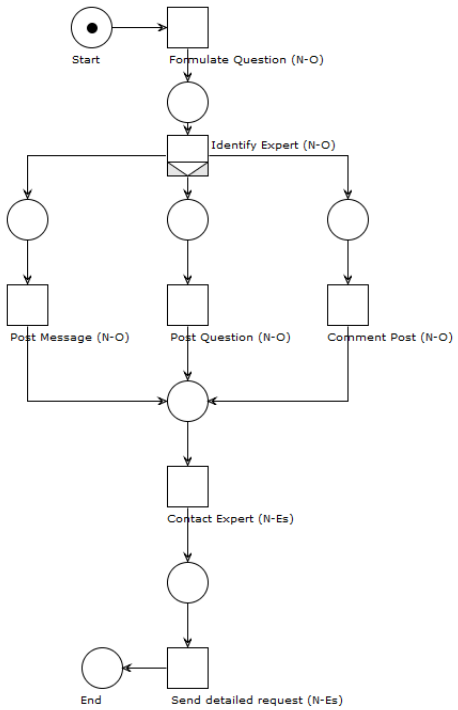

Fig. 1: Learning Process Model for Novice in Initiation Phase

Principal activities gravitate around observing and making contacts in the Initiation Phase of the learning process [19]. Ideally, this step constitutes an opportunity for the Novice to ask questions and get some help depending on the

requests while the Expert intervenes at this point to respond to such requests. On one hand, a Novice seeks help through performing a number of activities. These include FormulateQuestion, IdentifyExpert, PostQuestion, CommentPost or PostMessage, ContactExpert and SendDetailedRequest. On the other hand, the main activities as undertaken by the Expert during the same period of time include ReadMessages on the mailing lists/Chat messages, ReadPost from forums, ReadSourceCode as any participant commits code to the project, or CommentPost, ContactNovice and CommentPost.

3 Decision Point Analysis on Openstack mailing archives and internet relay chats

For the purpose of this experiment, we opt to focus only on the learning process involving the Expert in the initiation phase for illustrative purposes. Learning activities are classified between basic and instance activities. As depicted in Figure 2, the Expert performs a total 6 instance activities in this phase. The first 4 activities are part of the Observation basic activity, while the last 2 occur as the Expert tries to make contact with the Novice through ContactEstablishment. The letter E simply denotes Expert while O and Es respectively denote Observation and ContactEstablishment. We assume that these activities occur tentatively in this order:

1. ReadMessages(E-O) or ReadPost(E-O), CommentPost(E-O) or ReadSource Code(E-O)
2. ContactNovice(E-Es)
3. CommentPost(E-Es)

This succession order of activities simply suggests that the Expert gets involved in the learning process by starting with any of the three activities in (1) and follows the sequencing until step (3). The model offers the possibility of 3 traces as follows:

1. < ReadMessages(E-O), ContactNovice (E-Es), CommentPost(E-Es)>
2. <ReadPost(E-O),CommentPost(E-O),ContactNovice(E-Es),CommentPost(E-Es)>
3. <ReadSourceCode(E-O), ContactNovice (E-Es), CommentPost(E-Es)>

The FLOSS platform used in this analysis is OpenStack [10]. According to Wikipedia, "OpenStack is a free and open-source software cloud computing software plat-form. Users primarily deploy it as an infrastructure as a service (IaaS) solution. The technology consists of a series of interrelated projects that control pools of pro-cessing, storage, and networking resources throughout a data center—which users manage through a web-based dashboard, command-line tools, or a RESTful API that is released under the terms of the Apache License" [28].

We considered this platform mainly due to the availability of data needed for our analysis and also because it is still an active platform. We look at 2

repositories: the mailing archives and internet relay chats on which we can mine learning processes in both the Initiation and Progression phases.

The mailing archives database is made up of 7 tables that store data pertaining to compressed files (source_code file, bugs), the mailing lists as per group discussions and topic of interests, the number of messages exchanged as well as details of the individuals involved in these exchanges. This repository contains exactly 54762 emails exchanged between 3117 people who are registered on 15 different mailing lists. These emails were sent during a period of time spanning from 2010 to 2014. The first message recorded (the very first email sent) was at 10:34:23 on the 11th of November 2010 while the last email considered was sent at 12:16:22 on the 6th of May 2014. The length of the messages considered is of typical email length specifically with an average of 3261 characters, the longest email was of 65535 characters and the shortest message yields a single character length [16] [23].

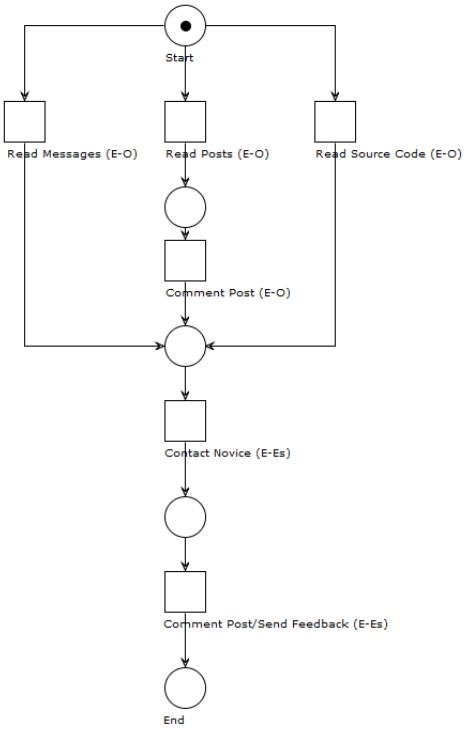

Fig. 2: Learning Process Model in Initiation Phase

We also consider the internet relay chat messages repository. Unlike the mailing archives dataset, this dataset is a database with only 3 tables. These tables contain details of chat messages between conversers; the channels people are

registered to or engage into communication on as well as the details about the people. The channels correspond to the topic of discussions on mailing archives. This repository contains more than 5 million chat messages, exactly 5603302, exchanged between 19247 people on a combined total of 30 channels/forums. From these chats messages, we eliminated those that could not be linked to senders. The final dataset was made up of 2142690 chat messages that we analyzed. Just like emails, these chat messages were exchanged over a period of 3 and half years [16]. The first message was sent on 2010-07-28 at 05:09:11 and the last message we considered was exchanged on 2014-04-09 at exactly 18:07:19. Furthermore, it is fit to mention the significant difference in the message length between the mailing archives and Internet Relay Messages repositories. In this dataset, the average length was 60 characters; the longest chat message reached 502 characters while the shortest message was of single character length [16].

4 Results

The first step of this experiemnt consists on discovering a model. This implies that we apply process mining discovery algorithms to find the actual behaviour as recorded in our derived event logs from the 2 chosen repositories. Although Figure 2 provides a generic representation of how the Expert's paths are anticipated to occur, we have the advantage of using the event log and looking at the actual learning behaviour exhibited by FLOSS participants in Openstack mailing archives and internet relay chat messages. We make use of the inductive miner [14] because of its ability to discover process models from event logs with infrequent behaviours.

One can notice in Figures 3 and 4, respectively depicting the Expert's paths in mailing archives and IRC messages, that decision points can be located and are identifiable through arcs present on the models (Petri nets). At first glance, we can directly note that the Expert takes more routes than the initial representation in Figure 2. The main reason for such a discrepancy can be explained by the volatility of FLOSS environments and specifically, the fact that the role of an Expert is not linked to an individual but rather any entity (participant) executing an Expert's learning activity at any point. In this case, on mailing archives for example, some people could execute only a single activity and disappear, while the situation might be slightly different with internet relay chat messages.

We can now replay the event logs on the discovered models in order to get more insights characterising the choice of individual activities in specific sequences as seen in Figures 3 and 4. After running the experiments with the decision miner, the resulting "decision tree" can be observed in Figures 5 and **??** respectively for the mailing archives and IRC messages. The resulting "decision tree" or rather process data-flow model is dependent on the attributes included in our event logs. In our case, the event log contains details about learning activities, learning state, role etc. Hence, Figure 5 shows that an Expert on mailing archives reads messages during the observation state. Practically,

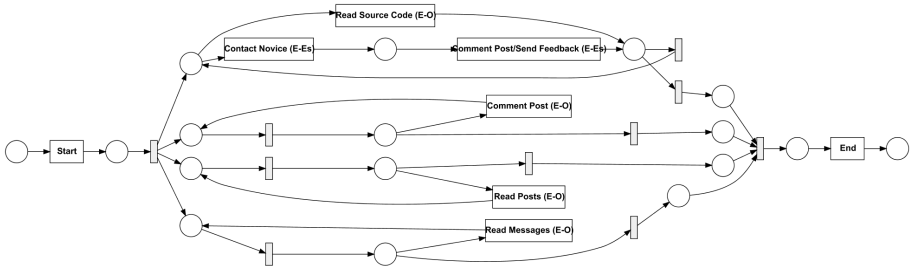

Fig. 3: Discovered Learning Process Model for Expert on mailing archives Repository

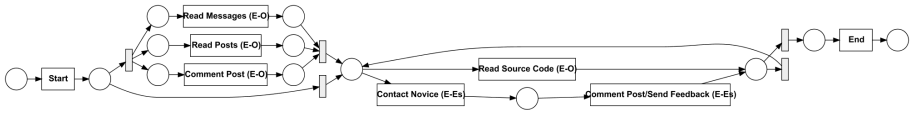

Fig. 4: Discovered Learning Process Model for Expert on internet relay chat messages Repository

the notation reads as follows : "LearningState==Observation, then the Expert would ReadMessages, ReadPosts and ContactNovice". In some instances, when "LearningActivity==ReadPosts, then the Expert would CommentPost". While interacting with the tool, different parameters and guards can be set to drill down and play around with the model as illustrated in Figure **??**. We could not include all of these details as we consider them beyond the scope of this paper.

In Figure **??** , we also chose to include the panel describing some data mining properties pertaining to a decision tree. On the left side of the panel, a confusion matrix is given with related scores that could help the analysts get detailed insights on a specif transitions or part of the model.

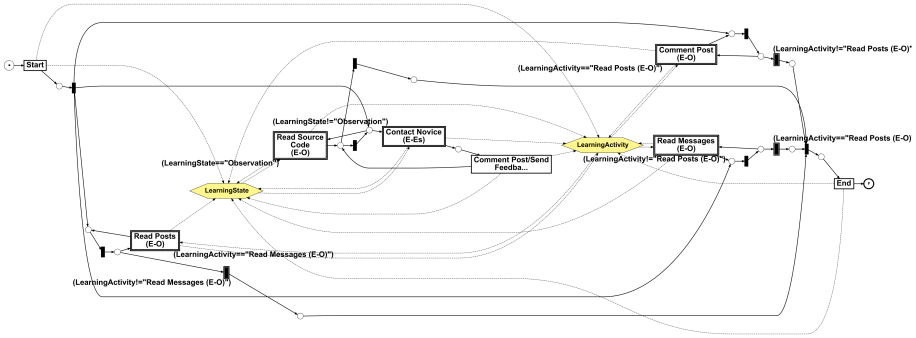

Fig. 5: Discovered Proces Data-Flow Model for Expert on mailing archives Repository

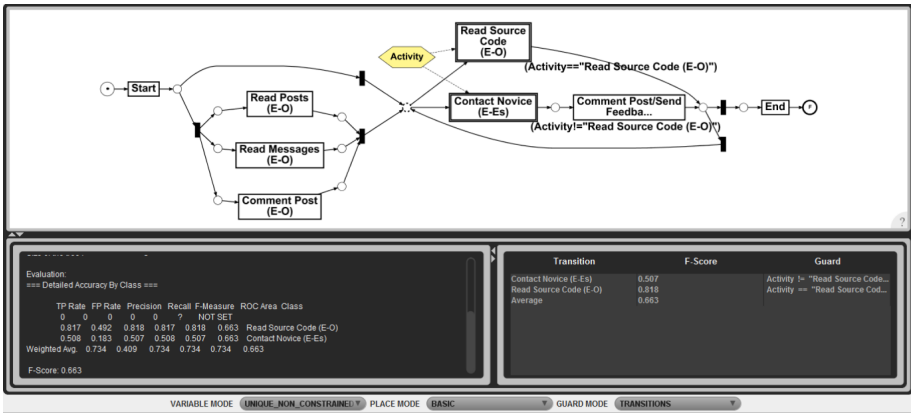

Fig. 6: Discovered Proces Data-Flow Model for Expert on IRC messages Repository with a detailed panel on confusion metrix

5 Discussion and Conclusion

A number of studies on FLOSS environments have laid a foundation regarding the potential for learning occurence in these communities. However, we believe that more could be done in terms of providing supporting empirical evidence and visualizations for learning processes in FLOSS environments. Learning participants follow different learning paths during the learning process and undesratnding this aspect of social interactions provides invaluable insights on learning processes in FLOSS communities.

In this paper, we set to make use of a process mining discovery technique called the decision miner in order to demonstrate how useful analyzing decision points on learning models in FLOSS environments is. For illustrative purposes, we made use of the datasets provided through the Openstack platform and we sought to first discover process models from event logs based on data from mailing archives and internet relay chat messages repositories. We then applied the decision miner to understand the occurence of each learning path on learning models.

The results, although not detailed enough, are indicative of a good prospect in understanding why ceratin activities are considered in certain combinations. Key to such an analysis is the inclusion of additional attributes that might shed lights on specific questions one needs to get answers to. For example, we might want to understand the profile of people undertaking specific combination of activities. Depending on experience, educational background, the choice and type of activities to be performed varies. In our case study, we excluded some of these details in the event logs but we got interesting perspectives based on other attributes such as learningstate and learning activity.

In our future work, we plan to extend this analysis by including additional personal details that can help provide a profile on the types of people involved in learning processes in the context of FLOSS environments.

References

1. van der Aalst, W.M.P.: Process Mining - Discovery, Conformance and Enhancement of Business Processes. Springer (2011)
2. Aberdour, M.: Achieving quality in open source software. IEEE Software 24(1), 58–64 (2007), http://dx.doi.org/10.1109/MS.2007.2
3. Cerone, A.: Learning and activity patterns in OSS communities and their impact on software quality. ECEASST 48 (2011), http://journal.ub.tu-berlin.de/eceasst/article/view/803
4. Cerone, A., Sowe, S.K.: Using free/libre open source software projects as e-learning tools. Electronic Communications of the EASST 33 (2010)
5. Dillon, T., Bacon, S.: The potential of open source approaches for education. FutureLab Opening Education Reports, http://www. futurelab. org. uk/resources/publicationsreports-articles/opening-education-reports/Opening-Education-Report200 (2006)
6. Fernandes, S., Barbosa, L.S., Cerone, A.: Floss communities as learning networks. International Journal of Information and Education Technology 3(2), 278 (2013)
7. Glott, R., Meiszner, A., Sowe, S.: Flosscom phase 1 report: analysis of the informal learning environment of floss communities. FLOSSCom Project (2007)
8. Glott, R., SPI, A.M., Sowe, S.K., Conolly, T., Healy, A., Ghosh, R., Karoulis, A., SPI, H.M., Stamelos, I., Weller, M.J., et al.: Flosscom-using the principles of informal learning environments of floss communities to improve ict supported formal education (2011)
9. Jaccheri, L., Østerlie, T.: Open source software: A source of possibilities for software engineering education and empirical software engineering. In: Emerging Trends in FLOSS Research and Development, 2007. FLOSS'07. First International Workshop on. pp. 5–5. IEEE (2007)
10. Jackson, K., Bunch, C., Sigler, E.: OpenStack cloud computing cookbook. Packt Publishing Ltd (2015)
11. Jensen, C., Scacchi, W.: Role migration and advancement processes in OSSD projects: A comparative case study. In: 29th International Conference on Software Engineering (ICSE 2007), Minneapolis, MN, USA, May 20-26, 2007 [11], pp. 364–374, http://dx.doi.org/10.1109/ICSE.2007.74
12. Krishnamurthy, S.: Cave or community? an empirical examination of 100 mature open source projects. First Monday 7(6) (2002), http://firstmonday.org/htbin/cgiwrap/bin/ojs/index.php/fm/article/view/960
13. LeBlanc, R.J., Sobel, A., Diaz-Herrera, J.L., Hilburn, T.B., et al.: Software Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering. IEEE Computer Society (2006)
14. Leemans, S.J., Fahland, D., van der Aalst, W.M.: Discovering block-structured process models from event logs containing infrequent behaviour. In: Business Process Management Workshops. pp. 66–78. Springer (2013)
15. Meiszner, A., Glott, R., Sowe, S.K.: Free/libre open source software (floss) communities as an example of successful open participatory learning ecosystems. UPGRADE, The European Journal for the Informatics Professional 9(3), 62–68 (2008)

16. MUKALA, M.P.: Process Models for Learning Patterns in FLOSS Repositories. Ph.D. thesis (2015)
17. Mukala, P., Buijs, J.C.A.M., van der Aalst, W.M.P.: Exploring students' learning behaviour in moocs using process mining techniques. Tech. rep., Eindhoven University of Technology, BPM Center Report BPM-15-10, BPMcenter.org (2015)
18. Mukala, P., Buijs, J.C.A.M., Leemans, M., van der Aalst, W.M.P.: Learning analytics on coursera event data: A process mining approach. In: Proceedings of the 5th International Symposium on Data-driven Process Discovery and Analysis (SIMPDA 2015), Vienna, Austria, December 9-11, 2015. pp. 18–32 (2015), http://ceur-ws.org/Vol-1527/paper2.pdf
19. Mukala, P., Cerone, A., Turini, F.: An abstract state machine (ASM) representation of learning process in FLOSS communities. In: Software Engineering and Formal Methods - SEFM 2014 Collocated Workshops: HOFM, SAFOME, OpenCert, MoKMaSD, WS-FMDS, Grenoble, France, September 1-2, 2014, Revised Selected Papers [19], pp. 227–242, http://dx.doi.org/10.1007/978-3-319-15201-1_15
20. Mukala, P., Cerone, A., Turini, F.: Ontolifloss: Ontology for learning processes in FLOSS communities. In: Software Engineering and Formal Methods - SEFM 2014 Collocated Workshops: HOFM, SAFOME, OpenCert, MoKMaSD, WS-FMDS, Grenoble, France, September 1-2, 2014, Revised Selected Papers [20], pp. 164–181, http://dx.doi.org/10.1007/978-3-319-15201-1_11
21. Mukala, P., Cerone, A., Turini, F.: Process mining event logs from FLOSS data: State of the art and perspectives. In: Software Engineering and Formal Methods - SEFM 2014 Collocated Workshops: HOFM, SAFOME, OpenCert, MoKMaSD, WS-FMDS, Grenoble, France, September 1-2, 2014, Revised Selected Papers [21], pp. 182–198, http://dx.doi.org/10.1007/978-3-319-15201-1_12
22. Mukala, P., Cerone, A., Turini, F.: An exploration of learning processes as process maps in floss repositories (2015)
23. Mukala, P., Cerone, A., Turini, F.: Mining learning processes from FLOSS mailing archives. In: Open and Big Data Management and Innovation - 14th IFIP WG 6.11 Conference on e-Business, e-Services, and e-Society, I3E 2015, Delft, The Netherlands, October 13-15, 2015, Proceedings [23], pp. 287–298, http://dx.doi.org/10.1007/978-3-319-25013-7_23
24. Papadopoulos, P.M., Stamelos, I., Meiszner, A.: Enhancing software engineering education through open source projects: Four years of students' perspectives. EAIT 18(2), 381–397 (2013), http://dx.doi.org/10.1007/s10639-012-9239-3
25. Rozinat, A., van der Aalst, W.M.: Decision mining in ProM. Springer (2006)
26. Sowe, S.K., Stamelos, I.: Reflection on knowledge sharing in F/OSS projects. In: Open Source Development, Communities and Quality, IFIP 20th World Computer Congress, Working Group 2.3 on Open Source Software, OSS 2008, September 7-10, 2008, Milano, Italy [26], pp. 351–358, http://dx.doi.org/10.1007/978-0-387-09684-1_32
27. Sowe, S.K., Stamelos, I.G.: Involving software engineering students in open source software projects: Experiences from a pilot study. Journal of Information Systems Education 18(4), 425 (2007)
28. Wikipedia: Openstack — wikipedia, the free encyclopedia (2016), https://en.wikipedia.org/w/index.php?title=OpenStack&oldid=716661364, [Online; accessed 24-April-2016]