An early empirical sentiment analysis of openSUSE Factory Developers and Users

Athanasios-Ilias Rousinopoulos and Gregorio Robles

[1] GSyC/LibreSoft
Universidad Rey Juan Carlos
Madrid (Spain)

athanrous@gmail.com, grex@gsyc.urjc.es

Abstract. *The Factory distribution is the current state of the development for the next openSUSE release (the next release name is 12.3 in March of 2013). The development releases[1] of the openSUSE distribution like Milestones or Release Candidates are snapshots from this distribution. There is constantly work going on in Factory. Every time a developer submits a fix, version update, new feature or new package it is built. So the Factory repository can be in any state and is a moving target. In this paper we analyze the evolution of sentiment (for a 27 month period) of the developers and users who use openSUSE Factory distribution, by mining data from the openSUSE Factory Mailing list*

1. Introduction

In this work we analyze the evolution of sentiment (for a 27 month period) of the developers and users who use openSUSE Factory distribution, by mining data from the openSUSE Factory Mailing list[2]. Our study sample is composed of developers and users who more often posted to the openSUSE Factory mailing list.

1.1. Assumptions

This study has been done under the following assumptions and conditions :

- The period of study is 27 months or 3 major releases of the openSUSE distribution
- The 3rd release period of our study was held back 3 months
- We consider as developers and users of openSUSE Factory (or contributors) the top 10 people who posted most in openSUSE Factory mailing list
- We selected only 10 people because they are the majority (core team) of the people who contributed most this period of time in the openSUSE Factory project. According to "Reductio ad absurdum' [3] method the results and deductions which came from the majority of an analyzed sample they do have validation in the rest of the sample as well.
- The sentiment analysis of each openSUSE Factory contributor is done by extracting a sentiment score for each month of the study period.
- As *sentiment state* of each contributor we refer to the classification of the *sentiment score* for this contributor. The *sentiment state* has 3 categories: positive, negative, neutral (see Chapter 4 for more information).

[1] http://software.opensuse.org/developer/en
[2] http://lists.opensuse.org/opensuse-factory/
[3] http://en.wikipedia.org/wiki/Reductio_ad_absurdum

Finaly as for the methodology part the sentiment analysis that has been applied is a text sentiment analysis being driven by machine learning techniques and affective word lists.

1.2. Goals

As goals of this study we define the following ones :

- Examine if Top 10 people are disappointed or not because of the delay mentioned in 1.1
- Answer to questions like :
  - Which is the most satisfied developer during the last 9 months?
  - Which is the less satisfied developer after a new release?
  - Are developers sentiment characterized by consistency?

1.3. More about Factory

Factory is built in its own *openSUSE:Factory* project on the openSUSE instance of the Open Build Service[4]. This is a huge repository of packages. Development, however, does not happen directly in the *openSUSE:Factory*, but in so-called devel projects. A devel project is a project where development happens for a specific group of packages, like `multimedia`, `GNOME`, `KDE` or `Kernel`. The relation of packages in the openSUSE:Factory project to packages in the devel projects is expressed in the meta data of the packages inside openSUSE:Factory.

Each devel project has its own set of processes, rules and communication channels that fits them best. The reference point for this information is the project description of their Build Service project. Devel projects are also subject to change because the world of FOSS[5] is constantly evolving. Certain software becomes obsolete, standards and defaults change, among others. That means devel projects can change names, get dropped, be newly created, or change content and direction, as can packages in devel projects.

The Factory project follows its own rules and roadmap without disturbing the official openSUSE Release. Apart from building software, contributors and users of openSUSE Factory do have other kind of responsibilities.

2. Related Work

In the last years, researchers have been working with sentiment analysis in many aspects. A majority of the research has been focused on the sentiment analysis in the web. Although there are many cases where scientists do study the multilingual sentiment analysis in more details [1] [2] [3] [4] [5]. In terms of sentiment analysis in the web, scientists focus more on social media sentiment analysis [6], e.g Twitter [7] [8] [9] rather than on more traditional platforms such as forums and mailing lists.

In a study hosted by the National Center for Biotechnology Information in the USA[6], mailing lists have been used for in-depth analysis of clinical messages by using natural language processing methods. Although it does not have a strong relation with sentiment analysis aspect, it is related with natural language processing methods and software (NLTK)[7] , which is the similar method and software used by our study as well.

[4] `http://openbuildservice.org/`

[5] Free/Open Source Software

[6] `http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3236668/`

[7] http://www.nltk.org/

Furthermore it mentions the power and the significance of the mailing lists by characterizing them as *a virtual community of practice that serves as an information hub with easy access to expert advice and opportunities for social networking*. The authors have mined 14,576 messages posted to an Internet mailing list from April 2008 to May 2009 and processed them with NLTK version 2.0.

3. Analysis

3.1. Input/Data Source

We used the openSUSE Factory mailing list archives from July 2010 to September 2012. Mailing list archives are available in mbox format and each mbox file includes messages for one month. In total, we have mined 17,470 messages and analyzed 4,176 messages from 270 mbox files during a 27 month period.

3.2. Data Extraction and Preprocessing model

In order to extract data from the mailing list we used a Mailing lists analyzer software called `MailingListStats`[8] developed by the GSyC/Libresoft research group[9] and available as free software.

`MailingListStats` is a command line based tool that downloads the mboxes in a directory where a database will be created. It stores all the information contained in the e-mails. Extracting the "Top 10 developers" is not enough for our research goal, because we need to analyze the body of the messages for all messages sent by the "Top 10 developers" from 15-06-2010 to 05-09-2012.

As a result we extract the message body text into 270 files (which is the result of having a file per month for each developer and the three period time). After the data extraction, preprocessing of the data is the next step of our model. Preprocessing is very important because the text of the message body differs from text in articles, books or even spoken language. The mined text includes many idiosyncratic uses, such as URLs, terminal commands, Linux distribution names, programming language code, system paths, Linux/Unix terminal commands, packages and repository names. It is necessary to preprocess and normalize the text. n Natural Language Processing practices, after the preprocessing stage, the text is tokenized for later processing.

3.3. Sentiment Model

In this article we study three periods of releases, the period after the openSUSE 11.3 Release until the openSUSE 12.2 Release (see Roadmap[10]). In other words we analyze the openSUSE Factory developers and users sentiments after three main releases. These three periods have not been chosen by chance, but under the fact that the release cycle for the 12.2 release has been postponed for almost 3 months.

Text sentiment analysis can be defined as *a text mining technique to analyze the sentiment of the writer or to the topic written about*. Furthermore sentiment analysis may use machine learning techniques. One often applied method is a naive Bayes classifier[11]

[8] `http://metricsgrimoire.github.com/MailingListStats/`
[9] `http://libresoft.es/`
[10] `http://en.opensuse.org/openSUSE:Roadmap`
[11] `http://en.wikipedia.org/wiki/Naive_Bayes_classifier`

where the algorithm is trained on a labeled data set. Within the Python the NLTK package[12] is a classic sentiment analysis data set analyzer, although general machine learning methods for sentiment classification are implemented as well.

Furthermore for sentiment analysis another method could be used, where word lists are annotated for their frequency and their valence, i.e., whether they are positive or negative. Although defining an opinion lexicon (a list of positive and negative opinion words) for annotating words and sentences seems to be a perfect theory for sentiment analysis, scientists claim that is far from sufficient for accurate sentiment analysis [11].

Our work and model is an 'early' empirical study of sentiment analysis because we only apply machine learning and NLP methods[13]. No other linguist method is being applied [14]. Moreover until now there is no official lexicon to tag computer science words and separate their meaning from the literal one. Apart from the opinion lexicon, a significant part of the sentiment model is the algorithm which has been used to tag and analyze the sentiment of the text files. The algorithm we have contains a list of positive and negative words [14] in the YAML format[15].

3.4. Algorithm

In terms of defining the algorithm which implements the sentiment model , in this section we explain each step of the algorithm :

- We define one (or more) dictionary of words (also called wordlist). The design of the dictionaries highly depends on the concrete topic where you want to perform the opinion mining. For example opinion mining about U.S Elections and opinion mining about the release of the latest Android is different. As a result the positive/negative expressions could be different but the context vocabulary is also quite distinct. In our case we defined one dictionary for positive words and another one for negative words.
- We decide the format of the text we are going to analyze and interact with. As our piece of code interacts with text, splitting, tagging, and extracting information from it there are several ways to define the structure of the text. Concerning the NLP and Tokenazation methods we have many options and ways to analyse the text. In our case we assume the following ones
  - Each text is a list of sentences
  - Each sentence is a list of tokens
  - Each token is a tuple of three elements: a word form (the exact word that appeared in the text), a word lemma (a generalized version of the word), and a list of associated tags.
- As in the previous step we have decided the structural shape of the processed text, we can start writing some code to read, and pre-process this text. With pre-process we mean some common first steps in NLP such as: Tokenize, Split into sentences, and POS Tag.

[12](Natural Language Toolkit), `http://www.nltk.org/`

[13]`http://en.wikipedia.org/wiki/Sentiment_analysis#Methods`

[14]`https://github.com/athanrous/text_sentiment_analysis/tree/master/dicts`

[15]`http://en.wikipedia.org/wiki/YAML`

- The next step is the basic text preprocessing, where the input is the text as a string and the output is a collection of sentences, each of which is again a collection of tokens.
- As we have a collection of sentences and we are using NLTK our forms and lemmas will be always identical. At this point of the process, the only tag associated to each word is its own POS Tag provided by NLTK.
- The next step is to recognize positive and negative expressions. To achieve this, we use dictionaries, i.e. simple files containing expressions that will be searched in our text.
- The recognition of positive and negative expressions is not enough for opinion mining. We have to tag the preprocessed text with the dictionaries defined before. Note that while tagging the text, the input is the previously preprocessed text, and the output is the same text, enriched with tags of type "positive" or "negative".
- The last step is the sentiment measurement of the sentiment tagged text. In our case we count how many positive and negative expressions we detected. For each 'positive' tag we measure it with '> 0' , negative with '<0' and neutral or no text found with '$= 0$' (see Table 1). In order to see the sentiment score of our text we just summarize the negative, positive and neutral tags found in the text.

All the scores for each developer and for each month were stored in a new database with the following columns: name (e-mail address), date, and score. This database was created in order to display and visualize our data. The source code of the algorithm is publicly available[16].

3.5. Display and Visualization

Display and visualization of our data is a significant part of our study. Without displaying the mined data it is impossible to evaluate, to confirm or decline the assumptions, and to extract any piece of information regarding contributors' sentiments. For displaying our data we have used the Python programming language, combined with scientific libraries (Matplotlib[17], SciPy[18]).

4. Evaluation

4.1. Primary evaluation

Evaluation of our data is the next step after the visualization. Table 1 provides information about how the scores have been classified. Positive sentiment means text with score greater than zero, whereas negative sentiment means text with score less than zero. Finally, score equal to zero means neutral sentiment or no messages have been posted by the user during this period of time. The number of messages that each contributor posted in the openSUSE Factory mailing list in Figure3.

Figure 1 provides the evolution of sentiment analysis across the 27 months period of time. Release times are annotated. For the last release (12.2), we included only the messages for the first 5 days of September 2012 as the release date was September 5th 2012, so we are not going to analyze what happened after the last release. The release

[16] https://github.com/fjavieralba/basic_sentiment_analysis
[17] http://matplotlib.org/
[18] http://www.scipy.org/

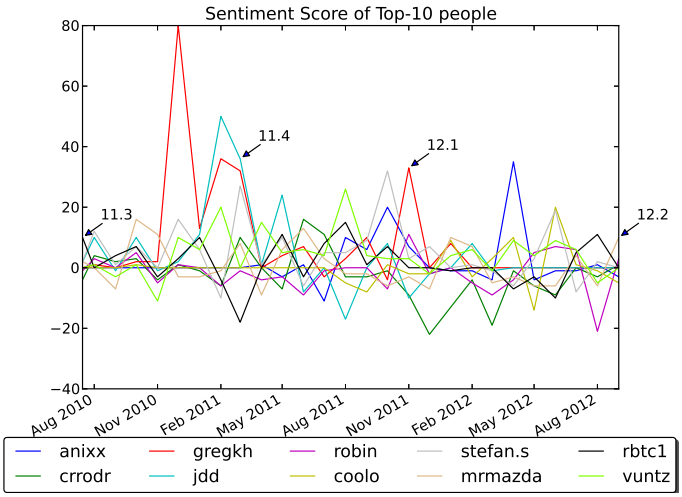

Figure 1. Sentiment score evolution through 3-releases period

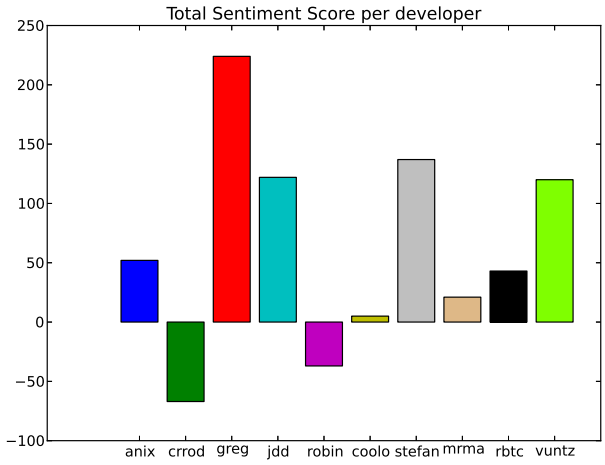

Figure 2. Total Sentiment score per Top-10 person

periods under consideration are thus: 11.3 to 11.4 (first period), 11.4 to 12.1 (second period) and 12.1 to 12.2 (third period).

By focusing and analyzing the evolution of sentiment score for the first period we obviously see that this period is being characterized by a high fluctuation. A majority of developers have a positive sentiment after the 11.3 release (see Figure 4). However, we can also see that only 30% of developers outreach the score of '20', and only one developer [gregkh] gets the maximum score of all at the middle of the first period. Two developers ([jdd] and [stefan.s]) raise their score during the months near to the 11.4 release. On the other hand, we see that [rbtc1] is not satisfied with the 11.3 release, as his scores are below 10 and he shows neutral sentiments during four months. Moreover an interesting a fact for [rbtc1] is shown during 11.4 release, when he gets the lowest score (-18) of all the developers during the first period. We would like to mention that a very low score during a month does not mean that the currently developer is the happiest or

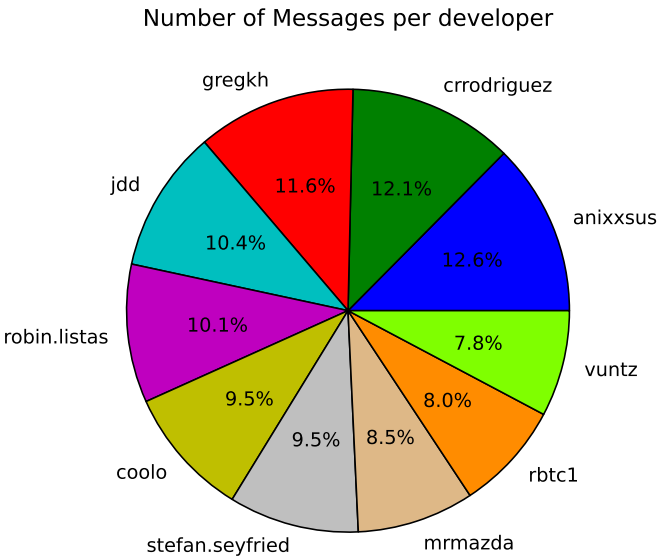

Figure 3. Number of total messages posted in the mailing list per Top-10 person

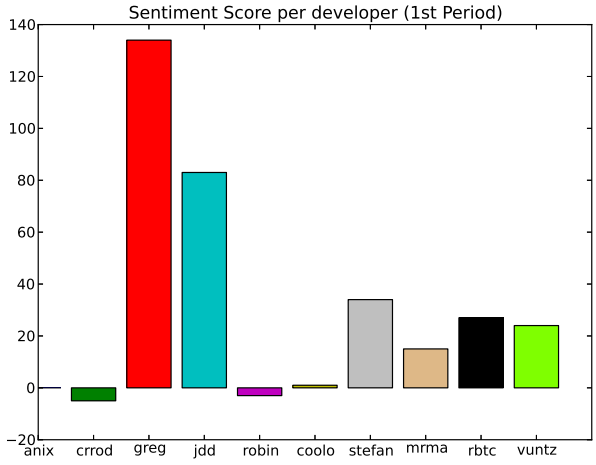

Figure 4. Sentiment score per contributor (1st Period)

most disappointed of all. By having a look at Figure 4 it is obvious that although, for example, he had the lowest score for one month finally turns out that the developer with the most negative sentiment for the first period is [crrod].

As for the second period, a majority of developers have a positive sentiment after the 11.4 release (see Figure 5). Just after the release only 30% of developers ([robin] [mrmazda] [crrodrig]) have negative sentiments. Two of these developers have the same amount of messages posted in the mailing list (132), where the other one ([robin]) has 30% less messages posted in the list. By having a look at Figure 5, we see that [mrmazda] and [crrodrig] do not overate 20 (as a score) and [robin] has the lowest sentiment score

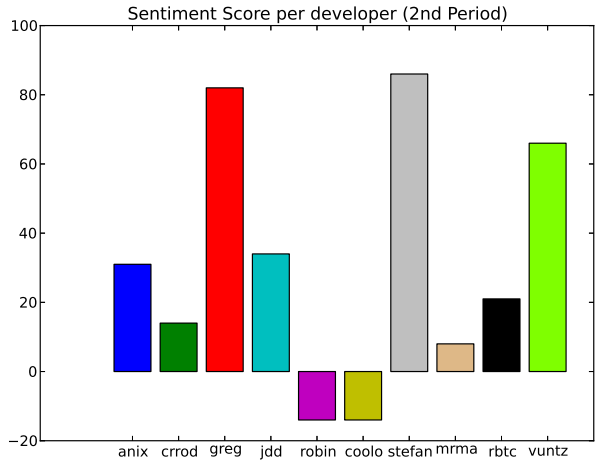

Figure 5. Sentiment score per contributor (2nd Period)

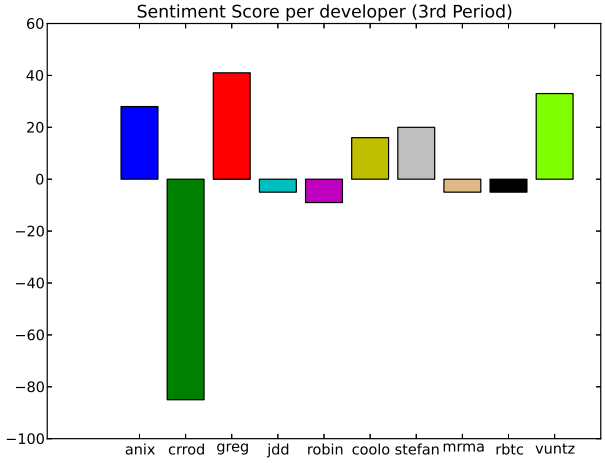

Figure 6. Sentiment score per contributor (3rd Period)

(same as [coolo]) for this period of time. Close to the 12.1 release time (Sept 2011 - Nov 2011), 50% of the developers feel happy as their score increases (we could observe the same effect in the first period), but 40% of developers do have negative sentiments during the release time. Only one developer has neutral sentiments [rbtc]. Furthermore, it has to be mentioned that during the timespan close to the 11.4 release the same amount of developers feel happy about the new release (5 developers), 3 developers have a neutral sentiment and only one seems not be happy for the new release. During this period only 9 to 10 developers are active in the openSUSE Factory list (remind that [anixx] posted 0 messages during the first period). According to Figure 5 70% overate the score limit of 20, and only one has score between 0 to 10 ([mrma]). Under these circumstances, it turns out that the fluctuation of sentiment score is higher than in the first period which means that developers and users felt happier with the 11.4 release.

This study included the third period due to a significant fact: the release date was

Score	Classification
> 0	positive
$= 0$	neutral or no text
< 0	negative

Table 1. Weights assigned to each metric

postponed for 3 months. In this third period, we see that the *delay fact* affected the developers' sentiments. During this period 50% of the developers had negative sentiments. Furthermore, [crrod] gets the lowest sentiment score (-85) for all the 3 period time of study.

However, if we analyze the sentiment score for this period, a total of five developers were sad while the other five were in general terms happy. In the dates close to the release date (July 2012 - September 2012), a propensity to neutrality arises as can be seen by another fact, as by the time of the 12.2 release 60% of the developers do have a neutral sentiments when during the previous releases only 20% and 10% of developers had neutral sentiments. In the meantime only one developer ([anixx]) had a very high score (35). In summary, it turns out that the last period of our study is characterized by negative fluctuation of the sentiment.

4.2. Sentiment prospensity

In section 4.1 we analyzed the total sentiment score of the contributors during the 3 periods of study. With the previous section analysis we have a general aspect of the sentiment score analysis. Although that seems filling our goals defined in Section 1.2 there is a gap that has to be filled out. Until now we now the sentiment score per contributor but we cannot define it more in depth in terms of negativeness and positiveness. For instance we have the following scenario : how can we assure that a contributor A with sentiment score +200 (300 positive score and 100 negative score) has the same feeling with contributor B who has the same score (+200) but with different 'origin' (400 positive score , 200 negative score)? In order to answer to this kind of questions and concern we defined a new term so as to feature with consistency the positiveness and negativeness of the sentiment score. The new term is called "*prospensity to sentiment*". As we measure positive and negative score we also defined two variables related to this term. Concerning the mathematical terms we define the variables as follow:

Prospensity to positive sentiment

$$Pr_{pos} = \frac{\sum_P osScore}{|\sum_N egScore|}$$

Prospensity to negative sentiment

$$Pr_{neg} = \frac{|\sum_N egScore|}{\sum_P osScore}$$

Cases :

$|\sum_N egScore| = 0$ only Pr_{pos} defined

$|\sum_P osScore| = 0$ only Pr_{neg} defined

4.2.1. Sentiment prospensity per period

As in Section 4.2 we defined the term of "sentiment prospensity" in this section we will see the display and visualization of the data concerning "sentiment prospensity". As in Section 4.1 we display the data per period. For each period we display two plots, one plot which visualizes the "prospensity to positive sentiment of the contributors" and the "prospensity to negative sentiment of the contributors"

First period As for the positive prospensity in Figure 7 we see that [greg] is the contributor with the highest Pr_{pos}. Having a look at Figure 4 we can clearly see that [greg] has the highest sentiment score during the same period of time. By examining with more details the Figure 4 and the Figure 7 we see that [jdd] has the second higher Pr_{pos} score and the second higher sentiment score. Under these assumptions we could define a propositional logic [19] between the sentiment score and Pr_{pos} score. Although this assumption seems the perfect way to explain the relationship between these two terms , we can accept it because in refsec:sentpros the scenario explained mention the different origin of the same score. Considering calculation of Pr_{pos} and Pr_{neg} score we have the following results :

Contributor A :

Pr_{pos} = 300/100 = 3

Pr_{neg} = 100/300 = 0.33

Contributor B :

Pr_{pos} = 400/200 = 2

Pr_{neg} = 200/400 = 0.5

We can confirm the non-propositional logic between the sentiment score and Pr_{pos} and Pr_{neg} score by focusing more in the plots related to the First period. For example the sentiment score for [stefan] is 34 and is the third higher score inside all the contributors whereas his Pr_{pos} = 4.4 and comes fourth in the ranking of developers (see Figure 7). Focusing more on the results that the plot represents, only one contributor [anix] has Pr_{pos} = 0. This happens because [anix] has sentiment score 0 the same period of time (see Figure 4 4). As a consequence we expect that the same developer will be caracterized by zero prospensity to negative sentiment as well.

[19] http://en.wikipedia.org/wiki/Propositional_calculus

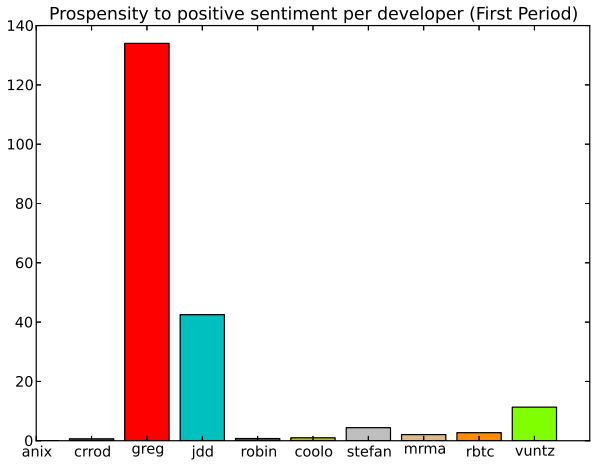

Figure 7. Prospensity to positive sentimet per developer (First Period)

Concerning negative prospensity (see Figure 8) 3 of 10 developers have zero prospensity to negative sentiment. Especially [coolo] has zero prospensity because during the first period he posted messages only for one month [with positive sentiment score]. As a consequence his $Pr_{neg} = 0$. In contrast to [coolo] , [greg] has $Pr_{neg} = 0$ not because of non posting messages but due to the fact that there is no negative score related to [greg] for the first period. It turns out that [greg] has maximum Pr_{pos} during the first period. Furthermore by examining the negative prospensity scores for those 3 contributors , we confirm the scenario defined in Section 4.2 and the definition of the term "prospensity to sentiment". As for the term we see that the score for those 3 contributors is exactly the same but the origin and the reason is totally different in each case.

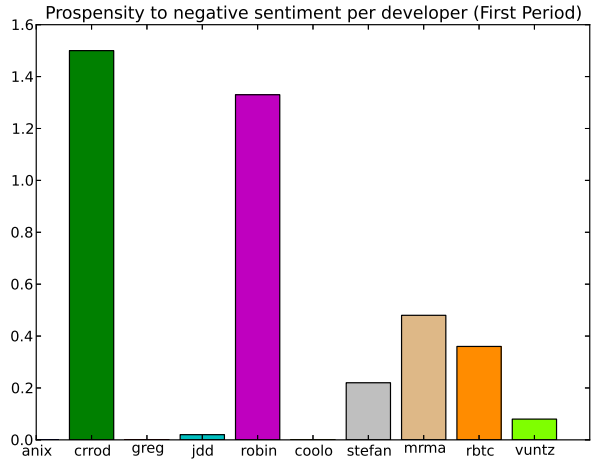

Figure 8. Prospensity to negative sentimet per developer (First Period)

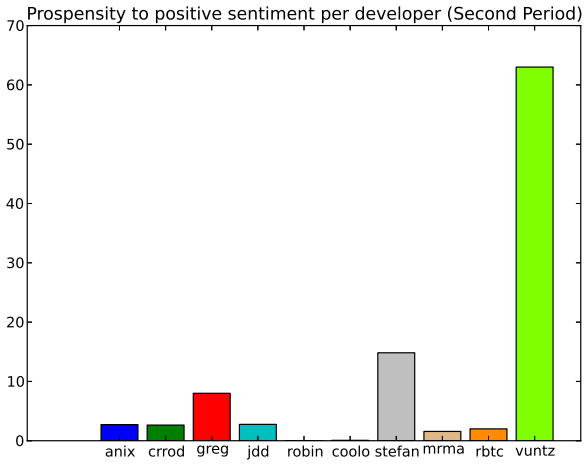

Figure 9. Prospensity to positive sentimet per developer (Second Period)

Second period Following the same approach as in Section 4.2.1 wee see that although [stefan] has the highest ranking in sentiment score, becomes second in the prospensity ranking.In contrast to [stefan] .[vuntz] comes third in the sentiment score ranking and first in the prospensity ranking (see Figure9). Furthermore we see that [robin] has $Pr_{pos} = 0$ because his sentiment score is negative. By examining the rest of the developers we will confirm the definition of the term "prospensity to sentiment".

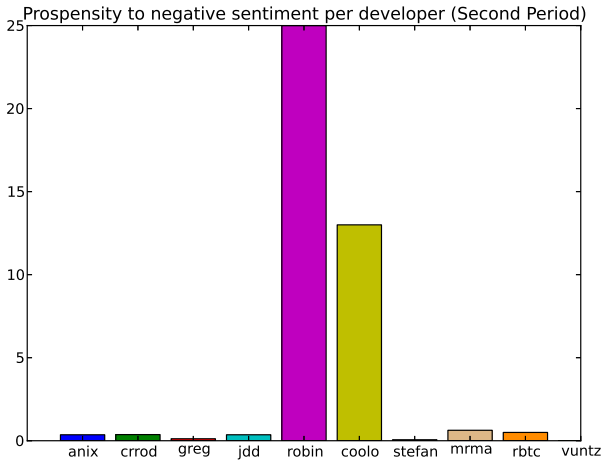

Figure 10. Prospensity to negative sentimet per developer (Second Period)

Having a look at Figure 10 [robin] has the highest ranking in Pr_{neg} and then comes [coolo]. The rest developer's Pr_{neg} does not overate 1 , and in case of [vuntz] it is 0. Turns out that during the second period of our study 80 % of the contributors are satisfied from the release of openSUSE and only 20 % of the contributors couldn't feel satisfied. The analysis of this period of time is a short one because we follow the same criteria and workflow so as to analyze the plots and the results like in the first period.

Third period As the third period lasts 3 months than the other ones we expect changes in the results and in prospensity score.

Concerning the positive prospensity we obviously see in Figure 12 that only [greg] has high score of Pr_{pos} , a truth that we can assume it from the Figure 6 as well. Only [crrod] has Pr_{pos} = 0 as in Figure 6 we can see that he obtained the most negative score among all the developers. In general we can see that 80 % has a low Pr_{pos} [less than 6] which can explain it as a possible dissapointment during this period.

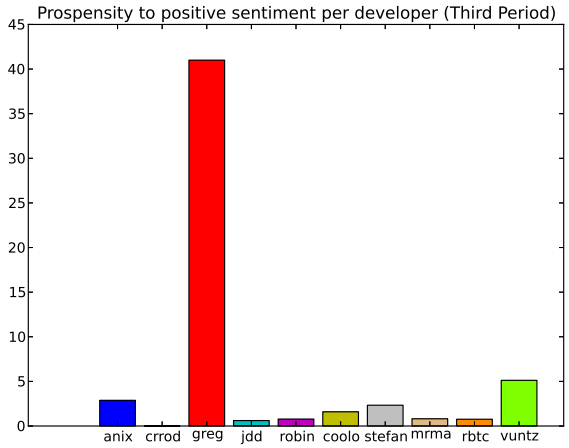

Figure 11. Prospensity to positive sentimet per developer (Third Period)

Although Pr_{neg} seems to be low according to 12 in general , [crrod] obtains the maximum score in Pr_{neg} among all the periods of our study. We could explain this prospensity with many ways. The best way to explain the prospensity is to examine the Figure 1 and concern the section 4.1 deductions.

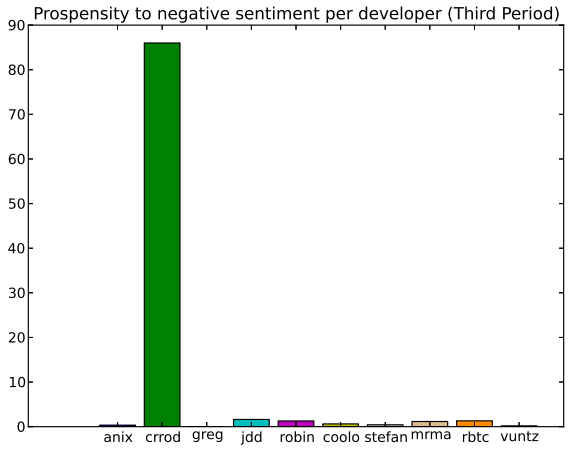

Figure 12. Prospensity to negative sentimet per developer (Third Period)

5. Conclusions

The aim of this study was to analyze the sentiment of the contributors in openSUSE Factory project before, at and after a major software release. We studied three periods of time in order to mine and analyze a larger sample of data. Our analysis and results show that the openSUSE Factory core contributors felt happy nearby and by the time of release (50% of contributors had a positive sentiment). Furthermore after the release, developers' sentiment score reduced by keeping its positive fluctuation although. At least, this is what we have observed during a normal release cycle.

The third period showed us that a possible delay in the release cycle of a distribution has negative affect on developer's sentiment as only 20% of developers were happy by the time of release, showing a 30% reduction in the amount of positive thinking developers who contribute to openSUSE Factory project. Under the assumption that the release lifecycle remains stable for all the release periods that we study, comes into sight that during the first two releases only 10% of the developers would change their sentiment. It turns out that under a stable release cycle model, developers have a propensity to positive sentiment rather than to negative one.

With the definition of the term "prospensity to sentiment" we focused and examined the evolution of the sentiment of the contributors. By following this methodology we are able to define the positiveness and the negativeness of a sentiment score by focusing on "the contributor's sentiment tendency " , which in our opinion will bring to the light better , more concrete results and deductions related to sentiment analysis.

In the near future, we would like to use sentiment analysis on a broader range of projects and situations, to see how usual development circumstances affect the sentiments of the members of a project. Of special interest would be the study of how the sentiments are spread through the community and how measures and methods could be introduced that minimize the effect of negative aspects and maximizes those ones that are positively conceived by developers and users.

6. Acknowledgements

The authors would like to thank the openSUSE community, for their support and comments on some questions raised related to this paper. Ilias-Athanasios Rousinopoulos has done this work being a master's student at the Free Software Master of the Universidad Rey Juan Carlos. The work of Gregorio Robles has been funded in part by the European Commission under project ALERT (FP7-IST-25809) and by the Spanish Gov. under project SobreSale (TIN2011-28110).

References

[1] Brett W. Bader, W. Philip Kegelmeyer and Peter A. Chew: *"Multilingual Sentiment Analysis Using Latent Semantic Indexing and Machine Learning"*. ICDMW '11 Proceedings of the 2011 IEEE 11th International Conference on Data Mining Workshops *Pages 45-52*

[2] Daniella Bal, Malissa Bal, Alexander Hogenboom, Frederik Hogenboom, Flavius Frasincar,Arthur Van Bunningen: *"Sentiment analysis with a multilingual pipeline", 2011*. WISE'11 Proceedings of the 12th international conference on Web information system engineering, *Pages 129-142*

[3] Alexandru-Lucian Gnsc, Emanuela Boro, Adrian Iftene, Diana Trandab, Cenel-Augusto Perez: "Subjectivity and Sentiment Analysis". WASSA '11 Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis. Pages 189-195

[4] Alexandra Balahur and Marco Turchi: *"Multilingual sentiment analysis using machine translation?", 2012*. WASSA '12 Proceedings of the 3rd Workshop in Computational Approaches to Subjectivity and Sentiment Analysis. Pages 52-60

[5] Jordan Boyd-Graber, Philip Resnik: *"Holistic sentiment analysis across languages: multilingual supervised latent Dirichlet allocation", 2010*. EMNLP '10 Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing. Pages 45-55

[6] Georgios Paltoglou and Mike Thelwall: *"Twitter, MySpace, Digg: Unsupervised Sentiment Analysis in Social Media", 2012* . ACM Transactions on Intelligent Systems and Technology (TIST) archive. Volume 3 Issue 4, September 2012 Article No. 66

[7] Younggue BaeHongchul Lee and Anam-dong, Seongbuk-gu: *"Sentiment analysis of twitter audiences: Measuring the positive or negative influence of popular twitterers", 2012*. Journal of the American Society for Information Science and Technology archive. Volume 63 Issue 12, December 2012. Pages 2521-2535

[8] Hassan Saif, Yulan He, Harith Alani: *"Semantic sentiment analysis of twitter", 2012*. ISWC'12 Proceedings of the 11th international conference on The Semantic Web - Volume Part I. Pages 508-524

[9] Hao Wang, Dogan Can, Abe Kazemzadeh, Franois Bar, Shrikanth Narayanan: *"A system for real-time Twitter sentiment analysis of 2012 U.S. presidential election cycle", 2012*. ACL '12 Proceedings of the ACL 2012 System Demonstrations. Pages 115-120

[10] Tanja Bekhuis, Marcos Kreinacke, Heiko Spallek, Mei Song, Jean A O'Donnell: *"Using Natural Language Processing to Enable In-depth Analysis of Clinical Messages Posted to an Internet Mailing List: A Feasibility Study"*

[11] Bing LiuN. Indurkhya and F. J. Damerau:*"Sentiment Analysis and Subjectivity.", Second Edition, 2010*

[12] Minqing Hu and Bing Liu:*"Mining and Summarizing Customer Reviews.", 2004*. Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2004), Aug 22-25, 2004, Seattle, Washington, USA

[13] Bing Liu, Minqing Hu and Junsheng Cheng: *"Opinion Observer: Analyzing and Comparing Opinions on the Web.", 2005*. Proceedings of the 14th International World Wide Web conference (WWW-2005), May 10-14, 2005, Chiba, Japan

[14] Bo Pang: *"Opinion mining and sentiment analysis", 2008*