"Peeling the Onion"

The Words and Actions that Distinguish Core from Periphery in Bug Reports and How Core and Periphery Interact Together

Héla Masmoudi[1], Matthijs den Besten[2],
Claude de Loupy[3,4] and Jean-Michel Dalle[1]
1. Université Pierre et Marie Curie, Paris, France.
masmoudi_hela@yahoo.fr; jean-michel.dalle@upmc.fr
2. University of Oxford, Oxford, UK.
matthijs.denbesten@oerc.ox.ac.uk
3. Syllabs, Paris, France
loupy@syllabs.com
4. University of Paris 10, MoDyCo Laboratory, Paris, France

Abstract. According to the now widely accepted "onion-model" of the organization of open source software development, an open source project typically relies on a core of developers that is assisted by a larger periphery of users. But what does the role of the periphery consist of? Raymond's Linus's Law which states that "given enough eyeballs all bugs are shallow" suggests at least one important function: the detection of defects. Yet, what are the ways through which core and periphery interact with each other? With the help of text-mining methods, we study the treatment of bugs that affected the Firefox Internet browser as reflected in the discussions and actions recorded in Mozilla's issue tracking system Bugzilla. We find various patterns in the modes of interactions between core and peripheral members of the community. For instance, core members seem to engage more frequently with the periphery when the latter proposes a solution (a patch). This leads us to conclude that Alan Cox's dictum "show me the code", perhaps even more than Linus's law, seems to be the dominant rule that governs the development of software like Firefox.

1 Introduction

What is the distribution of labor among members of the core and the periphery of a community engaged in distributed problem solving? We recently found preliminary indications of a stronger importance of the distinction between core and periphery than one might expect: bugs reported by core members are solved more rapidly when more duplicates are found while at the same time the number of duplicates associated with bugs does not seem to affect the speed at which bugs reported by peripheral members are fixed (Dalle *et al.* 2008). In this paper we report further investigations into the topic looking in particular at the relative size, characteristics, actions and

interactions between the core and the periphery for people dealing with Firefox bugs on Mozilla's Bugzilla issue tracking system.

The organization of open source communities has been compared to an onion before and, in fact, the "onion-model" has become widely accepted. Crowston and Howison (2005) sliced the onion – presenting us with a cross-section of open source communities in which one could distinguish a core and periphery of people involved in software development. In this view, members from the core are responsible for the brunt of the development while members from the periphery provide additional services such as spotting problems with the existing code. Raymond (1998) famously compared the role of people in the pheriphery to eyeballs – spotting bugs. However, if the pheriphery's role is to perceive and the core's role is to process, as a brain would do, then the mechanisms available to members of the periphery in order to capture the core's attention become crucially important. We suggest that the place where such problem-solving interaction between the core and periphery of a community like Firefox is most likely to occur is on the pages of its issue tracking system, which, in this case, is Bugzilla. In order to get an impression of what is going on on those pages, we have carried out a number of fishing expeditions – to use yet another metaphor. That is, not content with a cross-section of the onion, we ventured to "peel" the onion separating the actions and words used by the core from the periphery; furthermore, we "chopped" the onion, comparing and "tasting" the parts in order to characterize core and periphery specific actions and words; and finally we "fried" the onion by recoding the bug resolution actions and assessing their effect.

For our investigations, we take advantage of the fact that bug patching in the Mozilla and Firefox communities involves a large amount of textual exchanges, which are archived and publicly available for examination. As far as we know, apart from the interesting work of Ripoche and Sansonnet (2006), text-mining techniques have not really been used on this data archive. Below we show how to construct a corpus for content analysis by extracting bug-reports and discuss what we found using methods such as coding, conceptual interpretation, and text mining. What our findings show substantively, if preliminarily, is that the periphery as we define it mostly engages in "eyeball" activities of noticing bugs, providing contextual elements, and focusing the attention on bugs that might have been forgotten or assumed to be solved; and that its most direct interaction with the core is when it proposes specific solutions to the bugs that were reported.

2 Selection and preparation of bug reports

By selecting only the bug reports which could be traced to comments in the code repository, we make sure only to look at the discussions and interactions that have affected Firefox proper and cut out the noise of unproductive activity that is undoubtedly also going on in the system. In particular, we distinguish three elements in the bug reports. These are action, comments, and affiliations. Actions are things

like changing the status of the bug, assigning it to someone, or including contextual information with the report; comments are messages that people working on the bug convey to each-other via the issue-tracking system; and affiliations are the email-addresses from the people who carry out the actions or make the comments from which their origin as being part of the core or the periphery can be derived.

The investigations described below rely on a variety of corpora that were created on the basis of the bug reports. In order to create these corpora, first a list of bug report numbers was created from the bugs mentioned in commit-comments in a 2007 copy of the Mozilla CVS-code-repository for commits that were linked to files which were part of a Firefox, Firebird, or Phoenix branch. Having identified 37408 bugs this way, the corresponding bug-reports were retrieved from the Bugzilla issues tracker at Mozilla. The bugs span a period of approximately 10 years from first report to last activity. In order to distinguish between core and periphery among the contributors to bug reports, we checked who first reported the bugs and noted the status of the bug at the time of reporting. For bugs that are recognized as "New" straight from the start, we consider the reporter to be part of the core and for bugs that are labeled "Unconfirmed", the reporter is considered to be part of the periphery since apparently he or she did not possess the so-called CanConfirm privilege granted by cooptation (Dalle *et al.* 2008). In subsequent bug reports, the people responsible for actions or comments that have already reported another bug with status "New" previously are considered to be part of the core and coded "N" while people that have most recently reported another bug with status "Unconfirmed" are considered to be part of the periphery "U". A third group for which no core-periphery status can be determined using the method just described is classed as other or "O", but we shall see that "O" has a lot in common with "U" and so for most purposes this group can be considered to be part of the periphery as well.

3 Core Dominance

Our first investigation concerns the overall distribution of labor between core and periphery: how many people can we classify in either group and what is the proportion of activity that can be attributed to those people? Using the global corpus described above, we find that a proportion of 20-25% of the bugs are initiated by outsiders. Of the 6197 distinct email addresses that are associated with the opening of one or more of the 37k bug reports in our corpus, 1713 are marked as insiders and 5118 as outsiders while 634 switch from one to the other. Most of the outsiders only report one bug (3851); 620 report two; 386 report more than two. The numbers of bugs reported by insiders is more evenly distributed and averages about 16 reports. Considering all actions taken and comments contributed to the bugs reports overall, it turns out that about 85% can be traced back to insiders. Moreover, lengthier interactions seem to be associated with a higher involvement from insiders: see Figure 1.

We also plot the frequency of threads with given proportions of insider/outsider involvement (percentage of actions by core and peripheral members of the community). Figure 2 shows an interesting pattern according to which the frequency of threads depends linearly in log scale upon the proportion of insiders, for at least a large part of the spectrum.

When outsiders and insiders interact, they solve exponentially more problems provided the proportion of insiders is higher. It might be so that discussion between outsiders and insiders are easier and often more frequently when they "know" each other. Maybe an appropriate metaphor here could be the tables at a wedding, where discussions seem to be considerably more active at tables where people knew each other ex ante, compared to discussion between strangers, or discussions at tables where many people know each other and there are only a limited number of "newcomers".

4 Words that Core & Peripheral Members Use

Our second investigation is an attempt to qualify the activities of the core and the periphery. In particular, we want to find out whether the words that insiders tend to use are different from the word the outsiders use, hoping that this is indicative for a more general mindset.

Here, we look at a subset of the bug-reports in more detail. We consider two sets. The first, "cvs-sub", consists of 2000 bugs with bug id between 54452 and 73095[1]; the second, "cvs-mile", consists of the 694 bugs in the corpus which were associated with a target milestone specifying a version of Firefox (or its previous incarnations Firebird and Phoenix).

The reports in the corpus were tagged and subsequently partitioned with help of those tags using the following constructs:

- <R = [NUO] > to identify whether the event recorded was originated by someone from the core (N), or periphery (U/O);
- <a = [a-z] > to identify contributor 1-25 (a-y) and subsequent contributors (z) who participated in the bug resolution in order of appearance;
- <stat = {NEW, UNCONFIRMED, ASSIGNED, REOPENED, RESOLVED, CLOSED, …}> to identify the official status of the bug during the resolution process (cf. Bugzilla manual);

The type of analysis we do here can be described as textual content analysis. There are several tools available to perform this kind of analysis. Our tool of choice

[1] Note that these are 2000 consecutive bugs that were traced in the comments of the CVS log; the fact that the difference in bug id leaves space for about 20000 bugs suggests that 90% of bug reports on Mozilla's bugzilla are either resolved without affecting the code base or deal with code that is not related to Firefox.

is called Lexico[2]. We choose this tool because it is particularly helpful in estimating the likelihood of occurrences of words and other items and comparing these frequency-estimates among different parts of the corpus (Lamalle *et al.* 2003). A crucial metric in this type of analysis is what in Lexico is called *specificity*. This metric is an indicator of how specific certain terms are to the parts of the corpus in which they occur. The sign of the metric indicates whether terms are over- or under-employed in specific parts (Lebart and Salem 1994).

Table 1 and 2 give a list of the words that are most specific to comments from the core and the periphery respectively. The words that are identified as being specific to core or periphery are quite telling. The relative importance of words like "*We*" and "*I*" could be interpreted as being indicative of a sense of community where the plural is used or the lack there-of when the singular is prevalent[3]. There is a clear distinction between the people who are dealing with the technical issues under the hood and use the corresponding terms on the one hand and people who approach the black box of Firefox from the outside. Noteworthy as well is the relative importance of "Windows" and "NT" among the periphery. Finally, it seems that outsiders are disproportionately doing the marking of duplicates, which is understandable given that this is administrative work that benefits more to bug reporters than to people who spend most of their time trying to resolve a small number of bugs.

5 Peripheral Activities

Our third investigation delves a bit deeper by exploring in which phases of the bug resolution process the members of the core and periphery are particularly active. For this probe we take the corpus "cvs-sub" described in the previous section and discard all the comment text. We devide the corpus in parts that correspond to each of the stages delimited by the tag <stat =…> and we compare the relative frequencies of participation by people from the core and people from the periphery at each stage of the bug resolution process.

If we take for instance the partition of the corpus according to bug status, then we have seven parts corresponding to the possible modalities of status that a bug can take. In the typical bug status sequence from "Unconfirmed" to "New" to "Assigned", the partition "Unconfirmed" corresponds to the actions that were taken from the moment the bugs had achieved status "Unconfirmed" to the moment that its status was changed into "New". In addition to marking events like status changes and periods of inactivity, we also marked the identify of the person contributing to bug resolution in terms of how many people before him or her had already participated in the discussions and actions relating to that bug. Persons 1 to 25 were assigned letters a to y while z was assigned to all the ultimate latecomers: people

[2] http://www.cavi.univ-paris3.fr/Ilpga/ilpga/tal/lexicoWWW/index.htm

[3] Note that it is possible to use *stoplists* in Lexico in order to avoid dealing with tool words but in this case, tool words are very indicative and we decided to keep them.

with more than 25 other persons adding their bit to the resolution of a bug before them.

Starting with the stages of the bug resolution process as defined by Bugzilla manual, it is of interest to note that there are two peaks of activity for people from the periphery, as reported in Fig. 3. The first is, unsurprisingly, while the bug has status “Unconfirmed”, for this is the point at which the person from the periphery who has submitted a bug makes the case that this is a real bug in order to get it accepted as “New”. The second peak of activity is much later in the resolution process: at the point that the bug is declared “Closed”. A possible interpretation here, in line with other findings presented in this paper, would be that outsiders go on reporting duplicates of already closed bugs. Conversely, it might be so to that they could try to reopen bugs that, according to them were inappropriately closed.

Another interesting result, presented in Fig. 4, concerns the identity of people involved in the discussion. People who enter later in the discussion are less likely to come from the core than earlier on. When discussions around bugs tend to involve an increasing number of people, more people from the core tend to jump in and to contribute.

6 Interactions leading to bug resolution

Our fourth investigation looks at the effect of all activities and comments by members of the core and the periphery. We focus on the activities or groups of activities that distinguish bugs that are resolved at once from bugs that are resolved but then reopened again. Taking the whole corpus of 37000 bugs, we introduce an “alphabetic” coding. That is, we assign letters to represent events. Actions are encoded as follows:

- M – messages in general;
- C – messages signalling the creation of an attachment;
- D – messages identifying a bug duplicate;
- S – other bug status actions;
- A – addition of someone to CC-watch-list;
- Q – QA-contact;
- G - assignment of person who takes the lead in bug resolution;
- R – change in priority;
- V – change in severity;
- X – bug is declared resolved;
- P – attachment is of type “patch”;
- T – attachment is of type “text”;
- I – attachment is of type “image”.

Having this encoding crucially allows us to distinguish between the bugs in the corpus that were resolved just once and bugs that had to be reopened and resolved more than once. This allows us to compare the frequencies of different actions

depending whether bug were resolved only once (X) or more than once (XX), firstly for action unigrams (i.e. the frequency of occurrence, see Table 3) and secondly for the most and least frequent action bigrams (co-occurrences) in these bugs (Tables 4 & 5, respectively). Looking at the unigrams in Table 3, it appears that patches and priority actions are most frequent in X bugs, while duplicates and QA contact actions are least frequent. The data on action bigrams in Table 4 and 5 confirm these observations.

7 Interactions between core and periphery

We now turn to an enhanced alphabetic coding, in which all actions that originated from core members are left in capitals while actions from the periphery are recoded into lower case letters. Table 6 presents the frequencies of action unigrams for core and peripheral members, Table 7 presents their frequencies in relation to X and XX bugs, and Table 8 and 9 respectively present the most and least frequent action bigrams associated with X bugs. Clearly, patches are more frequent in X bugs, as are actions dealing with the priority of the bugs. On the contrary, duplicates are most frequent in XX bugs - an indication perhaps of their complexity - as is the provision of screenshots indicating the need for contextualization. Also changes in severity provided by peripheral members and changes in QA contact provided by core members are more frequent in XX bugs. The action bigrams strengthen the impression that patches are important. Moreover it appears that discussions about QA contact and assignment, and also discussions involving members of the periphery, tend to be associated with the XX bugs, which are declared resolved more than once.

Finally, Table 10 presents the probability of having an action from the core or the periphery following a given action from the core or the periphery. Strikingly, actions from the core are mostly followed by other actions from the core. This observation is less strong for bug duplicates, probably because of sequences of duplicates at the end of many bug discussions, sometimes after they are closed, provided by either core or peripheral members of the community; and for status actions, messages, and actions dealing with the severity of the bug, all of which could serve as entry points for peripheral members, for various reasons.

Similarly, but not less striking, actions from the periphery are generally followed by other actions from the periphery! This is most marked for assignee actions, but this is probably linked to the fact that new assignees act themselves immediately after they take the assignment. A similar feature can be observed for priority actions, which seems to trigger further actions from the periphery, maybe by the same contributor. On the other hand, this property is least marked for duplicates, messages, and severity actions, mirroring the latter observations, but also for QA actions and for patches. It might indeed be that QA actions by peripheral members do trigger a reaction from core members. Consequently, discussions around patches appear as a strong locus of interaction between core and peripheral members of

open-source communities, in that sense perhaps validating Alan Cox's (1998) dictum "show me the code" which stresses the importance of suggesting solutions for outsiders who want to be heard.

8 Conclusion

We believe that the collaboration and interactions between the core and the periphery of the community is an important aspect of open source development. Using econometric techniques we had already established that adherence to core or periphery matters in the case of Firefox (Dalle *et al.* 2008). In this paper, we have presented a cascade of techniques for textual analysis and alphabetic coding in order to shed more light on the interactions between insiders and outsiders in so far as they occur in the context of bug resolution related to Firefox. We found that most of the activities with respect to bug resolution are carried out by a small minority of core members. People in the periphery seem to content themselves with the reporting of bugs and the identification of duplicates while it is the people from the core who develop most of the solutions. There are some indications that contributions from the periphery are ignored. An exception is when these contributions involve "patches." This lead us to think that within Firefox "show me the code" is valued while volunteering to be an "eyeball" is simply taken for granted.

Thus, the explorations in text mining presented in this paper have yielded quite a few interesting if not puzzling observations that globally show that we still don't know enough about the nature and the subtleties of the interactions between core and periphery. Needless to say, the results presented here are still very preliminary and further investigations are still needed, using for instance vector spaces or hidden Markov models.

References

[1] A. Cox. Cathedrals, bazaars and the town council. Available at http://www.linux.org.uk/Papers_CathPaper.cs, 1998.
[2] K. Crowston and J. Howison. The social structure of open source software development teams. *First Monday*, 10(2), February 2005.
[3] J.-M. Dalle, M. den Besten, and H. Masmoudi. Channelling Firefox developers: Mom and dad aren't happy yet. In *Open Source Systems*, Milan, September 2008.
[4] C. Lamalle, W. Martinez, S. Fleury, and A. Salem. *Lexico 3, Outils de statistique textuelle. Manuel d'utilisation.* Université de la Sorbonne Nouvelle, 2002.
[5] L. Lebart and A. Salem. *Statistique textuelle*. 1994.
[6] L. Lebart, A. Salem, and L. Berry. *Exploring Textual Data*. 1998.
[7] E. S. Raymond. The cathedral and the bazaar. *First Monday*, 3, 1998.
[8] G. Ripoche and J.-P. Sansonnet. Experiences in automating the analysis of linguistic interactions for the study of distributed collectives. *Computer Supported Cooperative Work*, 15:149–183, 2006.

Appendix

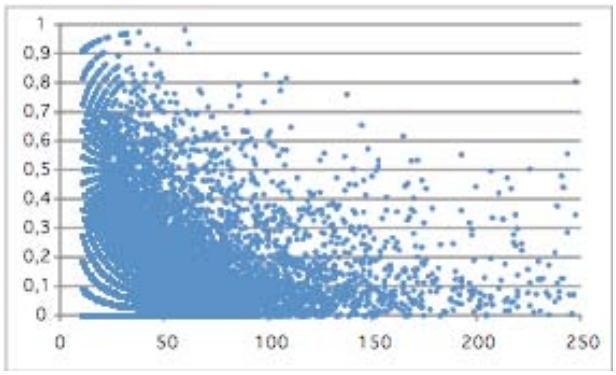

Fig. 1. Proportion of N given length of threads

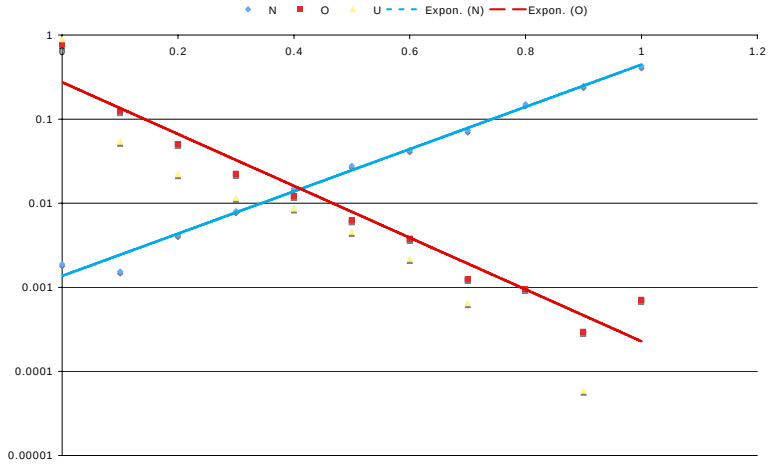

Fig. 2. Number of threads where there's a given % of N and O

Table 1. The words that are most specific for core members (N) and periphery members (O, U) in corpus "cvs-sub"

	N			**O**			**U**	
	Freq	Specif		Freq	Specif		Freq	Specif
We	6933	***	Mozilla	691	***	***	698	***
*	5970	***	*	268	***	Mozilla	382	31
Patch	5584	***	+	260	***	Bug	365	30
+	5009	***	message	304	28	duplicate	349	30

Line	3412	***	5	264	24	marked	347	27
Mozilla	1179	***	my	488	21	URL	174	23
Int	1615	49	mail	260	18	Windows	173	23
Trunk	1405	45	browser	254	17	page	309	20
Const	1447	42	text	348	16	bug	1218	18
Builds	1100	36	with	1200	14	has	600	18
Branch	1081	36	I	3350	13	been	443	16
Bytes	1283	35	page	337	12	text	276	14
Cpp	2508	33	www	173	11	of	1971	13
Fix	3344	32				as	993	13
Details	4614	31						
Checked	1493	28						
unsigned	997	28						
Created	3642	27						
attachment	4772	24						
JS	798	22						

Table 2. The words that are most specific for core members (N) and periphery members (O, U) in corpus "cvs-mile"

	N			**O**			**U**	
	Freq	Specif		Freq	Specif		Freq	Specif
+	3318	***	0	1915	***	***	870	***
attachment	2956	***	5	1276	***	+	179	***
Details	2941	***	Windows	1101	***	Bug	506	31
We	2200	***	Firefox	1080	***	Windows	619	29
0	2147	***	Mozilla	924	***	5	689	28
Mozilla	2142	***	Gecko	627	***	duplicate	451	28
Update	1751	***	U	620	***	marked	440	26
From	1547	***	rv	616	***	source	196	22
Content	1289	***	patch	388	***	Mozilla	484	20
Firefox	1140	***	***	351	***	U	329	20
Revision	1041	***	attachment	329	***	Firebird	181	19
5	931	***	details	320	***	Gecko	332	18
Windows	807	***	mozilla	316	***	has	653	17
Toolkit	778	***	we	272	***	been	540	17
Mozilla	642	***	+	168	***	NT	229	17
Xul	587	***	update	115	***	0	1024	16
Cvsroot	521	***	content	92	***	as	1027	14
Rv	515	***	From	68	***	rv	328	12
Gecko	366	***	toolkit	44	***			
U	323	***	revision	4	***			
NT	213	***	cvsroot	4	***			
Browser	2031	48	NT	398	47			
Done	693	46	I	3731	34			
=	902	45	7	382	30			
Rdf	538	43	my	499	29			

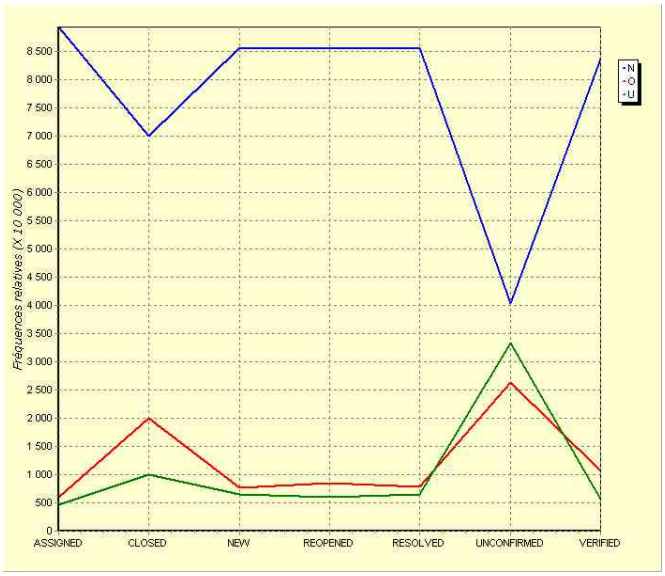

Fig 3. Relative frequencies of actions from core and periphery in different phases of the bug-resolution process

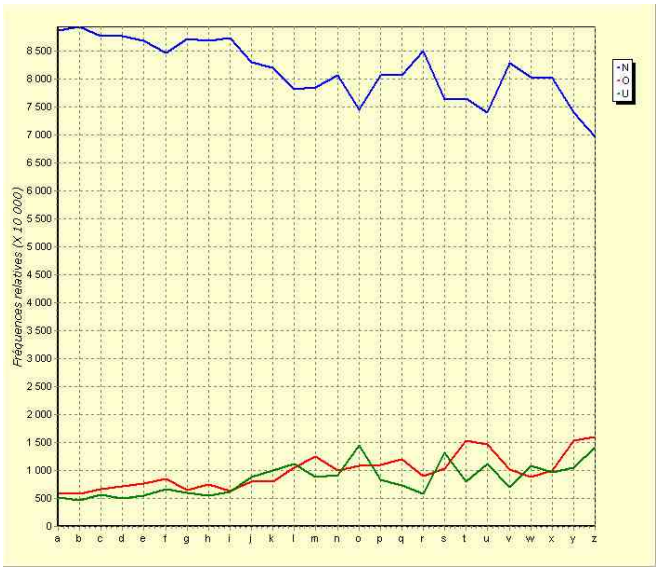

Fig. 4. Frequency that an action stems from someone from the core "N" or periphery "O", "U" for the n-th (coded by alphabetical letters) different actor involved in the bug discussion

Table 3. Frequencies of action unigrams for singly (X) and multiply (XX) resolved bugs

Action	X	XX	Number of occurrences
P	0.882	0.118	61702
C	0.871	0.129	79263
S	0.861	0.139	345335
R	0.856	0.144	10621
M	0.836	0.164	401267
A	0.831	0.169	124808
T	0.827	0.173	11081
G	0.826	0.174	28208
V	0.819	0.181	5317
I	0.813	0.187	3964
Q	0.8	0.2	9529
D	0.759	0.241	20360

Table 4. Most frequent action bigrams for singly (X) and multiply (XX) resolved bugs

Action	X	XX	Number of occurrences
CS	0.894	0.106	35 004
SP	0.890	0.110	22 143
AP	0.888	0.112	5 162
PC	0.882	0.118	59 324
MP	0.877	0.123	26 095
GS	0.868	0.132	5 144
CP	0.867	0.133	5 563
SS	0.866	0.134	117 826
RS	0.862	0.138	9 088
AS	0.861	0.139	10 453
SM	0.860	0.140	118 639
MS	0.854	0.146	158 665
CA	0.852	0.148	7 382

Table 5. Least frequent action bigrams for singly (X) and multiply (XX) resolved bugs

Action	X	XX	Number of occurrences
MM	0.820	0.180	139 161
GM	0.813	0.187	21 425
QS	0.804	0.196	5 368
MG	0.789	0.211	8 209
MQ	0.787	0.213	6 678
AD	0.786	0.214	14 818
DA	0.773	0.227	8 523
DM	0.769	0.231	6 450

Table 6. Frequencies of action unigrams by Core and Peripheral members

Action-Core	Number of occurrences	Action-periphery	Number of occurrences
A	71 443	a	16 986
C	46 363	c	7 325
D	15 159	d	2 607
G	19 927	g	1 690
I	1 880	i	892
M	269 849	m	42 155
P	31 044	p	10 198
Q	7 418	q	959
R	8 035	r	660
S	216 711	s	22 387
T	6 061	t	2 186
V	3 738	v	617

Table 7. Frequencies of action unigrams by Core and Peripheral members in singly (X) and multiply (XX) resolved bugs

Action	X	XX	Total
P	0.868	0.132	30216
p	0.866	0.134	10057
C	0.856	0.144	45373
r	0.854	0.146	644
R	0.839	0.161	7770
S	0.839	0.161	208468
c	0.832	0.168	7148
s	0.829	0.171	21506
g	0.819	0.181	1619
M	0.818	0.182	261032
q	0.813	0.187	938
A	0.813	0.187	70895
t	0.809	0.191	2163
T	0.805	0.195	5940
V	0.801	0.199	3636
a	0.800	0.200	16879
m	0.798	0.202	40612
G	0.795	0.205	19294
i	0.787	0.213	875
Q	0.782	0.218	7204
I	0.774	0.226	1858
V	0.764	0.236	483
D	0.750	0.250	14852
D	0.738	0.262	2552

Table 8. Most frequent action bigrams associated with singly(X) and multiply (XX) resolved bugs

Action	X	XX	Total
CS	0.884	0.116	16 104
pC	0.878	0.122	5 927
SP	0.878	0.122	9 780
PC	0.867	0.133	28 265
MP	0.862	0.138	12 565
RS	0.844	0.156	6 473
SM	0.842	0.158	70 211
SS	0.841	0.159	64 131
CM	0.835	0.165	15 074
MS	0.833	0.167	102 327
MR	0.832	0.168	5 230
AA	0.831	0.169	9 281

Table 9. Least frequent action bigrams associated with singly (X) and multiply (XX) resolved bugs

Action	X	XX	Total
GM	0.798	0.202	16 993
mm	0.787	0.213	6 793
AD	0.782	0.218	10 084
mM	0.778	0.222	9 263
Mm	0.775	0.225	7 927
MG	0.774	0.226	5 430
MQ	0.774	0.226	5 132

Table 10. Frequencies of action from the Core or the Periphery after given Core or Peripheral actions

Action Core	Proba Core	Proba Periphery	Action Periphery	Proba Core	Proba Periphery
A	0.915	0.085	a	0.499	0.501
C	0.942	0.057	c	0.406	0.594
D	0.857	0.135	d	0.523	0.471
G	0.963	0.036	g	0.304	0.695
I	0.958	0.042	i	0.406	0.594
M	0.895	0.069	m	0.505	0.476
P	0.965	0.035	p	0.645	0.355
Q	0.915	0.081	q	0.578	0.419
R	0.959	0.040	r	0.364	0.635
S	0.819	0.068	s	0.447	0.468
T	0.960	0.040	t	0.425	0.575
V	0.896	0.095	v	0.554	0.444