Building Reputation in StackOverflow: An Empirical Investigation

A. Bosu, C.S. Corley, D. Heaton, D Chatterji, J.C. Carver, N.A. Kraft
Replication by Justin Tomlinson

Introduction – StackOverflow is a central location for developers all around the world to get answers to their issues and to solve issues for other developers. The motivation as to why a developer would ask a question on StackOverflow goes without saying. They want an answer. However, what is the motivation for other developers to help and give answers to the questions posted on StackOverflow? The answer comes from StackOverflow's reputation system, which rewards users for being good contributors in the community. Of the 1.3 million users on StackOverflow, only about 443,000 users had answered a question, and only about 6,000 had a reputation score greater than 5000. How does one go about getting a large reputation score and how can it be done quickly? That is the topic of this research paper and the investigation done on data directly from StackOverflow as of August 2012 (MSR postGRESql data dump[4]). We look at four possible paths:

1. Strongest topic areas – How much effort is needed in a preferred area of interest?
2. Most reputed contributors and their impacts in different topic areas – How competitive are different areas of interest?
3. Times of day/week with fewer active contributors – What times are less competitive?
4. Contribution styles of the 10 fastest contributors to earn a reputation score of at least 20,000 – How did the best reputation seekers get high scores quickly?

Methodology – To begin, we define metrics that will be used throughout the investigation.

Accepted Ratio – Percentage of questions with an accepted answer

Unanswered Ratio – Percentage of questions with at least one answer, but none of the answers have been upvoted

No-Response Ratio – Percentage of questions without an answer

First Answer Interval – Time from when a question was posted to when it got its first answer

Accepted Answer Interval – Time from when a question was posted to when it got its first accepted answer

Metric	Original Value	Replication Value
Accepted Ratio	62.21%	62.21%
Unanswered Ratio	21.18%	21.18%
No-Response Ratio	8.69%	8.68%
First Answer Interval	14.98 minutes	9.27 minutes
Accepted Answer Interval	23.57 minutes	31.97 minutes

Note – The replicated first answer and accepted answer intervals are far from the original value because they had to be done using a different dataset, due to issues with the postGRESql data dump.

https://github.com/tomljr2/BuildingReputationInStackOverflow

Areas of Expertise – To create a more abstract overview of areas of expertise in StackOverflow, each tag with more than 10,000 questions is used to create a category of tags. There are 122 tags with more than 10,000 questions and this makes up 86% of all questions asked on StackOverflow. Then using a community detection algorithm program called Gephi [2][3] to cluster tags into different categories using a weighted undirected graph consisting of nodes (tags) and edges with weights based on how often tags are used together and a resolution value of 0.35, we get 14 different categories.

Results

Category	% of Questions	Accepted Ratio	Unanswered Ratio	Top tags
.NET	18.5%	65.0%	19.0%	c#, asp.net, .net
Java	16.1%	58.7%	23.5%	android, java, eclipse
Web	15.2%	64.3%	20.3%	javascript, jquery
LAMP	13.2%	62.1%	21.0%	php, mysql, arrays
C/C++	9.5%	66.5%	13.2%	c, c++, windows, qt
OOP	6.5%	67.1%	15.5%	oop, image
iOS	5.9%	61.6%	24.2%	iphone, ios
Databases	5.5%	67.6%	14.8%	sql, sql-server
Python	4.6%	67.9%	13.9%	python, django, list
Ruby	3.5%	65.9%	20.4%	ruby, ruby-on-rails
Strings	3.2%	72.0%	10.9%	regex, string, perl
MVC	2.0%	68.2%	18.0%	asp.net-mvc, mvc
Adobe	1.2%	57.6%	27.3%	flex, flash, actionscript
SCM	0.8%	68.9%	13.3%	git, svn

Note – Category names are the same as the ones in the original research paper.

An interesting result we can see from the categories is that web development (Web) and object-oriented programming (OOP) have their own categories. Naturally, LAMP should be encompassed by Web and Java, C/C++, Python should be encompassed by OOP. Since it is not, this likely means that web development and object-oriented programming are the most prominent topics on StackOverflow.

Level of Expertise Available in Different Areas – Knowing more about top users in different topics can help us understand how a user can be more successful in gaining reputation. These top users can be assigned one or more of the following badges.

Gold – A total score of 1000+ on 200 or more answers
Silver – A total score of 400+ on 80 or more answers
Bronze – A total score of 200+ on 20 or more answers

We define an "expert" as anyone who has a gold or a silver badge.

Results – We find that among all 1.3 million users on StackOverflow as of August 2012, there are 806 users with at least one gold badge and 1234 users with a silver badge, meaning there are a total of 2040 experts. These experts make up 0.5% of StackOverflow. Despite the small number of experts, they have a very large impact. They make up 29% of all answers on StackOverflow and 32% of all accepted answers.

We also find that the tag, c#, has the most experts, however, it also has the largest number of posted questions. Using a new metric, the Experts-to-Questions ratio, which is the number of experts per number of questions in a tag, we find a better method of evaluating competitiveness in a tag. A tag with a high Experts-to-Questions ratio will be more competitive and a tag with a small Experts-to-Questions ratio will be less competitive. For the sake of gaining reputation, it would be better to contribute to a tag with a small Experts-to-Questions ratio.

Ranking	Least Competitive Tags	Most Competitive Tags
1	flash	scala
2	facebook	r
3	ipad	delphi
4	apache	c#
5	excel	perl

Temporal Efficiency – Having less competition is good for gaining reputation as we have seen. However, there are ways to have even less competition. For us, we will look at the days of the week and the times of the day. Using queries into the MSR supplied postGRESql database, we find the percentage of questions posted, unanswered ratio, and accepted ratio for each hour of the day and each day of the week.

Results – One notable result that can be drawn is that even though questions are posted less often on the weekend, questions are more likely to be answered on the weekends. In fact, the unanswered ratio on Saturday and Sunday is around 3% lower than the rest of the days of the week. This could be useful for reputation seekers, because they should avoid answering questions on the weekends because they will have more competition to answer.

We also find that between 4:00 and 8:00 GMT, the accepted ratio drops below 59% on all days except for Sunday, and the unanswered ratio increases to be greater than 22% on all days except Saturday and Sunday. The research paper defines these as the “low-efficiency hours,” which happen to correspond to 23:00 to 3:00 ET. We also find that the most significant nationality amongst experts is the United States at 40%. This large set of experts may show that the data is skewed to be more reflective of American work hours, and so it is expected that late night and early morning will be less productive on StackOverflow. For a reputation seeker, these are the times that will have a smaller amount of questions, but also much less competition.

Proposed Strategies for Increasing Reputation Score – Among all users, only 1024 users had a reputation score of 20,000 or more. We can use these people to help us understand more about how to gain large amounts of reputation in StackOverflow. However, we also want to know how to do it fast. Instead of using all 1024 users, we just take the 10 fastest to reach a score of 20,000. From this, we can find information that can be helpful to us for gaining reputation.

Results

UID	First Score	Days to 20k	Original Answers	Replication Answers	Original Accepted Ratio	Replication Accepted Ratio	Top tags
938089	2011-10-09	64	489	1962	63.4%	66.2%	javascript, jquery
616700	2011-02-14	73	1004	1068	46.3%	46.2%	c, c++, java, linux
22656	2008-09-26	77	1184	21724	43.1%	57.2%	c#, .net, java
573261	2011-01-12	77	1085	1240	50.1%	50.0%	sql, mysql
224671	2010-01-07	77	895	3245	42.9%	55.3%	iphone, c++
335858	2011-11-20	84	926	2598	41.1%	45.6%	c#, java, c
157882	2009-11-01	85	1245	11935	34.5%	60.9%	java, jsp, html
95810	2009-04-25	85	1143	5613	34.2%	45.5%	python, sql, c++
922184	2011-08-31	91	563	896	46.0%	53.6%	c, c++, java
61974	2009-11-02	95	856	6269	39.3%	50.8%	c#, sql, regex

Note – The original paper created a script to identify the ten fastest users because of the lack of information on when a user reached a score of 20,000. For consistency, I used the same user id (UID) supplied by the paper to get my results.

Interestingly, most of these users average over 11 answers per day, however, the fastest user to reach a score of 20,000, who did it 9 days faster than the next fastest, only averaged 7.6 answers per day. However, this user also has the highest accepted ratio of all other users by a large margin. From this, the quality of an answer tends to be more important for how fast a user reaches a score of 20,000 than the quantity of answers posted. However, even if the quality of answers is not great, one can still reach a high reputation score, simply by answering a lot of questions.

Conclusions – A user looking to gain a large reputation score can do so in a variety of ways. We found that web development, object oriented programming, and .NET encompassed most of StackOverflow, so a user who is proficient in those areas will more likely grow reputation faster. We also find that there is a way to measure less competitive tags using the Expert-to-Questions ratio which provides us with a list of tags to contribute to or avoid for quickly gaining reputation. From looking at the hours and days of the week, we find that it is best to avoid American work hours to have less competition. Finally, we see that posting high quality answers will increase a reputation faster, but posting a lot of answers can increase a reputation score quickly as well. Using these methods that are outlined in this paper, a new user looking to gain reputation can do so quickly and improve the efficiency of StackOverflow as a whole.

References

[1] A. Bosu, C.S. Corley, D. Heaton, D. Chatterji, J.C. Carver, N.A. Kraft, “Building Reputation in StackOverflow: An Empirical Investigation,” 2014.
[2] M. Bastian, S. Heymann, and M. Jacomy, “Gephi: An open source software for exploring and manipulating networks,” 2009.
[3] https://gephi.org/
[4] http://2013.msrconf.org/challenge.php#challenge_data

https://github.com/tomljr2/BuildingReputationInStackOverflow