**AutoTriage(AT)**

**Revision History:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DATE** | **VERSION** | **CHANGE DESCRIPTION** | **AUTHOR** | **Areas Concentrated** |
| 09-Oct-2014 | 1.0 | Initial Analysis/Feasibility study/Initial Prototype | Rengaraj Selvaraj | Initial prototype |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Contents:**

**Currently how it is triaged**

**Inputs from senior resources**

**Time consumed**

**Why automate?**

**Our goal**

**Needs**

**Raw data from different source**

**Additional input prams**

**Prototype modules**

**Scheduler**

**Analytics/metrics**

**Utilities**

**Alerts**

**Countdown**

**Key points**

**References**

**Acknowledgements**

**Currently how it is triaged:**

**Inputs from senior resources:**

1. Experience

2. Error code

3. Agent complexity/type - service provider or custom agent.

4. Bandwidth of developers

5. Reopened bug - extra 2 days from reopened

6. End site response

7. Simple bugs 1 day – e.g.: chase bank migration

8. Complex bugs 403/413 - 5 days blindly

9. Revised bugs - ask the developer

10. DQ - 5 days blindly

**Time consumed:**

5 hours per week

**Why automate?**

1. Consumes time of senior resources

2. Manual task - no accurate results/based on assumptions

3. Dependent on experienced people - experienced people should concentrate on mentoring more

4. Chances of slippage in ETA or bugs being missed out

5. Boring task

**Our goal:**

1. Best accuracy in triaging/ETA

2. Identification of best resource

3. Best load balancing with respect to knowledge, bandwidth and work.

4. Best identification of redundant and duplicate bugs

5. Remove manual task, manual errors.

6. Free senior resources for mentoring and complex bug fixing.

7. Collect meaningful data sets from bugs’ Database

**Needs:**

**We need a very good bug report/template, this solves most of our**

**Problem!!**

**Raw data from different source:**

1. Bug id

2. Open date

3. Assignee

4. Reporter

5. Error code

6. Agent name

7. base/dependent agent name

8. Stack trace

9. White board

10. Resolution

11. Workflow status

12. Many more

**Note: Data is always last 1 year data and analytics is performed on it.**

**Additional input prams:**

1. Lines of code

2. Conditional statements

3. loops statements

4. Worst case scenarios

5. 1-4 on dependent and base classes

6. No. of versions in perforce

7. No of prod certified versions

8. Date of code push

9. End site performance - technical complexity, response/loading time

10. No of reported users, number of failed users.

11. Timeline for clearing similar bugs- similar based on error code/agent involved

12. Average experience of developers/ experience working on the agent bug and error code - this only given as suggestion based on band width of developer and no of bugs already in their plate.

13. If we have a developer who has no worked on certain type of bug but the bug has bigger ETA then - his name will be suggested so that he gets experience on that and the knowledge is distributed.

**Prototype modules**:

Bug DB

Schedule

Normalized data set

Data set creation

Classified data

Data mine raw valid data

Data Mine Raw necessary data

Extract all raw data

Normalized data set

Algorithm Block (score based, fine tune based on score)

|  |  |  |  |
| --- | --- | --- | --- |
| Cost based | Time based | Developer experience | Severity based |
| Priority based | De- risk based | Knowledge distribution based | Tag based |
| Comment/vocabulary based | Text based | Graph based | Resolution based |
| Role based | Band width based | Metadata based | Agent complexity |

Developer score list

Schedule

MLE

Developer name

White board data

Other data

Output final Feed data sets

**Scheduler:**

1. Morning-regular - every morning between 9.00 am to 11 am - every 20 mins.

2. Adhoc - trigger/configure new process anytime by developer, time independent

3. Evening-regular - evening between 7pm to 8pm for reports (manager, reviewer and developer perspective/analytics report) and alerts and EOD status

4. EOD post evening-regular - collection of raw data for analytics

5. Separate process - looks for missed bugs from above (runs between 11.00 am to 7.00 pm, ignores if already whiteboard updated or the bug already in queue of the other processes)

\*The content highlighted in green is not yet complete.

**Analytics/metrics:**

1. Average time, best time, and worst time - developer perspective

2. Count

3. Complexity

4. Nature - site change, technical challenge etc.

5. Predictive and proactive alerts (maintenance window, site change, new implementation, new release etc.)

**Utilities:**

1. Date - (PST, IST custom)

2. Parser

3. Cell data collector

4. Comment analyze

5. Firemen analyze

6. Error code/sum info/site analyze

**Alerts:**

1. Alert mails - time left for TAT, ETA, SLA breach (individual/lead/manager perspective), reports in xlsx, csv, pdf format

**Countdown:**

1. TAT, ETA, SLA

**Key points:**

1. Triage even when Bugzilla is down :) by accessing DB (read only)

2. Reopened bugs triaging i.e. revised ETA

3. Identification of correct team (NBB, hello wallet, outright, lifelock,xero,NSR,DAP,API,sustaining engineering,CS etc.)

4. Why we do not have automated crash reporting tools?

5. Why we do not have capture/replay technique tools?

**Accuracy Target (ETA, Developer name) : – 95%**

**Test cases:**

System performed vs resource performed

**References:**

[1] “Bugzilla,” http://www.bugzilla.org/.

[2] G. Jeong, S. Kim, and T. Zimmermann, “Improving bug triage with bug

tossing graphs,” in Proceedings of ESEC/FSE ’09, 2009, pp. 111–120.

[3] J. Anvik, L. Hiew, and G. C. Murphy, “Who should fix this bug?” in

Proceedings of ICSE’06, 2006, pp. 361–370.

[4] D. Cubranic and G. C. Murphy, “Automatic bug triage using text

categorization,” in Proceedings of SEKE’04, 2004, pp. 92–97.

[5] C. Sun, D. Lo, X. Wang, J. Jiang, and S.-C. Khoo, “A discriminative

model approach for accurate duplicate bug report retrieval,” in Proceedings

of ICSE’10, 2010, pp. 45–54.

[6] P. Runeson, M. Alexandersson, and O. Nyholm, “Detection of duplicate

defect reports using natural language processing,” in Proceedings of

ICSE’07, 2007, pp. 499–510.

[7] X. Wang, L. Zhang, T. Xie, J. Anvik, and J. Sun, “An approach to

detecting duplicate bug reports using natural language and execution

information,” in Proceedings of ICSE’08, 2008, pp. 461–470.

[8] P. J. Guo, T. Zimmermann, N. Nagappan, and B. Murphy, ““not my

bug!” and other reasons for software bug report reassignments,” in

Proceedings of CSCW’11, 2011, pp. 395–404.

[9] ——, “Characterizing and predicting which bugs get fixed: an empirical

study of microsoft windows,” in Proceedings of ICSE’10, 2010, pp. 495–

504. [Online]. Available: http://doi.acm.org/10.1145/1806799.1806871

[10] E. Shihab, A. Ihara, Y. Kamei, W. Ibrahim, M. Ohira, B. Adams,

A. Hassan, and K. Matsumoto, “Predicting re-opened bugs: A case study

on the eclipse project,” in Proceedings of WCRE’10, 2010, pp. 249–258.

[11] N. Bettenburg, S. Just, A. Schr¨oter, C. Weiß, R. Premraj, and T. Zimmermann,

“Quality of bug reports in eclipse,” in Proceedings of tEclipse’07,

2007, pp. 21–25.

[12] N. Bettenburg, S. Just, A. Schr¨oter, C. Weiss, R. Premraj, and T. Zimmermann,

“What makes a good bug report?” in Proceedings of FSE-16,

2008, pp. 308–318.

[13] N. Bettenburg, R. Premraj, T. Zimmermann, and S. Kim, “Extracting

structural information from bug reports,” in Proceedings of MSR’08,

2008, pp. 27–30.

[14] S. Breu, R. Premraj, J. Sillito, and T. Zimmermann, “Information needs

in bug reports: improving cooperation between developers and users,”

in Proceedings of CSCW ’10, 2010, pp. 301–310.

[15] P. Hooimeijer and W. Weimer, “Modeling bug report quality,” in

Proceedings of ASE’07, 2007, pp. 34–43.

**Acknowledgements**

Thanks to below people for providing valuable information on the triaging process and the bug review process

Amarendra Kumar <akumar7@Yodlee.com>

Pratibha Tiwari <PTiwari@yodlee.com>

Puneet Kumar Sharma <[psharma2@Yodlee.com](mailto:psharma2@Yodlee.com)>

Abhishek Agarwal <aagarwal1@Yodlee.com>