Datasys Congress
2021

Prediction Software Quality from Development and Release Factors

Authors: Rishita Mullapudi, Tajmilur Rahman, Joshua Nwokeji

Presenter:
Tajmilur Rahman PhD
rahman007@gannon.edu

Tajmilur Rahman PhD

Tajmilur Rahman, PhD, is an **assistant professor** in the department of Computer and Information Science at **Gannon University** in Erie, Pennsylvania, United States. Dr. Rahman received his doctorate degree in 2018 from Concordia University, Montreal QC, Canada.

His overarching research interest is to **investigate release engineering practices** in software systems. His research interests also include **software engineering** & **data science**, understanding the significance of software architecture for **long lasting software systems**, and providing tool support to the community to nurture **software feature architecture**. He is also interested in studies on software engineering education.

Dr. Rahman is the first author who extracted *feature-architecture*. His currently ongoing research works include, software release management, software quality prediction in rapid-release, predict potential architectural drift, and software engineering education.

rahman007@gannon.edu

Outline

- Software quality
- Importance of software quality
- Quality in terms of post release bugs
- Shorter version release cycles
- Development and release factors
- Data
- Methodology
- Preliminary results

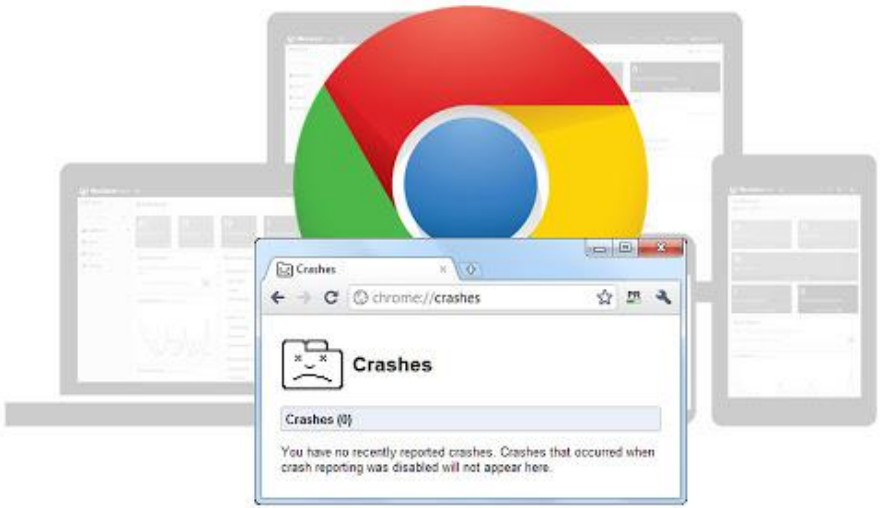
rahman007@gannon.edu

Software Quality

- Functional Quality
- Quality in Performance
- User Experience (UI) Quality
- User Experience - Post Release Issues

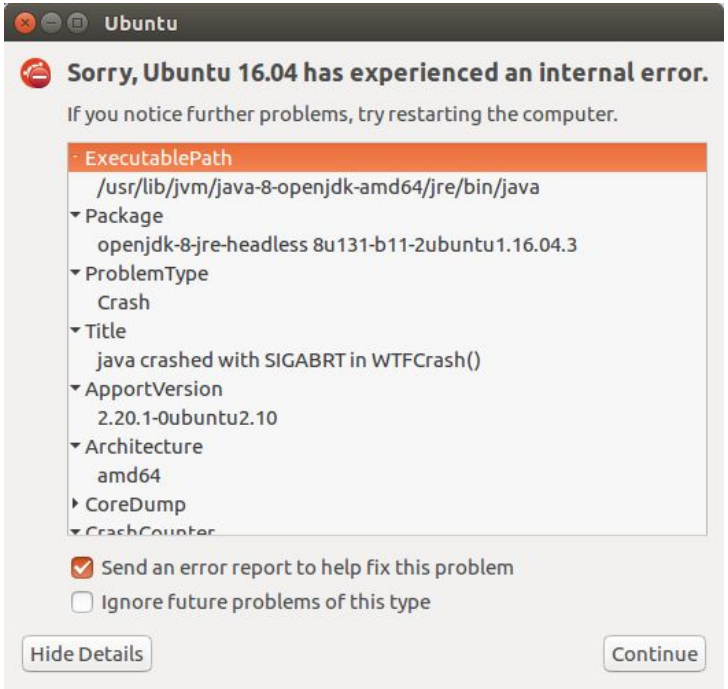
Software Quality - Post Release Bugs

- Software crash on the users' end
- Bugs reported by users
  - Example: Browser crash-report



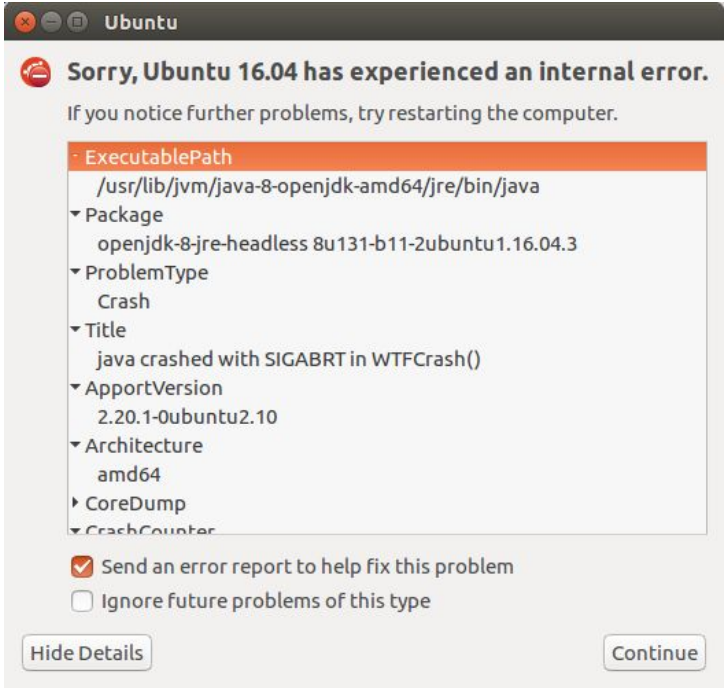
Software Quality - Post Release Bugs

- Software crash on the users' end
- Bugs reported by users
  - Browser crash-report
  - Operating System crash report



Software Quality - Post Release Bugs

- Reports are automatically collected from the end user



Software Quality - Why Important

- Indicates the quality of the released version
- Affects unser satisfaction
- Adds work load into the future sprints
- Additional effort slows down desired pace

Shorter Version Release Cycle

- Promised delivery of new version
- Every certain period of time
- Typically days or weeks
- Limited time for testing
- May lead to more post release bugs

Shorter Version Release Cycle

- Better to predict post release bugs
- Help developers be cautious
- Help prioritizing tests

Development and Release Factors

1. Number of Commits
2. Number of Churn per File
3. Number of Churn per Test File
4. Number of Churn per Configuration File
5. Number of Last Minute Churns
6. Number of Bug-fix Commits

Development and Release Factors

Number of Commits

- Indicates the volume of work done in a release
- Indicates activeness of the developers

Development and Release Factors

Number of Churn per File

- Indicates the volume of work in each file
- Indicates cluster of development effort
  - Amount of work for feature development
  - High churn per - may be a few features received a lot of work
  - Low churn per file - may be enhancement or modifications or small feature

Development and Release Factors

Number of Churn per Test File

- Indicates the volume of testing effort
- Also indicates presence of many features

Development and Release Factors

Number of Churn per Configuration File

- Indicates the volume of configuration effort
- Multiple platform release
- Infrastructure change

Development and Release Factors

Number of Churn at the Last Moment

- Indicates rush among the developers
- Last moment ~ Last one month of changes
- If the last commit before the release commit is May 31, then changes during May 01 - May 31 are considered as last minute changes

Development and Release Factors

Bug-fix Changes

- Amount of changes to fix bugs
- Identified from the commit messages
  - Patterns used: “Bug”, “Fix”, “Patch” etc.

Data

- Bug data
  - Eclipse Equinox bugs
  - https://bugs.eclipse.org/bugs/xmlrpc.cgi
- Historycal repository data
  - Github repository for Eclipse development
  - https://github.com/eclipse/rt.equinox.p2

Data

Version #	commit	# churn	# file	Conf Churn	# LM Churn	# TF Churn	m Bugs	M Bugs
3.0	10k	20k	500	20	10k	200	180	54
3.1	15k	25k	400	12	12k	100	223	78
3.2	12k	22k	450	21	15k	300	124	98

Classification

5 Decision Classes:

- Quality L1
  - Quality magnitude 0 - 5
- Quality L2
  - Quality magnitude 51 - 100
- Quality L3
  - 101 - 150
- Quality L4
  - 151 - 200
- Quality L5
  - 201 +

Quality Magnitude

Indicates the magnitude of post release bugs for a release.

The more high-impact bugs, the higher the magnitude is.

$$magnitudeM = hb \epsilon hb > 0 : hb * (mb + Mb) * 100/Tb$$

- hb <- High-impact bugs
  - Blocking and critical bugs
  - Will be applied only if it is > 0
- mb <- Minor bugs
- Mb <- Major bugs
- Tb <- Total bugs

Quality Magnitude

Magnitude of release version 3.4.0

Critical + Blocking bugs = 19

Minor + Major bugs = 26

Magnitude = 76.7

$$m3.4 = 19 * (26 * 100/644) = 76.7$$

Falls under **Quality Level 2** [51 - 100]

Quality Magnitude

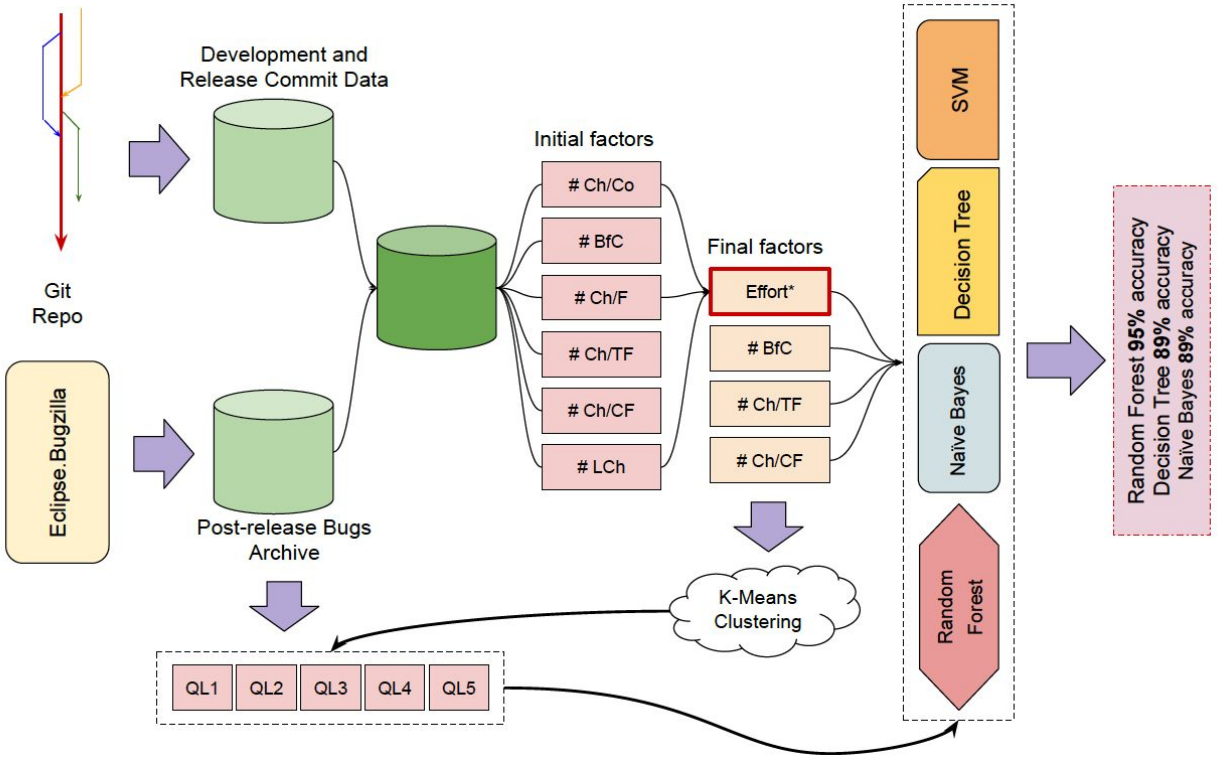
Version #	Min	Maj	Crit	Block	Total Bugs	M
3.4.0	6	20	12	7	644	76.7
3.5.0	14	17	5	8	349	115.0
3.6.0	3	11	5	1	180	47.0

Quality magnitude for release version 3.4.0, 3.5.0, 3.6.0

Classification Models

- Decision Tree
- Random Forest
- Naïve Bayes
- Support Vector Machine

Method At a Glance



In Progress

- 61% of the work is done
- Will apply K-Means clustering to find the classification thresholds
- Will construct the decision tree, random forest and other classification
- Compare their performance

Thank you for Being My Audience

Questions or comments are always welcome

Tajmilur Rahman PhD

rahman007@gannon.edu