Metrics Analysis on Continuous System Test

Sep/28/2016

Rakuten Inc.
Ecosystem Service Department
Delivery and Quality Solution Group
Manager
荻野恒太郎

http://corp.rakuten.co.jp/careers/engineering/

Rakuten

Agenda

- Background
- Metrics
- Analysis ①
- Analysis ②
- Analysis ③
- Conclusion

Rakuten

Agenda

- Background
- Metrics
- Analysis ①
- Analysis ②
- Analysis ③
- Conclusion

Rakuten

Background①：Change in Development Process and System Test

- From Waterfall to Agile

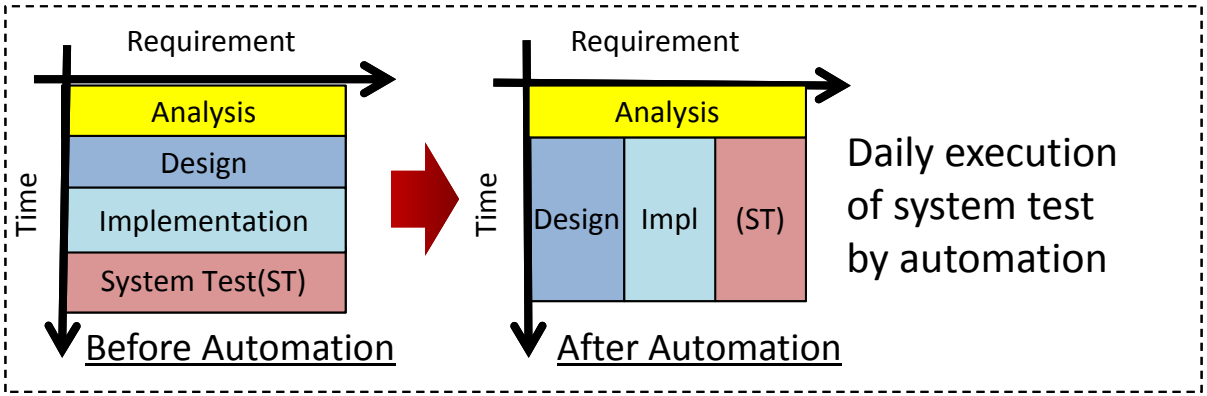
  Hiranabe Kenji, ”Role of Agile at the turning point of Software Engineering” SS2010.

- System Test Execution from early phase of the project

  Nagata Atsushi, “Approach of System Test in QA organization for Agile Development”, JASPIC2013.

- Continuous System Test

  Kotaro Ogino, ”Development process improvement by System Test Automation” JaSST’ Tokyo 2014.



Rakuten

Background②:Advantage of Continuous System Test

Myth in Test Automation

- Tradeoff relationship among quality, cost and delivery
  → Based on the assumption, independent QA from development process

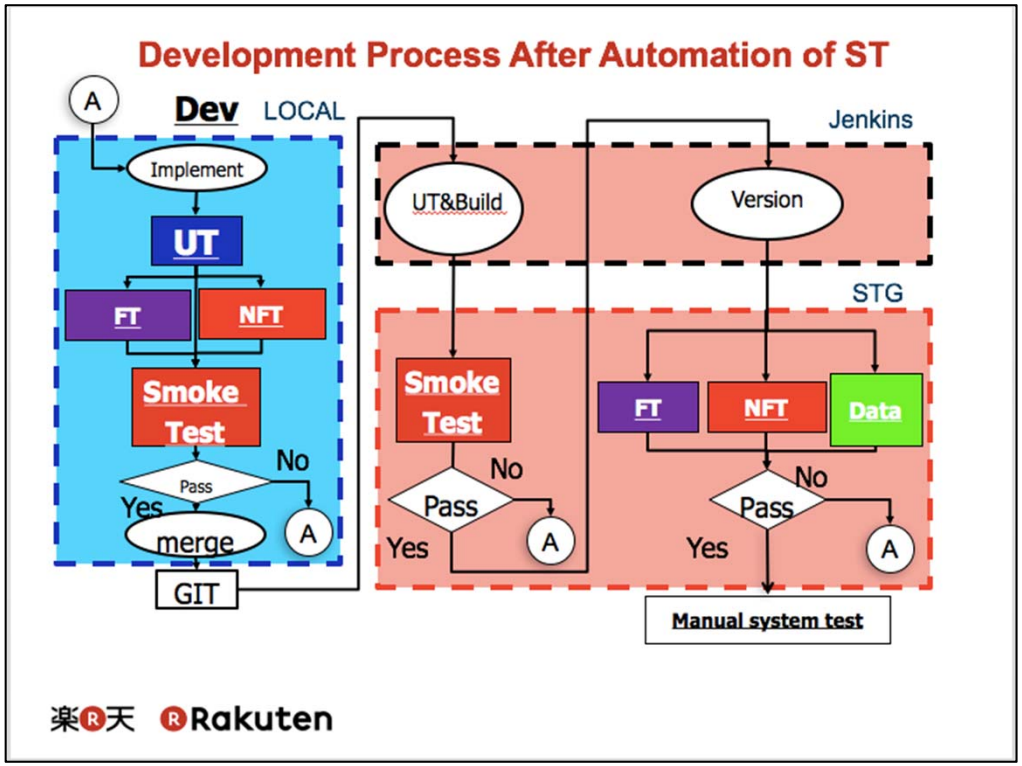
Continuous System Test

- Include system test in development process by automation

Rakuten

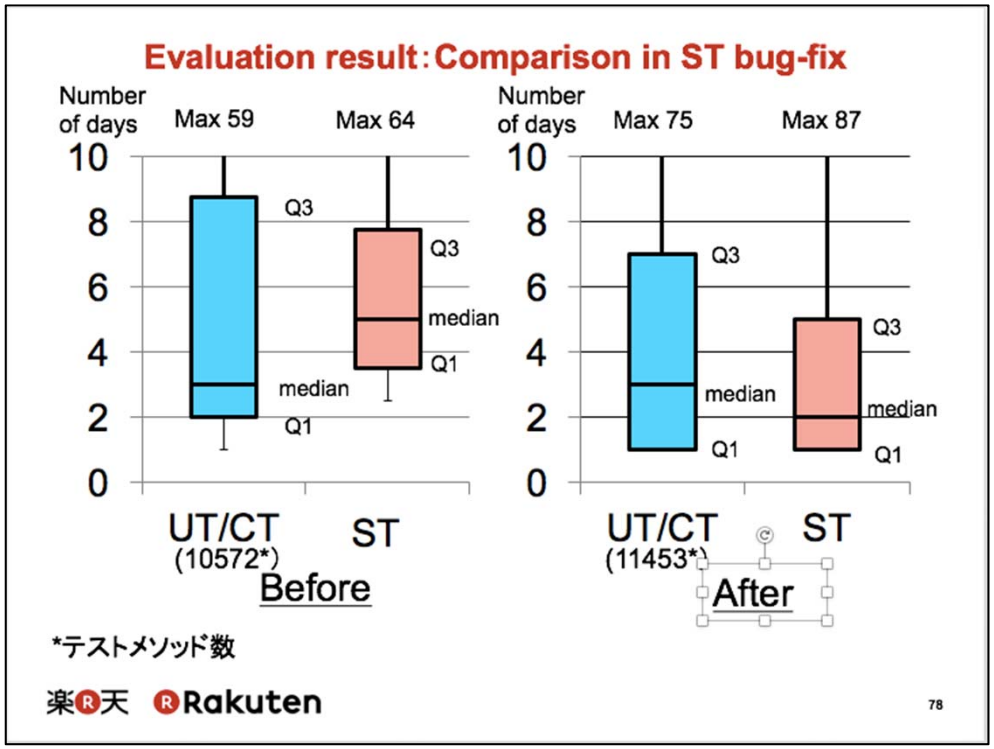
Background②:Advantage of Continuous System Test

System test in development process



Background②:Advantage of Continuous System Test

Bug-fix is improved.



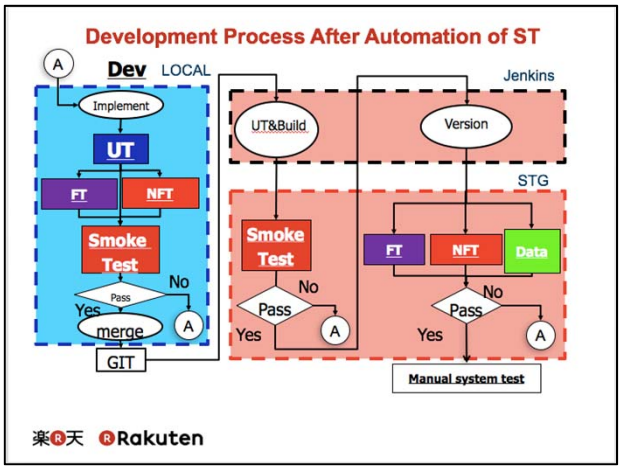
Background②:Advantage of Continuous System Test

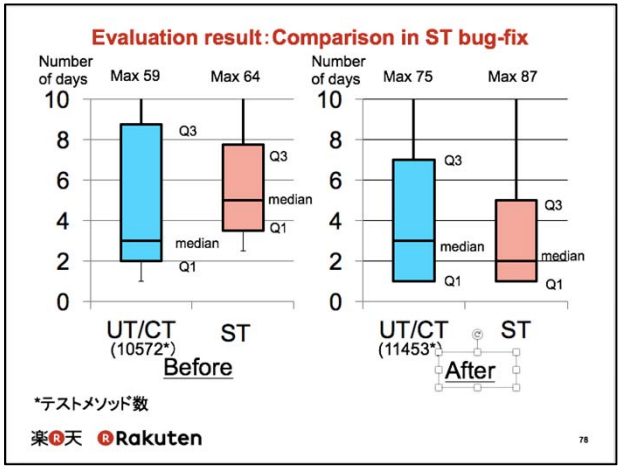
Myth in Test Automation

- Tradeoff relationship among quality, cost and delivery
  → Based on the assumption, independent QA from development process

Continuous System Test

- Include system test in development process by automation





Improvement in Bug fix = Improvement in Cost and Delivery

Objective and Approach of this report

Question for System Test

Q1: Is automated system test low quality?

Q2: How is system test in development process?

Q3: Any technique for better development?

Objective: To understand Dev and ST under continuous system test environment.

Approach: Metrics analysis

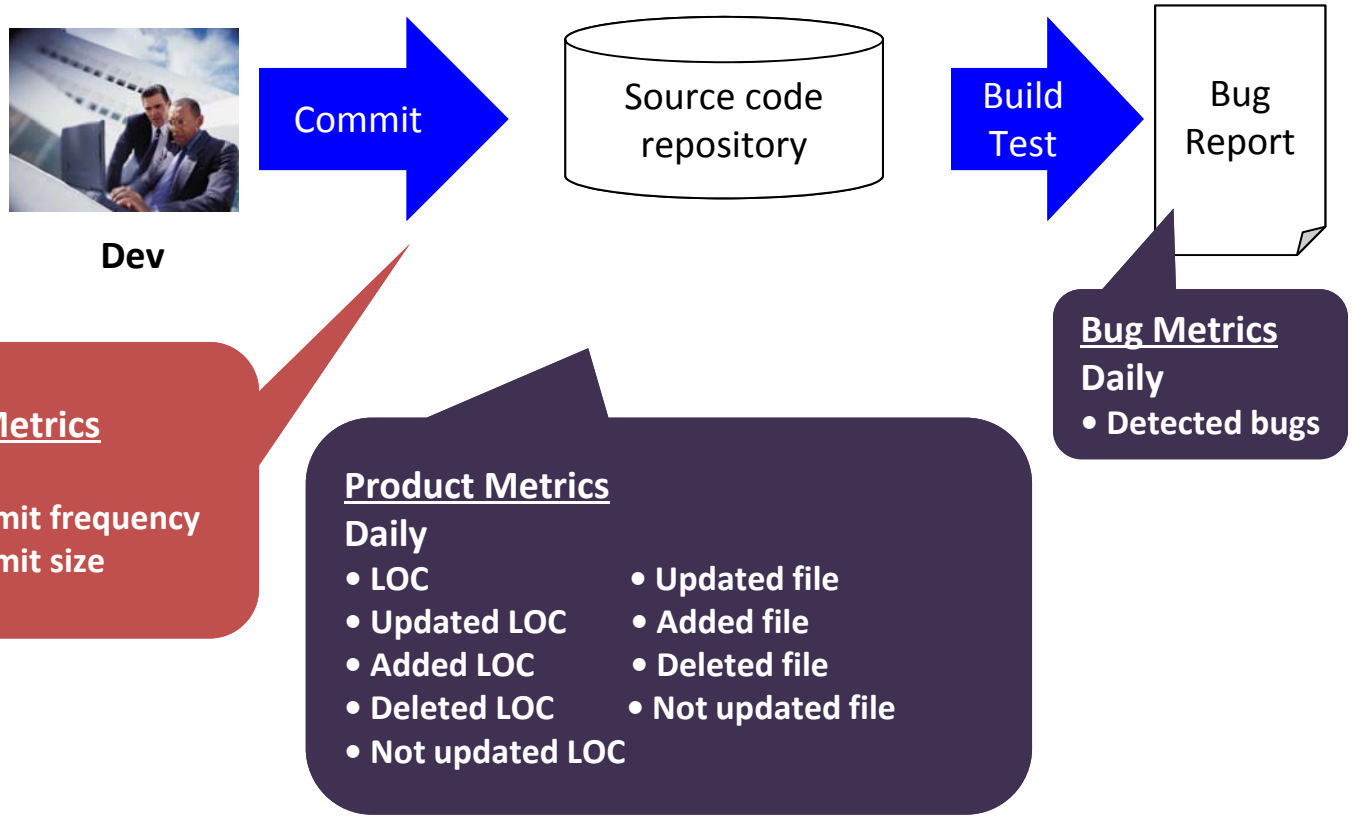
Rakuten

Agenda

- Background
- Metrics
- Analysis ①
- Analysis ②
- Analysis ③
- Conclusion

Rakuten

Metrics



Rakuten

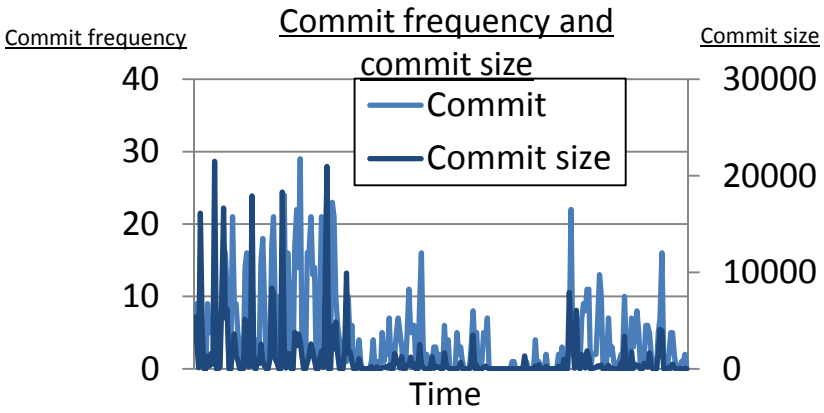
Metrics ollection methods

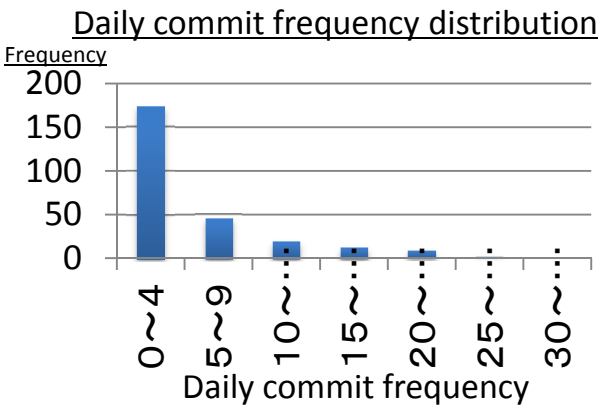
Group	Metrics	Collection method	Unit
Dev metrics	Daily commit frequency	git log (*1)	Number of times
	Daily commit size	git log (*2)	Number of lines
Product metrics	Daily LOC	cloc (*3)	Number of lines
	Daily updated LOC Daily added LOC Daily deleted LOC Daily not-updated LOC	cloc –diff (*4)	Number of lines
	Daily updated file Daily added file Daily deleted file Daily not-update file	cloc –diff (*4)	Number of Files
Bug metrics	Daily detected bugs	- Bugs detected in ST -Measured in created date of bug ticket - Duplicated bugs are deleted	Number of times

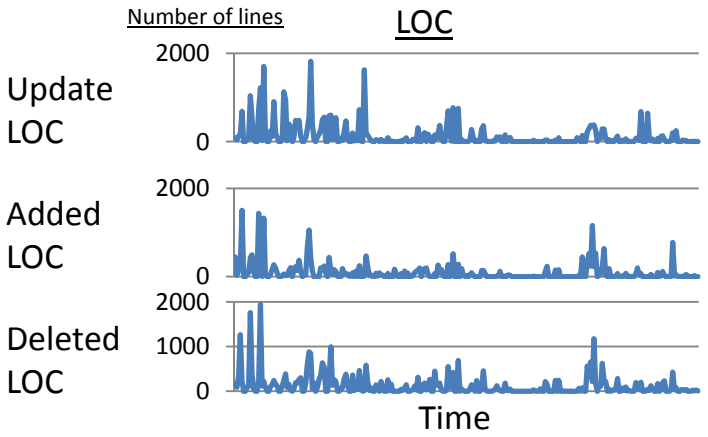
(*1) https://www.atlassian.com/ja/git/tutorial/git-basics#!log
(*2) Include the comments etc
(*3) http://cloc.sourceforge.net/
(*3) (*4) Java language, not including comments etc.

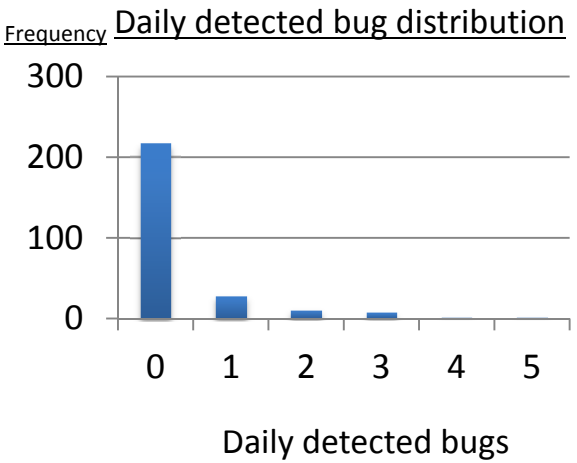
Rakuten

Measured metrics (2013 1/28~10/23)



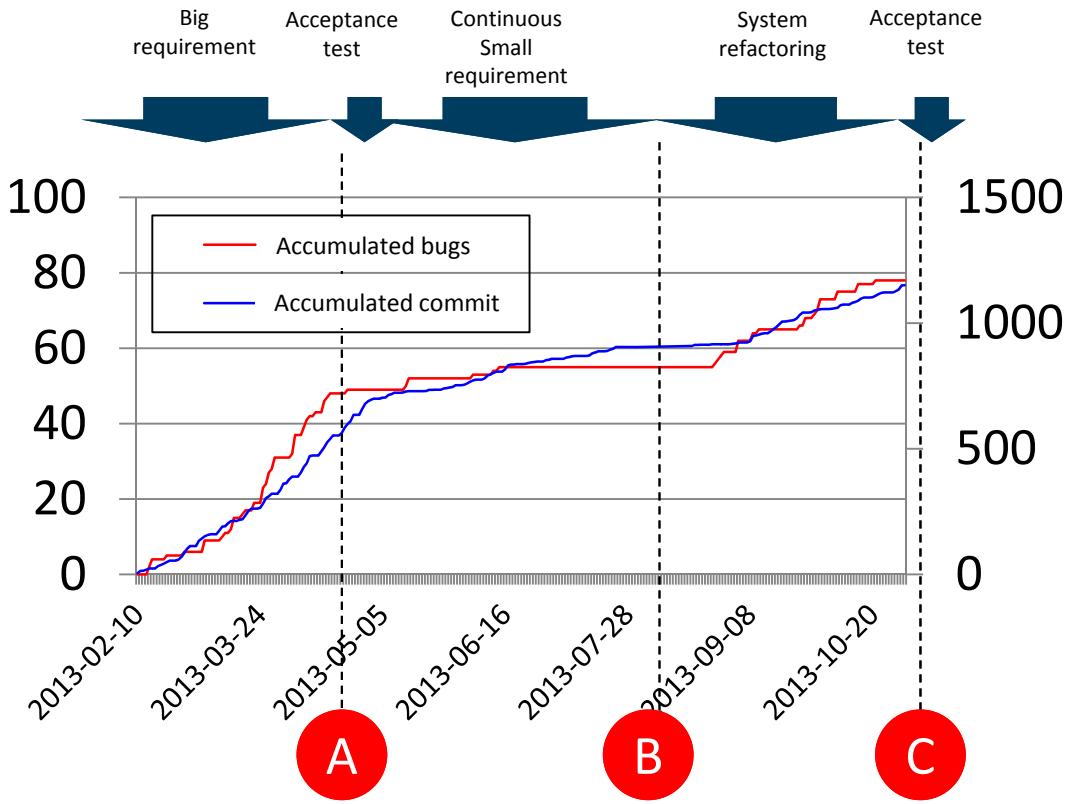






Rakuten

Analysis data and development phase



- Continuous development and testing
- Bug curve rapidly converged

Rakuten

Change in System Test Characteristics

	Traditional System Test	Continuous System Test
Timing of ST	After implementation	Parallel to implementation
Responsibility	Gate keeper of quality	Regression test
Adding test cases	Reliability curve at QA phase	Coverage at implementation phase
Change source code at test phase	Yes	No

Rakuten

Agenda

- Background
- Metrics
- **Analysis ①**
- Analysis ②
- Analysis ③
- Conclusion

Rakuten

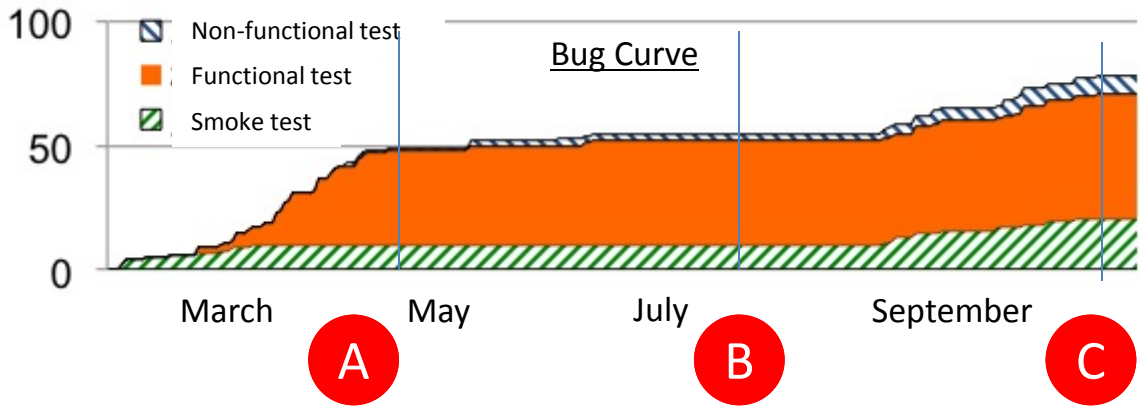
Analysis1: Evaluation on Automated System Test

Q1: Is automated system test low quality?

Objective of Analysis 1

to compare the test density and bug density with the industry statistics for investigating the test quality of target project.

- Considers the project as an incremental mini-waterfall
- Calculate the metrics at A.B. and C where the testing activities are stable



Rakuten

Analysis1: Evaluation on Automated System Test

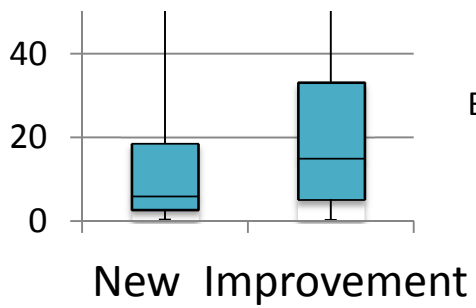
Metrics

- Bug density and Test Density
- Compare with the industry statistics provided by IPA(*1)
  - Minimum、P25, Median、P75, Maximum
    → Inside/outside the range between P25 ~ P75
  - Programming language Java
  - New development/ Improvement development

Industry statistics provided by IPA

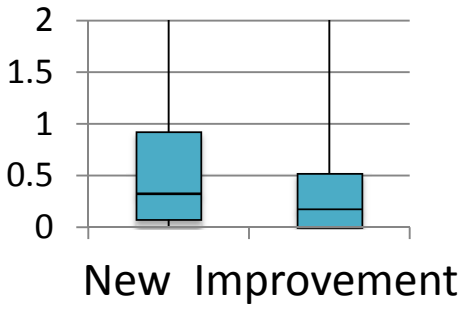
Test Density

Test density
= Number of test cases
÷ KLOC



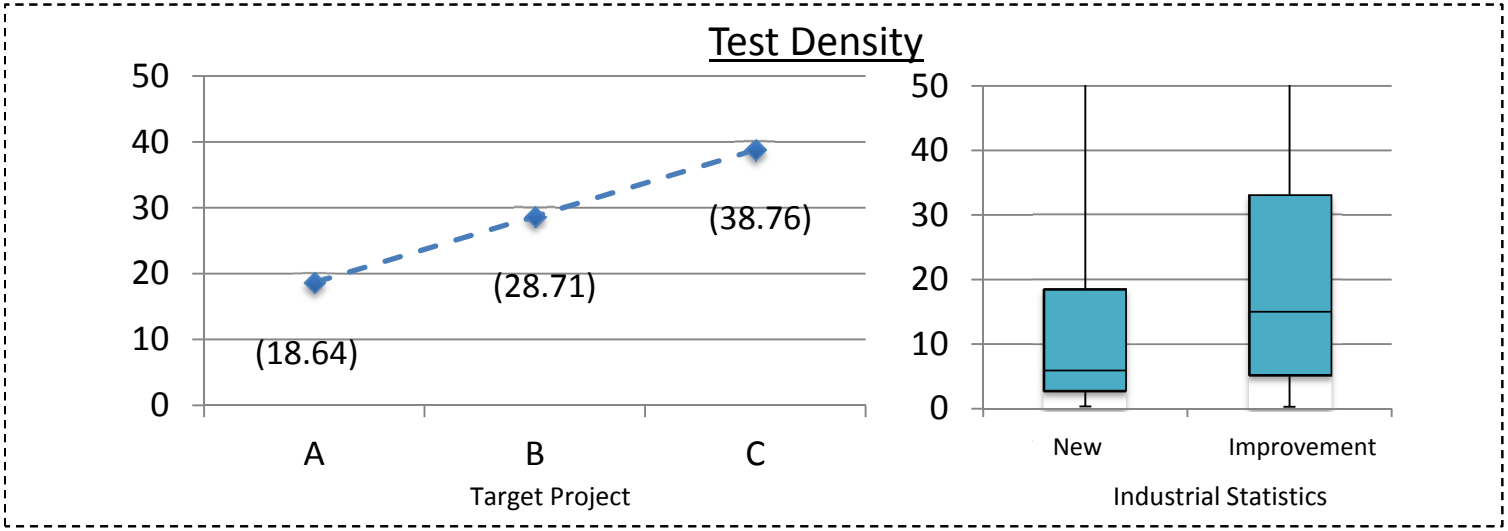
Bug density

Bug density
= Detected bugs
÷ KLOC



(*1) ”ソフトウェア開発データ白書 2012-2013
定量データ分析で分かる開発の最新動向”

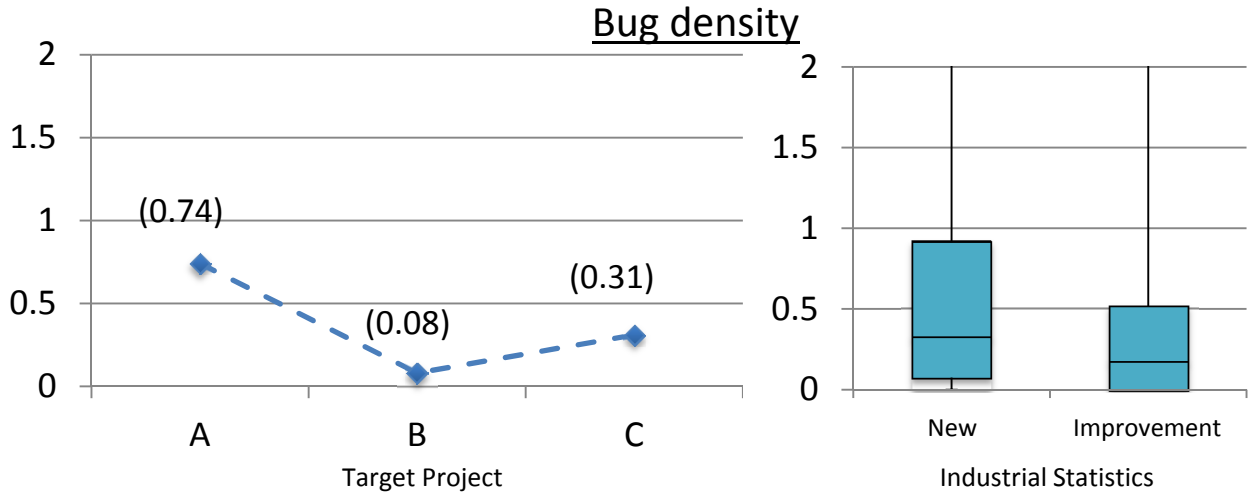
Analysis1：Test Density



Discussion:

- The number of test cases is normal to industrial statistics
- The test density has been improved
  → Adding the test cases becomes easier after implementation of framework
- The test density of C is bit higher than the industry statistics
  → Needs a guideline for the number and coverage of test cases in system test

Analysis1：Bug density



Discussion:

- Small bug density in phase B where small continuous requirements.
- Bugs are detected in phase 3 where no additional features is implemented
  → Detected the degrade of the system in refactoring phase.
- Bug density is inside the industry statistics
  → Considers the automated system test has a enough coverage

Agenda

- Background
- Metrics
- Analysis ①
- **Analysis ②**
- Analysis ③
- Conclusion

Rakuten

Analysis2：Relationship between Dev and Bug

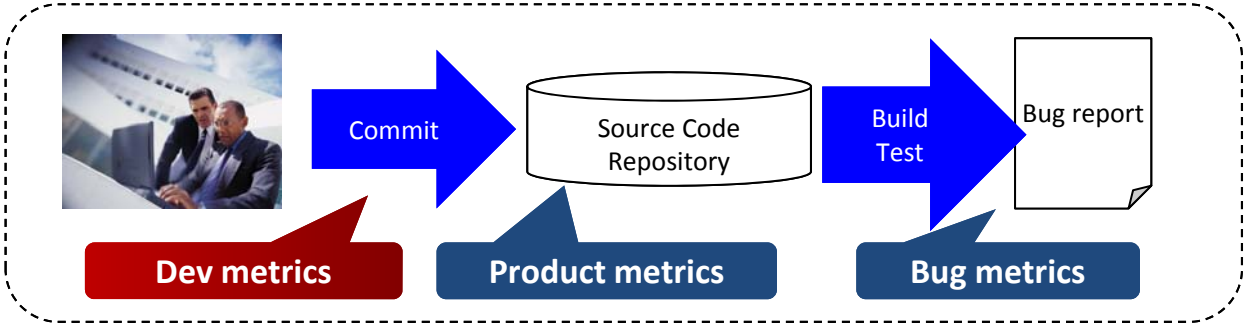
Q2: How is system test in development process?

Objective of Analysis 2

To investigate the relationship between bug and Dev metrics

Previous Research： Evaluation between bug and product metrics

- S Syed et al, “Open Source, Agile and reliability Measures”, ISQI, 2009
- Shimomura et al, “Evaluation of unit test quality risk using software metrics”, SQiP2013.

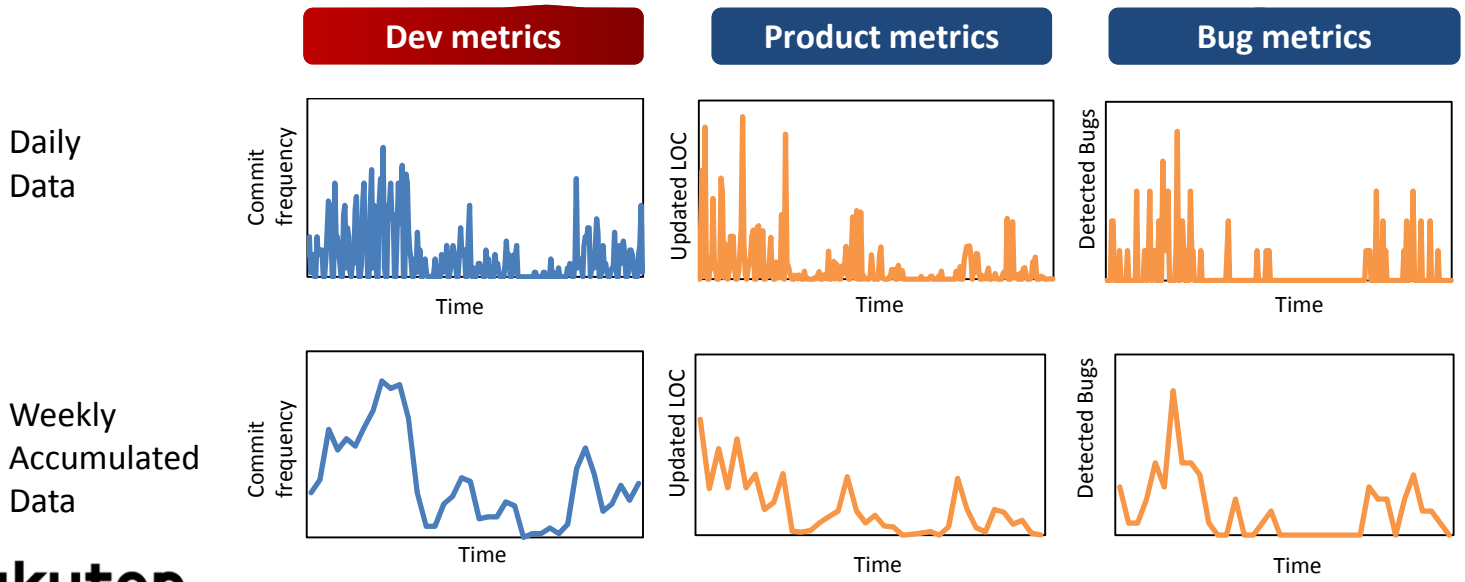


Rakuten

Analysis2：Relationship between Dev and Bug

Method

- Correlation between Dev and Bug metrics
  - Daily Dev and Product metrics
  - Weekly accumulated Dev and Product metrics

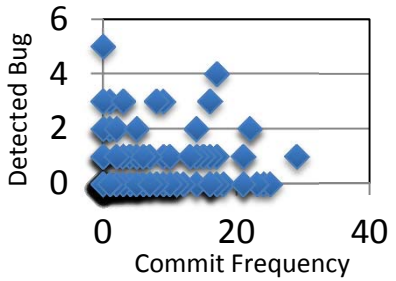


Rakuten

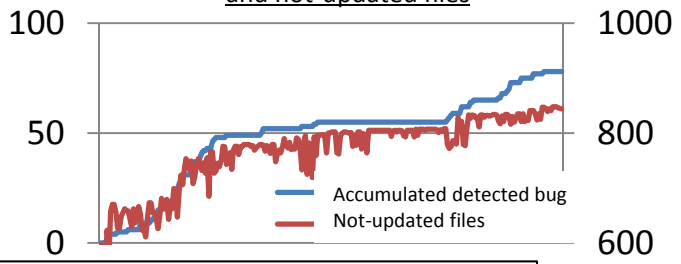
Analysis2：Correlation in Daily Data

Group	Explanatory Variable	Correlation Coefficient
Dev metrics	Commit frequency	0.19
	Commit size	0.06
Product metrics	Updated LOC	0.36
	Added LOC	0.17
	Deleted LOC	0.19
	Not-updated LOC	-0.17
	Updated file	0.20
	Added file	-0.09
	Deleted file	0.06
	Not-updated file	-0.19

Scatter Plot:
Commit Frequency VS Detected Bug



Time series of accumulated detected bugs and not-updated files



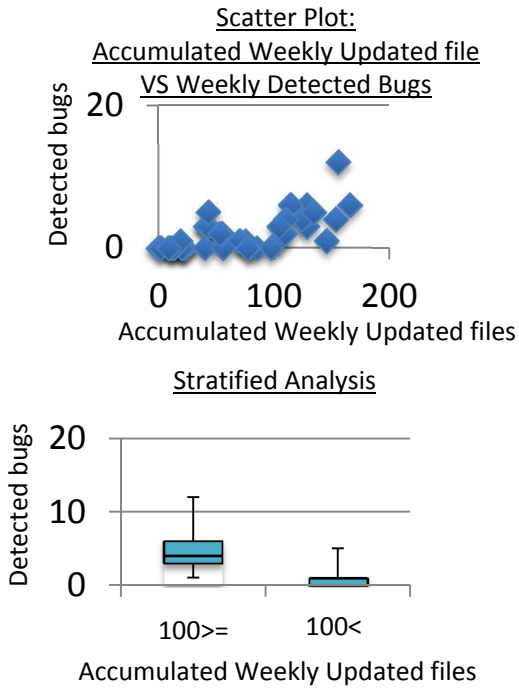
Discussion

- Low correlation for all metrics → Latent interval for integration bugs
- Any meaning that no-update on the files?

Rakuten

Analysis2：Correlation in Accumulated Weekly Data

Group	Explanatory Variable	Correlation Coefficient
Dev metrics	Weekly Commit frequency	0.47
	Weekly Commit size	0.33
Product metrics	Weekly Updated LOC	0.56
	Weekly Added LOC	0.42
	Weekly Deleted LOC	0.61
	Weekly Not-updated LOC	-0.29
	Weekly Updated file	0.66
	Weekly Added file	0.20
	Weekly Deleted file	0.33
	Weekly Not-updated file	-0.31



Discussion:

- Dev metrics has middle-level correlation, but lower than product metrics
- Accumulated updated files has the highest correlation

Agenda

- Background
- Metrics
- Analysis ①
- Analysis ②
- Analysis ③
- Conclusion

Rakuten

Analysis3：Analysis on Bug Curve

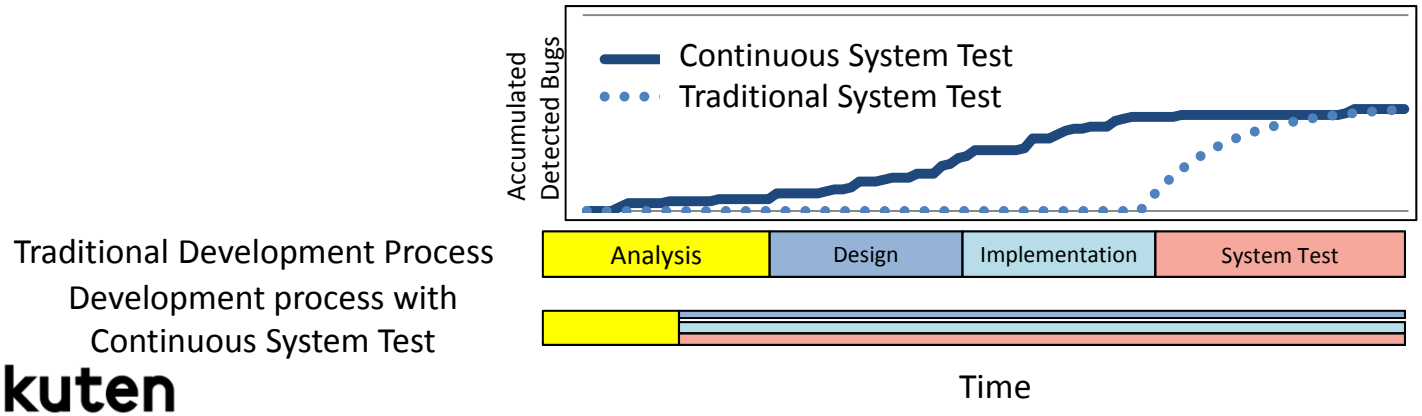
Q3: Any technique for better development?

Objective of Analysis 3

How to detect bugs earlier in continuous system test?
Investigate the reason why bug curve converged rapidly

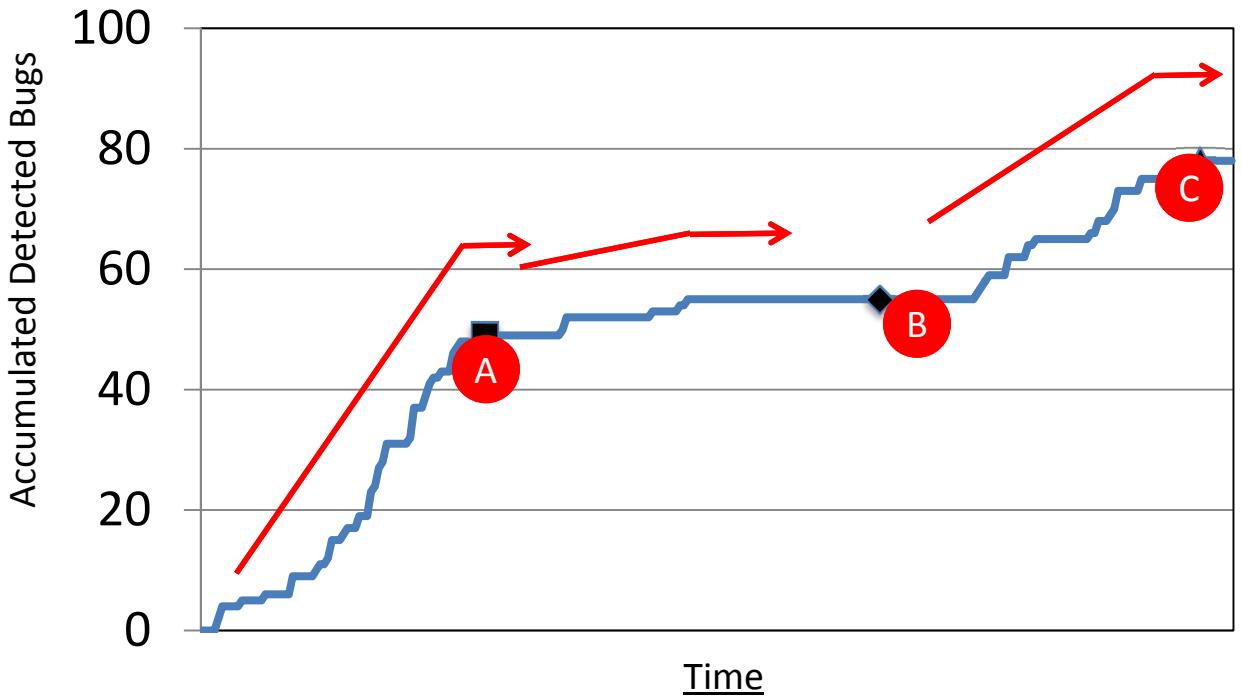
Reliability Growth Curve 【Software Reliability Model, Yamada Shigeru, 1994】

- Consider testing duration and number of detected bug.



Rakuten

Analysis3：Bug curve under Continuous System Test



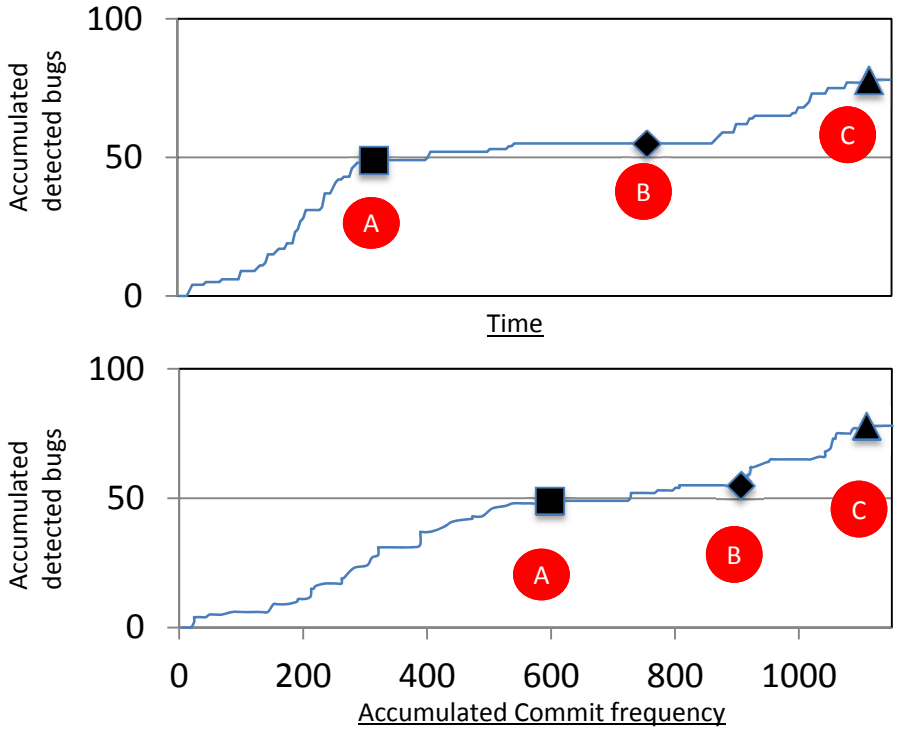
- Bug increase with the stable gradients
- The curve converged rapidly at phase end

Rakuten

Analysis3：Analysis on Bug Curve

Method 1

- Bug curve based on accumulated commit frequency

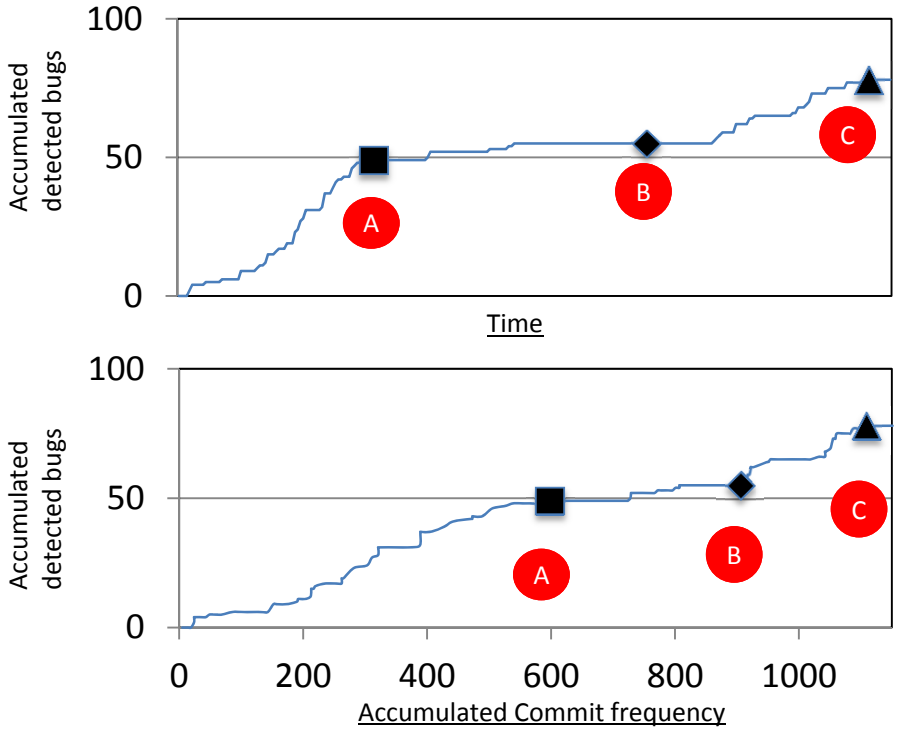


Rakuten

Analysis3：Analysis on Bug Curve

Method 1

- Bug curve based on accumulated commit frequency



Discussion:

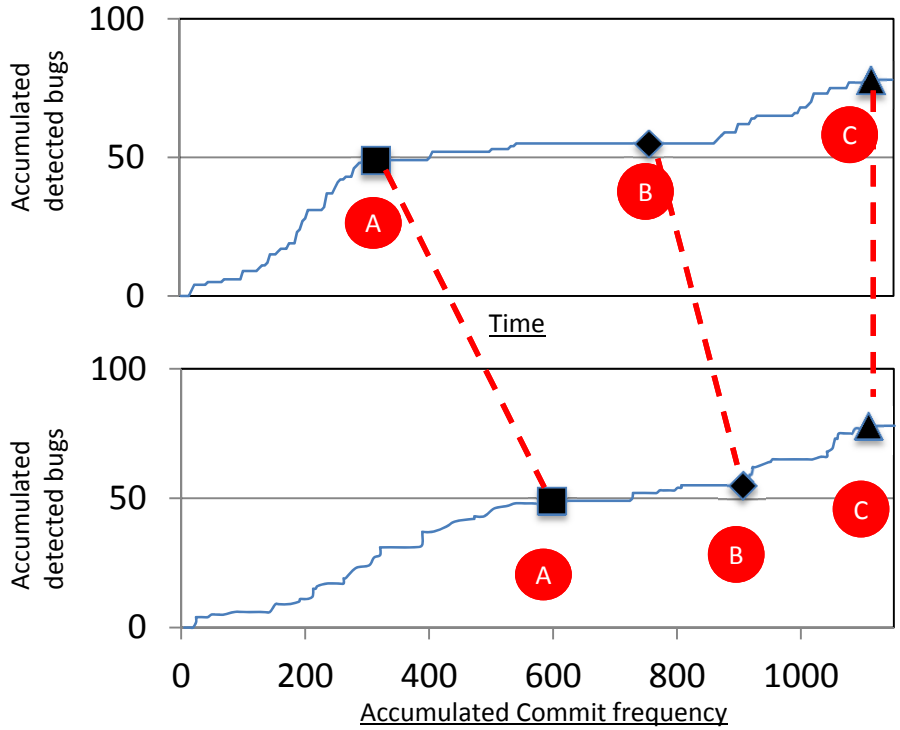
- A,B,C -> Not so much difference in duration
  Big difference in commit frequency
- When horizontal axis is time,
  the curve rapidly converged at phase end
- When horizontal axis is commit frequency,
  the curve converged smoothly
- Small converge makes big converge

Rakuten

Analysis3：Analysis on Bug Curve

Method 1

- Bug curve based on accumulated commit frequency



Discussion:

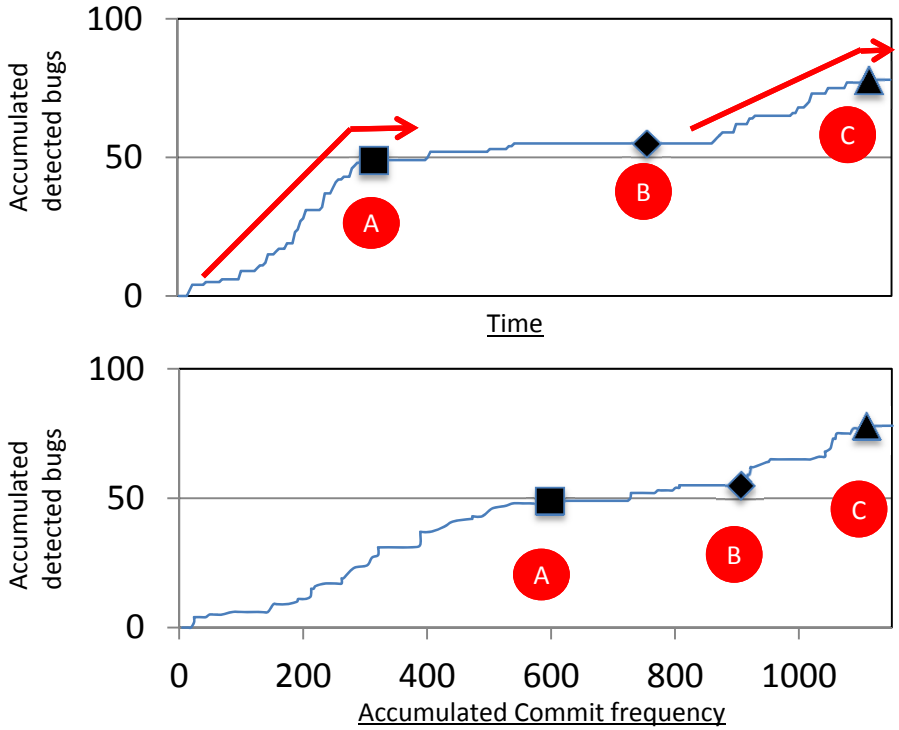
- A,B,C -> Not so much difference in duration
  Big difference in commit frequency
- When horizontal axis is time,
  the curve rapidly converged at phase end
- When horizontal axis is commit frequency,
  the curve converged smoothly
- Small converge makes big converge

Rakuten

Analysis3：Analysis on Bug Curve

Method 1

- Bug curve based on accumulated commit frequency



Discussion:

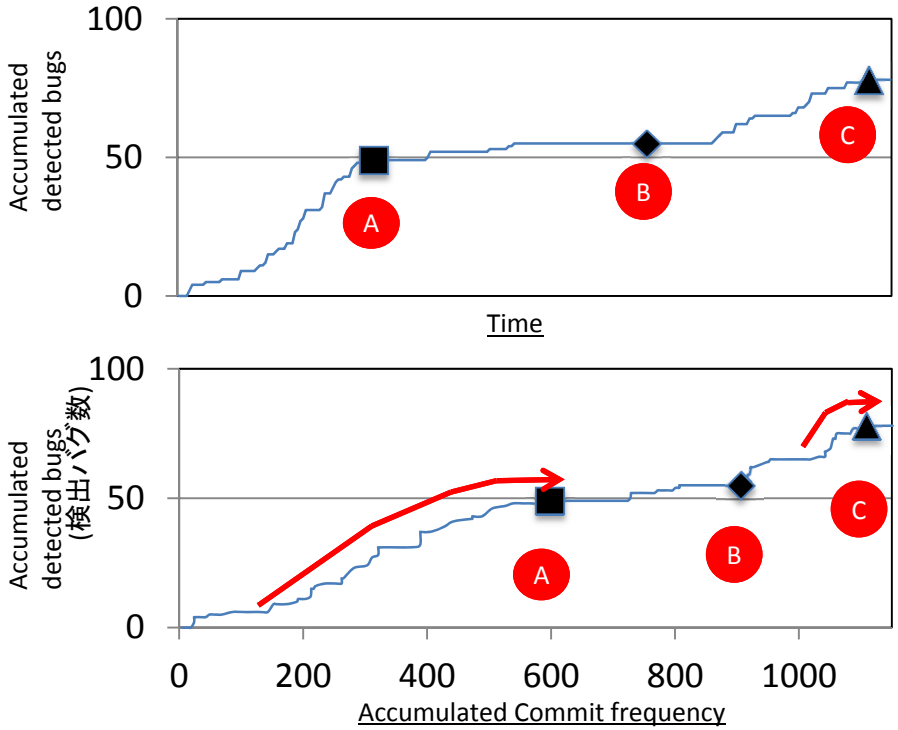
- A,B,C -> Not so much difference in duration
  Big difference in commit frequency
- When horizontal axis is time,
  the curve rapidly converged at phase end
- When horizontal axis is commit frequency,
  the curve converged smoothly
- Small converge makes big converge

Rakuten

Analysis3：Analysis on Bug Curve

Method 1

- Bug curve based on accumulated commit frequency



Discussion:

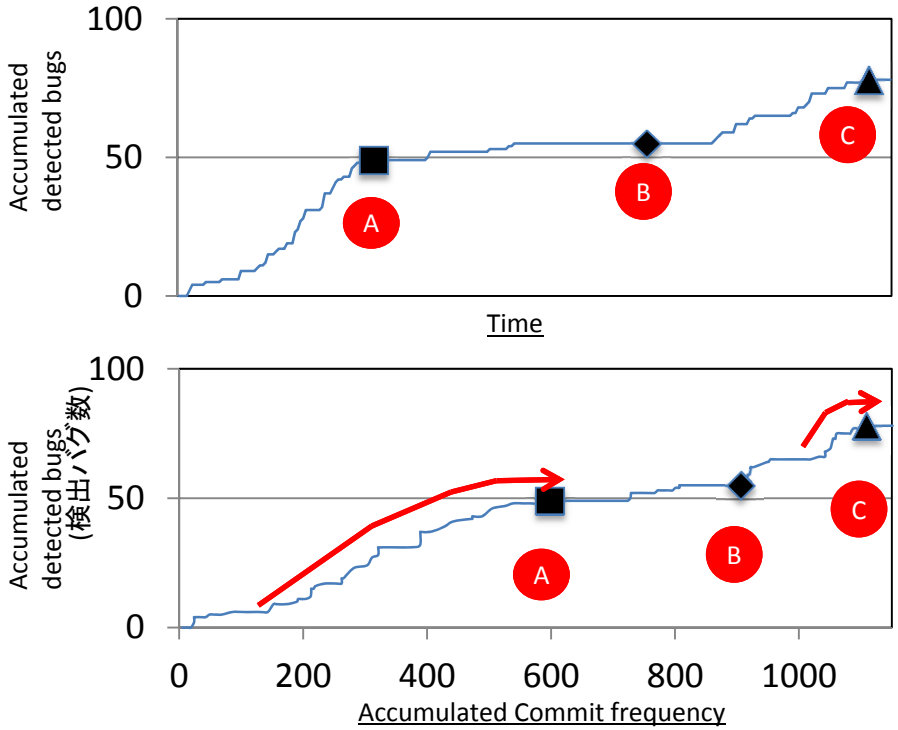
- A,B,C -> Not so much difference in duration
  Big difference in commit frequency
- When horizontal axis is time,
  the curve rapidly converged at phase end
- When horizontal axis is commit frequency,
  the curve converged smoothly
- Small converge makes big converge

Rakuten

Analysis3：Analysis on Bug Curve

Method 1

- Bug curve based on accumulated commit frequency



Discussion:

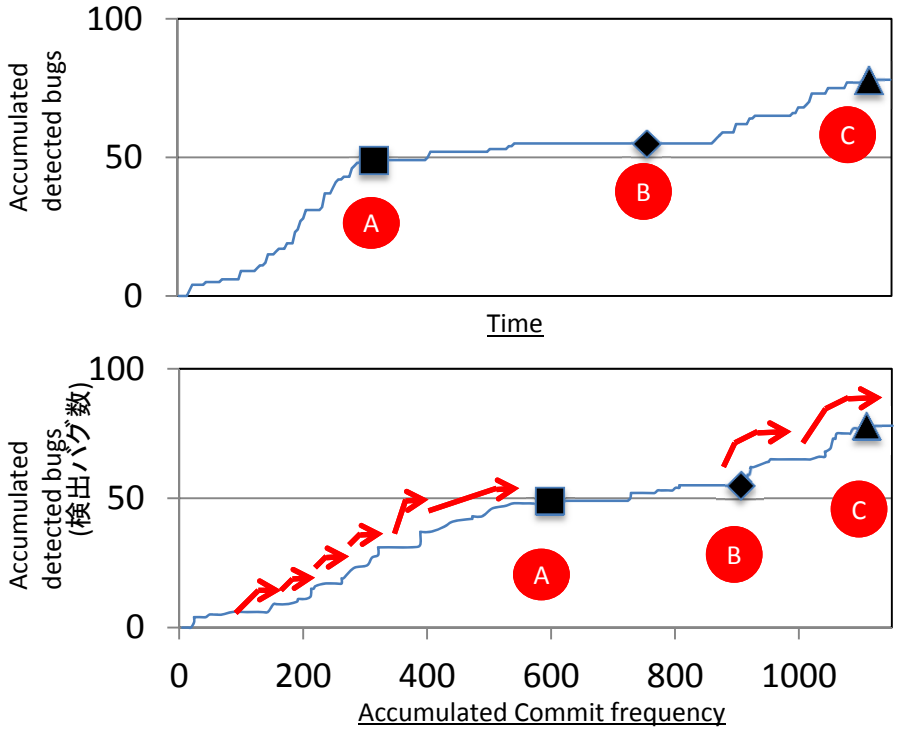
- A,B,C -> Not so much difference in duration
  Big difference in commit frequency
- When horizontal axis is time,
  the curve rapidly converged at phase end
- When horizontal axis is commit frequency,
  the curve converged smoothly
  -> It show the reduction of bugs in the commit of source code change.
- Small converge makes big converge

Rakuten

Analysis3：Analysis on Bug Curve

Method 1

- Bug curve based on accumulated commit frequency



Discussion:

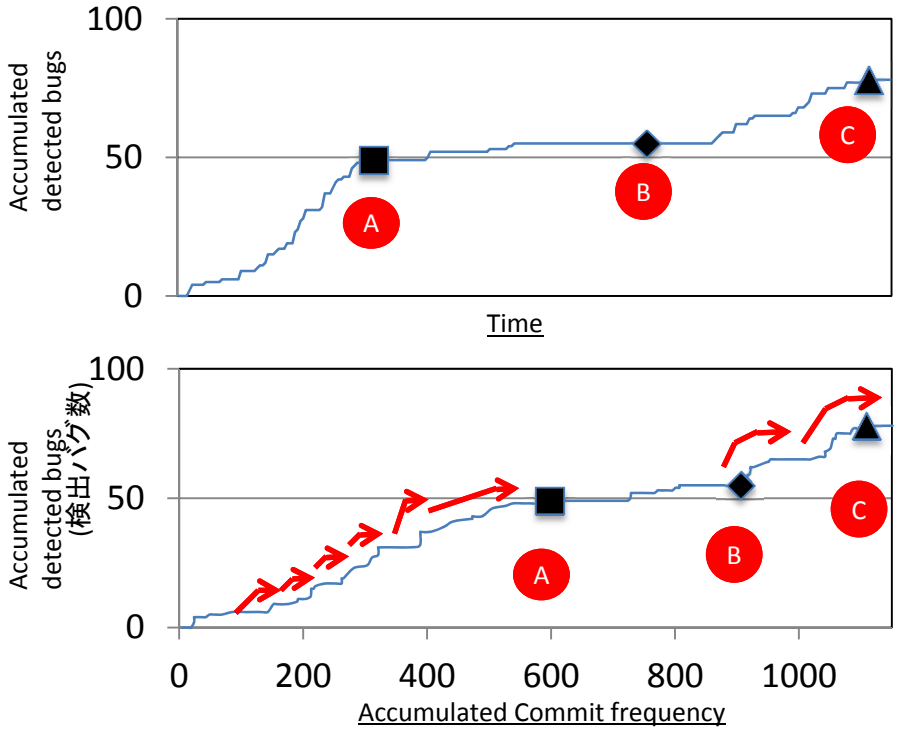
- A,B,C -> Not so much difference in duration
  Big difference in commit frequency
- When horizontal axis is time,
  the curve rapidly converged at phase end
- When horizontal axis is commit frequency,
  the curve converged smoothly
  -> It show the reduction of bugs in the commit of source code change.
- Small converge makes big converge

Rakuten

Analysis3：Analysis on Bug Curve

Method 1

- Bug curve based on accumulated commit frequency



Discussion:

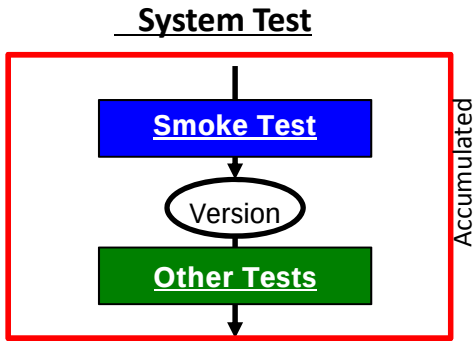
- A,B,C -> Not so much difference in duration
  Big difference in commit frequency
- When horizontal axis is time,
  the curve rapidly converged at phase end
- When horizontal axis is commit frequency,
  the curve converged smoothly
  -> It show the reduction of bugs in the commit of source code change.
- Small converge makes big converge
  -> Developers know the bugs immediately right after commit, then fix them.

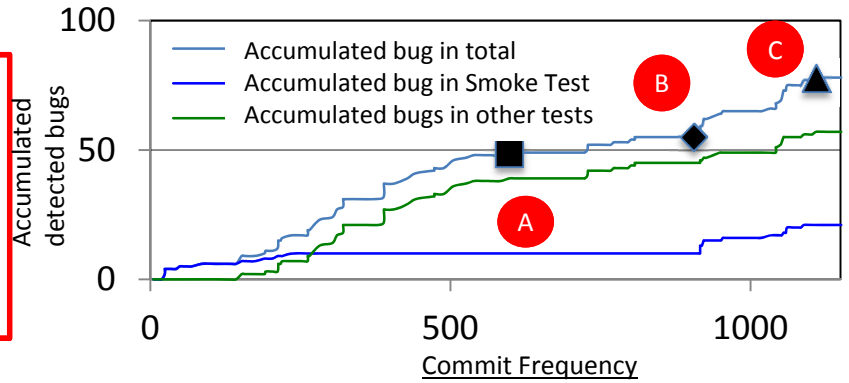
Rakuten

Analysis3：Analysis on Bug Curve

Method 2

- Bug curve analysis for each test type

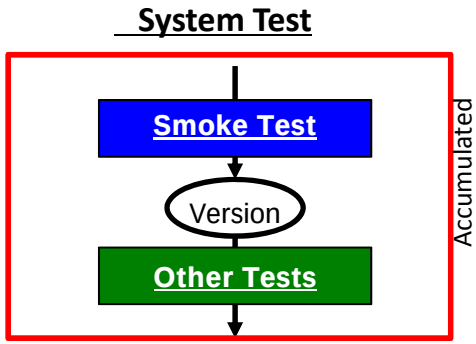


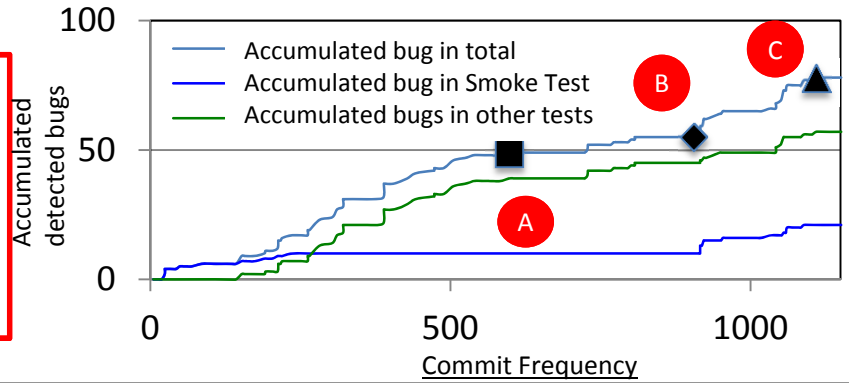


Analysis3：Analysis on Bug Curve

Method 2

- Bug curve analysis for each test type





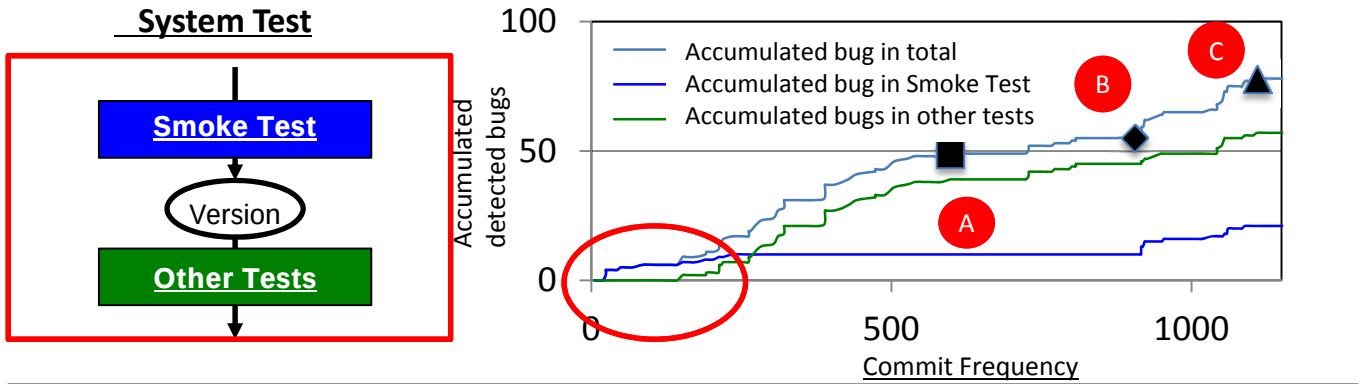
Discussion:

- The commit breaking the smoke test is happened only once in A
- 2 times in C (Detected bugs are both 10)
- In C, Total is converged right after converge of smoke test C

Analysis3：Analysis on Bug Curve

Method 2

- Bug curve analysis for each test type



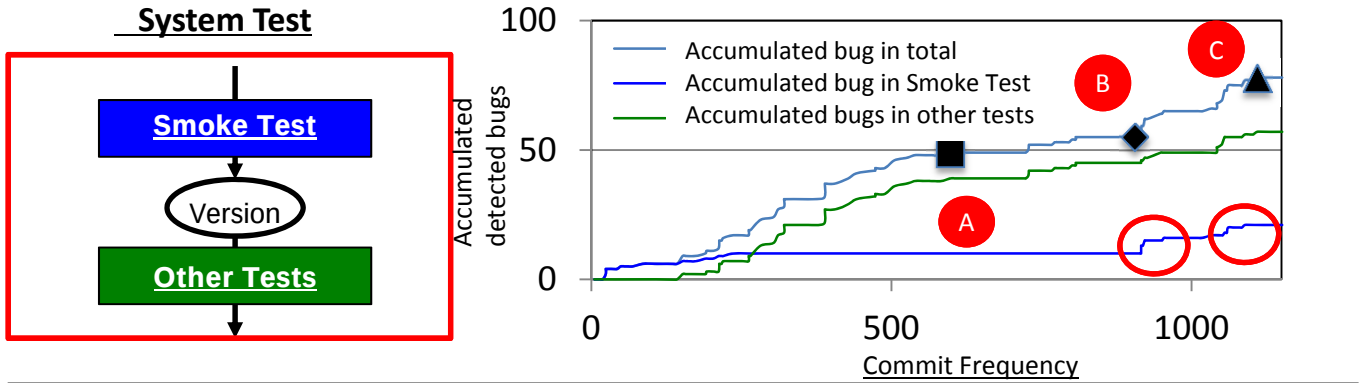
Discussion:

- The commit breaking the smoke test is happened only once in A
- 2 times in C (Detected bugs are both 10)
- In C, Total is converged right after converge of smoke test C

Analysis3：Analysis on Bug Curve

Method 2

- Bug curve analysis for each test type



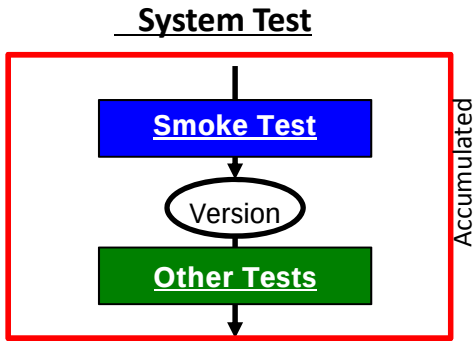
Discussion:

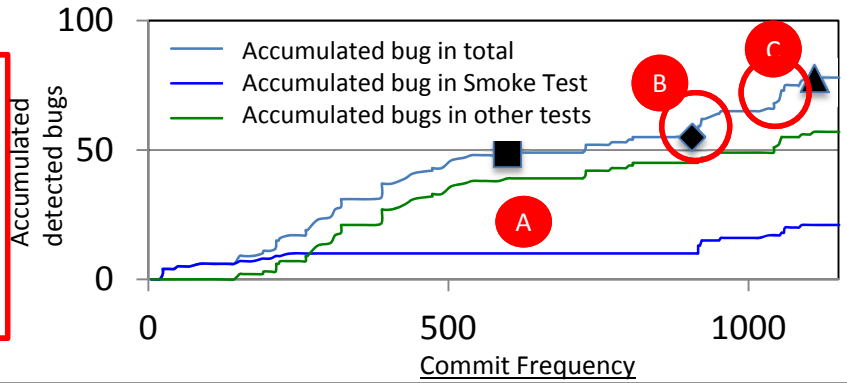
- The commit breaking the smoke test is happened only once in A
- 2 times in C (Detected bugs are both 10)
- In C, Total is converged right after converge of smoke test C

Analysis3：Analysis on Bug Curve

Method 2

- Bug curve analysis for each test type





Discussion:

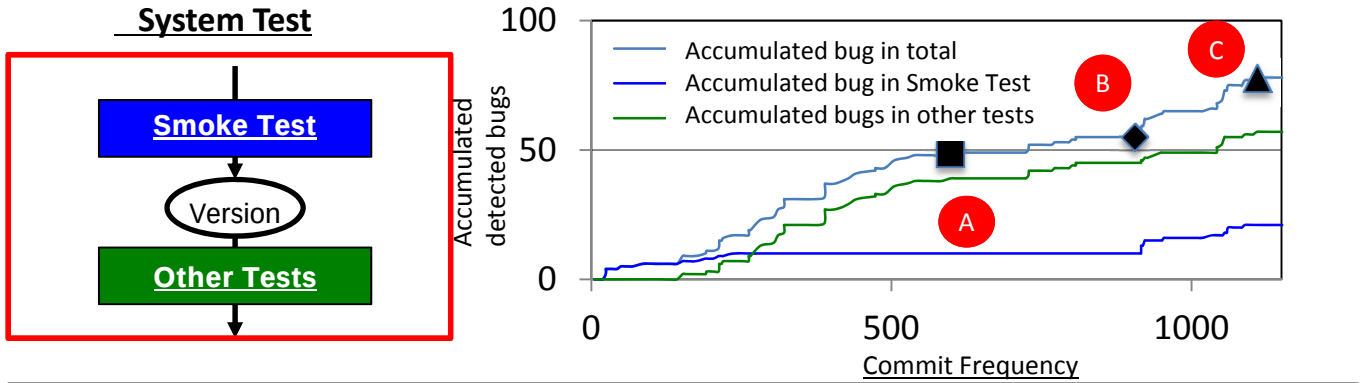
- The commit breaking the smoke test is happened only once in A
- 2 times in C (Detected bugs are both 10)
- In C, Total is converged right after converge of smoke test C

Rakuten

Analysis3：Analysis on Bug Curve

Method 2

- Bug curve analysis for each test type



Discussion:

- The commit breaking the smoke test is happened only once in A
- 2 times in C (Detected bugs are both 10)
- In C, Total is converged right after converge of smoke test C
  -> Implementation causing the smoke test break is divided for iteration.
  Smaller commit can make bug detection and fix earlier.

Agenda

- Background
- Metrics
- Analysis ①
- Analysis ②
- Analysis ③
- **Conclusion**

Rakuten

Conclusion：Question on Continuous System Test

Q1: Is automated system test low quality?

Q2: How is system test in development process?

Q3: Any technique for better development?

Conclusion：

To analyzed development, product and bug metrics for better understanding on continuous system test

Rakuten

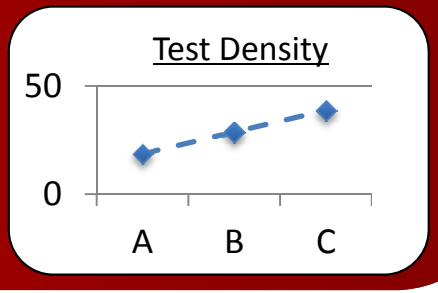
Conclusion：Answer for Q1

Q1: Is automated system test low quality?

Answer（From Analysis 1）:

Automated system test is not low quality

- However, the test density can be easily increased in automated test environment.
  -> need the guideline for the coverage of system test.



Rakuten

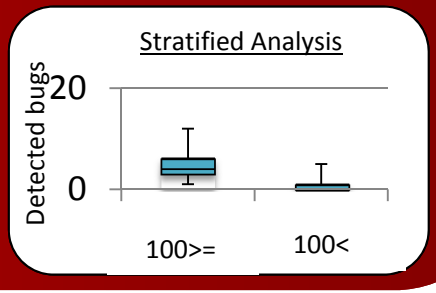
Conclusion：Answer for Q2

Q2: How is system test in development process?

Answer（From Analysis 2）:

Not only product metrics, but also Dev metrics has a relationship with bugs under the environment where system test is a part of process of development

- Bug ingestion relates to not updated files and updated files.



Rakuten

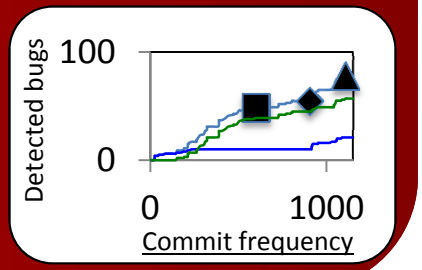
Conclusion：Answer for Q3

Q3: Any technique for better development?

Answer（From Analysis 3）:

Quick feedback to developers is important in continuous system test environment

- Commit type changed from feature to bug fix
- The bug can be detected earlier by dividing the commits which cause smoke test break into different iteration.

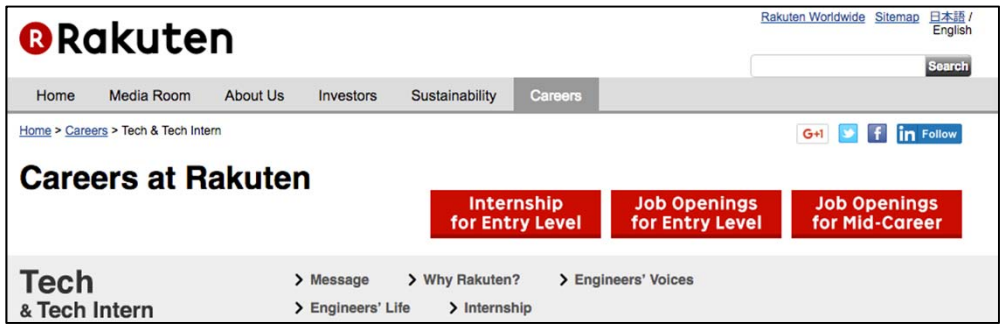


Rakuten

Further work

- Guideline for system test
  → Improvement of test under continuous system test
    - Prioritization of test cases
    - Prevent too much test cases
- Use Dev metrics for quality control
- More collaboration between Dev and QA

Long live testing



http://global.rakuten.com/corp/careers/engineering/

Rakuten