Yusuf Bahadur
3/4/20
Roopairs

Technical Debt is the cost of rework on the system caused by cutting corners and choosing an easy but limited solution.

During my last metrics posting, I was unable to set up SonarQube successfully on my machine and instead calculated the technical debt of our project by hand. Most of the technical debt last project arose from the RestApi.kt class which handled the majority of the api calls. In addition, our project lacked proper documentation which I estimated would take 15 hours to write.

Technical Debt for Metrics 1 calculated via Equations:

Equation:
RC = **k**(Cyclomatic Complexity) + **p** (Documentation)
Remediation Cost: **RC** [hours]
Development Cost: **DC** [hours]
Technical Debt = (RC / DC) x 100%

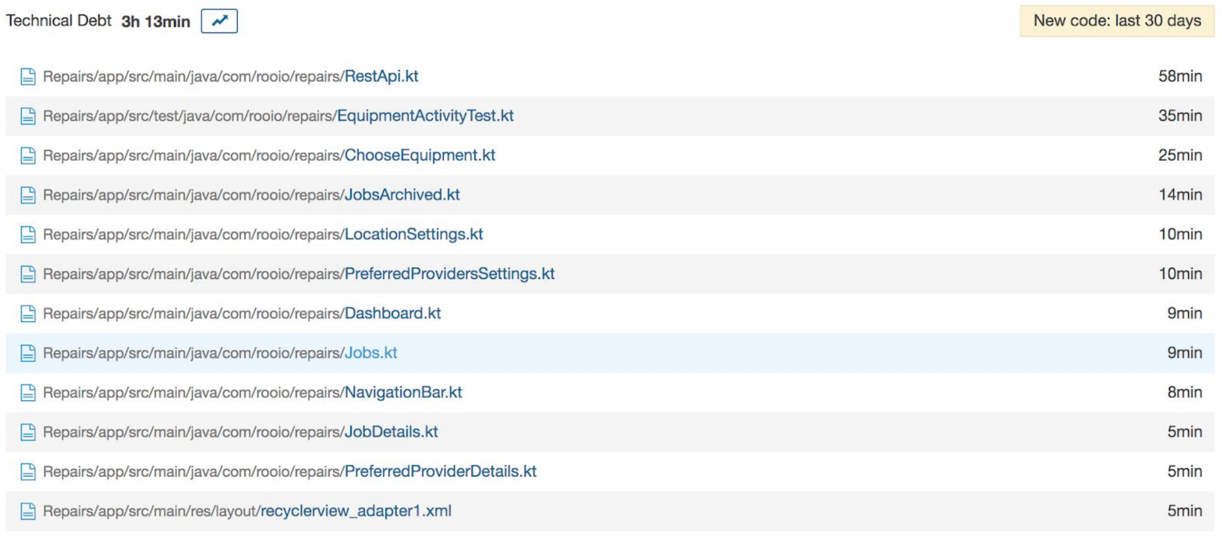
Calculations:
RC = 1.788
Remediation Cost: 8hrs (fix RestApi) + 15hrs Documentation
Development Cost: 363 Hours (Jira)

Technical Debt Ratio
Technical Debt = (23 / 363) x 100% = 6.33%

For the current metrics posting, I was able to successfully calculate Technical Debt via SonarQube. SonarQube estimates our projects total debt to be 3 hours and 13 min to date. This is less than Metric 1's estimate of 8 hours to refactor the RestApi class. However, in order to accurately compare Metric 1 to Metric 2 we can use SonarQube to see the difference between old technical debt vs new technical debt.

Technical Debt from SonarQube:

Overall Technical Debt: 3 hours 13 min

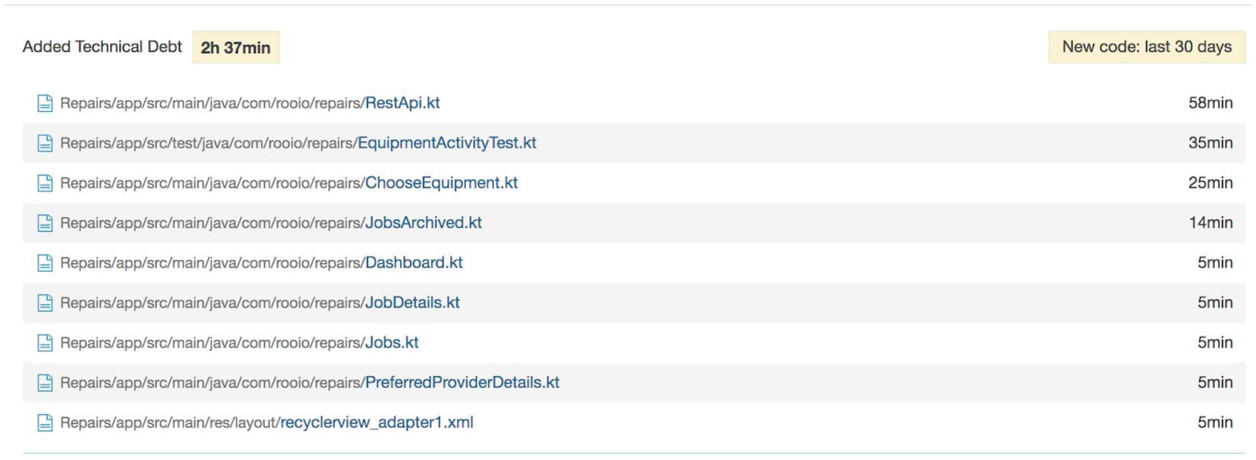


Technical Debt 3h 13min

New code: last 30 days

Repairs/app/src/main/java/com/rooio/repairs/RestApi.kt	58min
Repairs/app/src/test/java/com/rooio/repairs/EquipmentActivityTest.kt	35min
Repairs/app/src/main/java/com/rooio/repairs/ChooseEquipment.kt	25min
Repairs/app/src/main/java/com/rooio/repairs/JobsArchived.kt	14min
Repairs/app/src/main/java/com/rooio/repairs/LocationSettings.kt	10min
Repairs/app/src/main/java/com/rooio/repairs/PreferredProvidersSettings.kt	10min
Repairs/app/src/main/java/com/rooio/repairs/Dashboard.kt	9min
Repairs/app/src/main/java/com/rooio/repairs/Jobs.kt	9min
Repairs/app/src/main/java/com/rooio/repairs/NavigationBar.kt	8min
Repairs/app/src/main/java/com/rooio/repairs/JobDetails.kt	5min
Repairs/app/src/main/java/com/rooio/repairs/PreferredProviderDetails.kt	5min
Repairs/app/src/main/res/layout/recyclerview_adapter1.xml	5min

New Technical Debt from the Last 30 Days**:** 2 hours 37 min



Added Technical Debt 2h 37min

New code: last 30 days

Repairs/app/src/main/java/com/rooio/repairs/RestApi.kt	58min
Repairs/app/src/test/java/com/rooio/repairs/EquipmentActivityTest.kt	35min
Repairs/app/src/main/java/com/rooio/repairs/ChooseEquipment.kt	25min
Repairs/app/src/main/java/com/rooio/repairs/JobsArchived.kt	14min
Repairs/app/src/main/java/com/rooio/repairs/Dashboard.kt	5min
Repairs/app/src/main/java/com/rooio/repairs/JobDetails.kt	5min
Repairs/app/src/main/java/com/rooio/repairs/Jobs.kt	5min
Repairs/app/src/main/java/com/rooio/repairs/PreferredProviderDetails.kt	5min
Repairs/app/src/main/res/layout/recyclerview_adapter1.xml	5min

Given the new technical debt is from the last 30 days, we can assume that the original technical debt using SonarQube for Metric 1 would have been:

Metrics 1 Sonarqube Technical Debt = Overall Technical Debt - New Technical Debt
Metrics 1 Sonarqube Technical Debt = 3 hours 13 min - 2 hours 37 min
Metrics 1 Sonarqube Technical Debt = 36 min

Metrics 2 Sonarqube Technical Debt = Overall Technical Debt
Metrics 2 Sonarqube Technical Debt = 3 hours 13 min

Given Metrics 1 from SonarQube would have been 36 minutes of technical debt and Metrics 2 from SonarQube is 3 hours 13 min of technical debt, our team clearly has taken on a substantial amount of technical debt. A cause for this extensive addition of debt may be due to the faster pace of which our team is developing code.

It is important to note that SonarQube provides a technical debt ratio for our project at less than 0.1%. Given that this ratio is less than 5%, technical debt is not a huge concern for our project at the current state.