

Code Review Practices for Refactoring Changes: An Empirical Study on OpenStack

Eman Abdullah AlOmar ,Mohamed Wiem
Moataz Chouchen ,Ali Ouni

発表者 花田

<https://conf.researchr.org/track/msr-2022/msr-2022-technical-papers?#event-overview>

背景

- コードレビュー
 - ソフトウェアの品質を向上させ,知識を共有し,コーディング標準とガイドラインの遵守を保証するための技術.
 - 開発者はコードベースのコード変更を統合する前に
→リファクタリング活動のコードレビューについて議論.
- ベストプラクティスや成果,社会技術的な側面を探るために広く研究.

しかし

- リファクタリングがどのようにレビューされているか
 - その際に何を気にしているか
- ほとんど知られていない！

目的

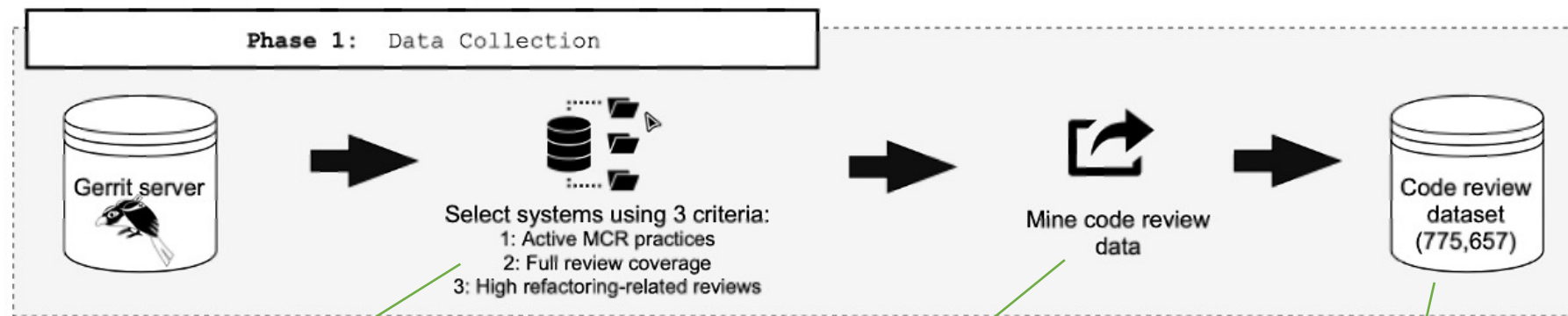
- 開発者が、提出されたリファクタリングコードを受け入れるか拒否するかを決定するための主な基準は何か？
- このプロセスを困難にしているものは何かを理解するための定量的および定性的研究を発表.

RQ

RQ1.リファクタリングレビューは非リファクタリングレビューと比較して,コードレビューの労力はどの程度なのか?

RQ2.リファクタリングレビューの判断に大きく関わる基準は何なのか?

データセット（データ収集）



三つの基準を定め、データ収集

- MCRの実践.
 - 多くのレビューが行われるシステムかどうか
- 完全なレビューカバレッジ.
 - コードレビュー全てを調査
- リファクタリング関連のレビュー数.
 - 十分なリファクタリング関連のインスタンスが必要

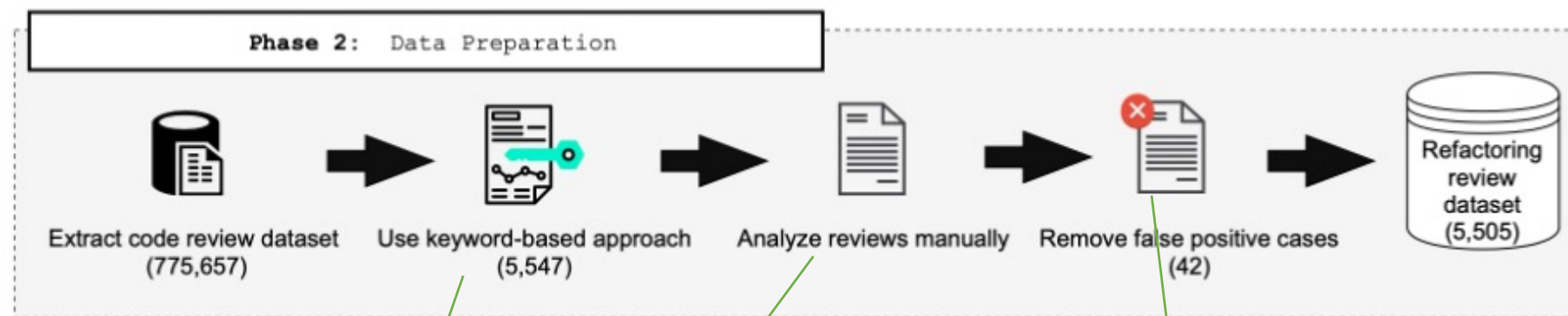
データマイニング

- Gerrit が提供するRESTful API7

収集したレビュー

- Closed
- 2012年12月から2021年4月まで
- OpenStackプロジェクトから合計775,657件のコード変更をマイニング

データセット（データ作成）



自動フィルタリング

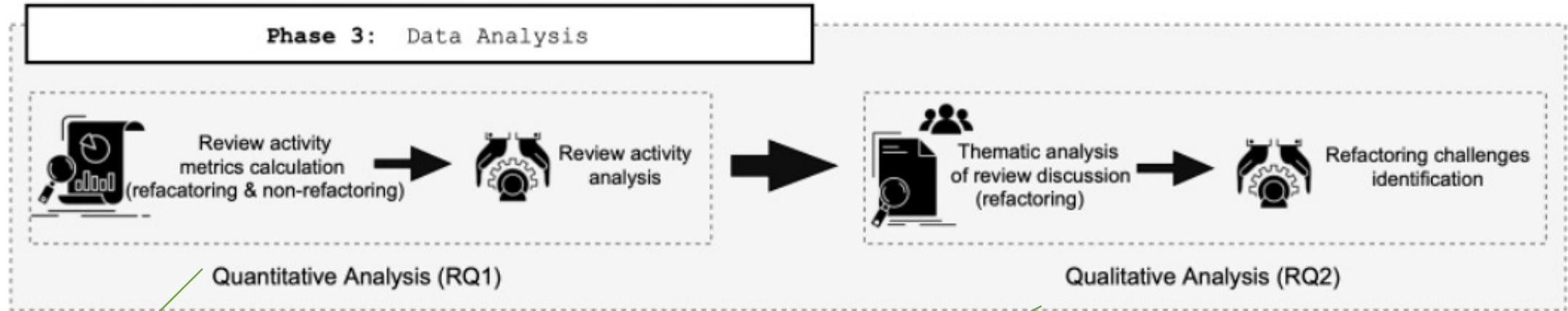
- コード変更のタイトルと説明の両方に refactor* というキーワードが含まれていないものをフィルタリング

手動フィルタリング

- データの正しさを保証するために、すべてを手動で検査（目視）

- 偽陽性ケースを削除（誤りが含まれる）

データセット（データ分析）



定量的なデータ分析

- リファクタリングと非リファクタリングのメトリクスを計算する際の変動が統計的に有意であるかどうか
- Mann-Whitney U検定[26]を使用
- 帰無仮説（リファクタリングと非リファクタリングのメトリクス値に変動がない）
- ノンパラメトリック効果量測定であるクリフのデルタ (δ) [23]を使用

- Negligible for $|\delta| < 0.147$
- Small for $0.147 \leq |\delta| < 0.33$
- Medium for $0.33 \leq |\delta| < 0.474$
- Large for $|\delta| \geq 0.474$

Romanoら[68]が報告した解釈

定性的なデータ分析

- リファクタリングコードのレビューの主要な特徴をテーマ分析
- プロセスの結果、6つのカテゴリと28のサブカテゴリからなる階層的な分類法作成
- この分類法の正確さと代表性に関する11の質問からなる調査を実施し、分類法の外部検証
- 調査結果についてフォローアップインタビュー

RQ1結果

RQ1.リファクタリングレビューは非リファクタリングレビューと比較して、コードレビューの労力はどの程度なのか？

→レビュー作業/メトリックに応じた小,中,または大きな効果サイズを伴う。
(Mann-Whitney U 検定を実施,統計的に有意な p 値が得られた)



リファクタリングのレビューが,コード変更著者とレビューアーの間でより長い議論を引き起こす。

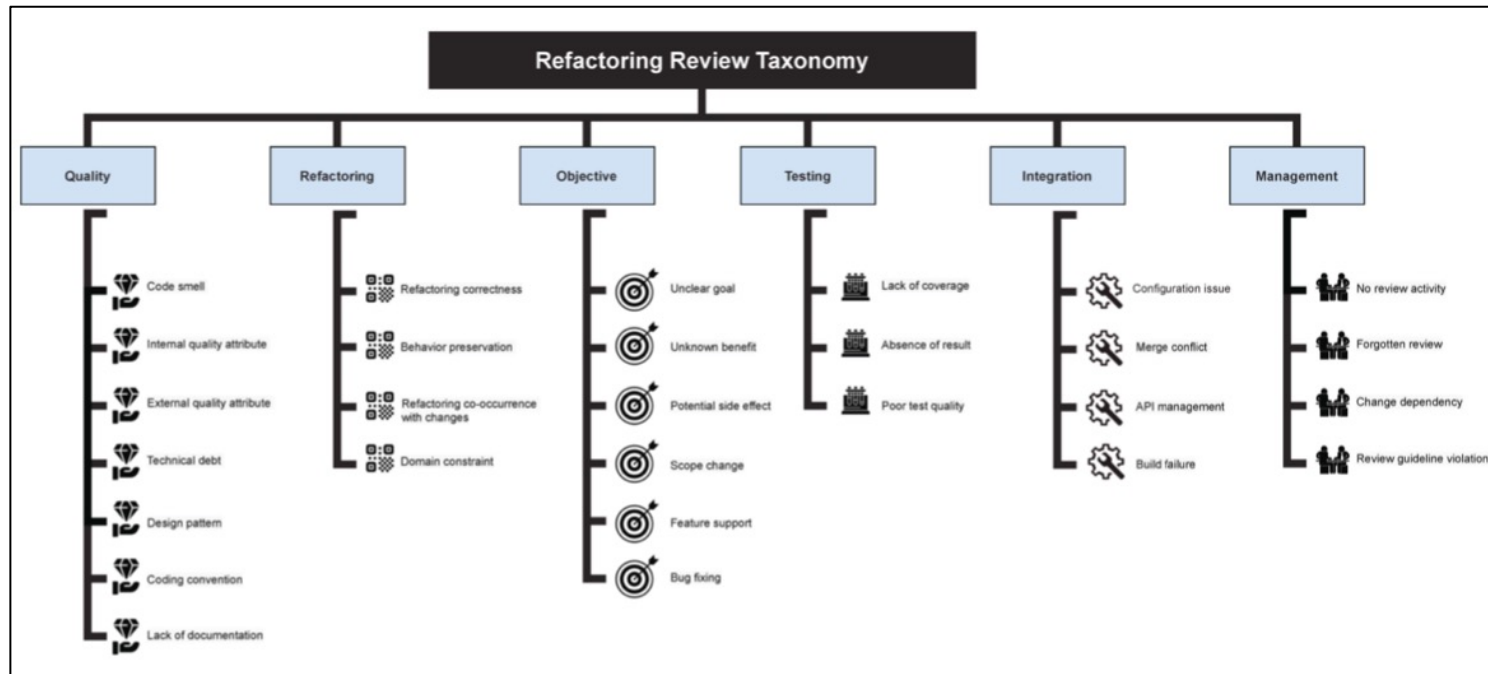


コードレビュー作業の観点から,解決までに著しく時間がかかる。

Table 3: Statistics of code review activity efforts.														
Metrics	Refactoring code review						Non-refactoring code review						Statistical difference	
	Min	Q1	Median	Mean	Q3	Max	Min	Q1	Median	Mean	Q3	Max	p-value	Cliff's delta (δ)
Number of reviewers	0	2	4	5.65	6	12	0	2	3	4.45	5	9	6.750458e-43	small (0.15)
Number of review comments	0	5	9	20.87	19	40	0	3	6	13.05	12	25	2.730193e-82	small (0.22)
Number of inline comments	0	0	0	6.61	5	12	0	0	0	3.26	1	2	2.066649e-98	medium (0.33)
Number of revisions	1	1	3	4.79	6	13	1	1	2	3.19	3	6	1.252695e-126	small (0.3)
Number of changed files	0	1	3	5.98	6	13	0	1	1	3.69	3	6	6.006899e-190	medium (0.33)
Review duration (seconds)	0	37	170.68	928.21	606.22	1458.68	0	12.85	75.98	738.93	351.03	852.99	2.572584e-66	small (0.23)
Length of discussion (characters)	0	168	451	3450.29	1563	3653	0	110	296	1962.39	1044	2436	6.966752e-42	small (0.15)
Length of description (characters)	62	171	264	327.61	388	711	56	124	218	276.97	354	699	4.071541e-41	small (0.13)
Code churn	-1	40	114	367.63	309	711	0	3	11	201.58	53	128	0.000000e+00	large (0.64)

RQ2結果

- テーマ分析技術を使ってOpenStackエコシステムを手動で調査し,レビュー基準の包括的でハイレベルなカテゴリーを作成



6つのカテゴリと28のサブカテゴリの表

カテゴリー分類 (RQ2)

カテゴリーの詳細

Category	Sub-category	Example (Excerpts from a related refactoring review discussion)
Quality	Code smell	<i>"I do understand the desire to refactor some code to eliminate duplicate code. The purpose of the common class was to contain all of the duplicate code between the 2 drivers. This seems like a half baked approach to refactoring to accomplish that goal, when the common class should have been used as a new base. Now with this patch there are 2 classes (base and common) that contain common code."</i>
	Internal quality attribute	<i>"I think there is still an inheritence problem with the Base2 class (see inline) and the way python handles multiple inheritance means we have to be careful when creating the UTF8 versions of the unittest classes."</i>
	External quality attribute	<i>"Looking through the series i found enough such code to see that it does not look easy readable."</i>
	Technical debt	<i>"I was trying to use exception.ARQInvalidState with assertRaises, but found we had some technical debts, so use Exception instead to check the error message."</i>
	Design pattern	<i>"I think the alternative should be what other refactor possibilities are. The proposed change indicates some of the methods that are being planned to be implemented. As an alternative to evolve this library in future we need another design pattern."</i>
	Coding convention	<i>"We've seen how difficult it is to arrive at a single naming convention that works for all OpenStack projects, but as Rally expands beyond benchmarking OpenStack alone, it is quite simply impossible to create a single name format that will work for literally everything."</i>
	Lack of documentation	<i>"Is there a documentation somewhere explaining how to define "extra" dependencies? It would be nice to have a link somewhere. IHMO dependencies are a very complex problem, and extra dependencies look even more complex to me."</i>

※表は省略しています

結論

- リファクタリングコードレビューの実践を理解
→研究コミュニティと産業界にとって極めて重要
- 本研究
 - リファクタリングレビューする際の基準を調査するために,量的および質的研究を実施
- 結果
 - リファクタリングのレビューには,リファクタリング以外の変更に比べて時間がかかること
 - レビュー基準の包括的でハイレベルなカテゴリーを作成
- 今後の課題
 - オープンソースと産業界のソフトウェア開発者を対象とした調査を実施する予定
 - コードレビューでリファクタリング活動を行う際の一般的なレビュー基準と特定のレビュー基準を調査する予定

所感

- 2名のオーサーがデータの分析を行っていたことに驚き,さらに,RQ2に約40日費やしていたと記述されていた事にも驚きました.

データセットについて

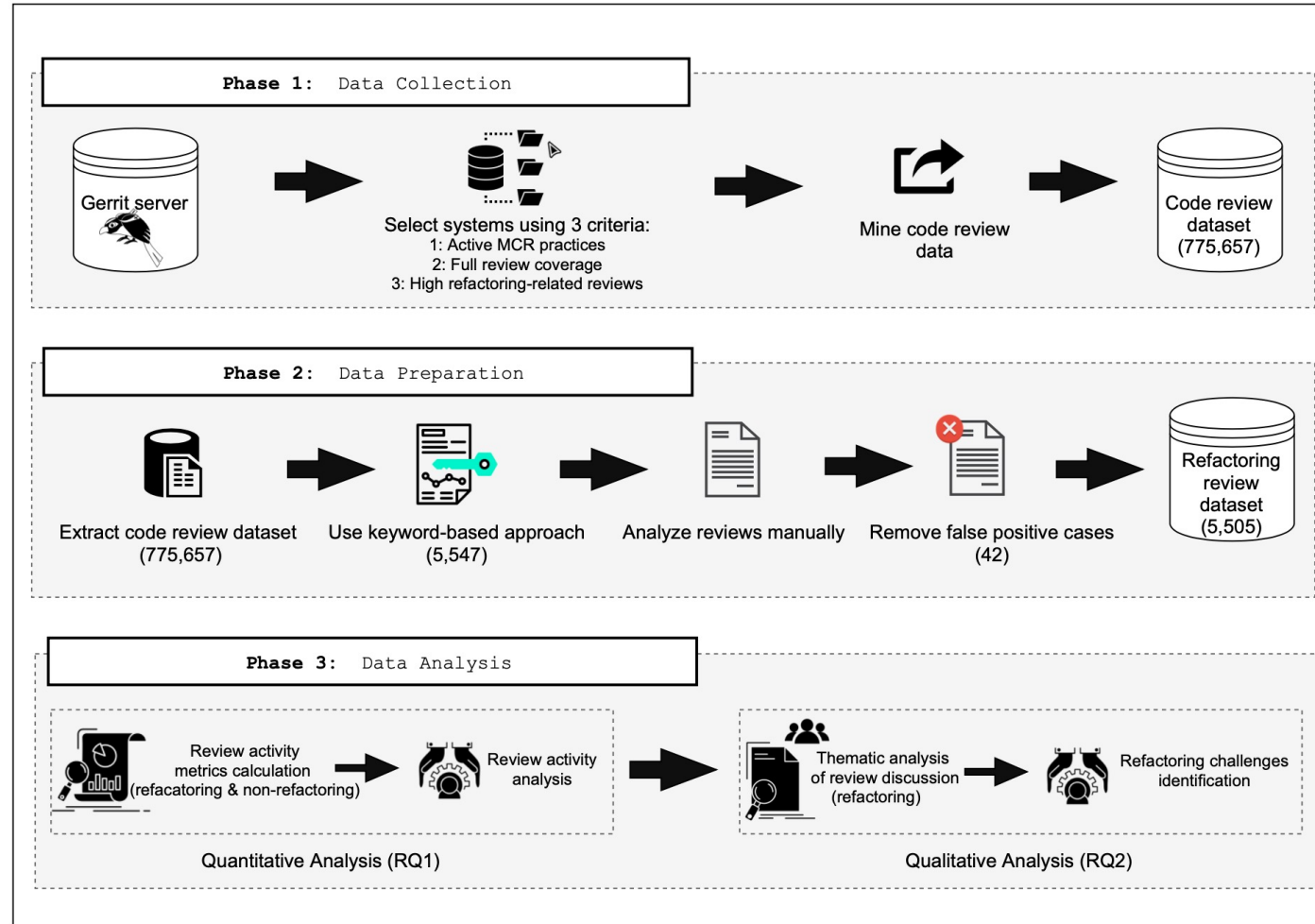


Figure 1: Overview of our experiment design.

関連研究

Table 1: Related work in refactoring-related code review.

Study	Year	Research Type	Research Method	Purpose	Evaluation Technique
Ge <i>et al.</i> [34, 35]	2014,2017	Tool-based code review	Case & formative study	Detect refactoring from non-refactoring changes	2 OSS & 35 developers
Alves <i>et al.</i> [13, 14]	2014,2018	Tool-based code review	User study	Inspect manual refactoring edits	3 OSS & 15 developers
Morales <i>et al.</i> [51]	2015	Empirical study	Case study	Understand the impact of code review on quality	3 OSS
Coelho <i>et al.</i> [24]	2019	Literature review	Systematic mapping study	Present refactoring solutions to MCR	N/A
Pascarella <i>et al.</i> [59]	2019	Empirical study	Case study	Study the effect of code review on code smells	7 OSS
Paixão <i>et al.</i> [56]	2020	Empirical study	Mining-based	Study refactoring in MCR	6 OSS
Pantiuchina <i>et al.</i> [57]	2020	Empirical study	Mining-based	Study refactoring in pull requests	150 OSS
Uchôa <i>et al.</i> [89]	2020	Empirical study	Mining-based	Study MCR & design degradation	7 OSS
Uchôa <i>et al.</i> [88]	2021	Empirical study	Mining-based	Predict design impactful changes in MCR	7 OSS
AlOmar <i>et al.</i> [3]	2021	Empirical & industrial case study	Case study & survey	Understand refactoring challenges in MCR	24 developers
Brito & Valente [20]	2021	Tool-based code review	User study	Detect refactoring during code review	8 developers
Kurbatova <i>et al.</i> [44]	2021	Tool-based code review	Mining-based	Enhance IDE representation of changes	4 OSS
Coelho <i>et al.</i> [25]	2021	Empirical study	Mining-based	Study refactoring-inducing pull requests	350 OSS
This work	2022	Empirical & case study	Mining-based	Understand refactoring practices in MCR	2,225 OSS

RQ1日本語

表3：コードレビュー活動努力の統計値

指標	リファクタリング・コードレビュー						ノンリファクタリングコードレビュー						統計上の差異	
	最小	Q1	中央値	平均値	Q3	マックス	最小	Q1	中央値	平均値	Q3	マックス	p値	クリフのデルタ (Hb)
査読者数	0	2	4	5.65	6	12	0	2	3	4.45	5	9	6.750458e-43	小 (0.15)
レビューコメント数	0	5	9	20.87	19	40	0	3	6	13.05	12	25	2.730193e-82	小さい (0.22)
インラインコメント数	0	0	0	6.61	5	12	0	0	0	3.26	1	2	2.066649e-98	中位 (0.33)
リビジョン数	1	1	3	4.79	6	13	1	1	2	3.19	3	6	1.252695e-126	小 (0.3)
変更されたファイル数	0	1	3	5.98	6	13	0	1	1	3.69	3	6	6.006899e-190	中位 (0.33)
レビュー時間 (秒)	0	37	170.68	928.21	606.22	1458.68	0	12.85	75.98	738.93	351.03	852.99	2.572584e-66	小さい (0.23)
議論の長さ (文字数)	0	168	451	3450.29	1563	3653	0	110	296	1962.39	1044	2436	6.966752e-42	小 (0.15)
説明文の長さ(文字数)	62	171	264	327.61	388	711	56	124	218	276.97	354	699	4.071541e-41	小 (0.13)
コードチャーン	-1	40	114	367.63	309	711	0	3	11	201.58	53	128	0.000000e+00	おおきい (0.64)

RQ2カテゴリー分類

Table 4: A taxonomy of the refactoring review criteria in Modern Code Review.

Category	Sub-category	Example (Excerpts from a related refactoring review discussion)
Quality	Code smell	"I do understand the desire to refactor some code to eliminate duplicate code . The purpose of the common class was to contain all of the duplicate code between the 2 drivers. This seems like a half baked approach to refactoring to accomplish that goal, when the common class should have been used as a new base. Now with this patch there are 2 classes (base and common) that contain common code."
	Internal quality attribute	"I think there is still an inheritance problem with the Base2 class (see inline) and the way python handles multiple inheritance means we have to be careful when creating the UTF8 versions of the unittest classes."
	External quality attribute	"Looking through the series i found enough such code to see that it does not look easy readable ."
	Technical debt	"I was trying to use exception.ARGInvalidState with assertRaises, but found we had some technical debts , so use Exception instead to check the error message"
	Design pattern	"I think the alternative should be what other refactor possibilities are. The proposed change indicates some of the methods that are being planned to be implemented. As an alternative to evolve this library in future we need another design pattern ."
	Coding convention	"We've seen how difficult it is to arrive at a single naming convention that works for all OpenStack projects, but as Rally expands beyond benchmarking OpenStack alone, it is quite simply impossible to create a single name format that will work for literally everything."
Refactoring	Lack of documentation	" Is there a documentation somewhere explaining how to define "extra" dependencies? It would be nice to have a link somewhere. IMHO dependencies are a very complex problem, and extra dependencies look even more complex to me."
	Refactoring correctness	"I think, this piece is incorrectly refactored . You should create neutron.conf.agent.linux with IP_LIB for linux and corresponding change for neutron.conf.agent.windows with IP_LIB for windows. Then you could import it, similar to this, how it was done previously."
	Behavior preservation	"Refactoring involves "behavior-preserving transformations", but this change does change behavior . Describing it as refactoring is misleading and suggests the change is less risky than it really is"
	Refactoring co-occurrences with changes	"I would have separate this change from your refactor . The value returned by this function is transferred to the compute manager and to the virt drivers. Some of them could have considered True/False/None to handle different behavior and even if that is not the case, to have this isolated can help reviewer to ensure all is OK."
Objective	Domain constraint	"Looking through the series i found enough such code to see that it does not look easy readable. It operates with StoragePolicy's properties which are initialized there and here we can guess only what they mean (e.g. external_boot). StoragePolicy knows all about disks, what kinds of disks exists, what disks are used by instance , etc. It would be consistent to move such operations into that class as well. Or at least to add an iterator over disks there and to use it"
	Unclear goal	"with regards to the refactor. The plan was to make a progressive refactor that would keep the old API around while the new one is adopted. The motivations behind this refactor are related to the lack of any kind of architecture and design in the current code. The current code, as it is, is not easily consumable by other services, which is the whole point of this library."
	Unknown benefit	"I would like understand what is the added value or improvement with these changes to l3-agent scheduler."
	Potential side effect	"It seems promising, but the change in pattern, if not thoroughly thought out, may lead to other unforeseen side effects ."
	Change scope	"Well I think lloSCSIDeploy name wins over lloPXEDeploy. This is no longer tied to a particular boot interface and can work with other boot interfaces (like virtual media when other driver is refactored). But I missed what Ruby pointed out, it's doing pxe-<something> for now. Needs to be refactored but may be better in another patch. We have enough content in this patch alone which refactors the generic upstream driver. So I think it's better to refactor in next patch."
	Feature support	"Isn't this a whole new feature , and not just a refactor? IMHO this should be its own spec[...] Yes, this is new feature and cannot be implemented by only this refactoring. This feature depends on the refactoring. I agree with you to create a spec"
Testing	Bug fixing	"I don't think some of this patch is needed since we merged Gage's fix. On the other hand, this change seems to be doing quite a bit of refactoring. Is there a reason to continue pursuing that refactor, or was it mainly to fix the bug ?"
	Lack of coverage	" Is there enough coverage on this to ensure that all changes are tested?"
	Absence of result	"I should read more closely... this was intended according to the commit message: "for patches that get 'rechecked' a lot, you'll see the whole history of test results on current patch."
	Poor test quality	"What do you think of continuing to clean up the test infrastructure as you've started here and create good examples of what to do? We could document how to write good tests so that authors and reviewers could ensure the quality of new additions, and require rewriting of existing tests only as needed rather than attempting the monumental effort of fixing the entire unit test tree."
Integration	Configuration issue	"IMO also some detailed configuration example and illustration is needed in etc/designate.conf - My concern is that, not specifically against patch but for whole project, we add all these new functionalities but people don't know how to use/configurate it. An example is option 'options' in many config sections. It varies a lot from different types of backend/pool_target, but we didn't document them well. People(at least me) need to read the source code to figure out how to configure that correctly . Maybe we should come up with a patch fix this in short future."
	Merge conflict	"This seems like it's mostly just going to cause merge conflicts with a lot of outstanding code."
	API management	"This seems to be a breaking change in the interface, meaning this new cli wouldn't work with the old api and vice versa. I think it would be appropriate if this new code (in the API) produced dag_execution_date on the action dictionary returned, so as to be backwards compatible with old clients, and this new client should probably sniff this for the new way, and revert to the old way if needed"
	Build failure	" Build failed (check pipeline). For information on how to proceed, see http://docs.openstack.org/infra/manual/developers.html#automated-testing ."
Management	No review activity	"This patch has been idle for a long time , so I am abandoning it to keep the review clean sane. If you're interested in still working on this patch, then please unabandon it and upload a new patchset."
	Forgotten review	"I see that this change has not been updated since April 23rd. I also see that there have been no responses from you to the questions Mark asked inline on the most recent patchset. Do you intend to continue working on this change?"
	Change dependency	"This is a significant enough change that I think it warrants discussion with the drivers team and a spec ."
	Review guideline violation	"This is much more readable than before, but I would like to see a satisfactory answer to the question of why this is necessary, before this is merged. As far as I am aware, there is no consensus from OpenStack on following this guideline ; do you have a spec for this work?"