

Don't Forget the Exception!

Considering Robustness Changes to Identify Design Problems

.....

Anderson Oliveira*, João Correia*, Leonardo Sousa[†], Wesley K. G. Assunção*[§], Daniel Coutinho*,
Alessandro Garcia*, Willian Oizumi*, Caio Barbosa*, Anderson Uchôa[‡], Juliana Alves Pereira*

*Informatics Department – Pontifical Catholic University of Rio de Janeiro (PUC-Rio), Rio de Janeiro, Brazil

[†]Department of Electrical and Computer Engineering - Carnegie Mellon University, Pittsburgh, United States

[§]Institute of Software Systems Engineering – Johannes Kepler University (JKU), Linz, Austria

[‡]Federal University of Ceará (UFC), Itapajé, Brazil

In Proceedings of 20th International Conference on Mining Software Repositories (MSR), 2023, pp.417-429

例外処理



例外処理はソフトウェアの堅牢性と信頼性を促進 [1][2]

例外処理に関する設計上の文書は提供されにくい [4][5]

－ 例外処理動作の実装が不十分 [1][7]

－ 設計者が例外処理を軽視しがちになる [8]

[1] H. Shah, C. Gorg, and M. J. Harrold, "Understanding exception handling: Viewpoints of novices and experts," *IEEE Transactions on Software Engineering*, vol. 36, no. 2, pp. 150–161, 2010.

[2] W. Weimer and G. C. Necula, "Exceptional situations and program reliability," *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 30, no. 2, pp. 1–51, 2008.

[4] F. Ebert and F. [5] R. de Lemos and A. Romanovsky, "Exception handling in the software lifecycle," *International Journal of Computer Systems Science and Engineering*, vol. 16, no. 2, pp. 167–181, 2001.

[7] D. Reimer and H. Srinivasan, "Analyzing exception usage in large java applications," in *Proceedings of ECOOP'2003 Workshop on Exception Handling in Object-Oriented Systems*, 2003, pp. 10–18.

[8] B. Jakobus, E. A. Barbosa, A. Garcia, and C. J. P. De Lucena, "Contrasting exception handling code across languages: An experience report involving 50 open source projects," in *2015 IEEE 26th International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 2015, pp. 183–193.

システムへの影響

| 例外処理の欠落は設計上の問題がある兆候^{[10]-[11]}

- 設計上の問題：非機能要件に悪影響を与える決定

| 設計上の問題に関する既存研究

- 設計上の問題をコードスメルを用いて検出^{[20]-[21]}
- 設計上の問題を示すコードスメルのパターンを分析^[23]



設計上の問題の検出にはより多くのスメルを考慮する必要



例外処理をスメルとどのように組合わせて解釈可能か不明

[10] R. Coelho, L. Almeida, G. Gousios, and A. Van Deursen, “Unveiling exception handling bug hazards in android based on github and google code issues,” in *2015 IEEE/ACM 12th Working Conference on Mining Software Repositories*. IEEE, 2015, pp. 134–145.

[11] J.Oliveira,N.Cacho,D.Borges,T.Silva,andF.Castor,“Anexploratory study of exception handling behavior in evolving android and java applications,” in *Proceedings of the 30th Brazilian Symposium on Software Engineering*, 2016, pp. 23–32.

[20] W. Oizumi, A. Garcia, L. D. S. Sousa, B. Cafeo, and Y. Zhao, “Code anomalies flock together: Exploring code anomaly agglomerations for locating design problems,” in *2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE)*. IEEE, 2016, pp. 440–451.

[21] L. Sousa, A. Oliveira, W. Oizumi, S. Barbosa, A. Garcia, J. Lee, M. Kalinowski, R. de Mello, B. Fonseca, R. Oliveira *et al.*, “Identifying design problems in the source code: A grounded theory,” in *Proceedings of the 40th International Conference on Software Engineering*, 2018, pp. 921–931.

[23] A. Oliveira, W. Oizumi, L. Sousa, W. K. Assunção, A. Garcia, C. Lucena, and D. Cedrim, “Smell patterns as indicators of design degradation: Do developers agree?” in *Proceedings of the XXXVI Brazilian Symposium on Software Engineering*, 2022, pp. 311–320.

用語

| ソフトウェアの品質を示す指標

1. 堅牢性スメル

- 例外処理に関するコードスメル ex)例外処理ブロックの中身が空

Catch Generic Exception	Method Throws Exception	Empty Catch Block	Catch NPE	Rethrows Exception	Throw New Instance Of Same Exception	Throw Exception In Finally	Exception As Flow Control	Throw NPE
-------------------------	-------------------------	-------------------	-----------	--------------------	--------------------------------------	----------------------------	---------------------------	-----------

2. 保守性スメル

- 設計上の問題を引き起こすコードスメル[23]

ex) Feature Envy, Intensive Coupling, Dispersed Coupling Long Method etc…

Design Problem	Smells Pattern
Ambiguous Interface	Long Method, Feature Envy, and Dispersed Coupling
Cyclic Dependency	Intensive Coupling and Shotgun Surgery
Concern Overload	Divergent Change, Feature Envy, God Class/Complex Class, Intensive Coupling, Long Method, and Shotgun Surgery
Fat Interface	Shotgun Surgery or Divergent Change, Dispersed Coupling, and Feature Envy
Misplaced Concern	God Class/Complex Class, Dispersed Coupling, Feature Envy, and Long Method
Scattered Concern	Dispersed Coupling, Divergent Change, Feature Envy, God Class/Complex Class, Intensive Coupling, and Shotgun Surgery
Unwanted Dependency	Feature Envy, Long Method, and Shotgun Surgery

| 堅牢性への変更

- 例外処理ブロックへの変更のこと [10]-[11]

目的 & Research Question

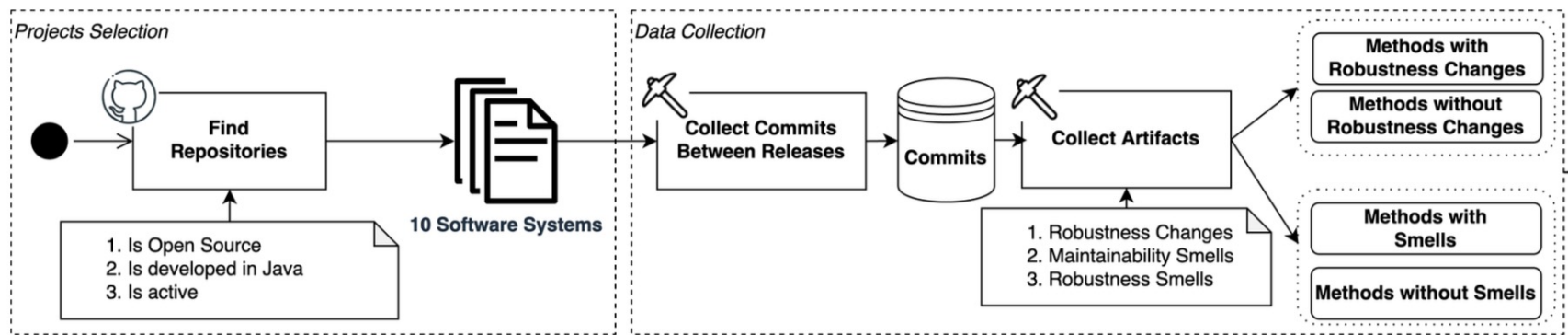
目的

設計上の問題を示す現象を,
堅牢性スメルと保守性スメルを組合せることで理解

| Research Question

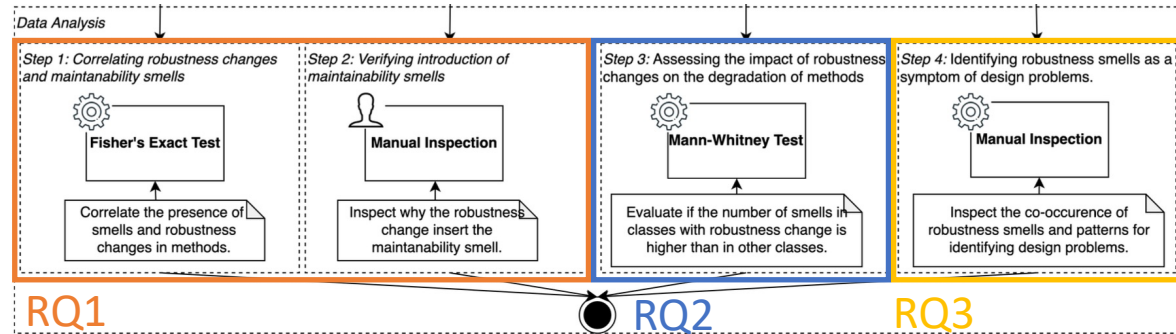
- RQ1：堅牢性への変更は保守性スメルとどの程度共存するか
 - 2つのスメルから設計上の問題を明らかにできる可能性
- RQ2：堅牢性への変更はメソッドの劣化にう影響するか
 - 保守性スメルを用いて調査
- RQ3：堅牢性スメルは、どのような設計上の問題を示すか

データ収集



Project	Start Release	End Release	Commits		Methods
			With Any Changes	With Robustness Changes	
apm-agent-java	v0.5.0	v1.28.0	1,025	484	9,949
dubbo	dubbo-2.6.12	dubbo-3.0.0	1,382	1,069	26,613
elasticsearch-hadoop	v1.3.0.M1	v8.0.0	1,120	503	3,894
fresco	v1.0.0	v2.6.0	1,165	791	7,700
netty	netty-4.1.31.Final	netty-4.1.75.Final	995	694	19,067
okhttp	parent-2.3.0	parent-3.14.0	586	455	5,933
RxJava	v1.0.10	v3.1.0	1,169	799	28,004
spring-boot	v2.7.6	v3.0.0	1,380	618	19,330
spring-framework	v5.3.24	v6.0.0	1,119	794	38,429
spring-security	5.1.0.RELEASE	5.6.0.RELEASE	1,426	690	10,062

分析手順



| **Step1** : 堅牢性への変更と保守性のスメルの相関

– 手作業でランダム抽出した206のメソッドを検査

| **Step2** : 保守性スメルを導入を追跡

– 堅牢性への変更により保守性スメルが導入されたケースを調査

| **Step3** : 堅牢性の変更がメソッドの劣化に与える影響

– Step2の変更内容分析

| **Step4** : 設計上の問題に関する堅牢性スメル of 識別

結果 | RQ1 (堅牢性への変更は保守性スメルとどの程度共存するか)

| 堅牢性への変更と保守性スメルの相関を計算

- 全システムで有意差あり(フィッシャーの正確率検定, $p < 0.05$)

保守性スメルが発生する確率が、
堅牢性への変化が
行われない場合よりも高い割合

Core System	Odds Ratio	SML + CH	SML + NoCH	NoSML + CH	NoSML + NoCH
apm-agent-java	4.755	50	941	99	8,859
dubbo	6.995	219	1,956	385	24,053
elasticsearch-hadoop	2.377	33	139	338	3,384
fresco	5.243	25	35	916	6,724
netty	6.902	45	63	1,778	17,181
okhttp	3.351	32	75	658	5,168
RxJava	5.938	48	105	1,991	25,860
spring-boot	2.192	10	62	1,320	17,938
spring-framework	4.504	31	73	3,302	35,023
spring-security	2.458	50	234	782	8,996

Odds Ratio: Odds ratio identified through the Fisher test

SML + CH: # of methods with maint. smells and robust. changes

SML + NoCH: # of methods with maint. smells and without robust. changes

NoSML + CH: # of methods without maint. smells and with robust. changes

NoSML + NoCH: # of methods without maint. smells and robust. changes

堅牢性への変更を行う際は、保守性スメルを意識する必要がある

結果 | RQ1 (堅牢性への変更は保守性スメルとどの程度共存するか)

| 赤枠3つの保守性スメルは特に堅牢性への変更で誘発

Software System	Odds Ratio							
	Brain Method	Dispersed Coupling	Feature Envy	Intensive Coupling	Long Method	Message Chain	Long Parameter List	Shotgun Surgery
spring-security	NR	11.002	2.477	NR	6.291	NR	NR	11.206
apm-agent-java	NR	6.952	8.153	9.058	13.431	2.482	NR	NR
dubbo	10.330	14.171	8.198	10.191	9.305	3.168	1.936	10.330
elasticsearch-hadoop	NR	7.214	2.968	4.607	3.971	3.002	NR	8.032
fresco	NR	9.299	8.878	NR	7.205	NR	2.780	NR
netty	NR	13.033	6.126	9.930	14.293	NR	2.461	NR
okhttp	NR	3.862	5.933	9.743	9.059	2.929	NR	9.011
RxJava	NR	10.049	9.820	NR	11.272	NR	3.387	21.686
spring-boot	NR	NR	NR	NR	NR	NR	3.682	NR
spring-framework	NR	15.411	4.517	NR	5.447	3.425	4.124	NR

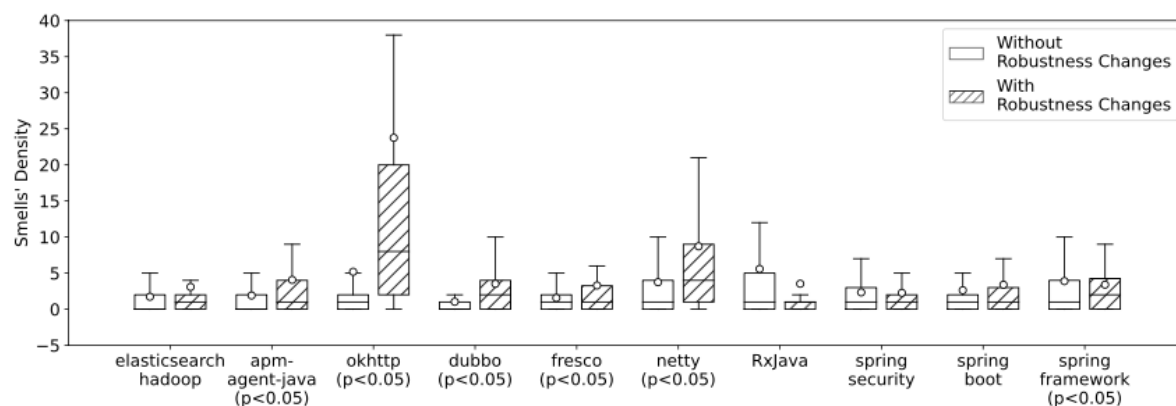
3種類の保守性スメルの影響を受けた206メソッドを持つクラスで
堅牢性への変更が行われたコミットを選択・評価（手作業）

RQ1

少なくとも74メソッド(35.9%)が
堅牢性への変更によって保守性スメルが導入

結果 | RQ2(堅牢性への変更はメソッドの劣化にどう影響するか)

| 堅牢性への変更の有無でメソッドの保守性スメル密度比較



- 6システムで、堅牢性への変更がされたメソッドの方が保守性スメル密度が高い (U検定, $p < 0.05$)
- 有意差がない場合でも変更がある場合は平均値, 中央値が高い

RQ2

堅牢性への変更を受けたメソッドはそうでないメソッドよりも保守性スメル密度が高い傾向

結果 | RQ2(堅牢性への変更はメソッドの劣化にどう影響するか)

| 目視の結果

- よく書かれた堅牢性への変更は保守性スメルが少ない
- 保守性スメルが多いメソッドは例外処理ブロックが不十分

| 適切な例外処理がされていないメソッドは適切にされているメソッドよりもより品質が低い

- LongMethods, DispersedCouplings, FeatureEnvy などが発生
- システムのモジュール性を阻害する可能性

結果 | RQ3(堅牢性スメルはどのような設計上の問題を示すか)

| 堅牢性スメルと保守性スメルが共存する場合を調査

Pattern	Robustness Smell								
	Catch Generic Exception	Method Throws Exception	Empty Catch Block	Catch NPE	Rethrows Exception	Throw New Instance Of Same Exception	Throw Exception In Finally	Exception As Flow Control	Throw NPE
Concern Overload	212	1	54	2	7	0	3	7	3
Cyclic Dependency	74	5	6	0	6	0	3	7	1
Fat Interface	171	2	6	0	1	0	0	1	1
Misplaced Concern	394	5	199	3	8	1	0	1	7
Scattered Concern	35	0	1	0	0	0	0	1	0
Unwanted Dependency	274	2	81	2	7	0	0	1	4
Total	1160	15	347	7	29	1	6	18	16

– 2種類の堅牢性スメルが保守性スメルと多く共存

Ex) try-catchが入れ子, Nullだけ返す例外処理ブロック

| しかし, 堅牢性スメルは直接設計上の問題に関係ない

– 保守性スメルが直接設計上の問題へ影響している可能性

まとめ

| 目的

- 堅牢性への変更と保守性スメルが設計上の問題を理解するのに役立つかを理解

| 結果

- 堅牢性への変更が保守性スメルを導入・悪化させる
- 堅牢性スメル(`EmptyCatchBlock`, `CatchGenericException`)は,保守スメルと併用すると設計上の問題を特定可能

所感

| RQ3の分析が薄い

- 結果もビミョーな出来
- 表の理解に時間がかかった

| 結局、堅牢性スメルの定義は？

- 堅牢性スメルや保守性スメルは一般的な用語？
- 初めて聞いた

補足資料 | 設計上の問題と保守性スメル

Design Problem	Smells Pattern
Ambiguous Interface	Long Method, Feature Envy, and Dispersed Coupling
Cyclic Dependency	Intensive Coupling and Shotgun Surgery
Concern Overload	Divergent Change, Feature Envy, God Class/Complex Class, Intensive Coupling, Long Method, and Shotgun Surgery
Fat Interface	Shotgun Surgery or Divergent Change, Dispersed Coupling, and Feature Envy
Misplaced Concern	God Class/Complex Class, Dispersed Coupling, Feature Envy, and Long Method
Scattered Concern	Dispersed Coupling, Divergent Change, Feature Envy, God Class/Complex Class, Intensive Coupling, and Shotgun Surgery
Unwanted Dependency	Feature Envy, Long Method, and Shotgun Surgery