A Study on Developers' Perceptions About Exception Handling Bugs

Felipe Ebert and Fernando Castor
Informatics Center
Universidade Federal de Pernambuco
Recife, PE, Brazil
{fe, castor}@cin.ufpe.br

***Abstract*—Several studies argue that exception handling code usually has poor quality and that it is commonly neglected by developers. Moreover, it is said to be the least understood, documented, and tested part of the implementation of a system. However, there are very few studies that attempt to understand developers' perceptions about exception handling, in general, and exception handling bugs, in particular. In this paper, we present the results of a survey conducted with 154 developers that aims to fill in this gap. According to the respondents of the survey, exception handling code is in fact documented and tested infrequently. Also, many of the respondents have had to fix exception handling bugs, in particular those caused by empty catch blocks or exceptions caught unintentionally. The respondents believe that exception handling bugs are more easily fixed than other kinds of bugs. Also, we found out a significant difference in the opinion of the respondents pertaining to the quality of the exception handling code: more experienced developers tend to believe that it is worse. We present a comprehensive classification of exception handling bugs based on the study results.**

***Index Terms*—exception handling; bugs; survey.**

I. Introduction

Several modern object-oriented programming languages implement exception handling. Nevertheless, developers tend to focus on the normal behavior of the applications and deal with error handling only during the system implementation, in an ad hoc manner. This practice creates a proper situation for the appearance of design faults (bugs) related to exception handling. Several studies [1], [2], [3] argue that exception handling code usually has poor quality and that it is commonly neglected by developers.

There is no study that attempts to understand developers' perceptions about exception handling bugs. In this paper, we present the results from a survey we conducted asking 154 developers about their practices and perceptions regarding exception handling bugs. To the best of our knowledge, this is the largest study to date that aims to uncover what developers think about exception handling.

Respondents of our survey believe that exception handling bugs are more easily fixed than other kinds of bugs and also that exception handling code is in fact documented and tested infrequently. Also, most responses have claimed that their use of exception handling stems mainly from an interest in improving code quality. Nevertheless, according to them, organizations seldom have policies regarding the documentation and testing of exception handling code.

There is also a significant difference in the opinion of the respondents pertaining to the quality of the exception handling code: more experienced developers tend to believe that it is worse. Furthermore, most of them have at some point fixed exception handling-related bugs. Common causes include empty `catch` blocks and exceptions caught unintentionally. Finally, based on the data we collected, we present a proposal of classification of exception handling bugs.

II. Methodology

In this section we describe the methodology of this study. It aims to answer four research questions:

- Do organizations and developers pay attention to exception handling?
- How commonplace are exception handling bugs?
- Are exception handling bugs harder to fix than other bugs?
- What are the main causes of exception handling bugs?

What is an exception handling bug? There is no widely accepted definition of exception handling bug. Our goal is to study bugs where exception handling is associated with the cause of the problem. Bugs in exception definition, exception throwing, exception handling, exception propagation, exception documentation, and clean-up actions (`finally` blocks) are all of interest. Therefore, an exception that is not thrown when it should be according to the expected behavior of the system is an exception handling bug. The same applies to a `catch` block that captures exceptions that it should not.

On the other hand, if a program performs a division by zero, that error will be trapped by the runtime system of the language (in the case of Java and C#, among others) and an exception will be thrown. The raising of the exception in this case is not the cause of the problem. From a bug fixing perspective, as soon as the bug is fixed, the part of the code where the bug manifested will not necessarily have a relationship with exception handling anymore. Therefore, we do not consider this to be an exception handling bug.

Survey. The survey was designed according to the recommendations of [4], [5] and our target population consists of developers with some experience in Java. After defining all

the questions of the questionnaire, we obtained feedback iteratively and clarified and rephrased questions and explanations. We had particular care in explaining what we meant by an exception handling bug. Together with the instructions of the questionnaire, we included some simple examples in an attempt to clarify our intent. Table I presents a summary of the questions of the survey. The complete list of questions as well as all the responses to the survey are available at the companion Web site[1].

The survey was sent to two distinct groups of people. The first one, with 96 responses, consisted mostly of Portuguese speaking developers in Brazilian companies and universities. The second group, with 58 responses, consisted of bug reporters and developers of Eclipse and Tomcat. We sent more than 4000 emails to reporters of bugs in the repositories of these two systems. In total we obtained 154 responses during a period of 2 months.

III. What Developers Say About Exception Handling Bugs

The main goal of the survey is to understand developers' perceptions about exception handling bugs. It is known that non-trivial systems usually have bugs that are difficult to find, stemming from complex control flow and overly general `catch` blocks [6] and from I/O operations [7]. However, even though the perceptions of developers about exception handling in general have been studied, at least on a small scale [3], to the best of our knowledge there are no studies that attempt to understand how developers regard exception handling bugs.

Respondents had on average between 7 and 10 years of software development experience (question 1). Most of them are currently working on medium-sized projects ranging between 50 and 100KLoC (question 2). Most respondents have developed systems using at least three different programming languages (question 3). In the remainder of this section we discuss the main findings of the survey based on the four research questions presented in Section II.

Do organizations and developers pay attention to exception handling? The survey included five questions (4, 5, 9, 18 and 19) whose goal is to determine whether developers worry about exception handling when they are not directly implementing the system, e.g., designing, testing, etc. For question 5, only 27% of the developers answered "yes". In a similar vein, 30% said that there are specific tests for exception handling code in their organization (question 9). And for question 4, 61% of the respondents said that none to little importance is given to the documentation of exception handling. Moreover, just 16% said that much or very much importance is given to exception handling documentation.

For question 18, about 40% of the respondents consider that the quality of exception handling code ranges between good and very good. Surprisingly, only 14% of the respondents consider that it is bad or very bad. This result correlates with developer experience. We found that the more experienced a respondent, the greater his/her tendency to consider the quality of exception handling code to be bad with the Student's T-test producing a p-value of 0.02478.

Inspired by the work of Shah et al. [3], we asked question 19. We provided them with a list of possible causes and gave them the opportunity to suggest additional ones. Table II presents the reasons more frequently cited by the respondents. Most of the respondents said that creating ways to tolerate faults and improving the quality of a functionality are the main reasons to use exception handling. One of the survey respondents provided a particularly interesting spontaneous answer for this question:

> *"exception handling is part of code flow - not using exception handling for some arbitrary reason would be like not using 'if' blocks, or not having your code compile."*

Only 17% of the respondents said they use exception handling for debugging purposes. In contrast, in the work of Shah et al. [3], which interviewed a group of 8 novice developers and 7 experts, most of the novice developers claimed to use exception handling mostly for debugging and because of language requirements. Expert developers, on the other hand, claimed that they use exception handling mainly to convey understandable failure messages. In contrast, in our study the most experienced and the least experienced respondents coincided in terms of both the most often cited reasons and the least often cited reasons (Table II).

Based on previous work discussing the problems with checked exceptions in Java [8], we expected a larger percentage of respondents to mention "language requirement" as a reason for using exceptions. Moreover, only 21% use exception handling because of organizational policies. This result and the previously discussed ones pertaining to testing and documentation suggest that organizations do not pay attention to exception handling, even though developers do. Finally, it is interesting to note that the 3 respondents who claimed not to use exception handling have professionally worked only with languages that implement exception handling and all of them have professionally worked with Java.

Developer experience does not seem to make a difference in the main reasons for the use of exception handling. For both the least experienced (until 5 years of development experience) and the most experienced (10+ years) respondents, improving the quality of a feature and creating ways to tolerate faults are the most common reasons to use of exception handling. Also, for both groups, organization policies and the need to debug the code are the least often cited reasons.

TABLE II
WHY DO DEVELOPERS USE EXCEPTION HANDLING

To create ways to tolerate faults	66%
To improve the quality of a functionality	63%
Importance of exception handling	53%
Language requirement	43%
Organizational policies	21%
To debug a specific part of the code	17%
Does not use exception handling	2%

How commonplace are exception handling bugs? To assess

[1]Address: https://sites.google.com/a/cin.ufpe.br/eh-bugs/

TABLE I
SUMMARY OF SURVEY QUESTIONS

Experience	1. For how long have you been a Java developer?
	2. What is the approximate size of the project you are currently working on (LoC estimate)?
	3. Which programming languages have you professionally worked with?
Context	4. In the design phase of your projects, what importance is given to the documentation of exception handling?
Documentation	5. Are there any specifications, documented policies or standards that are part of your organization's culture related to the implementation of error handling?
	6. How often are bugs related to exception handling reported at your organization?
	7. How often are bugs reported at your organization?
	8. Does your organization use any tool for reporting and keeping track of bugs?
Testing	9. Are there specific tests for the exception handling code in your organization?
Bugs	10. How often do you find bugs related to exception handling?
	11. How often do you find bugs that are not related to exception handling?
	12. Estimate the percentage of bugs related to exception handling code in your projects
	13. Have you ever needed to fix bugs related to exception handling?
	14. If you answered yes to question 13, please describe some of these situations
	15. Select the main causes of bugs related to exception handling you have ever needed to fix, analyze or have found documented
	16. What is the average level of difficulty to fix bugs related to exception handling?
	17. What is the average level of difficulty to fix other bugs that are not related to exception handling?
	18. What is your opinion about the quality of exception handling code in your projects compared to other parts of the code
	19. Why do you use exception handling in your projects?

how commonplace developers believe exception handling bugs to be, we asked them questions 10 and 11. The answers were given in a scale comprising: "never", "rarely", "sometimes", "most of the time" and "always". We put the answers in a numeric scale and used Student's T-test to check whether the answers are significantly different. According to the respondents, exception handling bugs are less frequently found than other bugs. We obtained a p-value of less than 0.0001. We also asked them to estimate the percentage of exception handling bugs in their projects (question 12). The mean estimate was 9.79% and the median was 5%.

Complementarily, we presented the respondents with the questions 6 and 7. Most of the respondents answered that exception handling bugs are reported "sometimes" (question 6) and other bugs are reported "most of the times" (question 7). The answers for the two questions differ significantly with a p-value of less than 0.0001.

Are exception handling bugs harder to fix than other bugs? Our study revealed that respondents consider exception handling bugs easier to correct than other types of bugs. 43% of the respondents consider exception handling bugs to be easy or very easy to fix (question 16). In sharp contrast, only 7% of the respondents say the same about other kinds of bugs (question 17). The answers for questions 16 and 17 were given in a scale comprising: "very easy", "easy", "medium", "hard" and "very hard". We put the answers in a numeric scale. Then we employed the T-test to analyze whether the answers for the difficulty of fixing exception handling and other bugs are significantly different. We found a p-value of less than 0.0001, thus, according to the respondents, exception handling bugs are easier to fix than other bugs.

What are the main causes of exception handling bugs? To uncover the main causes of exception handling bugs according to the survey respondents, we posed three questions (13, 14 and 15). Question 15 directly asked them about the main causes. The respondents were allowed to select zero or more causes from a list and could also suggest additional ones. The most commonly cited cause for exception handling bugs was the "lack of a handler that should exist". Followed by the causes "no exception thrown in a situation of a known error" and "programming error in the `catch` block".

To better understand developer perceptions about the causes of exception handling bugs, we asked questions 13 and 14. 83% of the respondents have had to fix an exception handling bug at some point (question 13) and 113 out of 154 survey respondents answered the latter question (question 14). The answers varied widely and many of them refer to specific technologies, frameworks and applications. Out of the 113, 19 had to fix bugs caused by empty `catch` blocks and 16 have fixed bugs stemming from exceptions caught unintentionally. The final result of this question is the exception handling bug classification presented in Table III.

TABLE III
EXCEPTION HANDLING BUGS CLASSIFICATION.

1. Lack of a handler that should exist
2. No exception thrown in a situation of a known error
3. Programming error in the `catch` block
4. Programming error in the `finally` block
5. Exception is caught unintentionally
6. Catch block where only a finally would be appropriate
7. Exception that should not have been thrown
8. Wrong encapsulation of exception cause
9. Wrong exception thrown
10. Lack of a `finally` block that should exist
11. Error in the exception assertion

There are diametrically different opinions on the subject of empty `catch` blocks. Even though a number of respondents consider them to be sources of bugs, some survey respondents seem to disagree:

> *"I'm also an Eclipse committer (on Platform/UI). On 4.2 we've changed how parts (e.g., editors and views) are rendered. Our new system silently swallows otherwise-uncaught exceptions. Tracing what happens when an EditorPart or ViewPart throw an uncaught exception is a teensy bit annoying."*

The citation above shows that the respondent does not want to know about problems in some parts of the system and

swallows exceptions to achieve that goal.

IV. Threats to Validity

Internal Validity. Our survey was conducted with an online and self-administered questionnaire. It is possible that respondents may have misunderstood the definition of exception handling bug and answered the questions based on a different understanding meaning. We tried to reduce the probability of this occurring by providing a clear definition of exception handling bug and some simple examples in the survey instructions.

External Validity. Our survey involved 154 respondents. This number is small and limits the generalizability of the results. Nonetheless, respondents of the survey came from different professional and cultural backgrounds. Furthermore, the largest study to date on the viewpoints of developers about exception handling [3] involved only 15 respondents (they were interviewed, instead of responding to a survey). Therefore, from a comprehensiveness standpoint, we can say that our study is an improvement over the current state-of-the-art.

Construct Validity. Our survey might not have covered all questions that we could have been asked of the respondents. Nonetheless, the final questionnaire was the result of several discussions between the authors, one of them a specialist in exception handling, and with a number of software developers and academics. Moreover, we run at least two small pilot studies before finally making the questionnaire public.

V. Related Work

The study most strongly related to ours was conducted by Shah et al. [3]. It involved 8 novice and 7 expert software developers and had the goal of understanding their viewpoints on exception handling. The authors conducted semi-structured interviews with developers and the results show that novice developers neglect exception handling until there is an error or until they are forced to address the problem due to language requirements. Furthermore, they do not like being forced by the language to use exception handling constructs. In contrast, the experienced developers think that exception handling is a very important part of development.

Marinescu [9] and Sawadpong et al. [10] analyzed bug reports from Eclipse. Marinescu [9] analyzed the defect-proneness of classes that use exception handling. Her study revealed that classes that throw or handle exceptions are indeed more defect-prone than others classes. Sawadpong et al. [10] aimed to determine if the usage of exception handling is relatively risky by analyzing the defect densities of exception handling code and the overall source code. They discovered that exception handling defect density of exception handling constructs is approximately three times higher than overall defect density.

None of the two aforementioned studies have also accounted for the perceptions of developers about exception handling bugs. Furthermore, none of them analyzes issues such as whether exception handling bugs are easier to fix nor what are their causes.

VI. Conclusions and Future Work

This paper presented a study on exception handling bugs based on a survey with software developers and researchers. The results show that exception handling code is not documented and tested frequently and also that developers assume that fixing exception handling bugs is easier than fixing other bugs. Also, developers claim that they use exception handling mainly to improve the quality of the systems they produce and also more experienced developers tend to believe that the quality of exception handling code is worse.

We also present a comprehensive classification of exception handling bugs based on the study results. The results of this study emphasize that the views of developers and organizations about exception handling bugs are in conflict. To improve the quality of software systems, these views must be reconciled so that exception handling code can receive more attention. The presented classification of exception handling bugs can provide assistance in that task, e.g., by working as a checklist for code inspections or a guide to the design of test cases.

In the future, we plan to conduct interviews with developers since very useful information in our study came from spontaneous answers provided by the respondents of the survey. Also, we are currently analyzing bug reports from two real systems so that we can compare what developers say and what they do in fact, when it comes to exception handling bugs.

VII. Acknowledgements

We would like to thank all respondents of the survey that took some precious time to answer our questionnaire. We also would like to thank the anonymous referees, who helped to improve this paper. Fernando is supported by CNPq (306619/2011-3), FACEPE (APQ-1367-1.03/12), and by INES (CNPq 573964/2008-4 and FACEPE APQ-1037- 1.03/08).

References

[1] F. Cristian, "Exception handling," in *Dependability of Resilient Computers*. Blackwell Science, 1989, pp. 68–97.

[2] D. Reimer and H. Srinivasan, "Analysing exception usage in large java applications," in *Proceedings of ECOOP Workshop on Exception Handling in Object-Oriented Systems*, July 2003, pp. 10–19.

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

[4] R. M. Groves, F. J. Fowler, M. P. Couper, J. M. Lepkowski, E. Singer, and R. Tourangeau, *Survey Methodology*, 2nd ed. Wiley, 2009.

[5] B. Kitchenham and S. L. Pfleeger, "Personal opinion surveys," in *Guide to Advanced Empirical Software Engineering*, F. Shull, J. Singer, and D. I. K. Sjoberg, Eds., 2008, pp. 63–92.

[6] M. Robillard and G. Murphy, "Static analysis to support the evolution of exception structure in object-oriented systems," *ACM TOSEM*, vol. 12, no. 2, pp. 191–221, April 2003.

[7] P. Zhang and S. G. Elbaum, "Amplifying tests to validate exception handling code," in *Proc. of the 34th ICSE*, June 2012.

[8] B. Cabral and P. Marques, "Exception handling: a field study in java and .net," in *Proc. of the 21st ECOOP*. Springer-Verlag, 2007, pp. 151–175.

[9] C. Marinescu, "Are the classes that use exceptions defect prone?" in *Proceedings of the 12th International Workshop on Principles of Software Evolution*, September 2011, pp. 56–60.

[10] P. Sawadpong, E. B. Allen, and B. J. Williams, "Exception handling defects: An empirical study," *9th IEEE International Symposium on High-Assurance Systems Engineering*, pp. 90–97, October 2012.