

The Effect of Field Data Integrity on Failure Reproduction and Fault Isolation

Madeline Hardojo, Sebastian Elbaum, and Zhimin Wang

Department of Computer Science and Engineering
University of Nebraska - Lincoln
Lincoln, Nebraska
{mhardojo,elbaum,zwang}@cse.unl.edu

Abstract

Failure reproduction and fault isolation are key activities in the post-release phase. As we develop techniques to exploit field data to accelerate and improve these activities, we must be aware of the impact of field data integrity. This work scopes and provides details about a study we are conducting to quantify the impact of users' contribution and filtering processes in the integrity of field data, and their effect on failure reproduction and fault isolation techniques.

1. Introduction

Learning from field failures is an essential step towards the development of more reliable software systems. Capturing the circumstances that led to the exposure of faults in the field could enable and accelerate failure reproduction and fault isolation. This fact is not overlooked by commercial efforts. Microsoft, for example, has developed an error reporting service to collect registry, processes, and machine data immediately after a failure [7] and Netscape relies on users to provide failure details through a structured report mechanism. Microsoft has also stated that error reporting helped them in addressing 29% of errors in Windows XP and more than 50% of errors in Office XP [1]. Furthermore, in a recent study, we found test cases derived from field data were able to detect 12% of faults not found by the in-house test suite [2].

The integrity of collected field data, however, is frequently questionable, which compromises its potential benefits for failure reproduction and fault isolation. Current practices for field data collection often render incomplete, ambiguous, and non-standardized information [6, 9]. For example, automatically collecting field data at the time of a failure can only provide partial information because the circumstances leading to the failure are not recorded and structured reporting mechanisms (e.g. Bugzilla) cannot compen-

sate for either the lack of users' knowledge or willingness to fill proper reports.

If we are to develop techniques that leverage field data for failure reproduction and fault isolation, we must have a clear understanding of the potential impact of field data integrity on those activities. In this work we start to frame a study aim at quantifying collected field data integrity and its effect on failure analysis activities. We are particularly interested in two factors that affect field data integrity: user contribution and the monitoring filtering processes.

The integrity of field data depends to a great degree on the attitude, resources, and knowledge of the customers. A recent Microsoft's report states that "a substantial percentage of customers" are willing to return error reports, but no data is made available to support that statement [4]. The customers who are not willing to return error reports, or do not have the resources to report them all the time, or just intentionally return spurious reports affect the integrity of field data and can adversely affect the failure analysis process.

When profiling a large deployed base the amount of captured field data can be astounding. In a preliminary deployment of the system we will be presenting in Section 3.1, we found that an average session data size is approximately 200 KB and there are about 5 sessions per day per user. Since we are aiming for a user population of a thousand users, the projected data will reach 30GB at the end of the study (30 days). Filtering processes can control the potential information overload by sifting the valuable data. For example, Hilbert and Redmiles in [5] suggested a filtering approach using usage expectations, i.e., developers' expectations about how users would use the system. Only when the monitoring agent detects that users' behavior deviates the usage expectations, the user data is reported back to the developers. This filtering process, however, can also affect field data integrity by discarding potentially valuable information for failure analysis.

In the following section, we state the research questions

we would like to address in this study. Section 3 describes the program, experimental setting, and the planned analysis to address our research questions. Section 4 makes the final remarks.

2. Research Questions

We have organized our research questions in two groups. The first group of research question is summarized in Table 1 and deals with the potential of collecting valuable field data for failure reproduction and fault isolation. The first row in the table contains questions about the raw potential of field data and serves for us to establish a feasibility upper bound. The questions about the potential of a failure reproduction technique (second row) is relevant because without a successful failure reproduction engineers have no way to know whether the failure is "real", whether it is important enough to be attended, and much less how to fix it. After a field failure has been replicated, engineers proceed with the isolation of the fault or faults leading to that failure. The questions listed in the third row attempt to assess the effectiveness and required effort of a fault isolation technique, which is important to enable an efficient correction or preventive actions to avoid failures [4].

Table 1. Questions regarding the potential of field data

Relevance	Question
Upper bound on the potential of field data	How many failures occurred in the field? How many faults were exposed in the field? What is the ratio against in-house failures? Is there any type of faults more likely to be found in the field? How is the distribution of failures across the user base?
Potential of a failure analysis technique	What is the percentage of failures it can reproduce? What effort is involved in such process? How accurate is the reproduction?
Potential of a fault isolation technique	What percentage of faults can be isolated using field data? How much field data is required to isolate a fault?

The second set of questions listed in Table 2 focuses on the effect of user contribution and filtering processes on the integrity of field data and on the effects on failure reproduction and fault isolation techniques.

Table 2. Questions regarding the effect of filtering processes and user contribution

Relevance	Question
Lack of integrity due to user contribution	What percentage of users is willing to send a failure report? What percentage of sessions is not collected due to resource constrains? How many failures go unreported?
Lack of integrity due to use of filtering technique	What percentage of sessions are filtered out? What percentages of failures occurred in the filtered out sessions?
Sensitivity of failure reproduction and fault isolation	How sensitive is each technique to the variation in data integrity due to filtering and user contribution?

3. Study Design

This section shortly describes the program and the experimental setting we will use to address the research questions.

3.1 Program Preparation

We chose MyIE as the experimental object. MyIE is a web browser that uses the core of Microsoft Internet Explorer (IE 5.0 or higher) but adds several features to IE (e.g., different front end, mouse gesture capability, popup filters, URL aliasing). Figure 1 shows a snapshot of MyIE. MyIE is a good experimental object because its source code is available for us to manipulate, it resembles a popular web browser which will make it easier for us to attract a considerable user base, and it is much larger than any program used for this type of study (67K size in LOC for all C and h files).

To study the potential of field data in relation to the in-house validation activities, we required an initial test suite and a set of faults. We generated a set of test cases for the program using first a category partition approach, and later complemented by white box testing. We automated the test cases using the Vermont HighTest generating a set of 176 test cases that takes approximately 100 minutes to execute on a PC with an Intel Pentium 4 1.8GHz processor, 256 MB memory, running Microsoft Windows XP Professional. The test suite yields 63.33% of block coverage and 70.6% function coverage.

One of the authors, closely familiar with the development of MyIE but not involved in the experimental goal or design, seeded 50 faults according to a procedure employed in previous studies of testing techniques [3]. We restricted the seeding to components which did not affect the GUI due to the failure detection capability. In other words, we



Figure 1. Snapshot of MyIE browser

are not interested in cosmetic failure. Our test suite exposed 7 of the seeded faults, which we removed from the program before deployment (as it would happen in practice).

3.2 Subjects

This study continues our series of studies on techniques that leverage field data for validation purposes [2]. As such, we want to address some of the threats to external validity from the previous studies by obtaining a larger number of participants to use MyIE.¹

Our approach to attract potential users includes gathering personal contacts (as in the previous study), setting and promoting a download page (that also includes logic to control the deployment and version management process)², and involving various newsgroups to promote the experimentation. We have identified 3 prospective newsgroups: software testing newsgroup (comp.software.testing), human factors in computer newsgroup (comp.human-factors), and research newsgroup (sci.research). Furthermore, we are promoting MyIE by listing its special features and characteristics. By doing this, we hope to convince potential participants to want to use the object program.

To entice participation we are offering a monetary price to the person that satisfies our criteria including: traversing certain program menu items, using special features of MyIE, referring another person to the study, and using the program at least once a day.

Although a price may be perceived as a threat to internal validity, the criteria to be satisfied is very general. Therefore, we are not restricting or encouraging certain behaviors from the participants. Furthermore, we expect for the price not to be the only reason why people participate in the study.

¹The participation of users in this study was approved in IRB #2004-01-126 FB dated 2/26/04

²The website can currently be viewed at <http://cse-sgodd-009.unl.edu:8080/mapstext/index.html>

3.3 Collection Mechanisms

To collect the desired data, we instrumented the source code of MyIE to provide block and menu trace information, users' input sequence, configuration information, and the state of users' computer when running the program (e.g., operating system, size of available physical storage space, memory, type and speed of processor).³

The collected data at each deployed site is packaged, time stamped, and sent to a central repository via a dedicated socket. Checks for connectivity are performed before transferring the data to the repository. In the presence of a failed connection, data is stored temporarily until the next usage and a counter is updated to keep track how many times a session failed to be sent. Once received at the collection site, the field data is unpacked and stored in their corresponding database table to allow easy access through queries statements, which provides flexibility to simulate different experimental scenarios.

The managing of failure scenarios requires a special handling in the data collection procedure. We overwrite the default windows settings to redirect protection errors through an exception handling filter to catch all program crashing failures. This allows us not only to manage the failure and direct information to our site, but also keep the integrity of the data collection buffer and allow us to complete our data collection.

To study the effect of users' contribution to field data integrity we modified the program to allow users to input whether or not they would send the collected data at the time the failure occurred (if the choice was given). At the beginning of the experiment it will be made clear to the subjects, however, that the collected data is central to the research questions and will always be sent to the repository. We are also aware that there are other valuable feedback information we could obtain from users through the same mechanism but decided against it to increase user acceptance of MyIE.

3.4 Planned Analysis

We are now going to shortly describe how we are planning to address each research question.

On assessing field data. The collection mechanisms we have put in place will provide all the data we require to determine the number and types of field failures, the faults triggering failures, and the distribution across the user base. Since we acknowledge that the potential of field data may be dependent on the in-house testing activities, we plan to simulate various in-house testing adequacy and compare the

³Note that we had to refrain from collecting navigational patterns due to privacy concerns.

potential gains achieved in the field when the deployed software is subjected to a certain level of adequacy during its in-house testing.

On replicating field failures. The failure reproduction techniques we are developing consist of imitating the state of the program before the failure, approximating user inputs and submitting it to the program, and checking whether the outcome consists of a failure of the type captured by our handler at the client site. Each one of those phases can employ different levels of detail and require distinct levels of tester participation, which leads to the creation of many techniques. For example, for inputs, one could consider users mouse clicks, user clicks and keyboard presses, all user inputs and lists of running process, or all user inputs, lists of running process, and device configuration. Some of these techniques can be fully automated while others require tester participation to fill some gaps (e.g., provide an input to traverse a specific path). We plan to assess each technique in terms of the percentage of failures replicated and the effort to apply them.

On isolating faults. We plan to start by assessing existing fault isolating techniques (e.g., minimizing input that cause failures [9], using predicate elimination strategies to isolate deterministic faults [6]). As with failure analysis techniques, we will vary the type and detail of field data supporting these techniques to evaluate their effectiveness. For example, we plan to manipulate the amount of failed sessions provided to a fault isolation technique to assess its accuracy.

On users contribution and filtering processes. To evaluate the user related effects without jeopardizing data collection, we plan to use the collected data on user feedback and connectivity to simulate those effects off-line. Simulation will also be employed to set scenarios with various filtering techniques (e.g., based on anomaly detection [3], based on distributed profiling [8], based on statistical sampling [2]). These simulations serve to create subsets of the field data by removing sessions that did not reach the repository. Then, failure reproduction and fault isolation techniques will be applied to measure their sensitivity to field data integrity.

4. Final Remarks

The work presented is still in progress. We would like to invite the workshop participants to suggest further questions and additional means to utilize the data we are gathering and the analysis we will be conducting.

References

- [1] S. Ballmer. Connecting with customers. <http://www.microsoft.com/mscorp/execmail/2002/10-02customers.asp>, Oct. 2002.
- [2] S. Elbaum and M. Harjojo. An empirical study of profiling strategies for released software and their impact on testing activities. Accepted to ISSTA, 2004.
- [3] S. Elbaum, S. Kanduri, and A. Andrews. Anomalies as precursors of field failures. In *International Symposium of Software Reliability Engineering*, pages 108–118, 2003.
- [4] J. Hamilton. Active server availability feedback. In *First Biennial Conference on Innovative Data Systems Research*. Online Proceeding, January 2003.
- [5] D. Hilbert and D. Redmiles. An approach to large-scale collection of application usage data over the Internet. In *International Conference on Software Engineering*, pages 136–145, 1998.
- [6] B. Liblit, A. Aiken, Z. Zheng, and M. Jordan. Bug isolation via remote program sampling. In *Conference on Programming Language Design and Implementation*, pages 141–154. ACM, June 2003.
- [7] Microsoft. Microsoft online crash analysis. <http://oca.microsoft.com/en/dcp20.asp>.
- [8] A. Orso, D. Liang, M. Harrold, and R. Lipton. Gamma system: Continuous evolution of software after deployment. In *International Symposium on Software Testing and Analysis*, pages 65–69, 2002.
- [9] A. Zeller and R. Hildebrandt. Simplifying failure-inducing input. In *Proceedings of the International Symposium of Software Testing and Analysis*, pages 135–145, Aug. 2000.