Regression Testing in Research And Practice

Xuan Lin
Computer Science and Engineering Department
University of Nebraska, Lincoln
1-402-472-4058
lxuan@cse.unl.edu

ABSTRACT

Regression testing is a costly but crucial problem in software development. Both the research community and the industry have paid much attention to this problem. However, are the issues they concerned the same? The paper try to do the survey of current research on regression testing and current practice in industry and also try to find out whether there are gaps between them. The observations show that although some issues are concerned both by the research community and the industry gay, there do exist gaps.

Keywords

Regression Testing, Software Engineering, Software Maintenance

1. INTRODUCTION

Regression testing is testing that a program has not regressed, that is, the functionalities that were working in the previous version are still working in the new version. During the maintenance of a software system or as the software evolves, the regression testing is the expensive but definitely necessary task. Since the cost of the software maintenance account for about two-thirds of the whole software, both project managers and researchers have to pay more and more attention to the regression testing. There have been a lot of researches on regression testing [1, 3, 4, 5]. Since regression testing is quite expensive, the researches mainly focus on how to reduce such cost. The topics include test selection, minimization, prioritization, etc.

However, are these research issues the same as what the people in industry really concern? Although there are hundreds of research papers on regression testing, the application of it in industrial environments has not been paid enough attention to and there has been little discussion on it. In [2], Onoma emphasized such problem, and gave one example that in practice some companies will rerun all test cases in regression testing while the research communities are still working on minimizing the test cases. There exist some gaps between research and practice! Several years have passed by after Onoma's article was published, a new survey must be done to report the new trend and practice of regression testing in industry.

The goal of this project is to address the gaps between current research and current application of regression in industry by studying the current research on regressing testing and the literatures on existing testing tools which are widely used in the industry. Once the gaps are addressed, we will be able to ask the question--- why this happens and new research direction and new guideline for the practice will be proposed.

The rest of the paper is organized as following: Section 2 is a briefly review of the current regression testing research literatures. In Section 3, the current popular commercial tools will first be introduced, and then some case studies are presented as they are some practices to apply the regression testing technology. Some interesting observations have been found according to Section 2 and Section 3, and they are presented in Section 4. Conclusions are given In Section 5.

2. Regression Testing Technology in Research Literatures

2.1 Regression Testing

Let P be a procedure or program, let P' be a modified version of P, and let T be a test suite for P. As described in [1], a typical regression test procedures is :

1. Select T' $\subseteq$ T, a set of test cases to execute on P'.
2. Test P' with T', establishing P''s correctness with respect to T'.
3. If necessary, create T'', a set of new functional or structural test cases for P'.
4. Test P' with T'', establishing P''s correctness with respect to T''.
5. Create T''', a new test suite and test execution profile for P', from t, T', and T''.

Different steps involve different important problems of regression testing. Step 1 involves the regression test selection problem: to select a subset T' of T, with which to test P'. Step 2 and Step 4 address the test suite execution problem: to efficiently execute the test suites and checking test results for correctness. Step 3 involves the coverage identification problem: to identify whether P' has new functionality which

require new test cases. Step 5 addresses the test suite maintenance problem: to update and store test information.

2.2 Regression Testing Technologies

Researchers had addressed the regression testing for years and research on regression testing includes a wide variety of topics which cover the above steps.

2.2.1 Test environments and Automation

Hoffman[8], Brown and Hoffman[9], Ziegler, Grasso, and Burgermeister [10] address the problem of test environments and automation of the regression testing process[1]. Their goals are to improve system quality and maintenance costs through systematic regression testing. They tried to define a general regression test process and tried to use scripts to automate test cases generation and execution. For example, [8] defines their own test script language which can be used to describe the test cases. Then they use the test program generator PGMGEN to generates test drivers in the C language.

2.2.2 Test Suite Management

Lewis[16], Hartmann and Robson[17], Taha, et al. [18], and Harrold et al. [19] focus on the problem of test suite management[1]. Whenever changes happen, part of the test cases will be selected from the original test cases, part of the test cases are obsolete and need to be deleted, and for the new functionality, new test cases should be added. All these changes should be managed during the regression testing process. [17] proposed a selective retesting tool. The tool has a test library which stores the test cases and test data, and it would automatically receive feedback as to the impact of the changes of software, including a complete reanalysis of the target system and the extraction of reusable test cases from the existing test library, and the determination of a subset of test data to revalidate the given changes. Besides, the tool would provide, if necessary, suggestions as to any additional tests that may be required to exercise the enhancements or new data. Finally, the tool will restore all the modified or new test cases and data into the test library. [18] maintains the test suites by using data flow analysis tool to partition the test cases into relevant, non-relevant and invalid classes. Thus, the cost of revalidating a program following modification will be reduced. [19] proposed a methodology to select a representative set of test cases from a test suite that provides the same coverage as the entire test suite. The selection was performed by identifying and then eliminating the redundant and obsolete test cases in the test suite. The representative set replaces the original test suite and thus, potential y produces a smaller test suite. The representative set can also be used to identify those test cases that should be rerun to test the program after it has been changed.

2.2.3 Reuse of existing test cases

2.2.3.1 Retest-all

When the program is modified, generally the testers have two main strategies to test the modified program. One is that select part of the test cases from the original test suites in order to reduce the cost. The other is to rerun all the original test cases which is known as retest-all strategy. [2, 21] introduce the retest-all methodology: re-use all previously developed test suite T, executing on the modified program P'.

2.2.3.2 Regression test selection

As described in 2.2.3.1, instead of rerunning all the test cases, the regression test selection technologies try to reduce the cost by picking a subset T' of T, and only run T' on the modified program P'. The assumption here is that the cost of the selection process will not outweigh the execution of the additional test cases (the test cases which are not selected by the technology).

[1] did a survey of regression test selection technologies. Totally 12 test selection technologies are studied, from data-flow technology to symbolic execution technology, etc.

2.2.3.3 Test case prioritization

Regression testing is expensive and in order to reduce such cost, researchers have done many works other than test selection technologies. [22, 23, 24] address the problem of the test case prioritization technology. They order (prioritize) the test cases by certain measures. Then in the regression testing cycling, the test cases will be used to test the modified program P' according to the order, so that the "better" test cases can run first. The goal of the prioritization is to increase the rate of fault detection (how quickly the test suite can detect the faults during the test process), or, increase the rate of code coverage (how quickly the test suite can increase the coverage of the program). For example, let t1, t2, t3 be the three test cases. Also assume that t1 has the coverage of 75%, t2 has the coverage 25% and t3 has the coverage of 50%. According to the second goal, the result of apply such technology is to run the test cases in the order of t1, t3, t2. And similarly, according to the first goal, the order of the three test cases will depend on their ability to expose the fault.

2.2.3.4 Test Suite Reduction

[25, 26, 27] focus on the test suite reduction technology. They try to permanently eliminating test cases from the test suite so that the cost of the future regression testing will be reduced and the size of test suite can be controlled. For example, [27] presents a technique to select a representative set of test cases from a test suite which provides the same coverage as the whole original test suite. They use data flow technique to analyze the coverage. They first identify test cases into three classes useful, redundant and obsolete, and then eliminate the redundant and obsolete test cases in the test suite. The rest representative test cases replace the original test suite. And thus, a potentially smaller test suite is produced.

3. Regression Testing in Practice

3.1 Commercial Regression Testing Tools

In this section, a set of leading regression testing tools will be briefly reviewed.

There are tons of testing tools nowadays and there are some other interesting tools, e.g., [33, 34, 35], which are not discussed in this paper. The main reasons why the tools are selected in this paper are:

1. These tools are widely used in the industry and have high reputation by the users.
2. The free trial version is available online or some important documents which can give enough details are available online.
3. The tools can support the regression testing well.

3.1.1 TestComplete

TestComplete is an automated testing environment for Windows applications. It provides extended support for testing Web pages, Web servers and projects created in Microsoft Visual C++, Visual Basic, Borland Delphi, C++Builder, Java and .NET development tools. It manages scripts to test applications externally, self-testing by the applications themselves, and many inside-outside forms of tests. It is oriented equally to unit testing and to functional testing, and provides superior support for daily regression testing.[28]

In true regression testing, all tests of all sizes are rerun and their results are accumulated, and nothing is thrown away. On each iteration, all existing, validated tests are run, and the new results are compared to the already-achieved standards. And normally, one or more additional tests are run, debugged and rerun until the project successfully passes the test.

Regression tests begin as soon as there is anything to test at all. The regression test suite grows as the project moves ahead and acquires new or rewritten code. TestComplete provides ways to manage these test cases and automate them.

3.1.2 QEngine

QEngine[29] is an extensive, platform independent Test Automation tool used for Web Functionality, Web Performance, Java Application Functionality, Java API, SOAP, Regression, and Java Application Performance testing. It supports testing of applications developed using HTML, JSP, ASP, .NET, PHP, JavaScript/VBScript, XML, SOAP, WSDL, e-commerce, traditional client/server etc. Tool is developed using Java which facilitates portability and multiple platform support.

The regression testing of QEngine includes

1. Interventionless automated daily build smoke test.
2. Built-in Error recovery for unattended testing.
3. Test suite can be split and run on multiple machines to reduce testing time.
4. Web Interface for monitoring and controlling test suite execution.
5. Comparison reports - Daily regression reports against a master report.
6. Transfer of reports to Central Server in Distributed Suite Environment.
7. Web Interface for monitoring and controlling Regression test bed.

We can see that the QEngine can automatically execute and maintain the test suites. And when do the regression testing, it also use the retest-all methodology..

3.1.3 Junit

Junit[30] is a well-known open source regression testing framework. Now in industry, a lot of companies are using junit to develop test cases and do the unit testing. It actually designs a framework to facilitate automatically executing the test cases and manage them.

3.1.4 Rational Functional Tester

IBM Rational Functional Tester is an advanced, automated functional and regression testing tool for testers and GUI developers who need superior control for testing Java, Microsoft® Visual Studio .NET and Web-based applications[31]. Verification points help to ensure there is no regression from one build of the application under test to the next. IBM Rational Functional Tester provides a wide range of verification points to test various aspects of the application, and it includes pattern matching support for tests in which the exact data response can be predicted.

3.1.5 Mercury WinRunner

Mercury WinRunner™ is a powerful tool for enterprise-wide functional and regression testing. WinRunner captures, verifies, and replays user interactions automatically, so users can identify defects and ensure that business processes work flawlessly upon deployment and remain reliable. WinRunner has several advantages, including:

- Reduced testing time by automating repetitive tasks
- Optimized testing efforts by covering diverse environments with a single testing tool
- Maximized return on investment through modifying and reusing test scripts as the application evolves

It has been used widely by many companies, such as Air Product, SmartMoney, Retail Europe, Xcel Energy, Abbott Laboratories, Johnson and Johnson. “Manual functional testing

of Nextel's sales application would have taken around 300 staff hours. But with Mercury's automated testing processes, the same project took less than 50 hours to complete. By performing regression testing in parallel on multiple machines at the same time, Nextel can now complete the same test in less than 10 hours - a 95 pecenet improvement compared to the original 300 hours", Dick LeFave, Senior Vice President and CIO, Nextel Communications.

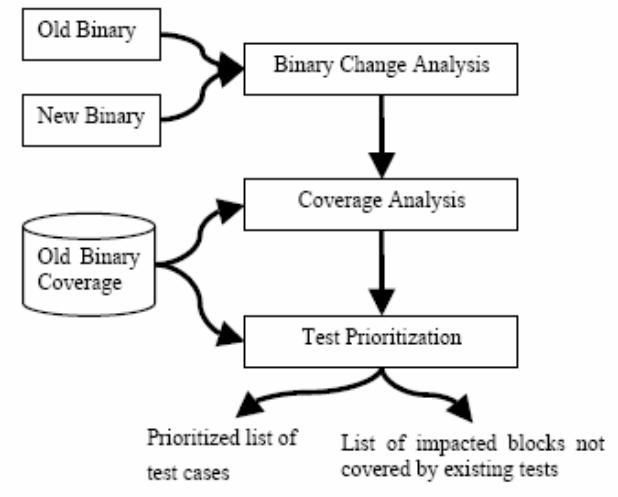
We can see that WinRunner also treat automatically executing regression testing as the highest priority.

3.2 Case Studies

In this section, several case studies will be done to show the effort to apply the regression testing technologies into the practice or the current regression practice in the industry.

3.2.1 Case Study1

In [11], Amitabh Srivastava and Jay Tiagarajan had tried to apply the prioritizing technology to the actual development environment. They have integrated and deployed their tool set in the real Microsoft development environment. The tool they developed is called Echelon, a test prioritization system. The distinguish feature of the tool is that it analyze the binary code while most the regression prioritizing technologies analyze the source code. The Echelon is part of the Magellan tool set. The core of the Magellan is a SQL Server database which stores test coverage information for each test. All the program binaries and the source codes are stored separately. Magellan provides a set of tools which are commonly needed during the test process. It includes test coverage collection tool, GUI interface to map coverage data source code and test migration tool to migrate coverage data from older version to the new one. Echelon is actually the prioritization system of the tool set. The following graph is the architecture of the Echelon system.



Echelon takes the binary of the old version as well as the new version along with the test coverage information as the inputs. Then as shown in the figure, Echelon uses a tool named BMAT to do the binary change analysis, that is, try to find the matching of the blocks in old binary and new binary. After that, Echelon try to determine which impacted blocks in the new version are likely to be covered by an existing test. Finally, Echelon use the similar prioritizing algorithms as [5, 6, 7] to prioritize the set of tests. When doing prioritization, Echelon only uses the coverage information. However, it claims that the other technology like the execution time can be easily added to the system to prioritize the tests.

Their work is a significant step to apply the prioritizing technology to the real development environment.

3.2.2 Case Study2

Seapine Software [12] has a process for requesting and managing changes to an application during the product development cycle. The process includes:

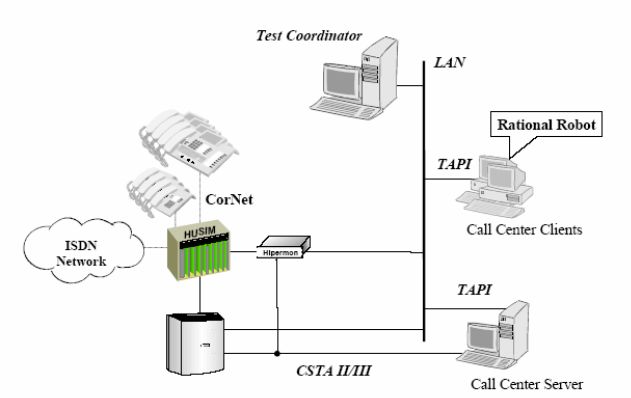
1. Collect change requests
2. Identify the scope of the next release and the scope of the next release and determine which change requests will be included in the next build.
3. Document the requirements, functional requirements, functional specification and implementation plans for each grouping of change requests.
4. Implement the change.
5. Test or verify the change. Unit testing is done by the person who made the change, usually the programmer. Function testing tests a functional area of the system to see that everything works as expected. Regression testing is system-wide to insure that all areas of the system still function as expected. This validates that the change caused no unexpected side effects and that the system still has the overall functionality it had before the change. Regression testing could consist of testing all functional areas of the system. User acceptance testing is done by testing is done by a select group of end users, usually run in parallel to the existing system. This is the final test before the system "goes live."
6. Release.

What need to emphasize here is that the regression testing is required to be system-wide. [2] also mentioned that in the industry, multi level regression testing is required to insure the software quality. Since in practice, large systems may be developed in different stages and by different teams. During the process, testing is involved in almost every stage, such as unit testing, functional testing, acceptance testing and field testing.

So in practice, regression testing is required to be embedded in every mature software development process, since for every change of the software, no matter how tiny it is, regression testing must be applied.

3.2.3 Case Study3

[13] presents a complex industry application, they exemplify on the basis of a concrete case study (Siemens' HPCO Application, a complex Call-Center Solution) how test engineers can now work with the Integrated Test Environment.



The above figure is one scenario regression test environment setting for the Call-Center Solution. We can see that even the simple scenario demonstrates the complexity of CTI platforms from the communication point of view because there are several internal protocols involved. This case study exposes the problem that in current industry practice, regression testing is intended to integrate with complex test environments. New methodology and technology should be developed to solve this problem.

3.2.4 Case Study 4

[14] shows a functional/regression testing for securities industry application. The tool they use is called Infosys ACCORD. It is an intelligent functional test automation solution that enables securities firms to cost-effectively test the functionality and improve the quality and performance of critical applications. ACCORD simplifies the overall testing processes of generation, translation and execution of test cases.

What is unique to ACCORD and we should notice is that it has an intelligent algorithm that optimizes the number of test cases while eliminating the redundant ones, thereby reducing cycle time and improving test quality. This shows the Test Suite Reduction Technology has been utilized in the real industry applications.

4. Observations

From Section 2 and Section 3, we can find several interesting relation between the research and practice of the regression testing.

4.1 Automation And Test Suite Management

The only practical way to manage regression testing is to automate it. If it is not automated, people will become numbed from running the same tests many times and seeing the same test results many times. It becomes too easy to overlook errors, which defeats the purpose of regression testing. So, for all the commercial tools, automation of regression testing are all supported. Meanwhile, in order to persuade uses to buy the tools, they all have the functionality to manage the test cases.

Researchers had addressed this problem more than ten years ago. [8,9,10,16,17,18,19,20]

4.2 Retest-all Or Test Selection

Researchers have done tons of work on test selection technologies. However, all those commercial tools described in Section 3 will retest all the old test cases while doing the regression testing. The reason might be that those commercial tools are general purpose, and those test selection technologies are hard to be automated in the general purpose tools. Also, some companies will retest all the existing test cases in regression testing [2], especially when the applications are real-time or safety-critical.

One thing need to be mentioned is that, as in this survey, the only resource I can use is the internet, and no companies have been contact, so it is also hard to say that more companies use retest-all technology than those using test selection technologies.

4.3 Test Suite Reduction

As mentioned in case study 4, the test suite reduction technology has been applied into real industry applications. And research community also has address this problem [25,26,27].

4.4 Test Prioritization

No commercial tools have been found to use this technology. However, some efforts have been done to try to apply the technology to the real development environment (Case Study 1).

4.5 Integration

Case study 3 gives a specific scenario that the test environment is complex and it is hard to do the integration. This case will be more and more common since nowadays the software has the tendency to be more complex.

Few researches have addressed this problem.

5. Conclusion

Regression testing is a costly but crucial problem in software development. There are a lot of researches addressing this area while in industry regression testing is also a crucial process. The paper is an initial work to survey both sides. And observations show that while there are some issues both sides are concerned, there are still some gaps between them. The gaps may be good

direction for the research, or more work should be done to try to apply the technology from lab to the industry.

6. REFERENCES

[1]. G. Rothermel and M. J. Harrold, Analyzing Regression Test Selection Techniques, IEEE Transactions on Software Engineering, V.22, no. 8, August 1996, pages 529-551.

[2].K. Onoma, W-T. Tsai, M. Poonawala, and H. Suganuma. Regression testing in an industrial environment.

Communications of the ACM, 41(5):81–86, May 1998.

[3]. G. Rothermel, S. Elbaum, A. G. Malishevsky, P. Kallakuri, and X. Qiu, On Test Suite Composition and Cost-Effective Regression Testing; ACM Transactions on Software Engineering and Methodology, V. 13, no. 3, July, 2004, pages 277-331.

[4]. G. Rothermel and M. J. Harrold, A Safe, Efficient Regression Test Set Selection Technique, ACM Transactions on Software Engineering and Methodology, V.6, no. 2, April 1997, pages 173-210.

[5]. S. Elbaum, .A. Malishevsky and G. Rothermel, "Test case prioritization: A family of empirical studies", IEEE Trans. Software Engg. , vol. 28, no. 2, pp. 159-182, Feb. 2002.

[6]. S. Elbaum, .A. Malishevsky and G. Rothermel, "Prioritizing test cases for regression testing", Proc. Int'l Symp. Software Testing and Analysis, pp. 102-112, Aug. 2000.

[7]. W.E. Wong, J.R. Horgan, S. London, and H. Agrawal, "A Study of Effective Regression Testing in Practice", Proc. Eighth Int'l Symposium Software Reliability Eng., pp. 230-238, Nov. 1997.

[8].D.Hoffman and C.Brealey. Module test case generation. IN Proceedings of the Third Workshop on Software Testing, Analysis, and Verification, pages 97-102, December 1989.

[9].P.A Brown and D. Hoffman. The application of module regression testing at TRIUMF. Nuclear Instruments and Methods in Pysics Research, Section A, . A293(1-2):377-381, August 1990.

[10].J.Ziegler, J.M. Grasso, and L.G. Burgermeister. An Ada based real-time closed-loop integration and regression test tool. In Proceedings of the Conference on Software Maintenance - 1989, pages 81-90, October 1989

[11]. A. Srivastava and J. Thiagarajan. Effectively Prioritizing Tests in Development Environment. In Proceedings of the International Symposium on Software Testing and Analysis, pages 97–106, July 2002.

[12]. http://www.seapine.com/

[13]. Andreas Hagerer, Tiziana Margaria, Olver Niese, Bernhard Steffen, Georg Brune and Hans-Dieter Ide, Efficient Regression Testing of CTI-Systems: Testing a complex Call-Center Solution, In Annual Review of Communication Volume 55, pages 1033-1040, Int. Engineering Consortium (IEC), 2001

[14].http://www.infosys.com/industries/banking/automated_order_management.asp?page=iemsl

[15]. http://www.mysql.com/news-and-events/success-stories/ti.html

[16].R. Lewis, D.w. Beck, and J.Hartmann. Assay – a tool to support regression testing. In ESEC' 89.2nd European Software Engineering Conference Proceedings, pages 487-496, September 1989.

[17]. J. Hartmann and D.J. Robson. Revalidation during the software maintenance phase. In Proceeding of the conference on Software Maintenance.

[18]. A.B Taha, S.M. Thebaut, and S.S. Liu. An approach to software fault localization and revalidation based on incremental dat flow analysis. In proceeding ot the 13th Annual International Computer Software and Applications Conference.

[19]. M.J. Harrold, R. Gupta, and M.L. Soffa. A methodology for controlling the size of a test suite. ACM Transactions on Software Engineering and Methodology.

[20]. W.E. Wong, J.R. Horgan, S. London, and A.P. Mathur. Effect of test set minimization on fault detection effectiveness. In 17th International Conference on Software Engineering.

[21]. H.K.N. Leung and L. White. Insights into regression testing. In Proceedings of the Conference on Software Maintenance, pages 60–69, October 1989.

[22]. S. Elbaum, A. G. Malishevsky, and G. Rothermel. Test case prioritization: A family of empirical studies. IEEE Transactions on Software Engineering, 28(2):159–182, February 2002.

[23]. G. Rothermel, R. Untch, C. Chu, and M.J. Harrold. Prioritizing test cases for regression testing. IEEE Transactions on Software Engineering, 27(10):929–948, October 2001.

[24]. W.E. Wong, J.R. Horgan, S. London, and H. Agrawal. A study of effective regression testing in practice.In Proceedings of the Eighth International Symposium on Software Reliability Engineering, pages 230–238, November 1997.

[25]. T.Y. Chen and M.F. Lau. Dividing strategies for the optimization of a test suite. Information Processing Letters, 60(3):135–141, March 1996.

[26]. M. J. Harrold, R. Gupta, and M. L. Soffa. A methodology for controlling the size of a test suite. ACM Transactions on Software Engineering and Methodology, 2(3):270–285, July 1993.

[27]. J. Offutt, J. Pan, and J. M. Voas. Procedures for reducing the size of coverage-based test sets. In Proceedings of the Twelfth International Conference on Testing Computer Software, pages 111–123, June 1995.

[28]. http://www.automatedqa.com/products/testcomplete/

[29].http://www.adventnet.com/products/qengine/index.html

[30]. http://www.junit.org/index.htm

[31]. http:// www-306.ibm.com /software/ awdtools /tester/ functional /features/index.html

[32]. http://www.mercury.com/us/products/quality-center/functional-testing/winrunner/

[33].http://www.finisar.com/nt/documents/QLogicTroubleshootsiSCSISANs.pdf

[34]. http://www.aptest.com/resources.htmlTest case prioritization: A family of empirical studies

[35]. http://www.devdirect.com/ALL/Testing_PCAT_1986.aspx