Volume 3, Issue 4, April 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Testing Embedded Oriented Software based on Software Engineering

Sonali Verma[1], Dhawaleswar Rao[2]
[1] M.Tech student, Lovely Professional University, Punjab, India.
[2] Assistant Professor, Dept of Computer Science & Engineering, Lovely Professional University, Punjab, India.

Abstract: ***Embedded software testing shares much in common with application software testing. Testing of the softwares which are used to run embedded applications for different industries is a very difficult and complex process. Embedded developers often have access to hardware-based test tools that are generally not used in application development. Testing is the last step in traditional software development. First of all we gather requirements, then execute high level design, then full detailed design, after that we create code, and then do unit testing, then integrate testing. The main focus is to provide bug free product in the embedded software market. This particular research has focused on the testing of embedded based software development. In this particular research, regression testing has been considered as to judge the reliability, usability and failure functions of the embedded based software in software engineering. We tested various phases of embedded systems software with regression testing.***

Keywords: Code Testing, Embedded Software, Software Engineering, Unified Modeling Language,

1. Software & Embedded Software Development

The term "software development" may be used to refer to the activity of computer programming, which is the process of writing and maintaining the source code, but in a broader sense of the term it includes all that is involved between the conception of the desired software through to the final manifestation of the software, ideally in a planned and structured process.[4] Therefore, software development may include research, new development, prototyping, modification, reuse, re-engineering, maintenance, or any other activities that result in software products[5]. Software can be developed for a variety of purposes, the three most common being to meet specific needs of a specific client/business (the case with custom software), to meet a perceived need of some set of potential users (the case with commercial and open source software), or for personal use (e.g. a scientist may write software to automate a mundane task). Embedded software development, that is, the development of embedded software such as used for controlling consumer products, requires the development process to be integrated with the development of the controlled physical product [7]. Embedded software's principal role is not information technology (i.e. it is not about information and the technologies related to providing information services), but rather the interaction with the physical world. It's written for machines that are not, first and foremost, computers. Manufacturers 'build in' embedded software in the electronics in cars, telephones, audio equipment, robots, appliances, toys, security systems, pacemakers, televisions and digital watches, for example. This software can become very sophisticated in applications such as airplanes, missiles, and process control systems [1]. Due to software bugs, the increased use of embedded software has, led to an increased failure rate in automobiles [2]. Embedded systems most often need real time programming. Real Time operating systems and their working are generally shown by two methods: 1. Finite state machine 2. Petri-Nets. But these two modeling methods have shown certain limitations as many sophisticated embedded systems are multiprocessor systems and the processes have short latencies [9].

2. Embedded Software Testing

One can argue that software is the dominant part of an embedded system, either as a final product (executable code) or during its development lifecycle (system modeling in specific languages and computation models). In both cases, software must be thoroughly verified to ensure product quality and reliability [10]. One can observe a growing number of academic and industrial works on the topic of embedded SW testing in the last four or five years, and this seems to be a good time for reflection: how exactly is embedded software testing different from traditional software testing? Is it an engineering or computer science problem? Does it need extra support from platform developers? What is the role of the SW engineer and of the designer in developing a high-quality software-based embedded application? Many authors suggest that, on top of the ordinary software testing challenges, software usage in an embedded application brings additional issues that must be dealt with: the variety of possible target platforms, the different computational models involved during the design, faster time-to-market and even more instable and complex specifications, platform-dependent constraints (power, memory, and resources

availability), etc [10]. On the other hand, current platforms are more and more powerful, and the specificities of the embedded application can help to reduce the search space during test generation: application domains, strong code reuse paradigm, use of less advanced programming language resources, and common availability of system models subject to or already verified with respect to specific properties, for instance [10]. Furthermore, a major part of the so-called embedded software does not depend directly on hardware, and one can argue that only a small percentage really needs to be tested together with the target platform, and this test is part of the platform design, not the system design. Application domains, strong code reuse paradigm, use of less advanced programming language resources, and common availability of system models subject to or already verified with respect to specific properties, for instance [10]. Furthermore, a major part of the so-called embedded software does not depend directly on hardware, and one can argue that only a small percentage really needs to be tested together with the target platform, and this test is part of the platform design, not the system design [10]

3. Application versus embedded testing

Embedded systems software testing shares much in common with application software testing. Thus, much of this two part article is a summary of basic testing concepts and terminology. However, some important differences exist between application testing and embedded systems testing. Embedded developers often have access to hardware-based test tools that are generally not used in application development.

Also, embedded systems often have unique characteristics that should be reflected in the test plan. These differences tend to give embedded systems testing its own distinctive flavor. This article covers the basics of testing and test case development and points out details unique to embedded systems work along the way.

4. Proposed Work

This research focused on providing solution for said problem by testing the embedded software problems while it is used under software engineering development.

This research focused on the Regression testing for the embedded software codes with help of UML based software.

Analysis of various embedded codes has been considered and analyzed. After this analysis, we started testing based on the analysis fetched to find the codes weak links and unused codes in total coding process.

These parameters considered with testing tool integration with these parameters. The testing of the software codes has been processed in UML based tool as shown in figure below (figure 1) and then processed through java coding to find the errors and weak links in codes.

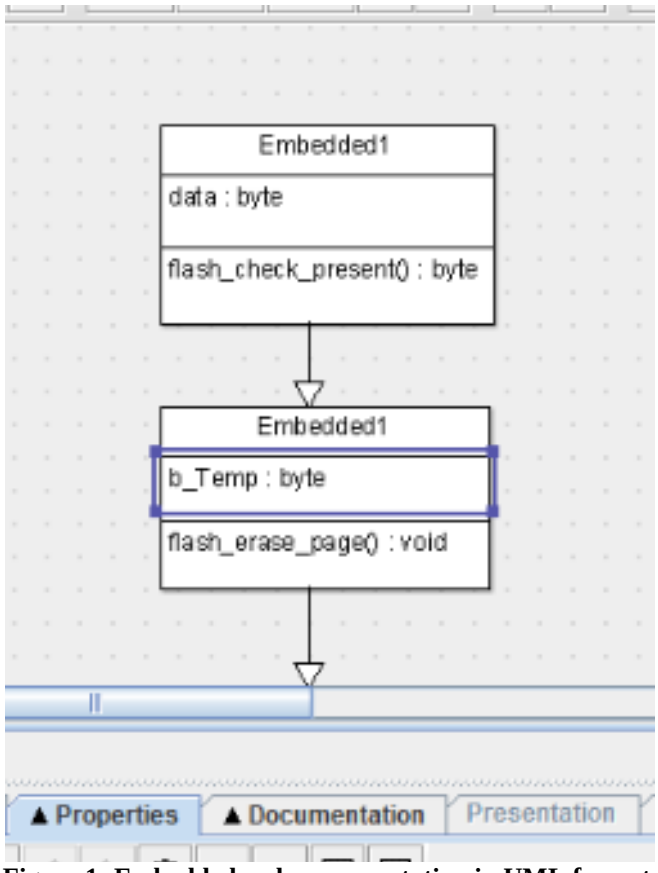

Figure 1: Embedded code representation in UML format

5. Results

To fulfill require experimentation we have done experimentation with regression testing.

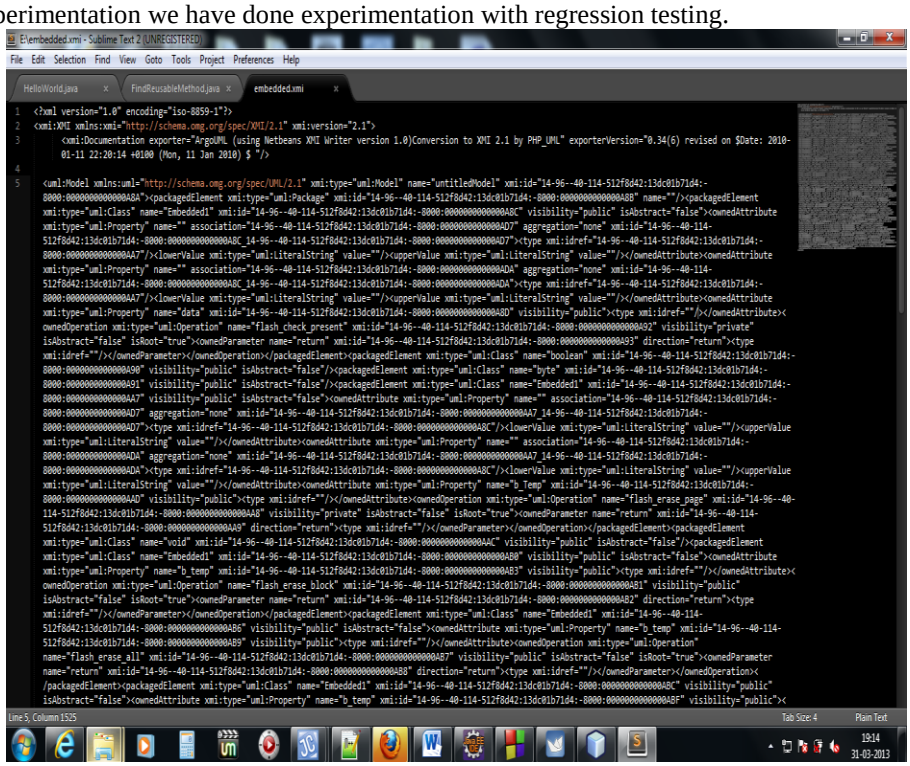

Figure 2: Embedded code representation

An Embedded code has been used in initial phases which present the working process in embedded code as shown in above figure (figure 2). Further this code is represented in form on unified modeling language with help of argo uml tool and uml diagram is obtained by converting embedded code into uml. Converted code is shown in figure below (figure 3).

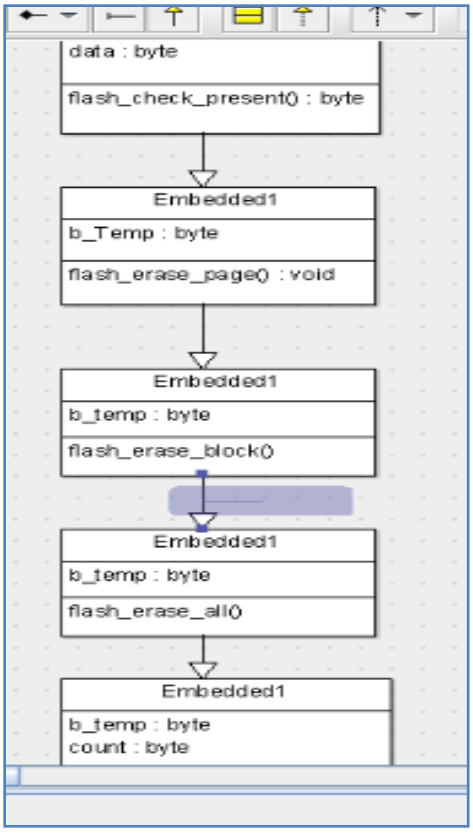

Figure 3: Converted code from embedded to uml diagram

Further the information in uml diagram is fetched with coding developed and proper sequence of the uml embedded code is represented in form of xmi coding as shown in below figure (figure 4).

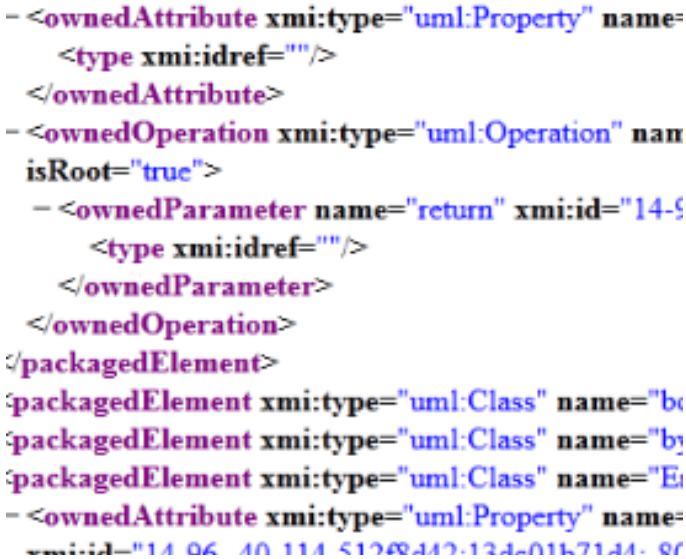

Figure 4: Demonstration of representation of embedded code in xmi coding.

Further process has done for finding errors in the code so that cost could be saved as described by basics of regression testing. The run time errors found in the embedded code is shown in figure below (figure 5a, 5b).

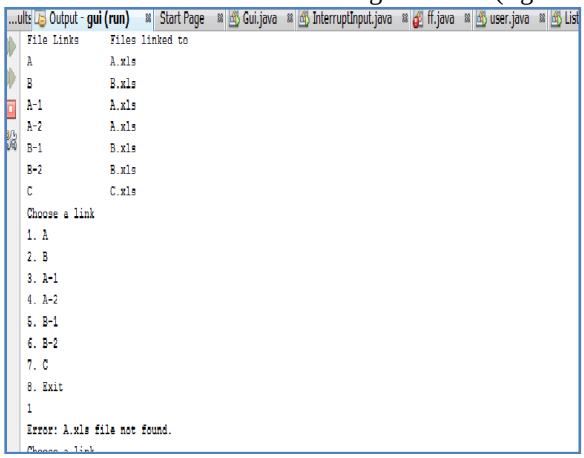

Figure 5a: Showing error of missing file

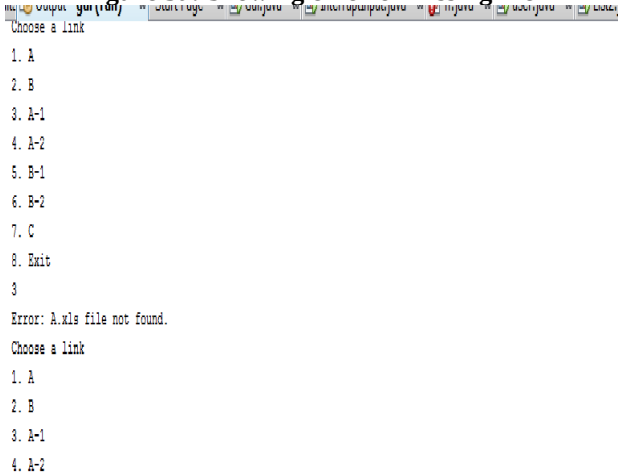

Figure 5b: Showing the detailed error of missing files which are corrupted in embedded code

Above errors show the regression testing for embedded code. According to the concept cost saving is done by finding errors while processing the codes

6. Conclusion

This Research will prove to be a good solution for saving resources while finding the weak link errors in the embedded software based codes. In this research, we have considered the codes based on embedded services and process with finding errors. In further experimentation we show the exact type of error found which will be useful in process of regression testing at larger scale. In future, this research can be enhanced by implementing the embedded software on live runtime devices and find the regression testing

References

[1] http://ptolemy.eecs.berkeley.edu/publications/papers/02/embsoft/embsoftwre.pdf.
[2] Azoff, Michael. "Embedded software is car manufacturing's biggest challenge." Ovum, 2 August 2011.
[3] http://www.bitpipe.com/tlist/Application-Development .html. "Application Development White Papers (Development of Software, Software Design, Designing Software, Software Engineering, Software Application Development, Enterprise Application Development, Platform Development, Software Development, Applications Development, Development) Software Downloads, Definition and Webcasts".
[4] "Application Development (AppDev) Defined and Explained", http://www.bestpricecomputers.co.uk/glossary/application-development.html.
[5] DRM Associates (2002). "New Product Development Glossary",http://www.npd-solutions.com/glossary.html.
[6] Steven P. Reiss (2001). Consistent software Evolution mandatory. Department of Computer Science Brown University.
[7] Joseph M. Morris," Software Industry Accounting", pp. 1-10, 2001.
[8] Alan M. Davis," Great Software Debates", Wiley-IEEE Computer Society Press, pp 125-128, October 8, 2004.
[9] Dr. Manuj Darbari, Hasan Ahmed," Software Engineering Practices in Embedded System Design Using Discrete Modeling Techniques", Proceedings of the World Congress on Engineering, Vol.1, WCE 2010, London, June 30 - July 2, 2010.
[10] Erika Cota," Embedded software testing: what kind of problem is this?", PPGC, EDAA, 2010.
[11] http://www.embedded.com/design/other/4212929/The-basics-of-embedded-software-testing.
[12] http://www.embedded.com/design/other/4212937/The-basics-of-embedded-software-testing--Part-2-?page=0.
[13] laser.inf.ethz.ch/.../01_Trends_in_Embedded_Software_Engineering.pdf. LAZER 2005.
[14] Maxime Perrotin, Eric Conquet, Julien Delange, Thanassis Tsiodras," TASTE: An open-source tool-chain for embedded system and software development", European Space Agency, ESTEC, Keplerlaan, 2010.
[15] Daniel Sundmark, Anders Pettersson, Sigrid Eldh, Mathias Ekman," Efficient System-Level Testing of Embedded Real-Time Software", M¨alardalen University, 2012.

