NOISE SUPRESSION SOFTWARE. AN ERRONEOUS PERCEPTION OF REALITY

Jose J. Garcia[1], Oscar Silveira[2]
[1]Florida International University, Miami, USA, JGarC002@fiu.edu
[2]Florida International University, Miami, USA, silveiro@fiu.edu

INTRODUCTION

In this article we attempt to deal with a misconception that may grow out of the use of software as ' that thing that helps us to solve everything without acknowledging that no software works without a material basis'. If there is not the material basis of an electronic circuit, then software might be considered of just an illusion of how things should be.

COMMON MISCONCEPTIONS ABOUT SOFTWARE

Let's first pose, from Rob Pirozzi's Common Misconceptions about Software Testing, the five most common misconceptions about how software testing differs from the activity of testing:

- **You can completely test the system**. The assert to this proposition is very simple. It is not possible to completely test any complex software system. The question to this may be, why ? Because of the software complexity under test, and what is of a high uncertainty, the data input. Further, the huge amount of different combinations of logical circuits, the system working in its network, and the several utilities which can be running in the background.
- **You can have "zero defect" or bug-free software**.The proper focus to this point is that perfection could be an "approachable " but never an achievable goal. What is perfection ?. Where is perfection ? Is there such a thing as perfection ? Instead, we should consider a software test to function in a way that in testing the circuits it performs well enough so that the software quality, is acceptable. Be aware that propaganda is always a sophism.
- **Software testing is exclusively a quality control activity**. Software testing is not just an activity of quality control, but of quality assurance of the overall development process. However, software testing is more than quality control and quality assurance. Software testing is a dynamic process in which the involved iterative process of parameters detection, execution and problem identification, is most of the times if not all, a probability, not a certainty.
- **Software testing slows us down.** Product release schedules are very much driven by a focus on time-to-market. This places an extreme pressure on the speed of development. To help to deal with this there have been great advances in software development platforms and tools that significantly accelerate the development process. In addition, software development methodologies such as Agile and Extreme Programming help to speed up software development. However, traditional testing has not kept pace with these advances, and has not experienced a corresponding increase in speed. This leads to the perception that testing slows us down, as it has not sped up as the development process has. Simply squeezing software testing to meet delivery schedules is not an adequate answer to this problem. This can result in delivering poor quality software with many defects that can lead to numerous *unforeseen and uncontrollable costs in time and money*. It is more important to address the issue by first accepting that testing is an important part of the process that takes a certain amount of time, time that must be built into the schedule. It is also important to investigate and adopt testing strategies and methodologies such as *Action-Based Testing* and *global test automation* (the integration of the latest test automation methodologies and technologies with global resource strategies) that help to speed up testing while improving test coverage. Literally.
- **Software testing is software development.** Software testing is not just a development process of the overall procedure of software generation. It is also part of a very separate and discrete activity. Using the terms software development, design or programming as if they were interchangeable terms is simply,

incorrect. They are not synonyms. One thing is development, the other testing, or programming a cybernetic system, whatever it be. They are not the same activities.

HARDWARE RELATED SOFTWARE

Adding to the above, software and hardware involves modeling of a combined system of reliabity in which individual hardware platforms and the software which is designed to those platforms are most of the times, independent of other hardware/software platforms, and the slightest variation to the hardware platform, or environment parameters affecting the hardware platform, can generate errors in software testing and performance. In software development the reliability block diagrams that are used of system hardware development and software related to this hardware development, accurately portray the interrelationship between the hardware platforms and the software executing on the platforms but, they are developed and used in estimating reliability metrics, not *deterministic* data. Further, we should stress the fact that, for complex structures, state diagrams are developed to accurately portray the unique interrelationships of the structure being modeled.

As a rule, software components for the individual hardware blocks do not fail independently. They fail, in association with the overall operational profile of the system being designed, and most of all, the software system must be modeled that way. Is there a way of posing perfection to the components of the system or, one hundred percent reliability?

References

Pirozzi, R., Common Misconceptions about Software Testing
http://www.logigear.com/newsletter/

Winograd, T, and flores, F, Understanding Computers and Cognition, Addison-Wesley Publishing . Company, Inc. I986

Pierce, Dr. Brian, DARPA Interference Multiple Access (DIMA), Strqtegic Technology Office.

Carlson, A, Communication Systems, An Introduction to Signals and Noise in Electrical Communication, Irwin/McGraw-Hill, third edition, 1986.

Vasilescu, G., Electronic Noise and Interference Signals, Principle and Applications, Springer Series on Signals and Communication Technology, 2004.

Noise and Random Signals.
http://web.nps.navy.mil/~smithrw/Ch4.PDF

Elements of an Electrical Communications System.
http://zone.ni.com/devzone/cda/ph/p/id/82

Garg, V. IS-95 CDMA and cdma 2000. Prentice Hall PTR, ISBN: 0-13087112-5, page 11.