Closing the Gap in Failure Analysis

Brendan Murphy
Microsoft Research
Cambridge UK
{Bmurphy, MarioGar}@Microsoft.com

Mario Garzia
Microsoft Corp
Seattle USA

Neeraj Suri
Dept of Comp Science
TU Darmstadt Germany
Suri@Informatik.tu-darmstadt.de

Abstract

There has been a growing gap, perceived or actual, between industrial and academia approach to failure analysis. A possible consequence of these differences is the decreasing participation of industry at reliability oriented conferences such as ISSRE and DSN (it should be noted that both conferences are attempting to address this decline). Also, very often the people from industry who do attend these conferences are from the research organizations rather than from the product groups. This position paper attempts to analyze this growing divergence between academia and industry in regards to failure analysis and makes a tangible proposal as to how the gap can be closed.

1. Computer Trends

The original ground breaking work in reliability (software and system level) originated from the telephone exchange manufacturers. The failure profile for these large primarily hardware systems were well understood and could be captured through mathematical models. The impact of humans on these systems could be largely ignored as their management operations were well defined, operators were well trained and the user interface was trivial (the telephone) which had limited impact on the behaviour of the exchange.

As the usage and failure profile of the original computer systems were similar to telephone exchanges, computer reliability research continued along the same research themes. But in the late 70's and early 80's things started to change as the computers' operating system increased in complexity, the ratio of system managers to machines decreased and the power of the end user increased with their interface no longer being restricted to punch card readers.

These changes to the systems usage profile also resulted in changes to its failure profile. A seminal 80's paper [1] from Jim Gray, then at Tandem, described these changing trends. Tandem systems were fault tolerant machines whose system managers and service support staff were some of the most highly trained in the industry. Yet even on these systems just over 30% of all failures were of a type that was being addressed by reliability research at that time.

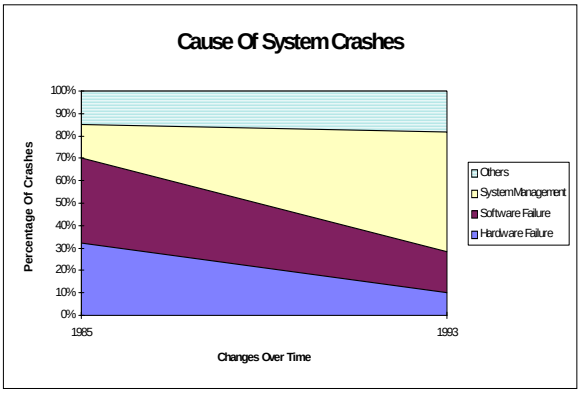

Figure 1: System Crash Trends [2]

A further study on Digital systems in the mid 90's confirmed that the underlying trend of systems failures was being driven by software and human errors (see Figure 1). This paper also highlighted that only 10% of all outages on computer systems, impacting availability, were as a result of system crashes, with the rest of the outages being caused by human induced shutdowns (i.e. operator shutdowns).

The underlying factors driving these trends are continuing. Server functionality is predominantly dependent upon the firmware and operating system rather than the underlying hardware. It is becoming increasingly difficult to characterize the internal fault model of the system as servers are primarily composed of components, manufactured by 3rd party suppliers. The components often contain their own firmware and driver software, with their hardware and the software being continually updated. While the management of

servers is continually being improved and simplified, companies often use this to decrease costs. This is reflected in changes in the ratio of systems managers to computers, whereas 30 years ago the norm was to have many people to manage a single system now often a single person manages 100s of systems.

Even for servers the role of the user is becoming increasingly powerful whereby their activity can impact the end systems. For example, in the past database development was often managed through four groups of people, system managers focusing on the hardware, managers for the operating system, managers focusing on the database configuration and the database developer. Now the total system can easily be managed by the developer. All of this has inevitably changed the failure profile of these systems.

The changes to the configuration and usage profile of servers are dwarfed by what is happening to client machines. The original client system was a dumb teletype input system which was a replacement for the card reader. Now a home PC can be as complex as most server systems with the PC having peripheral devices such as sound and video cards that are complex systems in their own right whose drivers consist of millions of lines of code. The configurations of these systems are also being connected in an infinite number of ways, with over 1,000,000 unique plug and play devices interfacing to Windows XP.

Initially as the usage and failure profiles of the systems changed both academia and industry continued to focus on traditional failures (system crashes brought about through hardware failures or due to the system not behaving to its written specification). Both assumed human errors were the problems of the end users and not the system manufacturers. This position was unsustainable within industry as even in the 80's most customer satisfaction surveys indicated that purely focusing on the classical failure classes would have limited impact on customer satisfaction.

Analyzing all system failures, irrespective of their causes, highlighted the problems associated with the traditional reliability approach. Reliability models no longer fit the failure data being captured from customer systems. Traditional approaches to validation such as fault injection only addresses a decreasing number of real life failures and often at only the lower system/hardware levels. For industry this meant moving from a predictive environment to an increasing fuzzy one, whereby making prediction on the quality of future products is becoming increasingly complex.

In addition to the changes in the causes of system outages, over the past two decades there has also been a dramatic change in the way software is developed and used. Software development today proceeds at a much faster pace than in the 80's, where there is a competitive need for frequent software updates and deployments. For example releases of web based software deployments can happen in weekly periods. In large and complex development projects, the multiplicity of components and dependencies means that the software is in almost constant flux. However when we look at the distribution of time between failures (see Figure 2) we find that while many failures happen in close proximity to a previous failure others can be separated by weeks, months and even years.

This presents a problem when trying to predict the reliability of software based on the very short evaluation and use periods resulting from the frequent updates. This problem is further complicated when we take into account the variety of usage scenarios possible for large scale deployments. For mass market software products such as Microsoft Windows, the number of customer usage scenarios can be very large, almost infinite. This further complicates software reliability predictions that now need to account for the different usage patterns. Both of these trends, frequent updates and usage scenario breadth, are in sharp contrast to the environment leading to the original research and models arising from telephony.

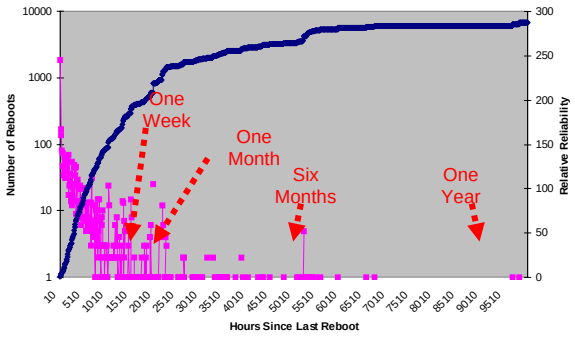

Figure 2: Time between Server Reboots [3]

For academia these trends presented fundamental issues. Academia traditionally took a very scientific approach to reliability research based on the existing data but the changes highlighted above require new data sets and new models. The development usage and failure profile of products in certain industries, such as embedded systems, Telco industries, nuclear power industries, railways etc still match the current reliability research and these industries were also willing to continue to work with academia. Whereas the other industries, specifically the computing industry, while bemoaning the direction of academic research, refused to release failure data. Thereby academic researchers were caught in a Catch-22 scenario, addressing an increasingly shrinking market,

but being unable to develop new theories and techniques due to a lack of data to train on. This inevitably hurts industry as less reliability research is meaningfully transferable to industry and the pool of available talent for recruitment into failure analysis is steadily decreasing.

2. Proposal

The major complaints of industry, regarding academia, are that they are working on problems that have insignificant impact on the end users and their business: the complaints of academia are that industry will not define the problems and will not provide data for academia to advance their research. To begin to address both of these issues a failure analysis workshop will be held in Microsoft Research labs in Cambridge UK in February 2007. Engineers from Microsoft and other system providers will describe the major drivers affecting system reliability, the current work being done to model and predict system reliability and where the key gaps lie. Academics will be invited to provide 2 page position papers on failure analysis which they will be able to present at the workshop.

Additionally failure datasets will be provided to the workshop participants to take away for future analysis. Due to privacy and contractual reasons the data will be anonymous both in users and in what software is running on the system. A framework will be provided for the researchers to share the methods used to analyze the data to promote the sharing of analysis techniques.

System providers will always be restricted in the data that can be provided to the research community. To get around this issue the workshop will also cover available tools and methods that can be used by the research community to capture and share their own data for future research.

It is hoped that this workshop will be the beginning of further events and activities to help close the gap between academic research and the needs of industry.

10. References

[1] J. Gray, "Why do computers stop and what can be done about it", *Proc 5th Symposium on reliability in distributed software and database systems*. California 1986.

[2] B. Murphy, T Gent, "Measuring System and Software Reliability using an automated data collection process", *Quality and Reliability Engineering International*, 1995.

[3] M. R. Garzia, "Assessing the Reliability of Windows Servers", *DSN Proceedings*, 2003.