5-2003

A Taxonomy Of E-Commerce Risks And Failures

Giridharan Vilangadu Vijayaraghavan

A TAXONOMY OF E-COMMERCE RISKS AND FAILURES

by

Giridharan Vilangadu Vijayaraghavan

A Thesis
Submitted to the Department of Computer Science
at Florida Institute of Technology
in partial fulfillment of the requirements
for the degree of

Master of Science
in
Computer Science

Melbourne, Florida
May 2003

We the undersigned hereby recommend that the attached document be accepted as fulfilling in part the requirements for the degree of Master of Science of Computer Science.

“**A Taxonomy Of E-Commerce Risks And Failures**”
a thesis by Giridharan Vilangadu Vijayaraghavan, May 2003

____

Cem Kaner, J.D, Ph.D
Professor and Thesis Advisor,
Department Of Computer Science

____

Pat Bond, Ph.D.
Associate Professor,
Department Of Computer Science

____

Muzzafar Shaikh, Ph.D.
Professor and Director,
Department Of Engineering Management

____

William Shoaff, Ph.D.
Associate Professor and Department Head
Department Of Computer Science

Abstract

TITLE: A Taxonomy Of E-commerce Risks and Failures

AUTHOR: Giridharan Vijayaraghavan

MAJOR ADVISOR: Cem Kaner, J.D, Ph.D.

Imagine being asked to test an e-commerce website. If you hadn't tested one before, where would you start? What experience would you draw on? Where would you look for more information? Even very experienced testers have blind spots when they try to generate test ideas for an application that they have not tested. This thesis presents a simple outline that will help you generate test ideas and limit your blind spots. The outline is the result of about 18 months of research on classifying e-commerce related failures and risks. The result has 45 top-level categories and 700+ examples of errors (potential issues to test for) under most categories. In many cases, there is also link to about 300+ examples of e-commerce defects that have been publicized in the press.

Using the list, you could pick a category of interest (such as accessibility or software upgrade), read descriptions of several types of problems that fit within that category, and so identify a few issues that would be appropriate to test for in your application. Based on feedback to the authors of *Testing Computer Software (*Kaner *et al.),* I believe that many testers will be able to use this list to identify potential problems that they would otherwise have missed.

Table of Contents

List of Figures

Acknowledgments

There are many people I would like to thank, for without them, this thesis would not be completed:

- Cem Kaner for being the greatest advisor and mentor one can have and for his motivation and guidance through this thesis process. Without his vision, encouragement and help this thesis would have never been possible.
- National Science Foundation and Rational Corporation for funding this research. This work was funded by NSF’s grant EIA-0113539 ITR/SY+PE: "Improving the Education of Software Testers."
- Dr. Pat Bond and Dr. Muzzafar Shaikh for serving on my committee.
- Dr. William Shoaff for being my academic advisor and for his support and understanding when I most needed it.
- The Florida Tech Department of Computer Science for their support and encouragement.
- George Hamblen, James Bach, Ross Collard, Sam Guckenheimer and Karen Johnson for their perspectives, ideas and support.
- All my lab mates at Kaner’s Lab (Software Testing Education Research Lab) for their help, opinions and ideas.
- Sunil Patel and Amit Singh for their timely help on the day of my thesis defense.
- Ajay Jha for helping me with the printing and submission of this thesis.

- My closest friends at Florida Tech, Voormila Satish and Nisha Pavithran for giving me the confidence that I can do this thesis and that the impossible is within my reach. I can’t thank them enough for encouraging me to embark upon this journey and their continued help and moral support without which this would have never been possible.
- My aunt and uncle in Melbourne, for being there as my family.

Dedication

This work is dedicated to the **One Almighty God**, Lord at Thirumala Hills, Lord Venkateshwara.

Sarva dharmAn Parithyajya MaamEkam SaraNam vraja
Aham Thva Sarva paapEpbhyO mokshayishyAmi maa sucha:
--Bhagavath Gita : 18.66

(Meaning) Totally relinquishing all dharmAs (completely
relinquishing the sense of agency, posssessiveness
and fruits of actions), seek Me alone as Your refuge.
I will free You from all sins .Grieve not.

Also dedicated to my Mom, Dad and Sister who are *always* there for me and who love me more than I can ever love them back.

Chapter 1: Introduction

…In the Headlines…

"…EBay's 22-hour crash in June cost the company more than $5 million in returned auction fees…"

"ESPN, which lost its fantasy baseball site for three days beginning July 11, conceded that it will have to compensate some of its 260,000 online players, who pay $30 each to play in the league…"

"Forrester puts the average cost of site downtime at about $8,000 per hour." [1]

1.1 The Nature of E-commerce Testing

It is easy to conclude from statements such as the ones above that the cost of a failure in an e-commerce system is formidable.

E-commerce testing involves testing high value, high risk, and high performance business critical systems. Generally, the testing process involves considerable effort in designing and integrating an effective risk management process to avoid or minimize the cost of failures.

The distributed nature of the underlying e-commerce applications also adds to the complexity in developing a comprehensive risk-based testing approach because it makes it difficult to exactly pinpoint where the risk of failure exists. Which application failed? Where did it fail? And when did it fail? E-commerce testing

[1] **Source: " The Cost Of Downtime" http://www.internetwk.com/lead/lead073099.htm**

involves testing the individual underlying components and web applications for potential risks and failures, along with the e-commerce “web site” as a whole.

A bug in an underlying vendor specific component, such as a Web server, transaction server or database system, may be missed or ignored. This is because the focus of testing might be on the site as a whole or on the code specifically written for the web application. Unfortunately, an inconspicuous bug in a third-party component might cause a serious failure when the system is in production. The failure may lead to bad publicity, lost reputation, customer confidence, and revenue.

1.2 Problem Statement

With so many different permutations and combinations in the ways an e-commerce site can fail, the process of brainstorming a list of test ideas, specifically for testing an e-commerce site, becomes extremely complicated. The brainstorming of test ideas is easier when:

- There is a body of knowledge available that educates a tester not familiar with the application under test, of the different ways the application has failed in the past or the different ways it can fail in the future. This could give the tester a starting point to write a test case that will test for failure. The different ways the application can fail constitute the failure modes for the application.
- A framework for idea generation exists: Idea generation is an activity that works more productively and produces better results when there is a framework on which ideas can be built on. The framework can be a collection of focused starting points, which give clues and inspire ideas in the minds of the testers. The framework could be an outline of a

> classification system that holds possible failure modes for an application. The framework for idea generation could be a "taxonomy of failures and risks."

This thesis provides both a list of potential failure modes in an e-commerce site and a taxonomy of failures to hold the failure modes.

1.3 Solution Approach

This thesis presents a simple outline that will help the tester generate test ideas and limit his/her blind spots. The outline is the result of 18 month's research on classifying e-commerce-related failures and risks. The result has 44 top-level categories and examples of errors (potential issues to test for) under most categories. In many cases, it also has links to examples of e-commerce defects that have been publicized in the press. Using the list, you could pick a category of interest (such as accessibility or software upgrade), read descriptions of several types of problems that fit within that category, and thus identify a few issues that are appropriate to test for in your application.

Based on feedback to the authors of *Testing Computer Software* (Kaner *et al*.), I believe that many testers will be able to use this list to identify potential problems that they would otherwise have missed.

I intend the outline to serve similar functions to the appendix in *Testing Computer Software (TCS)*:

- Help testers generate ideas;
- Help test plan inspectors check a set of tests for thoroughness and coverage;

- Help testers and other stakeholders identify risks during discussions of prioritizing the testing effort.

1.4 Organization of the Thesis

Chapter 2 “Risk-based Testing” provides an overview of the concept of applying risk analysis as a software testing technique.

Chapter 3 “Failure Modes and Effects Analysis (FMEA)” takes the concept of risk analysis a step further and provides an overview of the methodology of FMEA. It also contains a detailed literature survey on the application of FMEA from a software perspective (SWFMEA).

Chapter 4 “Taxonomies” provides an insight into the world of taxonomies. It takes the reader through different defect taxonomies/ bug taxonomies built by others and tries to analyze their objectives vis-à-vis the objectives of this thesis.

Chapter 5 “The e-commerce bug taxonomy” introduces the structure, content and objectives of the e-commerce bug taxonomy, which is the crux of this thesis.

Chapter 6 “E-commerce testing and shopping carts” contains a discussion on how e-commerce testing differs from traditional testing and then provides an overview of the different types of “shopping carts”.

Chapter 7 “List I: Component failures” contains the list of failure categories and failure modes that are based on component failures or non-qualitative failures such as database server failure, web server failure, memory leaks etc.

Chapter 8 " List II: Qualitative failures" contains the list of failure categories and failure modes that are due to the non-fulfillment of a qualitative attribute such as reliability, functionality, usability etc.

Chapter 2: Risk Based Testing

2.1 Overview

This chapter provides a high level overview of risk-based testing concepts that are related to the topic and goal of this thesis.

Ståle Amland states in his paper, “Risk Based Testing and Metrics” that “The objective of Risk Analysis is to identify potential problems that could affect the cost or outcome of the project. The objective of risk assessment is to take control over the potential problems before the problems control you” (Amland 1999)

D.W. Karolak (Karolak 1996) provided a risk analysis model in his book “*Software Engineering Risk Management*”.

Amland provides in his paper a “risk analysis activity model”, a modified version of Karolak’s risk analysis model to fit into the software test process.

It consists of the following six steps:

- Risk Identification
- Risk Strategy
- Risk Assessment
- Risk Mitigation
- Risk Reporting
- Risk Prediction

2.2 Risk Identification

This step is probably the most relevant step as far as this thesis is concerned because it tries to answer questions such as: “Is there risk to this function or activity? How can it be classified?” (Amland 1999)

This thesis through its taxonomy and collection of failure modes (risks) tries to provide the foundation for determining the answers to the above question.

Amland defines this risk identification “as the process of collecting information about possible risks to the application[2] and classifying it to determine the amount of potential risk in the test phase and in production (in the future).”

I am expecting that a tester or test manager who is performing the risk analysis for their e-commerce site to look for information about potential risks from the taxonomy and its collection of more than 700 failure modes. The failure modes are essentially a collection of possible risks to the system. In addition to that they might look at some failure mode from the list and get inspired to come up with a similar risk they think their system might face.

2.3 Risk Strategy

“Risk based strategizing and planning involves the identification and assessment of risks and the development of contingency plans for possible alternative project activity or the mitigation of all risks.” (Amland 1999)

[2] Changed the original statement from “project” to “application”

During this assessment, the tester or test manager will probably identify and prioritize what they think are the critical risks that will affect the application adversely. The list of e-commerce risks in this thesis will provide them a top-level view of many types of risks that they can look at, and if they find them critical and high risk they can include them in their priority list.

2.4 Risk Assessment

Amland defines this step as the step to "determine the effects (including costs) of potential risks". Since this step involves asking questions such as "What are the consequences if this particular function fails?" the step could be said to be similar to the effects analysis section of FMEA (Failure Modes and Effects Analysis) that is discussed in detail in the next chapter. This thesis provides a list of causes and symptoms of e-commerce failures in its taxonomy and risk/failure list and leaves the tester/test manager/ risk analyst to assess how that failure/risk affects their system.

2.5 Risk Mitigation

"Risk mitigation involves taking some immediate, pro-active step to reduce the probability or the impact of the risk" as defined by Rational's Unified Process. Setting up inspections are one way of mitigating risks by identifying high-risk functions and getting the testers to focus more on them to reduce the impact of failure when it occurs. During the inspection, testers who are sharp might look at the risk list provided in this thesis, find a pattern of higher occurrence in the failures for some functions, decide to assign them to the list of "high-risk functions," and design tests that concentrate on these functions more thoroughly.

2.6 Risk Reporting

This step involves reporting information found in the previous steps. Paul Gerrard's (Gerrard 2002) article on stickyminds.com, *'Risk-Based Test reporting'* is a good article to read more about risk reporting.

2.7 Risk Prediction

Predicting risks is possible by looking at the information collected in the previous steps. An idea about the type of risks, where, when and how they occur, makes the forecast of impending risks more accurate. According to Amland, information about the history and knowledge of previously identified risks helps to predict risks correctly.

This thesis contains listing of over 300 real life bugs that have affected e-commerce sites. This list can be a valuable asset in the hands of a tester who is looking for historical data about failures that have occurred in the past. The tester can base his predictions about possible new risks on them.

2.8 Risk-Based Testing: Doing it the "Outside-In" Way

James Bach wrote an article titled "*Heuristic Risk-Based Testing*" in the STQE (Software Testing and Quality Engineering Magazine, 11/99). In this excellent article, he suggests two approaches to do the process of risk analysis – the Inside-Out way and the Outside-In way. (Bach 1999)

In the Inside-Out way, the tester walks through the application and does a risk analysis by asking a number of questions such as “What if the function in this box fails? What would happen if it were broken? Can this function ever be invoked at the wrong time?”

In the Outside-In way he suggests that you “consult a predefined list of risks and determine whether they apply here and now.” He also exemplifies the approach by using the ISO 9126 standard for quality attributes. The taxonomy and the risk list provided in this thesis is one such predefined list of risks applicable to e-commerce sites. The list is broad, and hence a valuable tool in the hands of a new tester who decides to opt for the Outside-In approach for risk analysis.

2.9 Summary

This chapter summarizes the topic of risk-based testing and ties it to the relevance of the thesis and ties the taxonomy to it. The Risk-based testing takes a similar approach to the FMEA, which is the topic for the next chapter. Risk- based testing could be a good technique, especially when all critical functions need to be tested and the time available to test is short. It increases customer confidence since all the major bugs will be caught and the chances of massive failures are low.

Chapter 3: Failure Mode and Effect Analysis (FMEA)

3.1 Overview

Pentti & Atte (2002) in their research study introduce FMEA as:

"… One of the well-known analysis methods and has an established presence in the traditional reliability analysis. The purpose of FMEA is to identify the different failure modes of the components in a system, evaluate their effect on the system behavior and finally propose proper counter-measures to overcome these effects. The FMEA has become a generic standard that has wide applicability ranging from process management to the aerospace industry…" (p.3)

There are diverse interpretations, practices and standards, available for FMEA. Much of the discussion in this chapter is in respect to the experiences of applying FMEA to software systems/information systems.

The structure and some of the content in this chapter is based on the discussions in the research study: PASSI (Programmable Automation System Safety Integrity Assessment) belonging to the Finnish Nuclear Safety Research Programme.
(Pentti & Atte, 2002)

According to the study, "FMEA is well understood at the systems and hardware levels, where potential failure modes are usually known and the task is to analyze their effects on system behavior." (p.3)

With the increased implementation of system functions in the software level, there is an increased interest in exploring the prospects of applying FMEA on software-based systems. Also, it is generally deduced that a bottom-up technique, such as FMEA, is an effective way to identify and document a component that has failed or is malfunctioning.

3.2 History

The historical timeline and the discussions below are based upon history provided by Pentti & Atte (2002) with additional information from

www.fmeca.com (http://www.fmeca.com/ffmethod/history.htm)

FMEA can be traced back to the US military and to its military standard MIL-P-1629 developed about 50 years ago. Through the following decades, its use has been widespread across various fields such as automotive, space and health care. FMEA has absorbed methods and practices from the different industries it was adopted by. That has grown to become its strength.

Pentti & Atte (2002) chart the historical time line as:

1949: US Military

"The FMEA discipline was originally developed in the United States Military. Military procedure MIL-P-1629, titled 'procedures for performing a failure mode, effects and criticality analysis' dated on November 9th, 1949. The method was used as a reliability evaluation technique to determine the effect of system and equipment failures. Failures were classified according to their impact on the military mission success and personnel/equipment safety. The concept that personnel and equipment are interchangeable does not apply, for example in the modern manufacturing context of producing consumer goods, and therefore the manufacturers in different industries have established new sets of priorities, guidelines and standards of their own. However, military procedure MIL-P-1629 has functioned as a model for later military standards MIL-STD-1629 and MILSTD-1629A, which illustrate the most widely used FMEA procedures." (p.11)

1960's: Apollo Missions and Aerospace Industry

"Outside the military, the formal application of FMEA was first adopted by the aerospace industry, where FMEA was already used during the Apollo missions in the 1960's. Academic discussion on FMEA originates from the 1960's when studies of component failures were broadened to include the effects of component failures on the system of which they were a part. One of the earliest descriptions of a formal approach for performing a FMEA was given at the New York Academy of Sciences (Coutinho, 1964)." (p.11)

1970's: Software Failure Mode and Effects Analysis

"In the late 1960's and early 1970's several professional societies published formal procedures for performing the analysis. The generic nature of the method assisted the rapid broadening of FMEA to different application areas and various practices fundamentally using the same analysis methods. Along with the digital revolution, the FMEA was applied in the analysis of software-based systems and one of the first articles regarding software failure mode and effects analysis (SWFMEA) was given by Reifer (1979)." (p.11)

1980's: Automotive Industry and QS 9000

"In the early 1980s, United States automotive companies began to formally incorporate FMEA into their product development process. A task force representing Chrysler Corporation, Ford Motor Company and General Motors Corporation developed QS 9000 standard in an effort to standardize supplier quality systems. QS 9000 is the automotive analogy to better known standard ISO 9000. QS 9000 compliant automotive suppliers must utilize FMEA in the advanced quality planning process and in the development of their quality control plans. The effort made by the task force led to an industry-wide FMEA standard SAE J-1739 issued by the Society of Automotive Engineers' in 1994." (p.11)

1985: Software FMEA-IEC 60812

"Even thought there is no explicit standard for SWFMEA, the standard IEC 60812 published in 1985 is often referred when carrying out FMEA for software-based systems." (p.11)

3.3 The FMEA Approach

When a tester deals with a new software system, s/he must seek answers in a systematic manner for questions such as; what could go wrong with this software component or with the process that created it; how ugly is the bug going to look and what can be done in the future to prevent such bugs?

FMEA raises similar questions in a structured way. It raises questions about what could go wrong with a system or the process that was involved with creating the system. The effects analysis part deals with the consequence of the failure.

Some purposes of FMEA are relevant to the software-testing world such as:

3.3.1. Early Identification of Potential Design and Process Related Failure Modes.

K.H.Pries (1998) states in his SAE technical paper, “FMEA for software development,t” that a process or design can be changed early enough in the development if potential problems are identified. Boehm had stated “the cost of fixing a bug is higher in the later stages of software cycle than in the earlier ones.” (Boehm, 1976) If I were to draw a conclusion from the above statements, I would infer that FMEA could be used to brainstorm a list of potential failures and such a list can be used to fix design faults and expensive bugs that can be difficult to address at a later stage.

3.3.2. Serve as a Prelude to Test Design

“Pries also notes that software design FMEA is a listing of potential problems and is thus a listing of test cases. Pries points out that any effort put into the FMEA would be helpful in test case development. Pries further states that test descriptions developed from FMEA challenge the software product at its weakest points by testing for anomalies.” (Pentti & Atte, 2002, pg 12)

3.3.3. Prioritize a List of Potential Failures

The probability of occurrence of a failure mode, the severity of the effect of the failure mode, and the probability of detection of the failure mode determine the prioritization of failures. The prioritized list of potential failures essentially provides an ordered list of test cases to the tester. The decision to run a particular test case or the decision to fix a particular bug can be made after referring to the failure list.

In practice, the FMEA is achieved through a set of iterative process steps. The main steps are illustrated in fig.3.1

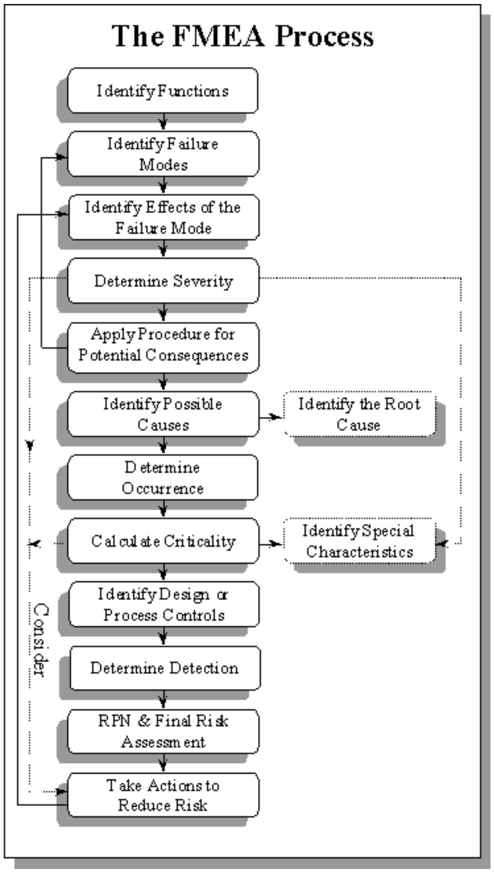

The FMEA Process
Identify Functions
Identify Failure Modes
Identify Effects of the Failure Mode
Determine Severity
Apply Procedure for Potential Consequences
Identify Possible Causes
Identify the Root Cause
Determine Occurrence
Calculate Criticality
Identify Special Characteristics
Identify Design or Process Controls
Determine Detection
RPN & Final Risk Assessment
Take Actions to Reduce Risk
Consider

Fig 3.1: Source: http://www.fmeca.com/ffmethod/fmeaproc.htm

As you can see, after the confirmation of the component that is going to be the subject of the FMEA, the next step is the identification of failure modes. Many different tools exist to help in the brainstorming procedure and to subsequently document the failure modes. But the quest of this master's thesis has been to facilitate the generation of failure modes through the use of fault taxonomies. In short, this thesis deals with creating a top-level fault taxonomy with a large number of categories and using it as an idea generation tool for brainstorming a list of failure modes.

3.4 Types of FMEA

Pentti & Atte (2002) in their study present this FMEA classification provided by SAG in 1996:

FMEA is generally classified as either a product FMEA or process FMEA

3.4.1 Product FMEA: "Product FMEA analyzes the design of the product by examining the way that the item's failure modes affect the operation of the product" (p.13)

3.4.2 Process FMEA: "The process FMEA analyzes the process involved in the design, building and maintaining a product by examining the way the failures in the manufacturing or service processes affect on the operation of the product." (p.13)

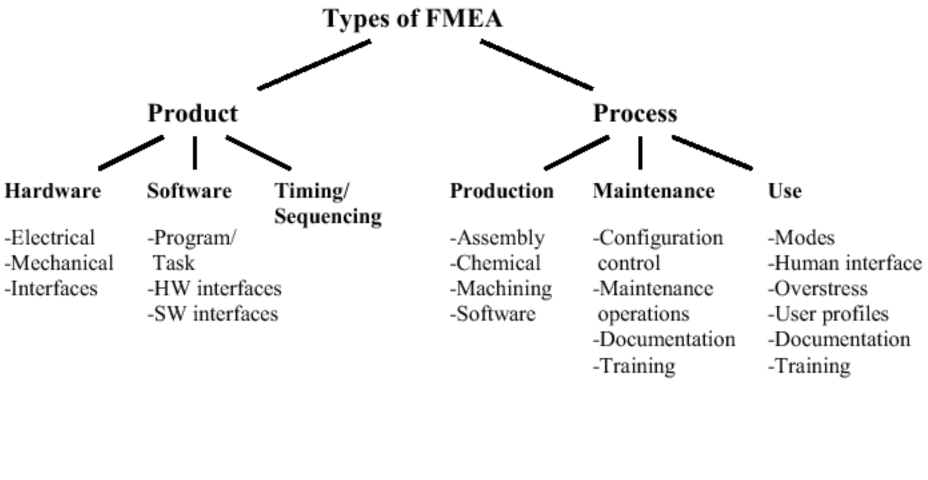


Fig 3.2 lists the processes and products within the software realm that fall under each category. (Source: “FMEA of software based automation systems” http://www.stuk.fi/julkaisut/tr/stuk-yto-tr190.pdf, pg 13)

3.5 Software Failure Modes and Effects Analysis (SFMEA)

Performing FMEA for a mechanical, automotive or electrical system is comparatively more straightforward compared to the process of performing FMEA on a software system because:

- There is historical data available about failures in these systems
- There is data provided about failures by the component manufacturers
- Test data may also be available for the components from their individual manufacturers
- There may be some inputs from individuals with operational experience about failure modes
- Mechanical and electrical components are expected to fail due to wear and tear or sudden spikes of stress and hence identification of related failure modes is more straightforward.

For software based systems, the situation is different because:

- There little historical data available because of the diversity of the components, proliferation of tools, and the ever-changing aspect of software technology
- Historical records are less frequently kept.

- The software industry itself is relatively young compared to the others and hence has less experience in cataloguing failures.
- Not much test data is generally available from the component manufacturers
- The Test engineers generally depend on personal knowledge or the about the software and does not generally find someone with a large amount of operational experience (unless the software has been in the market for a while).
- And most importantly, the software modules are "not perceived to fail, they only exhibit incorrect behavior." (Pentti & Atte, 2002)

The failure modes in SWFMEA need to reflect the incorrect behavior of the software and emphasize it. The steps in SWFMEA are mostly the same as described in fig 3.1

Pentti & Atte (2002) in their study quote Ristord, L and Esmenjaud (Ristord, L and Esmenjaud, 2001) and list their list of distinguishing characteristics between FMEA for hardware and software:

Hardware FMEA

- "May be performed at functional level or part level.
- Applies to a system considered to be free from failed components.
- Postulates failures of hardware components according to failure modes due to ageing, wearing or stress.
- Analyses the consequences of these failures at system level.
- States the criticality and the measures taken to prevent or mitigate the consequences." (p. 14)

Software FMEA

- "Is only practicable at functional level.
- Applies to a system considered as containing software faults, which may lead to failure under triggering conditions.
- Postulates failures of software components according to functional failure modes due to potential software faults.
- Analyses the consequences of these failures at system level.
- States the criticality and describes the measures taken to prevent or mitigate the consequences.
- Measures can, for example, show that a fault leading to the failure mode will be detected by the tests performed on the component, or demonstrate that there is no credible cause leading to this failure mode due to the software design and coding rules applied." (P.14)

3.6 Survey of Literature on FMEA

3.6.1 FMEA and the Software Industry, Software Development, Information Systems

There is very little information published on the use of FMEA for information systems (Banerjee, 1995). Almost all FMEA research has been directed toward manufactured products rather than information systems.

Mike Signor (2000) wrote a new tool called Failure Analysis Matrix (FAM) "The Failure Analysis Matrix: A Usable Model for Ranking Solutions to Failures in

Information Systems" and in his proposal conducted a very comprehensive survey of FMEA in Information Systems and the software industry in general. The same survey was also adapted by (Pentti & Atte, 2002) in their study for PASSI. The survey is summarized as follows:[3]

3.6.2 The FMEA in Information Systems

"Banerjee (1995) provided an insightful look at how teams should use FMEA in software development. Banerjee presented lessons learned in using FMEA at Isardata, a small German software company. FMEA requires teamwork and the pooled knowledge of all team members. Many potential failure modes are common to a class of software projects. Banerjee also pointed out that the corresponding recommended actions are also common. Good learning mechanisms in a project team or in an organization greatly increase the effectiveness of FMEA. FMEA can improve software quality by identifying potential failure modes. Banerjee stated that FMEA could improve productivity through its prioritization of recommended actions"

"Goddard (1993) described the use of SFMEA at Hughes Aircraft. Goddard noted that performing the SFMEA as early as possible allows early identification of potential failure modes. He pointed out that a static technique like FMEA cannot fully assess the dynamics of control loops."

Goddard (2000) stated that there are "two types of software FMEA for embedded control systems:

1. System software FMEA and

[3] With written permission to summarize and with suggested citation from Mike Signor through email dated 31st December 2002.

2. Detailed software FMEA.

System software FMEA can be used to evaluate the effectiveness of the software architecture without all the work required for detailed software FMEA. Goddard noted that system software FMEA analysis should be performed as early as possible in the software design process. This FMEA analysis is based on the top-level software design. Goddard stated that the system software FMEA should be documented in the tabular format used for hardware FMEA."

"**Goddard (2000)** stated that detailed software FMEA validates that the software has been constructed to achieve the specified safety requirements. Detailed software FMEA is similar to component level hardware FMEA. Goddard noted that the analysis is lengthy and labor intensive. He pointed out that the results are not available until late in the development process. Goddard argued that detailed software FMEA is often cost effective only for systems with limited hardware integrity.

"**Lutz and Woodhouse (1996)** described their use of Software Failure Modes and Effects Analysis (SFMEA) in requirements analysis at the Jet Propulsion Laboratory. SFMEA helped them with the early understanding of requirements, communication, and error removal. Lutz and Woodhouse noted SFMEA is a time-consuming, tedious, manual task. SFMEA depends on the domain knowledge of the analyst. They pointed out that a complete list of software failure modes cannot be developed.

"**Moriguti (1997)** provided a thorough examination of TQM for software development. Chapter 3.5 on Reliability Methods presented an overview of FMEA. The chapter pointed out that FMEA is a bottom-up analysis technique for discovering imperfections and hidden design defects. Moriguti suggested performing the FMEA on subsystem-level functional blocks. Moriguti noted that when FMEA is performed on an entire product, the effort often gets out of hand. Moriguti pointed out using FMEA before the fundamental design is completed can prevent extensive rework. Moriguti explained that when prioritization is emphasized in the FMEA, the model is sometimes referred to as Failure Modes, Effects and Criticality Analysis (FMECA).

“**Luke (1995)** discussed the use of FMECA for software. He pointed out that early identification of potential failure modes is an excellent practice in software development. It helps in the design of tests to check for the presence of the failure modes. Luke noted that software requirements not being met are failure modes. In FMECA, a software failure may have effects in the current module, in higher-level modules, and the system as a whole. Luke proposed that proxies such as McCabe's complexity value and historical failure rate be substituted for occurrence.

“**Pfleeger (1998)**, **Saeed, de Lemos, and Anderson (1995), and Fenelon and McDermid (1993)** briefly discussed the use of FMEA in software engineering. Pfleeger noted that FMEA works from known failure modes to unknown system effects. Pfleeger, Fenelon and McDermid pointed out that FMEA is highly labor intensive and relies on the experience of the analysts. Saeed et al. stated that FMEA uses inductive reasoning to start with a set of failure events and reason forward to possible effects.

“**Pries (1998)** outlined a procedure for using software design FMEA. Pries stated that software design FMEA should start with system or subsystem outputs listed in the Item and Function (leftmost) columns of the FMEA. The next steps are to list potential failure modes, effects of failures, and potential causes. Pries noted that current design controls could include design reviews, walkthroughs, inspections, complexity analysis, and coding standards.

“Pries argued that because reliable empirical numbers for occurrence values are difficult or impossible to establish, FMEA teams could set all occurrences to a value of 5 or 10. Pries noted that detection numbers are highly subjective and heavily dependent on the experience of the FMEA team.

“Stamatis (1995) presented the use of FMEA with information systems.

He noted that computer industry failures might result from software development process problems, coding, systems analysis, systems integration, software errors, and typing errors. Stamatis pointed out that failures might arise from the work of testers, developers, and managers. Stamatis noted that a detailed FMEA analysis might examine the source code for errors in logic and loops, parameters and linkage, declarations and initializations, and syntax.

“**Ammar, Nikzadeh, and Dugan (1997)** used severity measures with FMEA for a risk assessment of a large-scale spacecraft software system. They noted that severity considers the worst potential consequence of a failure, whether degree of injuries or system damages. Ammar, Nikzadeh, and Dugan used four severity classifications. Catastrophic failures are those that may cause death or system loss. Critical failures are failures that may cause severe injury or major system damage that result in mission loss. Marginal failures are failures that may cause minor injury or minor system damage that result in delay or loss of availability or mission degradation. Minor failures are not serious enough to cause injuries or system damage but result in unscheduled maintenance or repair.

“**Maier (1997)** described the use of FMEA during the development of robot control system software for a fusion reactor. He used FMEA to examine each software requirement for all possible failure modes. Failure modes included an unsent message; a message sent too early, a message sent too late, a wrong message, and a faulty message. FMEA causes included software failures, design errors, and unforeseen external events. Maier noted that for software failures, additional protective functions to be integrated in the software might need to be defined. For design errors, the errors may need to be removed or the design may need to be modified. Maier stated that unforeseen external events may be eliminated by protective measures or changing the design. Maier recommended that the methodology he presented be applied at an early stage of the software development process to focus development and testing efforts.

"**Bouti, Kadi, and Lefrançois (1998)** described the use of FMEA in an automated manufacturing cell. They noted that a good functional description of the system is necessary for FMEA. They recommended the use of an overall model that clearly specifies system functions. They suggested the use of system modeling techniques that facilitate communication and teamwork. Bouti, Kadi, and Lefrançois argued that it is impossible to perform a failure analysis when functions are not well defined and understood. They pointed out that failure analysis is possible during the design phase because the functions are well established by then. Bouti, Kadi, and Lefrançois also noted that when several functions are performed by the same component, possible failures for all functions should be considered.

"**Becker and Flick (1996)** applied FMEA in Lockheed Martin's development of a distributed system for air traffic control. They described the failure modes and detection methods used in their FMEA. The classes of failure modes for their application included hardware or software stop, hardware or software crash, hardware or software hang, slow response, startup failure, faulty message, checkpoint file failure, internal capacity exceeded, and loss of service. Becker and Flick listed several detection methods. A task heartbeat monitor is coordination software that detects a missed function task heartbeat. A message sequence manager checks message sequence numbers to flag messages that are not in order.

A roll call method takes attendance to ensure that all members of a group are present. A duplicate message check looks for the receipt of duplicate messages.

“**Stålhane and Wedde (1998)** used FMEA with a traffic control system in Norway. They used FMEA to analyze changes to the system. They noted that potentially any change involving an assignment or a procedure call could change system parameters in a way that could compromise the system’s safety. The FMEA pointed out code segments or procedures requiring further investigation. Stålhane and Wedde also stated that for an FMEA of code modifications, implementation and programming language knowledge is very important” (Mike Signor, 2000 pg. 74-96)

3.7 Standards Associated With FMEA

3.7.1 IEC 60812

“The International Electro-technical commission publishes IEC 60812 and it describes a FMEA and a FMECA. It describes how the analysis is implemented using FMEA or FMECA. It provides the following:

- The procedural steps necessary to perform an analysis
- Identify appropriate terms, assumptions, criticality measures and failure modes.
- Determine basic principles and ground rules
- Provide examples of the necessary forms.” (Signor 2000, pg 96)

3.7.2 SAE J 1739

“This standard outlines the product and process FMEA for plant machinery and equipment. It contains the following details

The topic of potential Failure Mode and Effects Analysis (FMEA) and gives general guidance in the application of the technique.

- Suggests approach on performing process/product FMEA on machinery
- Suggests evaluation criteria for severity, detection and occurrence of failure” (Signor 2000, pg 96)

3.7.3 MIL-STD-1629A

The United States DoD published this standard on November 24th 1980.

It provided the steps of performing the FMECA and evaluates and documents effects of functional/hardware failure on mission success, personnel and system safety.

The MIL-STD-1629A was cancelled by the action of the standard authority on August 4th 1998.” (Signor 2000, pg 96)

3.8 Examples of Failure Modes for Software Components

3.8.1 Hardware components

Failure modes can be brainstormed from operational experience.

Component manufacturer may give failure modes and frequencies for their product.

3.8.2 Software Components

(Reifer, 1979) gives the following general “list of failure modes based on the analysis of three large software projects:

- Comput ational
- Logic
- Data I/O
- Data Handling
- Interface
- Data Definition
- Data Base
- Other”

Ristord et.al. (2001) gives the following list of “five general purpose failure modes at processing unit level

- Operating system stops.
- Program stops with clear message.
- Program stops without clear message.
- The program runs, producing obviously wrong results.
- The program runs, producing apparently correct but in fact wrong results"

Lutz et.al. (1999b) "divide the failure modes concerning either the data or the processing of data. For each input and each output of the software component, they consider of the following four failure modes:

- Missing Data (e.g., lost message, data loss due to hardware failure)
- Incorrect Data (e.g., inaccurate data, spurious data)
- Timing of Data (e.g., obsolete data, data arrives too soon for processing)
- Extra Data (e.g., data redundancy, data overflow)."

For each event (step in processing), on the other hand, they consider the following four failure modes:

- "Halt/Abnormal Termination (e.g., hung or deadlocked at this point)
- Omitted Event (e.g., event does not take place, but execution continues)
- Incorrect Logic (e.g., preconditions are inaccurate; event does not implement intent)
- Timing/Order (e.g., event occurs in wrong order; event occurs too early/late)."

Becker et. al. (1996) gives the following "classes of failure modes:

- Hardware or software stop

- Hardware or software crash
- Hardware or software hang
- Slow response
- Startup failure
- Faulty message
- Checkpoint file failure
- Internal capacity exceeded
- Loss of service."

3.9 Table of FMEA Related Definitions:

The following definitions have been obtained from the definitions provided at www.fmeca.com (http://www.fmeca.com/ffmethod/definiti.htm)

Cause: "A Cause is the means by which a particular element of the design or process results in a Failure Mode."

Criticality: "The Criticality rating is the mathematical product of the Severity and Occurrence ratings. Criticality = (S) × (O). This number is used to place priority on items that require additional quality planning."

Current Controls: “Current Controls (design and process) are the mechanisms that prevent the Cause of the Failure Mode from occurring, or which detect the failure before it reaches the Customer.”

Detection: “Detection is an assessment of the likelihood that the Current Controls (design and process) will detect the Cause of the Failure Mode or the Failure Mode itself, thus preventing it from reaching the Customer.”

Effect: “An Effect is an adverse consequence that the Customer might experience. The Customer could be the next operation, subsequent operations, or the end user.”

FMEA Element: “FMEA elements are identified or analyzed in the FMEA process. Common examples are Functions, Failure Modes, Causes, Effects, Controls, and Actions. FMEA elements appear as column headings in the output form.”

Occurrence: “Occurrence is an assessment of the likelihood that a particular Cause will happen and result in the Failure Mode during the intended life and use of the product.”

Risk Priority Number: “The Risk Priority Number is a mathematical product of the numerical Severity, Occurrence, and Detection ratings.
RPN = (S) * (O) * (D). This number is used to place priority on items than require additional quality planning.”

Chapter 4: Taxonomies

4.1 Definition

"*Taxonomy* is the organization of a particular set of information for a particular purpose." (searchtools)

Whatis.com (whatis) explains the meaning of taxonomy by giving the meaning of the word from its root words, 'taxis' and 'nomos' where *taxis* in Greek means arrangement or division and *nomos* means law.

It further describes taxonomy "as the science of classification according to a pre-determined system, with the resulting catalog used to provide a conceptual framework for discussion, analysis, or information retrieval. In theory, the development of a good taxonomy takes into account the importance of separating elements of a group (taxon) into subgroups (taxa) that are *mutually exclusive, unambiguous, and taken together, include all possibilities*[4]. In practice, a good taxonomy should be simple, easy to remember, and easy to use."

4.2 The World of Taxonomies

Taxonomies have been in use for ages but the name of Carl Linnaeus (1707-1778), also known, as Carl von Linné or Carolus Linnaeus, is significant since he is often

[4] The goal of the taxonomy for e-commerce failures presented in this thesis was to develop a useful list of top-level categories where the emphasis was more on providing a structure that helps in generating a larger sets of failure modes/ test ideas than creating a smaller set of top-level mutually exclusive/unambiguous taxonomy. A discussion on the non-exclusivity and ambiguous nature of e-commerce bugs is provided in the next chapter.

called the father of taxonomy. He first published his system for naming, ranking, and classifying organisms in Systema Naturae (1735). Taxonomies are ever-evolving and far from reaching a complete state. This can be best seen in Linnaeus's own work the Systema Naturae, which he continued to revise and which grew from a slim pamphlet to a multi-volume work as more plant and animal specimens were sent to him from every corner of the globe.

Long before Carl Linneaus, there were many who worked on classifying and organizing objects into a pre-determined structure. Some of these works are documented and some others are lost making it difficult to attribute authorship to these works. But notable contributions to the field of taxonomy in the ancient world to the modern world are that of:

Aristotle (384-322 BCE), distinguished species by habitat and means of reproduction

Theophrastus (d. 287 BCE), Greek philosopher who identified five hundred plant types.

Charaka (1st century AD), India, classified 1500 medicinal plants into about 50 categories.

Bauhin, Gaspard (1560–1624), Swiss botanist and doctor of medicine, created an early classification of over 6000 plants by genus and species in his chief work, the Pinax theatri botanici (1623)

Voltaire (1694 – 1778), likened the hierarchical arrangement of species to political and religious hierarchies in the Philosophical Dictionary (1764)

Cuvier, Georges (1769-1832), French comparative anatomist who is considered the founder of functional anatomy, extended Linneaus taxonomy and listed over fifty thousand types of plants.

Sir Isaac Newton's classification of the heavenly bodies in Principia Mathematica (1687)

Taxonomies have been created and used widely from physical sciences to physical anthropology. In the business world we see a lot of talk on enterprise taxonomies and business taxonomies. In psychology we hear about personality taxonomies, gesture taxonomies and Krathwohl's affective taxonomy, educational psychologists use what is famous as Bloom’s taxonomy of educational objectives (1956). Many fields have used classification systems and taxonomies, but their applications and requirements have been different.

We have seen the word taxonomy also being used to describe the thesaurus or classification scheme, to organize information on the web.

In fact Sarah L. Roberts-Witti, in her article “Practical taxonomies: hard-won wisdom for creating a workable knowledge classification system” (Roberts-Witti, 1999) introduces the concept of taxonomies by describing Yahoo! as one of the largest and most familiar “knowledge classification systems in the world, fronted by a basic but highly useable taxonomy.” Apart from these knowledge taxonomies being a way of categorizing, indexing and cataloguing controlled vocabularies, they are also designed to provide users with methods to access the information they hold.

In conclusion, taxonomies have been present for a very long time and have evolved in both use and definition over the time. Just as Liz Edols concludes in her article, “Taxonomies are what?” (Edols, 2001) “While there is still some confusion about how to describe taxonomies, the basic concept is that it should be a useful one”

4.3 Taxonomies and Computer Science

Taxonomies have found wide applicability in areas of computer science where an organized and systematic approach of organizing is needed. For example, instances of the usage of the word ‘taxonomy’ in computer science can be found in an array of topics such as Taxonomy of human-computer Interactions, Taxonomy of computer system architectures, Taxonomy of computer inputs, Taxonomical classification of meta-data, Taxonomies in controlled vocabularies, etc

For example, parallel computing could be cited as one of areas of computer science where taxonomies have been used. Listed below are some examples of taxonomies created to classify computers in parallel computing:

4.3.1 Parallel Computer Taxonomies

Wasel Chemij (Chemij, 1994) in his M.Phil dissertation discussed in detail the taxonomies in parallel computing and also proposed his own proposed taxonomy. The following discussion on parallel computer taxonomies is based on his dissertation. It is recommended that the reader refer to the dissertation for a detailed discussion.

4.3.2 Flynn’s Computer Classification

In 1966, Michael Flynn developed a taxonomy (Flynn, 1966) of computer systems based on the notions of streams of instructions and data. It was his observation that one could describe, at a very coarse level, the degree of parallelism of a computer based on these two parameters: either the instruction or the data stream could be either serial or parallel. His taxonomy divided computers into four classes (Chemij, 1994, ch.7):

- SISD (Single Instruction Single Data)
- MIMD (Multiple Instruction Multiple Data)
- SIMD (Single Instruction Multiple Data)
- MISD (Multiple Instruction, Single Data)

4.3.3 Handler's Classification (1977)

“He proposed an elaborate notation for expressing the pipelining and parallelism of computers. Handler's taxonomy (Handler) addresses the computer at three distinct levels: the processor control unit (PCU), the arithmetic logic unit (ALU), and the bit-level circuit (BLC)” (Chemij, 1994, ch.7)

4.3.4 Shore's Taxonomy (1973)

“Shore proposed his taxonomy in 1973. It is based on the structure and number of the functional units in the computer. Shore's taxonomy has six categories each designated by a number. Type-I to Type-VI” (Chemij, 1994, ch.7)

4.3.5 Hockney and Jesshope's Structural Taxonomy

“Hockney and Jesshope developed an elaborate notation called Algebraic-style Structural Notation, (or ASN,) to describe parallel computers. This notation is the basis of their structural taxonomy (Hockney and Jesshope, 1998). The taxonomy is arranged as several trees, with a machine being described by all the labels in all the parent nodes all the way back to the root node.” (Chemij, 1994, ch.7)

4.3.6 IEEE STD 1044-1993: IEEE Standard Classification for Software Anomalies

This standard provides a standard approach to classification of anomalies found in software. It describes the process in a software life cycle as to how anomalies are recorded and processed. According to the IEEE standards web site, this standard has been withdrawn. According to the standard, the process is divided into four sequential steps interspersed with three administrative activities. The sequential steps are:

- Recognition
- Investigation
- Action
- Disposition

The three administrative steps applied to each sequential step are as follows:

- Recording

- Classifying
- Identifying impact

“Table 3c: Investigation classification scheme” in the standard contains a long causal list of generic software faults.

4.4 Taxonomy of Taxonomies

Some taxonomies are system specific (i.e. eccentric -- suitable for only one environment and application) and few others are generic and can be applied generally to a wide range of systems. Error taxonomies classify types of errors and error mechanisms. Learning taxonomies deal with required behaviors and types of learning. Functional taxonomies look at system functions.

Vulnerability taxonomy, Incident taxonomy, Attack taxonomy, deal with the classification of security bugs. The following section in this chapter summarizes the taxonomies that focus on the classification of security issues such as computer misuse and malicious attacks. During my literary survey I observed that though taxonomies have been created and used in various specializations within computer science, it has found maximum applicability and use with the computer security community. The next section surveys many of the security related “Computer attack taxonomies”.

4.5 Computer Attack Taxonomies (Lough 2001)

Many theses and dissertations have focused on creating, analyzing and comparing taxonomies for security issues. Notable among those are those from Purdue (Kumar, 1994, Aslam, 1995 and Krusul, 1998) and other dissertations

(Howard, 1997 and Lough, 2001). This survey follows closely the outline and content of the literary survey in Lough's dissertation (Lough 2001) on "Taxonomy of Computer Attacks with applications to wireless networks."

For more details about each work, refer to the individual dissertations and thesis.

The computer attack taxonomies concentrate predominantly on security issues. In my broader classification of e-commerce failures (which is more generic) security-related issues constitute only about three categories. But these taxonomies have been quoted in this literature survey because they are good examples of what computer-related taxonomies look like and what they were created for (objectives).

List of "Computer Attack Taxonomies" as listed by Lough (Lough 2001)

- Anderson' Penetration Matrix
- SRI Computer Abuse Methods Model (Neumann and Parker)
- Lindqvist and Jonsson's Extension of Neumann and Parker
- Jayaram and Morse's Network Security Taxonomy
- John Howard's CERT Taxonomy
- Sandia Laboratory Taxonomy
- Kumar's Classification and Detection of Computer Intrusions
- Aslam's UNIX Security Taxonomy
- Krusul's Taxonomy
- Bishop's Vulnerabilities Classification Scheme
- Richardson's extension to Krusul's Taxonomy

- Linde’s Generic System Functional Flaws
- RISOS Operating System Security Flaws
- Protection Analysis
- Landwehr’s Taxonomy (1994)
- Attansio’s Flaw Hypothesis Methodology (FHM) Penetration Characteristics
- Brinkley’s and Schell’s Computer Misuse Techniques
- Knight’s Vulnerability Taxonomy
- SRI Security Breaching Incidents
- Perry and Wallich’s Attack Matrix
- Parker’s Taxonomies
- Straub and Widom’s Motivation-Security Response Taxonomy
- Ristenbatt’s Methodology for Network Communication Vulnerability
- Du and Mathur’s Taxonomy
- Brian Marick’s Survey
- Chillarege’s Orthogonal Defect Classification

4.5.1. Anderson’ Penetration Matrix

James P. Anderson (Ande1980) developed a four-cell matrix that covers the types of penetrators, based on whether they are authorized to use the computer and data/program source.

His categories are:

A. External Penetration

B. Internal Penetration
  - The masquerader
  - Legitimate user
  - Clandestine user

C. Misfeasance

Anderson introduced an alternate taxonomy of threats to computers. He states that the *objective* of this study was to improve the computer security auditing and surveillance capability of the customer's systems.

4.5.2. SRI Computer Abuse Methods Model

Neumann and Parker (Neumann and Parker, 1989) published a series of their evolving model starting from 1989 through 1995. Their outline was based on a series of classes of computer misuse from their data of about 3000 cases over twenty years. They contain 9 categories and Neumann later extended the categories into twenty-six types of attacks.

Their nine categories are:

- External
- Hardware Misuse
- Masquerading
- Pest Programs
- Bypasses
- Active Misuse
- Passive Misuse
- Inactive Misuse

- Indirect Misuse

The *objectives* of this taxonomy were to “provide a basis for methodological threat analysis that assesses the significance of vulnerabilities in specific systems and networks. It is intended to increase the understanding of exploitable abuse techniques, and thereby to aid in reducing both the number of vulnerabilities and their seriousness.”

4.5.3. Lindqvist and Jonsson’s Extension of Neumann and Parker

Lindqvist and Jonsson extended Neumann and Parker’s model by expanding three of their categories- Bypass, Active Misuse, and Passive Misuse. Along with the extensions they added intrusion techniques and created a classification of intrusion results. They summarize their *objective* as a “step on the road to an established taxonomy of intrusions for use in incident reporting, statistics, warning bulletins, intrusion detection system etc.” (Lindqvist 1997)

4.5.4. Jayaram and Morse’s Network Security Taxonomy

Jayaram and Morse developed a taxonomy of security threats to networks. Their taxonomy consists of five categories

- Physical
- System Weak Spots
- Malign Programs
- Access Rights
- Communication- based

The *objective* behind this taxonomy was to list the class of security threats and mechanisms for meeting these threats in Internetworks. (Jayaram 1997)

4.5.5. John Howard’s CERT Taxonomy

Howard, in his PhD dissertation categorized the CERT incidents from 1989-1995. He created a new taxonomy with types of attackers, tools used, access information, results of the break-in and the objectives of the attack.

The *objective* behind this taxonomy was the “development of a taxonomy for the classification of Internet attacks and incidents, organization, classification, and analysis of incident records available at the CERT (R)/CC, and development of recommendations to improve Internet security, and to gather and distribute information about Internet security.” (Howard 1997)

4.5.6. Sandia Laboratory Taxonomy

This taxonomy is similar to Howard’s taxonomy with a few extra categories and with some categories merged. It was developed in Sandia National Laboratory.

Kumar’s Classification and Detection of Computer Intrusions (Kumar 1995)

The first of the Purdue taxonomies, it classifies computer intrusions on UNIX systems using the system logs and colored Petri nets.

The classes in his classification of intrusion attacks are:

- Existence
- Sequence
- Partial order
- Duration

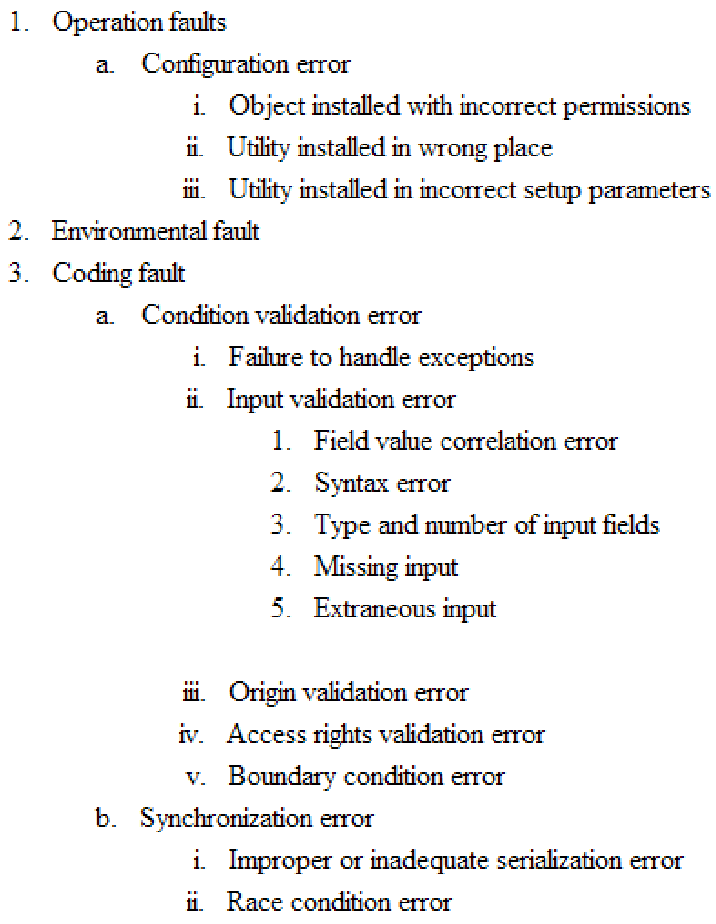
- Interval

4.5.7. Aslam’s UNIX Security Taxonomy

The second taxonomy to come from Purdue, Aslam (Aslam 1995)

builds a taxonomy of UNIX security flaws in his Master’s thesis. He also includes examples of different fault types.

His categories are:



1. Operation faults
    a. Configuration error
        i. Object installed with incorrect permissions
        ii. Utility installed in wrong place
        iii. Utility installed in incorrect setup parameters
2. Environmental fault
3. Coding fault
    a. Condition validation error
        i. Failure to handle exceptions
        ii. Input validation error
            1. Field value correlation error
            2. Syntax error
            3. Type and number of input fields
            4. Missing input
            5. Extraneous input

        iii. Origin validation error
        iv. Access rights validation error
        v. Boundary condition error
    b. Synchronization error
        i. Improper or inadequate serialization error
        ii. Race condition error

4.5.8. Krusul’s Taxonomy

The last of the Purdue taxonomies, Ivan Krusul’s Ph.D dissertation extends Aslam’s taxonomy and database. He created a new taxonomy that is detailed and complicated. His four basic categories are:

- Design
- Environmental Assumptions
- Coding faults
- Configuration errors

Objective:

“This dissertation presents a classification of software vulnerabilities that focuses on the assumptions that programmers make regarding the environment in which their application will be executed and that frequently do not hold during the execution of the program.” (Krusul 1998)

4.5.9. Bishop’s Vulnerabilities Classification Scheme

Bishop’s taxonomy (Bishop 1996) is a six-axis taxonomy for software vulnerability. Each of the axes contains a vulnerability classification. The axes are:

- The nature of the flaw
- The time of introduction
- The exploitation domain of the vulnerability
- The effect domain
- The minimum number of components needed to exploit the vulnerability

- The source of the identification of the vulnerability.

4.5.10. Richardson's Extension to Krusul's Taxonomy

Richardson at Iowa State University extended the Purdue taxonomies and developed his own taxonomy that is specific to Denial of Service (DoS) attacks. His taxonomy consists of three categorizations and has cataloged 630 attacks. He adds three more categories to Aslam's (Aslam 1995) taxonomy, specification weakness, implementation weakness and Brute force attacks. His second category is an extension of Krusul's taxonomy (Krusul 1998) and his third category is a collection of six mechanisms with the sixth one consisting of four sub categories.

Objective:

The purpose of this two-year study was to "study and understand network denial of service attacks so that methods may be developed to detect and prevent them." (Richardson 2001)

4.5.11. Linde's Generic System Functional Flaws

This taxonomy lists six classes to study for penetration results in the Flaw Hypothesis Model. He also lists the generic system flaws that were used in that penetration testing. His appendix includes a list of operating system attacks. (Linde 1975)

4.5.12. RISOS Operating System Security Flaws

Conducted in the 1970s to study computer security and privacy, RISOS consisted of seven generic fault categories. Its classification was generalized for the operating systems that were studied. It produced a classification of integrity flaws found in operating systems.

The seven fault categories were:

- Incomplete parameter validation
- Inconsistent parameter validation
- Implicit sharing of privileged/confidential data
- Asynchronous validation/Inadequate
- Inadequate identification/authentication/
- Violable prohibition/limit
- Exploitable logic error

4.5.13. Protection Analysis (PA)

This study was conducted in the 1970s at University of Southern California to study protection errors in operating systems (PA, 1978). It tried to establish a way of searching for security faults by formulating pattern matching techniques. If a sequence of code matched an error pattern, an error was discovered. The final draft of the project proposed four categories of faults based on syntactic structure.

4.5.14. Landwehr’s Taxonomy

Landwehr’s categorization was based on the genesis, location in the system software and time of introduction.

Their survey provides “a taxonomy for computer program security flaws, with an appendix that documents 50 actual security flaws. These flaws have all been described previously in the open literature, but in widely separated places. For those new to the field of computer security, they provide a good introduction to the characteristics of security flaws and how they can arise.” (Landwehr 1994)

My thesis also provides more than 200 actual e-commerce bugs and they too provide a good introduction to those who are new the characteristics of e-commerce bugs and how they can arise.

4.5.15. Attanasio's Flaw Hypothesis Methodology (FHM)

Attansio's Flaw Hypothesis Methodology (FHM) was more of a list than a taxonomy. But it lists areas of computers that may be attacked. The list lists sections on which they applied FHM. The list contains 16 characteristics.

Objective:

Attanasio discusses "a methodology for discovering operating system design flaws as an approach to learning system design techniques that may make possible greater data security." (Attanasio 1976)

4.5.16. Brinkley's and Schell's Computer Misuse Techniques

Their essay provides an overview of the vulnerabilities and threats to information security in computer systems. (Brinkley 1995)

It lists four areas of computer misuse.

- Theft of computational resources
- Disruption of computational resources
- Unauthorized disclosure of information
- Unauthorized modification of information in a computer

The last two categories are then further expanded into six classes

- Human error

- User abuse of authority
- Direct probing
- Probing with malicious software
- Direct Penetration
- Subversion of security mechanism

The historical survey of past failures provided in their essay is similar to the approach taken in this thesis to provide examples of past bugs.

4.5.17. Knight's Vulnerability Taxonomy

He defines five parts for vulnerability

- Fault
- Severity
- Authentication
- Tactic
- Consequence

He also classifies severity into six levels

- Administrator access
- Read restricted
- Regular user access
- Spoofing
- Non-detectability
- DoS

He also divides Tactic into five categories. Another salient feature of this taxonomy is that it maps vulnerability on the basis of time required to exploit from instantaneous to years.

Objective:

The author states that his work “will be used to define the forensic sciences stemming from computer crime, providing answers to the reasoning that hackers would use in a break-in.” He continues to say, “…by following the approaches given in this book, an investigator can mirror the tracks of a hacker’s logic as they intrude upon a computer network and understand the reasoning that goes on behind the attack.” (Knight 2000)

4.5.18. Beizer’s Bug Taxonomy

Beizer lists his taxonomy in his book (Beizer 1990). He divides his list into three types of bugs

- Bugs in ‘Design’ phase
- Bugs in ‘Implementation/Coding’ phase
- Bugs in ‘Maintenance phase’

He uses a 4-digit number to represent a bug and demarcate the levels.

Beizer provides his taxonomy in the book “Software Testing Techniques” which makes his taxonomy important in the context of this thesis, as it is another taxonomy created for testing purposes.

4.5.19. SRI Security Breaching Incidents

Published by SRI (Stanford Research Institute) in 1976, it lists 355 security breaches. They are divided into seven violation categories

- Intentional violations- Internal
- Intentional Violations- Computer department
- Intentional Violations- external
- Aura of computer
- Disaster
- Accidents
- Miscellaneous

The first three categories have seven sub-categories

- The major sub-categories are:
- Physical access
- System access
- Data manipulation-external
- Data manipulation- internal
- Misapplication of services
- Unprotected activities
- Physical theft

Almost all categories, except system access, are further divided into the same minor sub-categories. A total of seventy-one total violation categories were created.

4.5.20. Perry and Wallich’s Attack Matrix

They classified computer crimes into four categories

- Physical destruction
- Information destruction
- Data diddling
- Unauthorized use of computer services

They introduced a matrix with six types of computer crimes, six types of people using it, what damage could be done and what needs to be done to stop it. (Perry 1984)

4.5.21. Parker’s Taxonomies

He extended the three basic categories of confidentiality, integrity, and availability by adding two more categories authenticity and utility.

Parker lists eight primary functional vulnerabilities in computer systems and he lists 17 categories of computer abuse methods in his work (Parker 1989)

He also identified four main categories of computer crime in the chapter of his book Computer Security Reference Book (Parker 1992)

4.5.22. Straub and Widom’s Motivation-Security Response Taxonomy

Their taxonomy is not a taxonomy of attacks but a taxonomy of the types of attackers. It lists the motivation of the attacker. (Straub 1984)

4.5.23. Ristenbatt's Methodology for Network Communication Vulnerability

They define it as an expansion on the Data Link Vulnerability Analysis (DVAL). (Ristenbatt)

The four classes under it are

- Susceptibilities
- Interceptability
- Accessibility
- Feasibility

4.5.24. Du and Mathur's Taxonomy

Another taxonomy from Purdue, they provided a categorization of software errors that led to Security Breaches. Their scheme classifies errors by their cause, the nature of their impact, and the type of change, or fix, made to remove the error. (Du and Mathur 1997)

4.5.25. Knuth: Errors of TeX (Knuth)

TeX is typesetting software developed by Knuth. He cataloged the errors found during the development process and provided a classification system to hold them. Knuth's fault categories were:

- An algorithm gone awry
- A blunder or a mental typo
- A clean-up for consistency or clarity

- A data structure debacle
- An efficiency enhancement
- A forgotten function
- A generalization or growth of ability
- An interactive improvement
- A language liability
- A mismatch between modules
- A promotion of portability
- A quest for quality
- Reinforcement of robustness
- A surprising scenario
- Trivial typo

4.5.26. Brian Marick's Survey

He published a survey of software fault studies from software engineering literature. He reported faults that were found in production quality software. (Marick 1990)

4.5.27. Chillarege's Orthogonal Defect Classification

This classification method suggests using in-process measurement, i.e., the defect distribution, which can be used to measure the development process and highlight that part of the process that needs attention. (Chillarege, 1989)

4.5.28. Cem Kaner's Common Software Errors Appendix

Cem Kaner, in his book "*Testing Computer Software*" *2nd edition*, (Kaner, *et al.*) provides an outline of "common software errors." He lists about 400 bugs and classifies them into about 12 categories

- User interface errors
- Error handling
- Boundary-related errors
- Calculation errors
- Initial and later states
- Control flow errors
- Error in handling or interpreting data
- Race conditions
- Load conditions
- Hardware
- Source, version and ID control
- Testing errors

4.6 Classification of E-commerce Risks, Bugs and Failures: An E-commerce Taxonomy

This classification with about 44 top-level categories spans broadly to hold e-commerce failures and also serve as an outline to generate a list of potential failure modes (which may be considered to be a list of test ideas that can be used to generate test cases later). This classification scheme and its application to e-commerce functions such as a shopping cart, search engine or user authentication forms the crux of this thesis. The classification system will be discussed in detail in the next chapter.

Chapter 5: The E-commerce Bug Taxonomy

This taxonomy is the result of a year of research in collecting and classifying bugs in e-commerce sites and e-commerce software. The taxonomy, in its final form, has 44 risk categories and holds about 700+ potential failure modes. The taxonomy also points to about 300+ real life bug examples that have been publicized in the news media.

5.1 The Structure of the Taxonomy

The 44 risk categories make the main body of the taxonomy. Some of these categories have further sub-categories. The sub-categories were included to provide a more specific starting point to generate more focused test ideas and in some cases provided a broader picture of idea coverage. Each of these categories or sub-categories holds the collection of potential “risks” or “failure modes” specific to that category.

Within a given category, there is a list of:

- Potential failures.
- Information about causes of the failures, if we have that information.
- References.
- Examples of actual failures within the category’s theme that were reported in the trade press.

5.2 How to Use this Taxonomy

Generate new test ideas

The outline provides about 44 top-level categories with examples of errors under each one of them. The categorized outline inspires test idea generation if the tester considers the function-under-test. It then considers how the function would fail with respect to one of the categories. The tester, who has run out of good test ideas, can look for plausible failure modes in the risk list. The tester then creates tests looking for those types of failures.

Use the test ideas here, for applicable projects

The thesis fills in details for more categories of shopping cart problems and less extensively considers failure modes for some other functions. I have used the shopping cart as the example function because it has rich functionality (and thus many different failure modes). I have used an example function to make the application of the categories more concrete. The same types of problems will show up in many other types of e-commerce application functions.

This taxonomy is not exhaustive. I think the 44 top-level categories are sufficient, but below that level, you'll be able to add plenty of your own examples if you use the outline. I recommend that you take this thesis and customize it as you use it on different projects. The more tailored it is to your applications, the more long-term value it will have for you.

The tester who uses the taxonomy can sample from the list, selecting a potential problem for analysis.

The tester’s question should be whether the software under test could have a bug analogous to the one from the list.

If so, the next question is what type of test would expose this type of bug. Also, a tester unfamiliar with an aspect of the program can look for potential failure modes in the list, and then explore the program looking for those types of failures.

Audit test plans by inspecting for tests for potential errors

It’s difficult to find the blind spots in a long test plan. There is so much detail available that it’s hard to see what’s missing. A list of potential failure modes provides you with an independent cross-check. To audit the plan, consider each category in turn. Ask yourself whether failures within that category are possible in the software under test. If so, what they might look like. Pick two or three possible failures and then check the test plan. Does it have tests that would catch these problems? If not, you’ve found a hole in the plan.

As teaching material to assist in the training of testers who are new to e-commerce testing

The clear structure of the outline, the detailed list of possible risks and failures, and the concrete examples can help new testers broaden their understanding of the range of problems open to their discovery. You can also use them as anchoring points for discussions. For example, some test groups set up a weekly lunch meeting to improve their knowledge or skills. At one of these lunches, you might talk about how to recognize accessibility errors. Another day, you might talk about how to recognize update-installation side effects. This outline can help you focus those discussions. A Test manager, training new testers, can walk the group through selected examples from the risk list in order to convey to the trainees the breadth of their work.

As a presentation tool for explaining to managers the different types of failures that can occur in an e-commerce site

Managers don’t necessarily understand the breadth of the scope of risks involved in e-commerce projects. Use this outline to develop a list of the types of issues that can be tested for. Cross-reference to published examples of errors that have shown up in the trade press (use mine or find your own). A well-organized presentation based on demonstrable risks can go a long way toward getting you adequate funding.

5.3 Discussion: Conformance

Conformance of the e-commerce taxonomy to the “Definition and Requirements of a Taxonomy”

As seen in the previous chapter the word “taxonomy” has been used in so many different contexts, that it is hard to pinpoint exactly one definition or define the requirements of what constitutes and qualifies to be called as a taxonomy.

Nonetheless, Lough in his dissertation provides a list of properties, which he has compiled from the lists provided by Howard, Lindqvist, Amoroso, Krusul and Bishop. His combined list contains about 18 properties that a taxonomy should possess, including accepted, appropriateness, comprehensible, completeness etc to mutually exclusive, unambiguous and useful.

The e-commerce taxonomy is appropriate, comprehensible, specific, and probably useful for the purpose it was created. It will possibly be accepted as a good beginning by the testing community and with possible customizations for individual needs, it may end up meeting its objectives.

The e-commerce taxonomy is far from being exhaustive or complete despite its mammoth span of categories and large number of failure modes. The e-commerce world and its systems are too diverse to have a single universal taxonomy that generalizes all potential failures across its different product lines, OS families and environments. The failure modes in this taxonomy only represent a sample of possible failures that the system can face. The tester is then further encouraged to conjure up other possible failures which are similar or different from the ones in the list and develop tests for them.

Taxonomists generally state that in a proper taxonomy, each category must be mutually exclusive to each other category. That is, the categories must not overlap. In the e-commerce world, it is common to find that multiple causes or symptoms for a bug may cause the bug to be overlapped across multiple risk categories. Hence the e-commerce taxonomy will probably never exist as a mutually exclusive taxonomy. During the early design days of the taxonomy, some people had pointed out that the inclusion of an “Others” category will render the taxonomy complete. Every bug will have a category in which to reside and if it does not, it will fit into the “others” category. Lough quotes Dr. Carl Landwehr, in suggesting the same idea but continues to question the usefulness and validity of the “other” category and calls its use debatable.

A taxonomy should always be expandable when a new category of risks is identified. An ever-evolving taxonomy tends to be more useful and current. As for the purpose of testing, it is very difficult to build a perfect taxonomy in the first attempt and available test time. But, a simple and broad categorization that is able to raise specific questions in the minds of the testers is sufficient for the purpose of using a taxonomy as an aid for test idea generation.

In summary a good taxonomy for testing purposes has enough detail for a motivated, intelligent newcomer to the area to be able to understand it, and is broad enough to raise at least a few issues new to someone with moderate experience in the area. A good taxonomy is a useful tool for informing a tester who is new to the area about the types of problems to be tested for.

5.4 How the Taxonomy/List Was Developed

I brainstormed a first draft top-level list (I thank James Bach, Cem Kaner for their assistance), and based on that I created a basic outline with about 10 categories, which formed the basis for further work. The categories partially passed the mutual-exclusivity test but were not broad enough. They did not completely cater to the needs of a tester because the emphasis was more on providing a framework to generate test ideas. I would assume here, that a tester will be able to generate better test ideas when the categories are specific and increase the number of test ideas when there are more categories. The next step was the generalization step of the reports of individual bugs I had collected. Some took shape as causal categories and others as symptomatic. But basically, just as my advisor Dr. Cem Kaner had summarized the goals in one of his emails, "The goal is to find a nugget that will arise under a wide enough range of circumstances that it is worth including in the list."

Searched electronic bug databases (such as bugnet.com and cnet.com) for examples.

Bug databases (Bugtraq, Bugnet, etc), security advisories (CERT, MITRE, Securityfocus, etc), bug columns in magazines, newsgroups are good sources for actual bugs.

Also Internet magazines such as eWeek, Cnet.com, Zdnet.com publish major e-commerce site outages, glitches and sometimes the cause for the outage as perceived by the industry experts or as released by the concerned e-commerce site.

Searched open source software for bug databases for specific products. These gave us examples and indications of the types of bugs possible.

Open source software sites are useful from the point of view that they make their bug databases public. It will be hard to find the complete list of the types of bugs that plague e-commerce databases/servers from commercial software sites such as Oracle or Sun. However, the bug database from a popular open source software site such as apache.org, which is available in the public domain, is useful.

Brainstormed additional types of problems.

ISO-9126 categories were adapted with some modifications for the qualitative categories of the taxonomy. The final draft had about 60 categories. The process of changing, adding, removing and merging categories continued, however, until the present version of the taxonomy was created. The failure modes for the categories were inspired by ideas from troubleshooting guides, user-groups, talking to people who have tested these applications before, real-life bugs, vendor white papers, and risk forums.

Circulated the list and the outline for peer review.

The preliminary taxonomy and sample failure modes were circulated among peers, test managers, and users and any bug/issue/problem suggested by them went into the list.

Chapter 6: E-Commerce Testing and Shopping Carts

In the first chapter I discussed the nature of e-commerce testing and how this thesis helps in improving this type of testing. But, the learning curve involved with the transition is high. This chapter tries to list some of those factors involved with the transitioning to e-commerce testing from other platforms. It then moves on to a discussion of e-commerce ‘shopping carts,’ which have been used in coming chapters, such as in the example “function-under-test” to generate failure modes and test ideas.

6.1 The Problem of Transitioning to E-commerce Testing from Other Platforms

E-commerce testing involves numerous challenges, such as new or unfamiliar technologies, technology-driven business processes or logic that the tester doesn’t understand how to test. Learning the necessary information or skills may be complex but can be simplified with strategies for quickly generating test ideas or test plans that address potential risks. I believe that an outline of well-researched potential failures can help flatten the steep learning curve involved with e-commerce testing.

It may be worthwhile for any tester who is moving into the e-commerce-testing arena to remember the following issues:

- **“Test Global and Test Distributed”:** E-commerce systems are truly global in spirit and structure. The different underlying systems may be on different continents, but they appear to integrate seamlessly over large, distributed and non-homogenous business networks and other communication channels. Risk analysis and test planning should allow for potential problems caused, for example, by a side effect of a software upgrade on a transaction server physically located in Holland on a transaction taking place between a customer in Korea and a retailer in the US.

- **3 Ms where 1st M: Multiple Platform 2nd M: Multiple Clients (Browsers) 3rd M: Multiple customer profiles:** This will be a considerable change for a tester who comes to e-commerce testing from other traditional testing fields. In the stand-alone or older client servers, the user’s platform, the client type and the nature of the user might be well known to the developers and testers. E-commerce systems involve more uncharted territory, with greater diversity of operating systems, browsers and other system software and hardware.

- **The user profile varies greatly in terms of age, gender, taste and usage**: Testing business software has become as configurationally complex as testing consumer software, but with more serious consequences in the event of failure.

- **Learning to count in “web years”:** Changes and updates are inherent in E-commerce sites. Content and target platforms change quickly, without much time for planning and regression testing of each change. This can be challenging for testers coming from traditional business applications. The e-commerce tester must learn to generate effective sets of test ideas rapidly.

- The risk of testing in a “not-so-representative test environment”

- In an e-commerce world, creating a completely representative test environment is often impossible. With a limit on how much can be actually simulated in a test lab, there is real risk of not knowing how the application will behave in some environments in the field.

6.2 The Sample ‘Function’: An E-commerce Shopping Cart

Overview

The familiar metaphor of a shopping cart that is present in many e-commerce sites has an important function of keeping track of the user’s state while he/she is “shopping”. A simple cart may just maintain a list of items that the user places in it and maintains state until the user finishes shopping and exits the system (by closing the browser). Complex and advanced carts have more sophisticated functionality such as real-time credit card processing and real-time order tracking.

As e-commerce sites grow in size and popularity, they tend to add more and more features to their shopping carts. Shopping carts have grown from simple state tracking functions to a highly sophisticated and creative piece of software offering a flexible range of user options.

The diversity, the creative imagination and technological innovation that have gone into the design of these next-generation-shopping carts, make them fascinating and challenging to test. Some issues that you may want to consider before testing a shopping cart are:

- The same issue of testing under "web years" discussed about general e-commerce testing applies here. Hence testers will have to reckon with testing under reduced time and sudden spikes in workload.
- A shopping cart may not scale. It may work well for 100 users but not for 1000. Testers should estimate performance standards early in testing.
- They are prolific and vary greatly in terms of design, size, complexity, and underlying technology. Hence no standard best practices exist that can provide a single-point reference on how to efficiently and comprehensively test a shopping cart.
- Because of the rapid evolution and change in their design and features, it is common for testers to encounter legacy-shopping carts built with outdated technology.
- In contrast, the rapid changes in design and functionality might be a challenge to the less tech-savvy tester, as they add more learning pressure on the tester.

I thank Karen Johnson for sharing her experiences on testing a shopping cart and for her inputs on some of the issues/bugs mentioned here.

6.3 Different Types of Shopping Carts

Testing each of the shopping carts can be very different because of the difference in the way they have been built and hosted. Except for some in-house shopping carts, shopping carts tend to have a large number of 3rd party components, which are sometimes beyond the scope of the testing group.

- **3rd party-built and -hosted shopping carts** are generally a cheaper alternative. They are employed by small-scale e-commerce sites with fewer staff to design and maintain the system. Sometimes the 3rd party host may be the same as your site-hosting provider. But 3rd party hosting wrenches much of the control away from the tester since the bulk of the components are not in the tester's domain. But the positive side of this type of shopping cart is that the cart is smaller, less complex and generally easier to test.

- **Out-of-the-box shopping carts** are customizable, pre-fabricated, and ready for deployment. Installation, customization, and configuration bugs are some of the common types of issues that testers will encounter when testing shopping carts of this genre. Though many serious security holes have been caught due to bad settings and bad configurations in out-of-the-box carts, much of the deeper level of testing becomes the responsibility of the vendor who originally developed the cart software.
- And finally there are many **free shopping cart scripts** available for download in **CGI or ASP script** sites. Very simple and small-scale e-commerce sites tend to use these carts. From the tester's perspective, the risk is in the script code. While some of these scripts are well done, amateurs hastily develop others.

Chapter 7: List I: Component Failures

7.1 Database Server Failure

A database server is software that manages data in a database. It updates, deletes, adds changes, and protects data. Database servers provide both the access control and concurrency control. So while testing a shopping cart, if you find empty catalogs, unpopulated data fields and authentication problems, then you should check the database server. Some of the issues discussed here are based upon the discussion in the paper "Managing Database Server Performance within an Electronic Commerce Framework" (Martin, 1999). Listed below are some ways a shopping cart can fail when the database server goes wrong:

Failure Modes

- Unable to load or populate data in the product catalog.
- Unable to load or populate order data in the shopping cart.
- Unable to load or populate customer profiles.
- DB server failure may lead to a complete failure of data retrieval in the system since DB server manages/serves the data in the system.
- Increase in response time during "browse" transaction. Browse transaction generates high frequency, random, sequence of queries on the database server.
- The "shopping cart" transaction fails to update/load the billing details/price in the basket. Shopping cart transaction places medium weight, high frequency read/write operation.

- Increase in response time to load/update billing details, price lists and total in the basket.
- Failure or delay to commit the customer order to the database in the "Buy transaction".
- User-registration failure, unable to execute read-write process during user registration.
- Search process fails to execute since DB server failure may cause failure of read-only search processes to fail.
- Increase in "search" time may indicate performance problems in Database server.

7.2 Cache Server Failure

Cache servers are used as intermediaries for web requests and retain previously requested copies of resources. The use of a cache server is to handle common requests locally and improve site performance by better speed up and reduced overhead on the web servers.

The common issues discussed here can be found in more detail in “Known HTTP Proxy/Caching Problems” (Dilley)

Failure Modes

- Cache may return an outdated shopping cart document if the header is misrepresented or last modified date is omitted.
- If shopping cart content is dynamic in nature, then cache server will not be able to serve new content.

- If caching proxy server fails during a shopping session, sometimes the browser fails to bypass the server and may need to be reconfigured and shopping cart state may be lost.
- If shopping cart uses any form of encoded response, the proxy might cache it and send it to a non-encoding capable client.
- Sensitive shopping cart content may get cached by interception proxies that break client cache directives like "No cache" or "Must revalidate".
- The cache server may end up blocking some methods used by the shopping cart software. Because the method contained in the request is unknown to the proxy so instead it generates the default HTTP 501 Error as a response.
- Shopping carts that use IP address to track state of the cart, may fail because interception proxies at ISP level may alter client's IP to that of the proxy itself.
- A caching proxy mesh might break HTTP content serialization resulting in the user getting older content when the shopping cart page loads.

Examples of Related Bugs and Issues

Is Web caching bad for the Internet?
http://www.cnn.com/2000/TECH/computing/04/18/web.cache.idg/

Known HTTP proxy/caching problems
http://www.wrec.org/Drafts/draft-ietf-wrec-known-prob-03.txt

7.3 DATABASES

The approach used here to classify different databases can be found in more detail in “Oracle9i Database Administration: Recover Databases ” (Oracle, ch.32), and as per the guide database failures are classified as:

- Database statement-failure:
- Database-Instance Failure:
- Database-User-Process Failure:
- Database-Media Failure:

7.4 Database Statement-Failure

Definition: “Statement failure occurs when there is a logical failure in the handling of a statement”. (Oracle, ch.32)

Failure Modes

- User may be attempting to issue a statement referencing a table in the shopping cart that does not exist.
- A user may be attempting to issue a statement referencing a table in the product catalog, user database that they have do not have permission to access.
- Flawed statement or flawed query used by the web developer may make shopping cart data inaccessible to the user.
- Inability of a user to submit information that is to be stored into a database because of inadequate table space allocation for the user/operation.

- Flawed statement/query may lead to in-correct addition/deletion of items in the basket.
- Incorrect access of tables may lead to incorrect computations/calculations of shipping/taxes.
- Failure to clearly specify required fields, optional fields and edit permissions may lead to problems when data is being written back into the tables.
- Inefficient queries on the shopping cart tables.

7.5 Database-Instance Failure

Definition: “Instance failure occurs when a problem prevents a database instance from continuing to run. An instance failure can result from a hardware problem, such as a power outage, or a software problem, such as an operating system crash. Instance failure also results when you issue a SHUTDOWN ABORT or STARTUP FORCE statement.” (Oracle, ch.32)

Failure Modes

- The number of simultaneous connections allowed is less that the maximum number required by the system for shopping cart transactions.
- Power outage when shopping cart database is being accessed and no recovery routines exist.
- Check for issues where using a product database and multiple item forms together would cause an error.

Examples of Related Bugs and Issues

Database glitches at Walmart.com

http://www.internetnews.com/ec-news/article.php/4_739221

7.6 Database-User-Process Failure

Definition: "A process failure is a failure in a user, server, or background process of a database instance such as an abnormal disconnect or process termination" (Oracle, ch.32)

Failure Modes

- Risk of user being unable to return to shopping cart after navigating away from the page since contents of cart not been saved.
- User is unable to add/delete/modify contents of the basket.
- Client PC hangs during shopping cart transaction and user state not saved/retrievable.
- Failure of the shopping cart database to rollback process on detection of user process failure.

7.7 Database-Media Failure

Definition: "An error can occur when trying to write or read a file on disk that is required to operate a database. This occurrence is called media failure because there is a physical problem reading or writing to files on the storage medium." (Oracle, ch.32)

Failure Modes

- Insufficient system memory for the shopping cart database
- Disk failures/hard drive crashes, and other irreversible media corruption of the shopping cart database may cause complete loss of data.
- Corruption of shopping cart database backup.

7.8 Error Messages/ Exception Handling

Provided below is a detailed list of errors that you might encounter in an e-commerce site with a shopping cart. It might be useful to test for appropriate error messages. Testers should find this list useful to test a shopping cart site for error handling and check if the error handler handles these common errors. It has also been sub-categorized for ease of use on the basis of the kind of errors the system has been designed to handle.

Failure Modes

7.8.1 Error Handling - Quantity

- Ability to erratically checkout an empty shopping cart and check if error message is displayed.
- Ability to add negative numbers to the quantity field. Check for appropriate error handling
- Accepts decimal entries for quantity but ignores the decimal point and either accepts the first or last digit alone, so 7.0 may be interpreted as 7 or 0 and no error handling exists to prompt or correct the error.

- Accepts decimal entries for quantity but again ignores decimal point and accepts the quantity comprising of both the digits, so 7.0 may be interpreted as 70. And no error message to prompt or correct the error.
- Quantity field not size-constrained and no error message to prompt user of acceptable values or data range.
- An over-sensitive error handler may not let a user increase/decrease/edit the quantity field at an editable stage and may risk rendering the data entry final.

7.8.2 Error Handling - Registration Forms

- Forms requiring registration information, shipping address information, billing address information employ script based entry validation to validate entries but sometimes the scope of the script exceeds its limit and generates an error message for entry fields outside the limit of the script or optional fields.
- Some address fields contain two parts, address 1 and address 2 in order to accommodate lengthy addresses. But some error handlers count both fields as compulsory and generate error messages to users who leave address 2 empty (because their address is short and fits right into the first one!)
- Long addresses may get truncated and no error message or routine exists to warn the user about the size constraint.
- Lack of error routine to check for a valid US zip code in the address section.
- Check for trigger-happy error messages that sometimes pop up to a non-US shopper’s dismay, to validate an empty US zip code.
- Error message pops up informing the user of incomplete information entry but does not point to the field where error exists.

7.8.3 Error Handling - Interaction and Transaction

- **“An Internal server error”** may be displayed without any fix to the user, sometimes this error, which may be due to a missing term in the URL, can be fixed by appending a term, like “&reference” to the address.
- **“Inventory module error message”** may be displayed with no explanation to the user; sometimes this error occurs when two users access the last item and the inventory control tries to update the order so that only one user gets access to the item.
- If you encounter an **“ODBC error message”** when you click checkout, you may be missing your "session ID". An error handler should be enabled to handle this common error or should provide help to customers with a simple fix to these errors.
- **“Timeout error messages”** If any routine exists to check the time of inactivity and auto times out any shopping cart, the existence of timeout routines should be communicated to the user beforehand.

7.8.4 Error Handling - Payment/Credit-Card

- An incorrect expiration date (be sure to use a two-digit year, such as "02") and supporting error message to prompt the user.
- **“Invalid card number error message”** if the card processing is a real time event in the cart, then user may be prompted to enter the number again or try a different card
- Inconsistency between the address in the billing section and the address in the card. Check for a user-understandable error message.

- Browser version too old to support card processing/secure protocols, user must be pointed to the browser issue and left in a limbo with a clueless message pointing to card error instead of browser incompatibility issue.
- **Invalid ABA (**American Banking Association) **number error message**: If shopper is paying by check, he/she must supply valid checking account and "ABA" numbers.
- Site does not support the card used by the customer; provide a message beforehand about the type of cards the site supports.
- **“Temporary network error messages”:** A temporary network problem may cause a data transmission error between the credit card processor and your bank.
- Check if alternative error handling exists when third party billing agents fail.

7.8.5 General error messages

- Unable to understand error message; cryptic & undecipherable error messages, especially in secure areas of the shopping cart, may make users abandon their cart in panic.
- A common mistake in a shopping cart error-handling system is displaying machine errors or compilation errors to the user instead of understandable error messages, which are consistent with the language of the site.
- Persuasive VBScript or JavaScript error message boxes that pop up on an erratic entry but don’t close on clicking “OK!”
- Error-handling routine re-directs you to another page for explaining the error but provides no way to return back to the original state of the shopping cart.

- Loads a pop-up error message box, but a 404 “page not found” error displays in the error pop-up!
- Over-enthusiastic exception handling: generates error message even after the error has been corrected or error message pops up for correct entries due to failed script-based validation routine.
- Error message box or an action to close the error box causes illegal operation or illegal memory reference in the browser software. This causes the browser to close in the middle of a transaction.
- Error boxes written in scripts not supported or incompatible with browser type.
- Typos, grammatical errors in error messages that change the meaning of the intended error message
- Illegible error message: A combination of the color scheme of the message box and the font size and color may cause the legibility of the error message to degrade.
- Security problems caused by bad error handlers: Sometimes error messages pose serious security risks by exposing sensitive data like port numbers, line number of internal code, type of server and internal configuration of systems. Mixing machine-communicated errors and error-handling system may simplify the process of writing error messages but the risk of a security lapse runs high when such error messaging systems unintentionally channels out internal and sensitive data.

Examples of related bugs and other issues

Problem with "Hotwire.com": lack of a useful error message

http://www.phototour.minneapolis.mn.us/essays/hotwire.html

7.9 Human Error

Though human judgment and perception is far superior to that of any machine, the human tendency to err is always a risk. All shopping cart-centric e-commerce systems involve some human action and intervention in the form of data entry, data upgrade, system upgrade, and system design. The common human errors in the shopping cart are incorrect price entry and erroneous handling of back end processes. Below are some common risks that exist due to human errors:

Failure Modes

7.9.1 Human Error on the Retailer Side

- Risk of price glitches: incorrect price entry, incorrect data feed, incorrect database configuration and all other forms of incorrect human data entry
- Quantity glitches, incorrect entry of numeric inputs, input in wrong format
- System time incorrectly set, all time stamps on order placements out of sync
- Administrator forgot to restart the web server or shut it down by mistake
- Back-end human error: wrong item sent, or package inter-changed
- Shopping cart configured incorrectly
- Administrator erased custom settings by mistake
- System reset to default by mistake
- Security breaches and system security compromises due to deliberate or non-deliberate human action
- Forgot to backup the files

- Corrupted the configuration file by mistake
- Erased data or deleted files by mistake
- Physical failures introduced to the shopping cart system and its underlying hardware, due to bad handling, accidental damage caused by human action
- Human error in entering the correct e-mail address when sending confirmation of order placement (in non-automated systems)
- Typos, grammatical mistakes, and incorrect language structure usage in content pages
- Any large-scale human disaster or man-made disaster that causes physical damage to underlying e-commerce system.
- Transaction aborted due to non-intervention of required personnel

7.9.2 Human Error on the Customer Side

- Incorrect selections, incorrect navigation, incorrect understanding of the shopping process
- Adding the wrong quantity, filling up information in the wrong fields, filling up incorrect information, specifying wrong shipping address prevent the e-commerce system from delivering the items purchased through the shopping cart.
- Entering the wrong data type, entering in the wrong format (eg., Date), selecting the wrong shipping options
- Deliberate or non-deliberate abortion of the transaction process.
- Loss of shopping cart state and subsequent abandonment of shopping cart due to erroneously closing the browser.

- Entering wrong credit card number or selecting wrong credit card type or entering the expiration date in the wrong format or order.
- In-correct usage of the shopping cart functionality, like pressing the confirmation button multiple times or clicking on selection buttons multiple times, causing errors in the order placement.
- Trying to access the shopping cart in an incompatible underlying environment or using an older incompatible version of the browser, or having scripts and cookies disabled.
- Do not have/has not installed the required plug-ins or media software required to view the shopping cart catalog.
- Wrong shipping methods requested for Alaska, Hawaii, Puerto Rico, and international addresses. Only UPS Second Air, FedEx 2Day, and USPS Priority Mail deliver to these addresses.
- The billing and shipping addresses are reversed
- Wrong e-mail address entered

Examples of Related Bugs and Other Known Issues

Ashford.com flaw allows "free" purchases
http://news.com.com/2100-1017-233806.html?legacy=cnet

IBM customers buy $1 laptops in site snafu
http://news.com.com/2100-1017-235771.html?legacy=cnet

Pricing mistake prompts Buy.com rush
http://news.com.com/2100-1017-221397.html?legacy=cnet

AOL nightmare: ordered a digital camera from AOL, received McAfee Office 2000 instead!!
http://www.zdnet.com/anchordesk/talkback/talkback_229502.html

United to honor dirt-cheap online ticket fares
http://www.itworld.com/Tech/2409/CWSTO57853/

Amazon.com hit with pricing glitch
http://www.computerworld.com/industrytopics/retail/story/0,10801,47949,00.html

Attache cases go for a penny a piece after pricing glitch at Staples.com
http://www.computerworld.com/industrytopics/retail/story/0,10801,57891,00.html

Customer outrage prompts Amazon to change their price testing policy
http://www.computerworld.com/industrytopics/retail/story/0,10801,50153,00.html

Amazon charging different prices on some DVDs
http://www.computerworld.com/managementtopics/ebusiness/story/0,10801,49569,00.html

The price isn't right: A keying mistake set the price at $26.89 instead of $299.
http://thestandard.net/article/0,1902,24690,00.html

Amazon glitch spurs shopping spree

http://www.usaicorp.com/cc/clips/2000/080200/amazonglitch.htm

Coding glitches main culprit in e-tail fire sales

http://news.com.com/2100-1017-244280.html?legacy=cnet

Price goofs in e-commerce

http://www.augustachronicle.com/stories/053101/fea_124-3979.shtml

7.10 Risks Due to Calculation Errors

A shopping cart has various calculations and computations like discount calculations, billing calculations, shipping and handling calculations and tax calculations. Summarized below are some common risks due to calculation and computation errors that cause shopping carts to fail.

Failure Modes

7.10.1 Discounts/Coupons and Special Offer Calculations

- Coupons in the online world are generally a set of numbers that accord a pre-determined discount. Common errors are incorrect sequences of numbers, a mistakenly swapped set of numbers (denoting a different product and different discount)
- Coupons being accepted by the system after their expiration date
- An infamous bug allowing the same customer to use the coupon multiple times causes the total price has been completely discounted.

- Coupon functions okay, but the billing system does not honor the coupon code and continues to charge the full and non-discounted price.
- Coupons with conditions are also error-prone. Sometimes the conditions that make the coupon valid have errors and make the coupon redeemable under all conditions!
- Some coupon codes that are still in the development stage and not yet been open for public view get “crawled” by search crawlers. And the public gets away with some free shopping!
- One other bug mentioned in the list below highlights how a user could not place a discount over her credit voucher. This was because the system could process either a discount or credit but could not process both together.
- Errors in “quantity available” or “in stock value” displayed in the catalog. This may be due to incorrect computation of inventory stock value. The risk is there will be a delay in shipping the order or the order may never be delivered.
- Check for all discount options. Discount by total percentage may work but discount by total weight may not work.
- Check for issues caused by duplicate items with quantity discounts.

7.10.2 Pre-checkout/Check-out Calculations

- Some shopping carts show the total only after checkout, but show incorrect and incomplete calculations when checked in the pre-checkout stage.
- Some shopping carts display only the price of the item in the pre-checkout stage but omit all other additional costs like shipping and taxation. They tend to show a large cumulative price when user is checking out.

- Hidden costs not shown in the pre-checkout stage
- Multiplication errors when multiplying prices in real numbers with integer quantities and subsequent error in displaying the total price.
- Rounding errors in checkout figures
- Currency conversion errors when more than one type of currency is accepted; subsequently, conversion rate tables may be inaccessible or outdated
- Tax calculations errors.
- Taxes are applicable only to the items, but like one of the bug examples quoted in this section, sales tax was applied to shipping costs.
- Even when an order has been placed for multiple copies of the same item, tax maybe mistakenly applied only to one copy due to the system's internal calculation code.
- When the issue of state taxes is encountered, common problems include wrong application of 'state' or 'city' factor to the tax calculations.

7.10.3 Shipping Calculations

- Some sites that interface with other sites such as UPS to calculate 'shipping costs' skip calculations on shipping when the other site is down and may not provide accurate total costs.
- Again, sites that interface with external sites for accessing their shipping tables may incorrectly compute shipping costs when large quantities of heavy items are ordered. This error occurs due to the maximum limit on the weight that is available in the cost calculator tables. For example, The UPS site allows calculation of shipping costs for packages with a maximum total

weight of 100 lbs. Any order above that weight may be miscalculated since the calculation will continue to be based on 100 lbs.

- International shipping is another error-ridden area. Most international air shipping costs keep changing and also vary with respect to destination. Generally the risk remains that the table used for calculation is outdated.
- Most carts calculate on the basis of price threshold, weight threshold, quantity threshold, line item threshold and sometimes no charge. Risk is high for error to occur due to using the wrong model for calculation. A single very heavy shipment may then cost much less if quantity is mistakenly used for the purpose of calculation.

Examples of Related Bugs and Issues

Spring forward leaves eBay behind
http://www.computerworld.com/industrytopics/retail/story/0,10801,59222,00.html

Glitches let net shoppers grab free goods
Botched coupon deals let shoppers waltz out with free or nearly free deals
http://news.com.com/2100-1017-242811.html?legacy=cnet

Macys.com says no to unauthorized coupon codes
http://ciscomp.com.com/2100-1017-251548.html?legacy=cnet

Shoppers seize unauthorized discounts at Macys.com
http://ciscomp.com.com/2100-1017-251334.html?legacy=cnet

AltaVista credits players $1 after contest error
http://news.com.com/2100-1017-242970.html?tag=mainstry

Staples.com nailed again by its own Net coupons
http://ciscomp.com.com/2100-1017-244220.html?legacy=cnet

Online shopping, glitches and gotchas
http://www.nwfusion.com/newsletters/techexec/2001/01156806.html

7.11 Risks Due to Software Upgrade Errors

Due to the dynamic nature of their content, web stores and shopping carts undergo frequent updates, upgrades and changes. But these frequent changes tend to frequently break things and cause havoc when the site opens up for business after the upgrade.

Listed below are some of the risks posed by software upgrade in shopping carts and e-commerce systems.

Failure Modes

7.11.1 Software Upgrade on the Server Side

- A common error is the failure to back the web-store before upgrade.
- Accidentally over-writing the product database file during upgrade.
- Non-removal of staging files before upgrade may lead to corruption of the shopping cart.

- Failure to update or reset correct file permissions in the shopping cart after upgrade process, this causes some pages to show “Unauthorized to view” errors when the user clicks on a catalog page.
- Many software upgrade processes look for folders with standard names. For example, CGI based shopping carts look for standard CGI directory path. Any deviations from the standards pose the risk of an incomplete install/upgrade.
- Some upgrades corrupt the shopping cart by changing the default file types to newer file types. The newer file type may not be compatible with clients that use it.
- Files upgraded successfully but did not to make changes go”live" after upgrade.
- Failure to check the OS compliance of host server before the upgrade
- Failure to verify the host server's software and hardware requirements before upgrade
- Insufficient disk space available for the shopping cart upgrade process and the upgrade stalls before completion
- Failure to update older and outdated content before an upgrade or site re-design
- Risk of mistakenly listing outdated and discontinued products by over-writing new files with older ones.
- “We ran two programs at the same time that will not run together”,
- Upgrades performed without checking inter-compatibility between existing or newer software processes within the system.

- Post upgrade “internal glitches” have prvented orders from being processed in shopping carts, they generally occur because of new but mismatched data feed installs, convoluted linking due to addition of new links within the shopping cycle, older links not removed and new links installed without targets.
- Upgrades to some parts of the system, may cause selective failures in dependent or related sections of the system. A common issue has been upgrades to client information databases, causing user authentication failures due to lockouts and denial of access to login processes.
- A fix to one bug causes another! A common problem in conventional software too.
- A “newer look” or “fresh look” after an upgrade may not always mean an error free look for the site. “Newer look” changes the GUI and functionality and this leads to newer problems both in terms of functionality, usability and technical glitches leading to blackouts.
- Another important risk is the risk of security problems that are caused by poor installation and incomplete installation that result in some security features being turned off.
- Software upgrades sometimes sets all options to ‘default’ automatically after the installation is complete. This in turn may overwrite any existing customized options This leads to change in e-commerce system behavior and settings.

7.11.2 Client Side Response to Server Side Software Upgrade

- Browser incompatible with the new upgraded server side shopping cart

Examples of Related Bugs and Issues

Amazon endures third holiday outage

http://www.ecommercetimes.com/perl/story/5870.html

Webvan stalls on the way to Thanksgiving dinner

http://news.com.com/2100-1017-248798.html?legacy=cnet

E*Trade users locked out of trading

http://news.com.com/2100-1017-221117.html?legacy=cnet

Problems hit E*Trade for third day

http://news.com.com/2100-1017-221192.html?legacy=cnet&tag=rltdnws

Software glitch affects DoubleClick's domestic clients

http://www.atnewyork.com/news/article.php/8471_441871

Walmart.com runs into glitches

http://news.com.com/2100-1017-249390.html?tag=prntfr

When Buy.com redesigned its Web site on April 26, it mistakenly listed between 4,000 and 7,000 discontinued laser discs for $1.11 apiece.

http://news.com.com/2100-1017-225527.html?tag=rn

Dangerous shop service if installed the right way

http://exploiter.virtualave.net/9904-exploits/hhp-WebShop.txt

Yahoo introduces email bug after attack
http://news.com.com/2100-1023-236686.html?legacy=cnet

7.12 Document Confidentiality

How secure is the e-commerce site? Is it safe to give my credit card number? Can someone get my order details and my personal information?

Document confidentiality means, "protecting private information from being leaked to third parties" [Stein, 2000]. Compromises on this issue lead to serious security related failures. This category deals with issues like credit card information leaks, order information leaks, account information leaks, etc.

Shopping carts with advanced features provide direct linking with credit card processing agencies using secure protocols such as SSL or SET (Secure Electronic Transaction). But we need to remember that these secure protocols are also prone to failures and attack by malicious elements and can cause sensitive data loss.

Cryptography is a key technology that is used for protecting the system against such leaks. Testers testing shopping carts that encrypt data in bill payment transactions need to know some simple ways in which cryptography fails.

Failure Modes

- The risk of the cryptography algorithm failing because it contains patterns from the plain text and the algorithm can be guessed.
- The risk of the decryption key being guessed and hence may succumb to attacks such as brute-force attack.

- Risk of using lower bit keys to encrypt data. The lower the number of bits the easier it is to crack the key. 128 bits and higher are considered safe.
- Loss or corruption of a private key.
- A key is compromised but there is a failure to replace or remove the compromised key.

Cryptography is a well-published topic and lots of information is available on the web on encryption. It may prove futile for testers to test every aspect of cryptography, due to the complexity of the subject. But a basic idea of the risks associated with violating a few basic rules, such as safe key selection or failure to replace corrupted keys, will help in validating the security in the transaction stage of the shopping cart.

Apart from encrypting the data transfers to protect information, “Document Confidentiality” also involves physically safeguarding files and documents that contain sensitive and confidential information.

Refer to the bug example, “Shopping Carts Expose Order Data,” where a poorly installed shopping cart exposes the order ‘log’ file with names, addresses, and credit card numbers in a world-readable format. People could search for these log files just by entering simple key words from any search engine!

Failure Modes

- Risk of exposing directories that hold sensitive files and allowing an external user to access the directory or folders from the web!
- Risk of setting improper read and write permissions to these files allowing any external user to access and modify these important files.

- Risk of mistakenly configuring the e-mail list server to include sensitive customer information or attach confidential files in public e-mail listings and postings.
- Script errors that let users edit their URL by changing a few visible parameters, like order number, granting them access to other users' records!
- Poor configuration of shopping carts may cause an attacker to gain entry to classified information (refer to the examples for more details).
- Risk of unfixed bugs or new bugs in databases and server software may open up serious security holes (refer to the example bugs). The site www.bugnet.com hosts a long list of security bugs in this category.
- Check for issues such as where the shipping section shows "billing information" in non-secure customer e-mail messages.
- Check for situations where, instead of just the last four digits, all numbers of the credit card are exposed.

Examples of Related Bugs and Other Issues

E-Commerce Fears? Good Reasons

http://www.wired.com/news/ebiz/0,1272,44690,00.html

Shopping carts expose order data

http://www.internetnews.com/ec-news/article.php/4_102621

Shopping Carts exposing CC data

http://exploiter.virtualave.net/9904-exploits/cybercash.cc.txt
http://exploiter.virtualave.net/9904-exploits/perlshop.cc.txt
http://exploiter.virtualave.net/9904-exploits/shopping.cart.cc.data.txt

Expert finds hole in shopping carts

http://zdnet.com.com/2100-11-514435.html

HQ for exposed credit numbers

http://www.wired.com/news/ebiz/0,1272,44613,FF.html

Which? under fire over security scare

http://news.bbc.co.uk/1/hi/sci/tech/1402222.stm

Qwest glitch exposes customer data

http://online.securityfocus.com/news/431

O'Reilly leaks geeks' info

http://online.securityfocus.com/news/408

United Airlines

Frequent fliers who logged onto United Airlines' Web site got a look at other people's mileage plus account information for more than 12 hours.

http://www.siliconvalley.com/docs/news/tech/072275.htm

Hacker posts credit card info

http://www.wired.com/news/technology/0,1282,33539,00.html

7.13 System Security

7.13.1 Password Security (Password disclosure vulnerabilities)

- Compromised password file: Test for location, protection and safety of /etc/password file. It is generally a soft target and can be easily stolen/compromised.
- Root directory of the password file compromised: Check if the shadow password file/ root directory is secure and protected. Test your protected password database for its ability to withstand attack by known password crackers.
- Password stored in a plain readable cookie: “Save User ID and Password to this computer” option saves user ID and password in a cookie in clear text, making it readable by any user.
- Unencrypted password files and indexable by search scripts: Test for common password file extensions. Use of known password file extensions such as .pwd makes the file an easy target for hacker scripts that search for common password file types.
- Password visible as plain text in the URL: Test for sensitive variables such as passwords in the URL when using the GET method. The practice of passing such variables in the URL may expose the authentication credentials to attackers.
- Existence of unknown default logins: Test for existence of default logins and passwords such as “admin” and password: “letmein”.
- Forgetting to remove dummy logins created during design/test phase: Test for dummy logins created for testing purposes but not destroyed after the

testing is done. Many hackers have found that getting into a site is as easy as typing “test” for login and “test” for password!

- Password in plain text in the backup configuration files: Test the configuration backup file, for presence of logins/passwords. There are known cases referenced in the example section where admin passwords have been accidentally written into the configuration backup file in plain text when the administrator backed it up.
- Compromised passwords due to cross-scripting attacks: Test all common elements of functionality in a form page for existence of inserted malicious code that can cause cross-scripting attacks. These may result in an attacker accessing the target user's cookies (including authentication cookies containing password information), if any, associated with the site. An attacker could also access data recently submitted by the target user via Web form to the site, or take actions on the site acting as the target user.
- Encryption function failure: Test for efficient use of DES encryption functions (crypt) to protect the password. For example, in one bug, the first two characters of the password were in plain text! This effectively reduced the key space to 42 bits and made it easy to conduct a brute-force password guessing attack on the rest of the key space.
- Non-existence of scripts or protection mechanisms: Test for existence of routines/mechanisms that will enable “Account Lockouts” in the event of a brute force guess password attack.
- Misuse of account protection mechanism for password authentication: An attacker might generate a random username and check if it is a valid user-name depending upon the error message returned. For example, "The user has entered an invalid password" indicates that a user name is a valid user name, while the error message "The user name could not be found"

indicates that a user name is not a valid one. Try testing for existence of account lockouts for repeated entry of incorrect/non-existent logins.

- Password leaks in log analyzers: Log file analysis tools have leaked administrator password information in the past due to bugs present in them. Test the tool for known bugs publicized in security advisories.
- Password leaks through alternative text files capable of storing passwords: Passwords may be mistakenly stored in non-password plain text files, such as in the case of JustAddCommerce shopping cart software. Here, the passwords were available in rtf.log files. So it may be worthwhile to test for alternative non-encrypted file types that are capable of holding passwords.
- Weak dynamic password generator algorithm: Dynamic password generators or temporary password generators may cause a security breach if their password generator algorithm is weak or based on some guessable parameter, such as current requester's session ID or IP address. A good example of this failure is the case of a Web-based bulletin board where one user could send a request for a password reset for another user. Since the temporary password was based on the requester's session ID, the requester could guess the new password of the victim's account by guessing the generated password from his/her current session ID.
- Default configuration vulnerability can give a malicious user access to the root directory. Even if the root directory's default is reset, when default passwords for other groups exist, then a malicious user can log in with one of the non-root accounts, rename their '.profile' file, and then telnet to the system to obtain a UID of 0 (i.e., root access).
- SQL-injection-attack may be used to create a legal database query in certain cases, and a user can enter a system without a password. A good example from the bug list to explain this is the bug in AdMentor banner ad rotation

script, which lets a user input login: ' or "=‘ and password: ' or "=‘ which creates a query that looks like: SELECT row FROM table WHERE login = " or "=" and lets a user enter without a password!

- Check for weakly encrypted admin passwords.

7.13.2 Cross-Site Scripting Vulnerability

- Exploit: “On the final checkout page, save the HTML to disk (keeping browser open to maintain session) and edit the ACTION= portion of the form to direct the data back at the shopping cart instead of to verisign. The exact URL should match that to which verisign would submit a validated order. Save the edited HTML, reload in your browser, and submit bogus credit card information with your order. Since there is no authentication between Verisign and the shopping application, the shopping application will think that the card was authorized, and so it will finalize the order.”(Reference: http://lists.insecure.org/vuln-dev/2002/Jan/0066.html)
- Exploit: “Sign up for a free demo PayFlow Link account at Verisign. While in demo mode, this account will "validate" almost any credit card information submitted to it as long as the card# meets basic format, expiration date hasn’t expired, and amount <= $100. This demo account should be configured to send the confirmation information to the exploitee's shopping system. Then perform a similar HTML edit of the final checkout page as above, only this time, change the hidden form tag to direct the payment to the demo PayFlow Link account. Save the HTML, reload in your browser, and submit bogus credit card info.” (Reference: http://lists.insecure.org/vuln-dev/2002/Jan/0066.html)
- A remote attacker could embed malicious JavaScript within user information fields when registering as a new customer. When an

administrator views the customer listing, the script would be executed allowing the attacker to control administrative tasks. Refer to the example of SunShop Shopping cart where this type of security risk exists.

- Improper validation of user input can lead to cross-site scripting vulnerability. An attacker might include a malicious script within a URL he/she posts. The attacker can steal data/passwords/ run arbitrary code on a victim who clicks on the link.

7.13.3 Denial-Of-Service Attack

- Specially formed URLs intended to cause the CPU to rise to 100% usage may be used. An illustration of this form of attack is the use of http://target/cgi-bin/c32web.exe/ShowProgress on Cart32 shopping cart. This results in a 100% consumption of host's CPU and leads to a DoS on the site. The system needs to be re-started to resume service.
- Test for vulnerabilities in web servers where URLs sent to the server are appended with a large character string to cause a buffer over-flow resulting in a DoS attack on the web server.
- Test for potential failures where DOS devices such as 'con', 'com1', 'com2', etc are specified as the files to be opened. When the application "dll" tries to open these objects, a denial-of-service may occur.

7.13.4 Virus and Worms

Virus attacks have always been one of the major threats to security. In the past viruses such as the Nimda virus, SirCam virus,'I Love you bug' and others have wreaked havoc and e-commerce sites have lost millions in virus related losses.

The list of viruses that can affect an e-commerce site is long and exhaustive. Hence I have provided some news articles/ news media clips in the bug list that offer some insight into the incidents of virus attack on e-commerce related software. The best way to know if the any of the applications in your e-commerce system is prone to a virus attack is to search one of the current virus advisories or virus information centers for any reported activity. Some of the current and well-updated virus advisories/ information centers are:

- Symantec Security Response: (http://securityresponse.symantec.com/)
- The McAfee AVERT Virus Information Library: (http://vil.nai.com/vil/default.asp)
- Computer Associates Virus Information Center: (http://www3.ca.com/virusinfo/)
- Virus Bulletin:(http://www.virusbtn.com/)
- The WildList Organization International (http://www.wildlist.org/)

7.13.5 Browser Vulnerabilities

The browser archive at evolt.org (http://browsers.evolt.org/) lists almost all known browsers and allows you to download them. Some of these browsers are very specialized and virtually unknown to general users. Of all known browsers, Internet Explorer from Microsoft and Netscape (Now owned by AOL-Time-Warner) have continued to be dominant in the browser market and together make up for the bulk of the market share.

The list of security holes/vulnerabilities/flaws in different browsers is innumerable and exceeds the scope of this sub-category. Hence, I have limited my discussion on browser vulnerabilities to MS-IE and Netscape. Many organizations create a list of browsers they support and they limit their tests to those alone. The best way to test if the customers to your e-commerce site are going to face any security glitch/ privacy glitch while shopping at your site is to first get a list of common browser types used by your customers (from the server logs.) Then search and create a list of vulnerabilities those browsers may have by referring to the different vulnerability advisories.

Glitches caused by browsers such as exposing of credit card numbers due to a cookie glitch in the browser or serious system security breach such as ability to execute malicious code through a browser buffer over-run etc have hampered customer trust and confidence about shopping safely on e-commerce sites. Provided below are examples of vulnerabilities that have been seen in IE and Netscape and links to other resources that provide detailed information about browser related security issues.

7.13.5.1 IE Vulnerabilities (IE 5.01, IE 5.5, and IE 6.0)

The Internet Explorer is widely acclaimed as the most popular and leading browser software in use. But, it has also led the list of having probably the largest list of security vulnerabilities in most of its versions. These vulnerabilities have affected millions of e-commerce/ web users across the globe as IE holds the largest market share. The article below lists six of the most recent vulnerabilities. The list is just a minute sample and far from being complete. Refer to the security advisories listed later in the section for other vulnerabilities and for other versions of IE.

The article “Beware of critical new IE vulnerabilities” at http://www.zdnet.com.au/newstech/security/story/0,2000024985,20263902,00.htm lists these six of the latest IE vulnerabilities as critical:

- "IE 5.5 or 6.0 had a HTML buffer overrun flaw that could allow attackers to gain user-level access to any computer that connects to a malicious web site or that opens a HTML e-mail."
- "Buffer Overrun in HTML directive (CAN-2002-0022) — this critical threat can allow an attacker complete access to both servers and client systems if users visit a malicious site or open an HTML e-mail. This is user-level access, so the risk depends on the permissions granted to the particular user. Note that IE 5.01 is not vulnerable to this flaw."
- "GetObject scripting command spoof (CAN-2002-0023) — this is a moderate threat on servers and a critical threat on client systems. It allows an attacker to read files on the vulnerable system either through a malicious web site or HTML e-mail. It will not allow the attacker to alter files."
- "Download dialog spoofing via content-type and content-id fields (CAN-2002-0024) — this is a moderate threat on servers and client systems. It will cause the wrong filename to appear in a file download dialog box, which could trick users into downloading a malicious file. Microsoft says that this is not the same vulnerability discussed in MS01-058."
- "Application invocation via content-type field (CAN-2002-0025) — this is a moderate threat on both servers and client systems that allows a malicious web page to cause existing files to execute on a vulnerable system. Although Microsoft lists this as a moderate threat, I consider it critical because it could be used to initiate a reformat on a hard drive, for example."
- "Script execution (CAN-2002-0026) — this is another moderate threat on both servers and client systems. It allows a malicious web site to cause IE 5.5 and IE 6.0 to bypass protections against running arbitrary scripts. IE 5.01 is not vulnerable to this flaw."

- “Frame domain verification via document.open (CAN-2002-0027) — this is a moderate threat on servers and a critical threat on client systems. It's a variation of the vulnerability discussed in MS01-058 and poses threats similar to the GetObject scripting spoof, including the ability to read files on vulnerable systems. IE 5.01 is not vulnerable to this flaw.”
- “Download dialog spoofing via content- type and content-id fields—IE 6.0 has a particularly dangerous default setting that causes files to run rather than to be saved.”

One way to stay informed about flaws in IE that could cause serious security issues to the users of your e-commerce site could be by subscribing to the Microsoft Security Bulletin, from the company. You can also get information about getting the corresponding patch from the bulletin. Major security advisories such as Common Vulnerabilities and Exposures (CVE), The CERT® Coordination Center, SecurityOffice.net, SecurityFocus.com, SecurityTracker.com and Securiteam.com also publish updates on latest security flaws and vulnerabilities in browsers.

7.13.5.2 Netscape Vulnerabilities

Netscape Security News Archive at (http://wp.netscape.com/security/notes/?cp=sciln) is probably a good place to start looking for vulnerabilities and other security related issues that have affected the Netscape browser. The site archives all security flaws prior to the Netscape 7.0.

Listed below are a few examples of vulnerabilities from the archive:

- “XMLHttpRequest Vulnerability
  A flaw that could potentially allow a malicious web site to read files stored on a user's computer has been discovered in Netscape 6.1 through 6.2.2 versions of the Netscape browser.”

- "Cookie Vulnerability
  A flaw that could potentially allow a malicious web site to read the cookies that another site has stored on a user's computer has been discovered in Netscape 6 through 6.2 versions of the Netscape browser."

- "SmartDownload Exploit
  A potential exploit was discovered for Netscape SmartDownload version 1.3 in which a buffer overflow could potentially be used to execute malicious code on a user's computer. The potential exploit affects Netscape 4.x or Internet Explorer Browser users with SmartDownload 1.3 installed on their computer."

- "JavaScript Cookie Exploit (May 2, 2000)
  An exploit was reported for Netscape Communicator 4.72 and earlier in which a hostile site can read the links in a user's bookmark file and some attributes of HTML files if the user's profile name and the Communicator installation directory path are known to the hostile site."

The WWW Browser Security & Privacy Flaws website at (http://www.cen.uiuc.edu/~ejk/browser-security.html) also lists some browser related security flaws.

7.13.6 Errors: Input Validation, Access Control, Buffer Overflow, Authentication, Configuration

- Price validation scripts absent in shopping cart code: "A hacker puts $100 worth of items in a shopping cart and then saves the Web page to a local hard drive. He or she then modifies the price to $10 and resubmits the page. If the shopping cart is improperly coded, it might not double-check the prices and allow the price change upon resubmission." (Reference: http://www.informationweek.com/story/IWK20020221S0025).

- Ability to append variables to URLs with malicious intent: Appending specific variables and values to URL strings such as http://target/cgi-bin/cart.pl? allows remote users to perform certain actions.(Reference: http://www.cve.mitre.org: CVE-2000-0252)

"vars" will display the configuration settings of the application that include the username and password used for credit card transactions.

The "db" string will list the entire database file containing all items in the shopping cart.

- “May allow remote users to modify shopping cart contents by requesting a certain URL with altered variables. A new item will appear in the shopping cart on the website with the latest manipulated data.”(Reference: Dansie Shopping Cart 3.04 Multiple Vulnerabilities at CVE-2000-0252)”
- “Many popular shopping cart applications that use a Microsoft Access database backend do not properly restrict access to the shopping cart database file (.mdb). This could allow a remote attacker to send a specially crafted URL request to the server to download the database and gain unauthorized access to sensitive customer information” (Reference: http://www.iss.net/security_center/static/9816.php)
- Check for vulnerabilities in perl and cgi scripts. One potential error is that of variables failing to check input or perform access validation. Refer to example, Hassan shopping cart version 1.18 or earlier, in which a remote attacker can request a URL containing "dot dot" (/../) sequences to traverse directories and read arbitrary files on the Web server.
- In a buffer over flow attack, a remote attacker can overflow a buffer and execute arbitrary code on the system by sending a long query string. Test for existence of constraints on inputs and test by inputting long query

strings. Refer to the buffer overflow bug in PDGSoft Shopping Cart version 1.50, which is vulnerable due to a redirect.exe script & changepw.exe script.

- Badly configured shopping cart could expose the order log file and the configuration file that may include the system's admin username and password in plain text.
- Configuration file maybe exposed due to poor configuration. The configuration file may contain sensitive data, such as the admin password in plain text format. Refer to bug in quickstore that exposed admin password due to poor configuration settings. In the case of EZMall 2000 shopping cart, it could expose the order log file, which contains sensitive information about the purchase activity on the vulnerable site.
- Check for vulnerability where a remote attacker can modify hidden form fields to specify arbitrary executable files located on the server. These files may be executed when the form is submitted. This vulnerability exists in Carello Shopping Cart for Windows NT and Windows 2000. Check the example bug list for details.
- Check for file-over-writing vulnerability. This means the ability for an attacker to rename files (for which he knows the name and path) to incorrect file name extensions and then sending an HTTP request to download the files. Test by renaming a known file password.txt to password.txt1 and request password.txt1. If the download file/save it to disk box appears, you have just found a vulnerability! Example bug in the list: Pacific Software, Carello File Duplication and Source Disclosure Vulnerability
- Does your shopping cart CGI executable produce an error that could lead to a debugger page? If so, you might have a potential vulnerability. Check the

debugger page for server configuration details, environment settings, list of programs etc.

- Test the shopping cart page source. In the view source file, is there any javascript function visible to the user that passes any sensitive variable? Check if the modification of any of those variables affects data displayed on the page or price of the item? Check the bug on “Java Internet shop vulnerability” for an example of this bug.
- Check if input data is filtered for symbols such as ‘/’ or ‘dot dot’ (..). An attacker can append these symbols to a URL to traverse directories/ retrieve files that are normally not accessible.
- Diagnostic tools included with shopping software with default invocation names such as shopdbtest.asp pose problems when attackers use them for purposes not intended. For example, the diagnostic tool in VP-ASP disclosed information with regards to the location of the database that contained the storage and configuration files named shopping400.mdb and shopping300.mdb by default. The file contained a person’s information including the credit card details in a plain text format!
- Test for some basic and straightforward design vulnerabilities such as storing of sensitive data in a world-readable, un-protected and un-encrypted Microsoft access database.
- Test if non-HTML files types are accessible by specially constructed http requests, such as http requests appended with null file extensions, which may bypass file extension checks.
- Check error messages for the details they display. For example, requesting a non-existent template or file can result in a poorly configured error message handler to divulge details such as full path to the working directory.

7.13.7 Execution of Arbitrary Code

- Check if specially crafted URL requests can be sent with meta characters to execute arbitrary commands on the server with elevated privileges. Look at the bug in “Shop Plus cart” for an example of this type of attack.
- Check for existence of a secret password backdoor. Cart32 shopping cart version 2.6 was tested and a secret backdoor password "wemilo" was found! An attacker could exploit this vulnerability to obtain sensitive information including passwords and customer information, such as credit card numbers. The attacker could also execute arbitrary code.
- Check for Authentication Bypass Vulnerability, which provides a remote attacker to bypass authentication by providing the program with a filename that exists, or by inserting a null character. This may result in enabling an authenticated attacker to execute arbitrary code sometimes via shell meta characters in the kill parameter. Test the program code that directly passes parameters to “system” commands for lapses that can allow a user to pass a kill command via the system command.
- Check shopping cart installations that use “sample code”, or “sample pages” that come with the package. Sample code may have some inherent vulnerability. An example is MiniVend version 4.04 shopping cart and earlier that comes with a sample storefront containing a vulnerable piece of code. Refer to the bug in the bug list.
- Test the scripts provided for search in the shopping cart for flaws. Search scripts have access to content, the files in the web server and if input is not validated, it can become an easy gateway to execute arbitrary code.
- Check load-page scripts for lack of input validation. These are another potential gateway to executing arbitrary code.

Examples of Related Bugs/Issues

Shopping-cart glitch could give hackers a discount

http://www.cnn.com/2000/TECH/computing/02/04/shop.glitch.idg/

IE flaw puts credit card info at risk

http://www.zdnet.com.au/newstech/security/story/0,2000024985,20261796,00.htm

E-shoppers rattled by U.S. security snafus

http://www.ecommercetimes.com/perl/story/4544.html

The form tampering vulnerability

http://www.webtechniques.com/archives/1998/09/webm/

High risk Apache exploit circulating

http://www.internetnews.com/dev-news/article.php/1369501

Swarm of Yahoo bugs raises security questions

http://news.com.com/2100-1023-240982.html?tag=rn

Bugs bust open 'unbreakable' Oracle 9i

http://zdnet.com.com/2100-1104-831204.html

Security flaws found in PHP

http://www.internetnews.com/dev-news/article.php/982841

Vulnerabilities in Microsoft SQL Server
http://www.cert.org/advisories/CA-2002-22.html

Apache web server chunk handling vulnerability
http://www.cert.org/advisories/CA-2002-17.html

Multiple vulnerabilities in Microsoft IIS
http://www.cert.org/advisories/CA-2002-09.html

Multiple vulnerabilities in Oracle Servers
http://www.cert.org/advisories/CA-2002-08.html

Buffer overflow in Microsoft Internet Explorer
http://www.cert.org/advisories/CA-2002-04.html

Multiple vulnerabilities in many implementations of the Simple Network Management Protocol (SNMP)
http://www.cert.org/advisories/CA-2002-03.html

Microsoft Internet Explorer does not respect content-disposition and content-type MIME headers.
http://www.cert.org/advisories/CA-2001-36.html

Multiple vulnerabilities in several implementations of the Secure Shell (SSH) protocol
http://www.cert.org/advisories/CA-2001-35.html

Buffer overflow in System V derived login

http://www.cert.org/advisories/CA-2001-34.html

Oracle9iAS web cache vulnerable to buffer overflow

http://www.cert.org/advisories/CA-2001-29.html

Nimda worm

http://www.cert.org/advisories/CA-2001-26.html

Continued threat of the "Code Red" worm

http://www.cert.org/advisories/CA-2001-23.html

W32/Sircam malicious code

http://www.cert.org/advisories/CA-2001-22.html

Cisco IOS HTTP server authentication vulnerability

http://www.cert.org/advisories/CA-2001-14.html

Superfluous decoding vulnerability in IIS

http://www.cert.org/advisories/CA-2001-12.html

Buffer overflow vulnerability in Microsoft IIS 5.0

http://www.cert.org/advisories/CA-2001-10.html

Sadmind/IIS worm

http://www.microsoft.com/technet/security/bulletin/MS00-078.asp

Netscape allows Java applets to read protected resources

http://www.cert.org/advisories/CA-2000-15.html

Inconsistent warning messages in Internet Explorer

http://www.microsoft.com/technet/security/bulletin/ms00-039.asp

http://www.cert.org/advisories/CA-2000-10.html

Malicious HTML tags embedded in client web requests

http://www.cert.org/advisories/CA-2000-02.html

Widespread attacks on Internet sites

http://www.cert.org/advisories/CA-1995-18.html

Taming the wide open web

Web sites are vulnerable in many different ways. Here's what you need to know to make yours as secure as possible.

http://palmosadvisor.com/Articles.nsf/aid/ANDRM01

A frenzy of hacking attacks

http://www.wired.com/news/infostructure/0,1377,34234,00.html

Scripting flaw threatens web servers

http://zdnet.com.com/2100-1105-945502.html

File-name flaw threatens PGP users
http://www.zdnet.com.au/newstech/security/story/0,2000024985,20267999,00.htm

Apache security flaw discovered.
http://www.linuxmax.net/news/00807.html

A security bug was found in software used by millions of web sites. Private experts alerted users and the FBI's computer security division.

PHP flaw leaves two million servers open.
http://www.itweek.co.uk/News/1133816

NETGEAR FVS318 Firewall Router username/password disclosure

FVS318 Firewall/VPN/Router stores usernames and passwords in plain text when a backup of the configuration is made.
http://www.securiteam.com/securitynews/5TP0Y008AQ.html

CGI-Telnet Perl Script for web servers discloses password file to remote users
http://www.securitytracker.com/alerts/2002/Sep/1005319.html

Web Server 4D may disclose passwords to local users.
http://www.securitytracker.com/alerts/2002/Sep/1005286.html

Opera Web Browser may disclose passwords typed into an HTML Form to local users.
http://www.securitytracker.com/alerts/2001/Nov/1002797.html

iPlanet Web Server publishing feature allows remote users to conduct brute force password guessing attempts.
http://securitytracker.com/alerts/2002/Jan/1003156.html

EServ Web Server discloses password protected files and directories to remote users
http://securitytracker.com/alerts/2002/Jan/1003173.html

JustAddCommerce E-commerce Software discloses user passwords to local users
http://securitytracker.com/alerts/2002/Feb/1003644.html

Messaging Board lets remote users gain access to other user bulletin board accounts
http://securitytracker.com/alerts/2002/Feb/1003422.html

Oracle Application Server PL/SQL Module for Apache has buffer overflows that allow remote users to execute arbitrary code and gain access to the server.
http://securitytracker.com/alerts/2002/Feb/1003451.html

CGI Online Worldweb Shopping (COWS) E-Commerce System discloses user information and order data to remote users and also permits cross site

scripting attacks.
http://securitytracker.com/alerts/2002/Jan/1003309.html

Cross-site scripting vulnerability: flaw leaves online Citibank customers vulnerable.
http://www.eweek.com/article2/0,3959,141472,00.asp

Bank closes web security hole
Fleet fixes credit card site flaw after customer discovers, reports breach.
http://www.pcworld.com/news/article/0,aid,74964,00.asp

Apache Tomcat vulnerable to cross site scripting via passing of user input directly to default error page
http://www.kb.cert.org/vuls/id/672683

Lotus Domino Server R5 vulnerable to cross site scripting via passing of user input directly to default error page
http://www.kb.cert.org/vuls/id/642239

IBM WebSphere vulnerable to cross site scripting via passing of user input directly to default error page
http://www.kb.cert.org/vuls/id/560659

CERT® Advisory CA-2000-02 malicious HTML tags embedded in client web requests
http://www.cert.org/advisories/CA-2000-02.html

Shopping-cart-form-tampering (4621)

Form tampering possible in several web-based shopping cart applications
http://www.iss.net/security_center/static/4621.php

Dansie Shopping Cart 3.04 multiple vulnerabilities
http://online.securityfocus.com/bid/1115

The Make-a-Store OrderPage shopping cart application allows remote users to modify sensitive purchase information via hidden form fields. (CAN-2000-0101 (under review))
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0101

The SalesCart shopping cart application allows remote users to modify sensitive purchase information via hidden form fields.
(CAN-2000-0102 (under review))
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0102

The SmartCart shopping cart application allows remote users to modify sensitive purchase information via hidden form fields.
(CAN-2000-0103 (under review))
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0103

The Shoptron shopping cart application allows remote users to modify sensitive purchase information via hidden form fields.
(CAN-2000-0104 (under review))
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0104

**The EasyCart shopping cart application allows remote users to modify sensitive purchase information via hidden form fields.
(CAN-2000-0106 (under review))
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0106**

**The Intellivend shopping cart application allows remote users to modify sensitive purchase information via hidden form fields.
(CAN-2000-0108 (under review))
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0108**

**The WebSiteTool shopping cart application allows remote users to modify sensitive purchase information via hidden form fields.
(CAN-2000-0110 (under review))
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0110**

**The shopping cart application provided with Filemaker allows remote users to modify sensitive purchase information via hidden form fields.
(CAN-2000-0123 (under review))
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0123**

**The Check It Out shopping cart application allows remote users to modify sensitive purchase information via hidden form fields.
(CAN-2000-0134 (under review))
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0134**

The @Retail shopping cart application allows remote users to modify sensitive purchase information via hidden form fields.

CAN-2000-0135 (under review)
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0135

The Cart32 shopping cart application allows remote users to modify sensitive purchase information via hidden form fields.
CAN-2000-0136 (under review)
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0136

The CartIt shopping cart application allows remote users to modify sensitive purchase information via hidden form fields.
CAN-2000-0137 (under review)
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0137

BugTraq Mailing List, Wed Aug 07 2002 - 03:22:51 CDT, "MidiCart Shopping Cart Software database vulnerability" at
http://archives.neohapsis.com/archives/bugtraq/2002-08/0074.html

Multiple vendor e-commerce shopping cart information disclosure vulnerability
http://online.securityfocus.com/bid/2299/info/

Midicart ASP remote customer information retrieval vulnerability
http://online.securityfocus.com/bid/5438

Alan Ward A-Cart web accessible database file vulnerability
http://online.securityfocus.com/bid/5597

Shopping cart program leaves back door open
http://www.internetnews.com/ec-news/article.php/4_340591

Hassan Consulting's shopping cart directory traversal
http://www.iss.net/security_center/static/5342.php

PDGSoft Shopping Cart version 1.50 is vulnerable to a buffer overflow in the redirect.exe script
http://www.iss.net/security_center/static/4545.php

PDGSoft version 1.50 is vulnerable to a buffer overflow in the changepw.exe script.
http://www.iss.net/security_center/static/4546.php

Dansie Shopping Cart version 3.0.4 contains a backdoor in the cart.pl (Perl) application, which could allow a remote attacker to execute arbitrary commands on the Web server.
http://www.iss.net/security_center/static/4975.php

SunShop Shopping Cart version 2.5 and possibly other versions are vulnerable to cross-site scripting
http://www.iss.net/security_center/static/8840.php

PDGSoft's Shopping Cart misconfiguration exposes config and order files
http://www.iss.net/security_center/static/3857.php

QuikStore shopping cart system: This misconfiguration could expose the configuration file, which contains the plaintext administrator password. An attacker could use this password to compromise the system. http://www.iss.net/security_center/static/3858.php

Hassan Consulting Shopping Cart remote command execution http://www.iss.net/security_center/static/7106.php

Carello Shopping Cart hidden form fields could be used to call arbitrary executable files http://www.iss.net/security_center/static/9521.php

Cart32 shopping cart allows remote access to client lists and admin functions

BugTraq Mailing List, Thu Apr 27 2000 21:30:37, "Re: Alert: Cart32 secret password backdoor (CISADV000427)" at http://online.securityfocus.com/archive/1/57566

@stake, Inc./Cerberus Information Security Advisory CISADV000524b, "Carello Web file overwriting vulnerability" at http://www.atstake.com/research/advisories/2000/advcarello.html

Cart32 shopping cart allows remote access to server installation details http://www.iss.net/security_center/static/4398.php

BugTraq Mailing List, Wed May 31 2000 10:52:00, "Java Internet Shop Vulnerability" at http://online.securityfocus.com/archive/1/62793

BugTraq Mailing List, Wed Sep 05 2001 12:06:56, "ShopPlus Cart" at http://online.securityfocus.com/archive/1/212155

BugTraq Mailing List, Tue Dec 18 2001 - 07:51:05 CST, "Aktivate Shopping System Cross Site Scripting Vulnerability" at http://archives.neohapsis.com/archives/bugtraq/2001-12/0194.html

UkR Security Team - Advisory 03.04.2001, "uStorekeeper(tm) Online Shopping System - Runtime Script - 'arbitrary file retreival' vulnerability" at http://online.securityfocus.com/archive/1/173721

Delphis Consulting Plc Security Team Advisory DST2K0036, "Price modification possible in CyberOffice Shopping Cart" at http://online.securityfocus.com/advisories/2686

BugTraq Mailing List, May 27 2002 8:54AM, "VP-ASP shopping cart software." at http://online.securityfocus.com/archive/1/274230

Delphis Consulting Plc Security Team Advisory DST2K0035, "Credit card (customer) details exposed within CyberOffice Shopping Cart v2" at http://online.securityfocus.com/advisories/2685

Vuln-Dev Mailing List, Jan 21 2002 3:57PM, "Security holes in COWS (CGI Online Worldweb Shopping)" at http://online.securityfocus.com/archive/82/251570

SecuriTeam Security News Archive, 22-July-2002, "ClickCartPro Security Vulnerability (Misconfiguration)" at http://www.securiteam.com/securitynews/5DP0T0K7PY.html

Xato Network Security, Inc. Security Advisory XATO-112000-01, "MULTIPLE VULNERABILITIES WITH CART32 SHOPPING CART" at http://www.xato.net/reference/xato-112000-01.txt

BugTraq Mailing List, Tue Jul 11 2000 - 16:19:43 CDT, "Akopia MiniVend Piped Command Execution Vulnerability" at http://archives.neohapsis.com/archives/bugtraq/2000-07/0150.html

BugTraq Mailing List, Mon Oct 09 2000 03:45:41, "Security Advisory : eXtropia WebStore (web_store.cgi) Directory Traversal Vulnerability" at http://online.securityfocus.com/archive/1/138495

BugTraq Mailing List, Mon Feb 12 2001 - 10:51:38 CST, "Commerce.cgi Directory Traversal" at http://archives.neohapsis.com/archives/bugtraq/2001-02/0208.html

BugTraq Mailing List, Mon Mar 11 2002 - 05:33:37 CST, "CaupoShop: cross-site-scripting bug" at http://archives.neohapsis.com/archives/bugtraq/2002-03/0114.html

BugTraq Mailing List, Mon Jul 15 2002 - 15:56:40 CDT, "Again NULL and addslashes() (now in 123tkshop)" at http://archives.neohapsis.com/archives/bugtraq/2002-07/0166.html

BugTraq Mailing List, Mon Jul 15 2002 - 15:56:40 CDT, "Again NULL and addslashes() (now in 123tkshop)" at http://archives.neohapsis.com/archives/bugtraq/2002-07/0166.html

BugTraq Mailing List, Jan 14 2002 11:36PM, "Web Server 4D/eCommerce 3.5.3 Directory Traversal Vulnerability" at http://online.securityfocus.com/archive/1/250231

BugTraq Mailing List, Jan 14 2002 11:35PM, "Web Server 4D/eCommerce 3.5.3 DoS Vulnerability" at http://online.securityfocus.com/archive/1/250242

BugTraq Mailing List, Mon Mar 25 2002 - 03:10:52 CST, "dcshop.cgi anybody can delete *.setup for database" at http://archives.neohapsis.com/archives/bugtraq/2002-03/0302.html

NERF group security advisory #2, "Multiple vulnerabilities in web-shop 1C: Arcadia, in module tradecli.dll" at http://www.nerf.f2s.com/txt/adv02.txt

SecuriTeam Windows NT Focus Archive, 22-Feb-2002, "Rich Media E-Commerce Stores Sensitive Information Insecurely" at http://www.securiteam.com/windowsntfocus/5XP0N0A6AU.html

Cgi Security Advisory #5, "VirtualCart Shopping Cart" at http://www.cgisecurity.com/advisory/5.txt

BugTraq Mailing List, Sun Feb 27 2000 00:42:35, "EZ Shopper 3.0 shopping cart CGI remote command execution" at http://online.securityfocus.com/archive/1/48580

BugTraq Mailing List, Mon Dec 17 2001 - 16:30:49 CST, " Agoracgi v3.3e Cross Site Scripting Vulnerability" at http://archives.neohapsis.com/archives/bugtraq/2001-12/0177.html

Schwab's site could be vulnerable http://www.cnn.com/2000/TECH/computing/12/08/schwab.cost.idg/index.html

7.14 Risks Due to Memory Leaks

These are the curse of many a site and not surprisingly also a major cause of many e-commerce site crashes. These also cause deteriorating performance of many e-commerce sites. Shopping carts are complex systems that use scripting code, run on a server, need an underlying OS and also need a browser on the client side to function. Hence a memory leak that occurs in any of the above components can cause the shopping cart to fail indirectly due to memory leaks!

Just as this article on Web testing in ExtremeTech (Extreme Tech) puts it, “Memory leaks can range from irritating to debilitating. A merely irritating memory leak might involve a component growing until it crashes and re-spawns. This is still bad for the site, as a crashed component forces everyone to wait until it restarts. The worst-case scenario is a component that uses up more and more system memory (or worse, kernel memory) without exiting, until the entire system finally grinds to a halt."

The common issues discussed here is a subset of a larger set of memory leak issues and bugs that can be found (in more detail) at Labmice.net (http://www.labmice.net/troubleshooting/memoryleaks.htm)

Failure Modes

7.14.1 Issues Due to Memory Leaks in Scripting Code

- Some scripts have maximum static string size and their violation causes memory leaks. Hence shopping cart pages that use heavy and long scripts to add navigational or functional capabilities may violate these size constraints. This results in the browser generating "Out of Memory" errors when the cart pages are viewed.
- Bugs in the Script DLLs may cause memory leaks under specific conditions such as when the limit on the number of loops within a script is exceeded.
- Some inbuilt functions such as 'string format' functions (as in VBscript) in common script languages have known memory leak problems. Frequent use of these functions in a high-volume web environment such as ASP-enabled shopping cart page may lead to serious memory leak problems.
- Enough cannot be said about the negative effects of sloppy programming practices, the root cause to most memory leaks. Badly written scripts that lock up resources can be as lethal in shopping cart pages as anywhere else.
- Since shopping carts are chiefly database driven, resource locking is a big risk where a database record or a file becomes unavailable for prolonged

periods of time. This can happen when a particular site component has an exclusive hold on it.

- Older version of constituent components in the scripting environment can also be a potential cause to memory leaks. An example is using an older version of perl interpreter engine in a perl-based shopping cart.
- Some design-level decisions can also save the e-commerce site from potential memory leaks. One is to adopt a modular scripting framework where independent classes can be tested for memory leaks.

7.14.2 Issues Due to Memory Leaks in Browsers

- IE 5.0 and 5.01 have had memory leak problems when images were re-sized using DHTML (Dynamic Hyper-Text Markup Language). Because it is common to use scripts and DHTML to format images in shopping cart catalogs, care should be taken not to trigger memory leaks. This can potentially freeze the cart page, and cause performance problems and loading errors.
- Some browser methods like “StartDownload” consume excessive memory and do not return them back to the system. Hence shopping carts that offer downloadable files; product, or software code browsers; and also host large sized image files, are in danger of potential risks of memory leaks. These are triggered by calling these risky browser methods.
- Browsers may load and access several libraries. Shopping carts that spawn duplicate product browsers and pop-ups multiply the total RAM consumption. Because these browsers have known memory leakage problems associated with their versions, they may cause the system to come to a cranking stop, freezing any potential transaction midway.

- Memory leaks in IE 3.0x: These versions (3.0/3.01a/3.01b) have progressive memory leaks; one manifestation slows down the performance and response of the browser slowly over a period of time. Thus media-rich or heavy data content shopping cart pages face serious performance issues.
- Another famous memory leak bug in IE 3.x is that it caches page information when using the <Form Method=POST> tag and fails to free the memory until the application is shut down. This is a serious problem because many carts use GET and POST methods in their forms.
- Issues due to memory leaks in underlying operating system
- Undeleted threads are a major source of memory leaks in operating systems. Thus if the underlying OS of an e-commerce system starts leaking memory due to undeleted threads, the hosted shopping cart can begin to fail due to lack of memory available for its functioning.
- Any third-party process that may be running on the operating system may cause unreleased memory. This may indirectly cause the shopping cart to fail due to underlying OS failure because of lack of sufficient memory.
- Some standard system libraries in older versions of operating systems may leak memory; the risk of not upgrading to newer underlying operating systems in e-commerce sites transforms itself into a risk leading to memory leak.
- In shopping cart sites, which have user-written server side plug-ins created by server programming APIs, the user code may introduce serious memory leaks.

7.14.3 Issues Due to Memory Leaks in Server

- Incorrect use of multithreading in Web server software can be a problem and may end up in a memory leak. Because a shopping cart sits on a web server, a web server failure renders a shopping cart non-operative.
- According to the different IIS FAQ lists and MS knowledge base articles, Inetinfo process in Internet Information Server (IIS) may leak memory when using SSL.
- Some Web servers hosting software downloads or shopping carts offering large documents, such as eBooks and media files as downloadable products, have a high risk of suffering from memory leaks. These leaks can be equal to the size of the file being uploaded/downloaded if transfer is aborted prematurely. The cause may be due to some methods like Request.BinaryRead being called by ASP or other similar scripts.
- When trying to access member accounts in a shopping cart, small memory leaks may occur when a lookup of the current domain name is performed. A pointer to the domain name may be saved in a global location without freeing the previous domain name already stored there.

Examples of Related Bugs and Issues

IIS memory leaks
http://www.iisfaq.com/MemoryLeaks/

Memory leaks in OS
http://www.labmice.net/troubleshooting/memoryleaks.htm

More on resource leaks

http://www.willows.com/listarchives/dev/twindev-1998-jul/0155.html

7.15 Risks Due to Insufficient Capacity Planning

Failure modes

7.15.1 Risks Based on the Number of Users and Usage

- Shopping cart performance degrades due to increase in site users disproportionate to existing capacity.
- No increase in the number of users, but increase in the activity of the users, (increase in terms of catalog page hits, latency time, or increase in usage of search activity), increases in shopping cart update cycles. The increase in such heavy resource consuming activities may upset the capacity planning equation, which may be based on the number of users and not usage.
- The most common cause of sudden load, which causes deficiency in system capacity is the seasonal increase in customers especially the "holiday shoppers." Test shopping cart for performance and scalability under realistic loads.
- Increase in the number of transactions involving third-party components like billing cycles, credit card authorizations and account transfers, where the insufficiency in the capacity of the third party systems will indirectly cause the shopping cart and the e-commerce site to stall.
- Resource consumption also depends on the stage of the shopping cycle. For example: The checkout stage uses more pages, more CPU, more DB transaction cycles and more server utilization than the catalog 'browse'

stage. One must plan for sufficient capacity and availability for all stages of the shopping cycle, keeping in mind the requirement changes at each stage.

7.15.2 Risks Based on Computing Infrastructure

- CPU insufficiency may be a big risk if there is an excessive demand placed on CPU by the Web server or the database server. Web servers especially tend to consume more CPU cycles than a system's corresponding database server.
- If the shopping cart spawns a new process every time a user invokes it, and no mechanism exists to limit the maximum number of shoppers, then soon the processes will choke the available CPU and cause the entire system to slowly crash.
- An operation may cost less in terms of resource consumption. If the frequency of that operation is high, however, then very soon there is a capacity insufficiency risk. Generally, product pages and search pages are of moderate cost but, when search page operation is very frequent, soon, it may become the largest resource consumer despite a lower cost.
- Shopper capacity is determined by the underlying operating systems. For example, according to this paper, [De Klerk] "Win NT reaches CPU utilization of 96.40% at shopper's load of 1000 while Win 2000 reaches CPU utilization of 72.89% at shopper's load of 1000."
- If any shopping cart operation like basket load or catalog load is memory intensive, then the underlying Web server may run into memory deficit very soon.
- If any shopping cart operation forces the Web server's page-able process to page to disk, it will adversely affect the performance of the Web server.

7.15.3 Risks Based on Site Content Complexity

- Network capacity may become a bottleneck if shopping cart uses high static content like large images and static HTML.
- Poor site design, where heavy elements and heavy content pages are called more often than the lighter ones, causes resource consumption to become unevenly distributed and resource consumption becomes very high. Higher demand should have ‘light’ content; conversely, pages with lesser demand can be made ‘heavy’.
- Advertisements retrieved from ad databases, customizations to fit shopper’s choice, ActiveX control driven Menus and Java-based menus are complex site components. They may affect the capacity of the system and pose a risk to the functioning of the shopping cart.

Examples of Related Bugs and Issues

Crashing success for the Web?
For Online Retailers, a Make-or-Break Year Could Find Sites Overloaded
http://abcnews.go.com/sections/business/TheStreet/onlineretail991202.html

Charles Schwab Web site crashes
http://www.binarythoughts.com/article.cfm?StoryID=237

Encyclopaedia Britannica's new Web site crashes
http://www.infowar.com/p_and_s/99/p_n_s_102299e_j.shtml

E-tail sites crash over holiday weekend
http://news.com.com/2100-1017-249048.html?legacy=cnet

Webvan running out of Thanksgiving goodies (may also go into process failure)
http://news.com.com/2100-1017-248881.html?legacy=cnet

Customers locked out of Virgin megastore's online sale
http://news.com.com/2100-1040-230643.html?legacy=cnet

7.16 Web Server Failures

A Java Definition from the Java Glossary (Java Glossary):

“The Web server is the software that provides services to access a network, e.g., the Internet. A Web server hosts Web sites, supports HTTP and other protocols, and executes server-side programs.”

A Web server failure is probably the most commonly assumed failure by users who experience a failure while surfing the Internet. Web servers have evolved from simple software that serves pages. Today they are sophisticated software that also hosts and supports numerous protocols, depending upon functionality, and may also run functional scripts. From the testing point of view, a Web server can display a multitude of symptoms depending upon its function, its failure and how it fails.

During the initial research of this thesis I undertook a survey of available servers from different vendors. I reviewed their trouble shooting guides, white papers, technical support documents and open bug databases containing common bugs and issues. I would recommend that approach to any tester whose requirement is to write tests for a very specific brand of Web server such as IIS or Apache. Look up the product’s site for known issues and trouble shooting guides for lists of common symptoms for potential failures. Some descriptions for the bugs were sometimes very succinct. The tester may have to know the server terminologies really well to understand them.

Provided below is a set of example failure modes for a general Web server.

Failure Modes

- Overflowing static buffer
- Remote users can execute UNIX shell commands in UNIX web servers
- Remote users can download CGI script executables
- Web server aborts during startup
- Memory leakage
- Header confusion - some headers appear twice, some never appear
- Incorrect handling of file names and types
- Incorrect file permission on the web server
- Incorrectly configured DNS
- Unable to update the DNS
- Missing secondary DNS lookups
- Some service daemons may be down, like ftp, and http may work but telnet does not work
- Exceed the maximum simultaneous connection limit
- Connections take significant amount of time to close- slow connections
- Failure of server side processes such as scripts, cgi or servlets
- Incorrect permissions on the scripts and server side executables

- Perl or shell scripts throw compilation errors
- Newer versions of interpreter run on older versions of scripts that may break the script
- Serves improper headers
- Outputs malform HTML
- Incorrect firewall or router configuration might cut access to the Web server
- Firewall resets after power outages to older incorrect configurations
- The HTTP daemon does not start at startup.
- Some web server processes have crashed or are not responding
- Server timeouts
- Resource intensive processes may cause server requests to timeout
- The machine on which the server is running may run out of resources such as CPU, memory, etcetera
- A non-Web server process may consume all system resources, choking the processes.
- Fast-growing log files consume all the disk space and a full disk may choke the Web server
- Inadequate Web server capacity
- Web servers buckle under peak load

The white paper titled “Why is My Web Site Down?” (http://www.freshwater.com/white_paper/article.htm) to which I referred for writing this category, is a good read for testers testing for server failures.

Examples of Related Bugs and Issues

Apache Web server chunk handling vulnerability

http://www.cert.org/advisories/CA-2002-17.html

Multiple vulnerabilities in Microsoft IIS

http://www.cert.org/advisories/CA-2002-09.html

Scripting flaw threatens Web servers

http://zdnet.com.com/2100-1105-945502.html

EServ Web server discloses password-protected files and directories to remote users

http://securitytracker.com/alerts/2002/Jan/1003173.html

IBM WebSphere vulnerable to cross-site scripting via passing of user input directly to default error page.

http://www.kb.cert.org/vuls/id/560659

AOL home page glitches irk users

Glitch resulted from a server upgrade

http://news.com.com/2100-1023-827901.html?tag=dd.ne.dtx.nl-hed.0

Apache Web server flaw found

http://www.pcworld.com/news/article/0,aid,101952,00.asp

7.17 Network Failures

A reliable network is key to the success of any e-commerce site. If the site is frequently offline for network maintenance, then it does not bode well for the maintenance staff and the business.

Network failures could be at both the ends of an e-commerce system-- the client side and the server side. The server side network issues are probably more in the control of the testers/troubleshooters than the client side one. A network issue could range from a total loss of connectivity to intermittent connectivity and even performance problems. The network risks that any IT infrastructure faces apply to an e-commerce infrastructure too. Listed below are few of the common risks/ failures that one can see in any network environment.

Some of reading, which helped in compiling this list includes an Internet presentation slide from Avici systems, titled, "Causes of Downtime"5 and network troubleshooting guides provided by various vendors such as Cisco, Novell, etcetera.

Failure Modes

- Node or link inoperative
- Node or link does not function
- Node or link continues to operate but incorrectly

[5] Building a Reliable and Scalable

Internet (**http://www.ieice.org/hpsr2002/documents/archives/IEEE_HSPR_mtg_Final.pdf**)

- Incorrectly configured node
- Failures in underlying telecommunication switching systems
- Router failures
- Router table fails
- Router fails to scale to larger network implementations
- DoS attack on router
- Router fails to boot
- Device cannot establish IP communication with another device across a router, but can establish IP communication locally
- Network adapter fails at the server site
- Switch interface fails
- Transmit and receive cable pairs mismatch
- Load balancer fails
- Network hardware failures/ link failures
- Copper cables damaged/cut/ corrosion/magnetic interference
- Fiber cuts in fiber optic cables
- Cable breaks in Ethernet cables
- Network congestion
- Line card failures
- Slow restoration time
- Congestion resulting from re-routing

- Inefficient traffic engineering mechanisms
- Inefficient traffic engineering mechanisms
- Link failures and congestion
- Fiber cuts
- Line card failures
- Slow restoration time
- Congestion resulting from re-routing
- Inefficient traffic engineering mechanisms
- Issues related to network operation
- Issues due to platform upgrades
- Issues related to capacity expansion
- Configuration errors.
- Issues arising out of link expansion.
- Connectivity issues
- Loss of connectivity
- Intermittent connectivity
- Timeout
- Performance issues
- Duplicate addresses.
- Consistently high utilization rates

Examples of Related Bugs and Issues:

Router glitch cuts Net access

http://news.com.com/2100-1033-279235.html?tag=rn

AOL suffers e-mail lapse

http://news.com.com/2100-1001-204929.html?tag=rn

Network glitch hits Slashdot

http://news.com.com/2100-1001-268959.html?legacy=cnet

Domain name glitch plagues users

http://www.wired.com/news/technology/0,1282,19342,00.html

Hacking group reveals 'Net protocol security glitch'

http://www.nwfusion.com/news/1999/0812hack.html

Microsoft fixes wireless glitch

The technology giant has unofficially released an update to help prevent its wired and wireless hubs from disconnecting computers from the Internet

http://news.zdnet.co.uk/story/0,,t269-s2125745,00.html

WorldCom network troubles delay Internet

http://www.informationweek.com/story/IWK20021003S0012

7.18 Hardware Failures

Hardware failures can occur anytime and anywhere. The only way to control the risk is by designing the hardware architecture with fewest possible single points of failure. In an e-commerce system, a hardware failure can broadly be classified as any of three types, depending upon where they occur: Server-side hardware failures, client-side hardware failures and third-party hardware failures.

Server-side failures are more within the realm and control of the testers and maintenance personnel, but the impact of hardware failures at a third-party service provider or at a client's site will seriously affect the completion of any transaction that is underway. Listed below are some common failures and risks impacting hardware systems. I have generalized them to cover the three types.

The three types of hardware failures were three separate categories in the original design of this taxonomy and I would encourage testers to consider them to brainstorm test ideas. Brainstorming where the failure occurs and how it impacts the system will be useful to prioritize on the test ideas.

Failure Modes

- A shopping cart may not function and pages may not be served if the hardware that runs any of the following servers fail:
- Application server hardware
- Back-up server hardware
- Cache server hardware
- File server hardware

- Memory error may corrupt data on the hard disk and all shopping cart data, including the executable, web pages, scripts may be lost
- Non-recoverable disk failures and RAID (Redundant Array of Inexpensive Disks) failures
- Resource conflicts: Peripherals that try to use the same interrupt requests, DMA channels or I/O addresses may cause hardware resource conflicts; the e-commerce system may experience increased delay and even gradual failure
- High central processing unit (CPU) utilization by some processes and subsequent failure due to insufficient CPU
- RAM failures, paging faults, memory leaks in server hardware (refer to category on memory leaks for further failure modes)
- Inappropriate hardware configuration and problems due to insufficient resources. For example, creating the same hardware configuration used for a Web server for a back-up server. A back-up server may probably need more memory/disk space.
- Hardware time outs, which may lead to user session time outs during shopping sessions
- The shopping cart program tries to write to the wrong storage device
- The shopping cart program tries to read from the wrong storage device
- The shopping cart program may try to write into a wrong disk sector, over-write some other file on the disk and it may crash or time out
- Hardware fails due to over-heating

- Physical theft, physical damage, damage to server side hardware due to natural disaster, hardware damage due to water seepage/incorrect temperature control/ high humidity
- Hardware controllers may get corrupted
- System timing: Cache server hardware failures because access time is too fast to handle
- Power loss, power surge, intermittent power outages may cause short-circuits in mother board
- Modem failures due to power surges
- Ethernet card problems due to incompatibility with older machines
- Router failures
- Hardware-based load balancer fails
- Backup generators fails
- UPS fails to start upon power failure or starts only after the system reboots due to power loss
- Cables may get cut/damaged, Internet access devices may fail and the Internet access may get cut .

Examples of Related Bugs and Issues:

Power problem triggers hardware failure
http://www.lyonware.co.uk/PDFs/DT-CASE/NetSam.pdf

Massive e-commerce service recovers from lengthy technical glitch
http://www.cnn.com/2000/TECH/computing/12/03/nexchange.up/

eBay recovers from outage

A hardware failure that took the site off-line for almost 11 hours yesterday
http://www.internetnews.com/ec-news/article.php/4_550801

Amazon fixes daylong hardware glitch
http://news.com.com/2100-1017-273042.html?legacy=cnet

Hardware glitch plagues Hotmail
http://news.com.com/2100-1023-223476.html?tag=bplst

Tech glitch brings Napster down
http://www.cnn.com/2000/TECH/computing/10/04/napster/

Hardware glitch knocks Yahoo groups offline
http://www.netscapeworld.com/nl/itw_today/03052002/

Hardware glitch hits Yahoo e-mail
http://zdnet.com.com/2100-1105-851533.html

7.19 Navigation Failures

One of the most difficult problems in an e-commerce site design is navigation. If the navigation is bad, then:

- The user cannot find the content they are looking for quickly.
- They get lost and don’t know where they are on the site.

- They lose context with the logic of the Website.
- They get frustrated and you lose a valuable customer.
-

Listed below are some common failure modes for the navigation failure category.

Failure Modes

- Illogical placement and use of “next” or/and “back”/ “previous” buttons
- Bad design and selection of navigation structures or using non-linear navigation for linear segments of content
- Menu does not provide access to all segments of content.
- Content blind spot: some content not accessible by any of the navigation structures/paths.
- Information “buried” too deep; too many navigational “clicks” are required to get to the desired content segment.
- Navigational menu requires plug-ins that is incompatible with the user’s browser.
- Inconsistent and unclear navigational aids
- Unable to dynamically change navigational structures to keep up with non-unique URLs generated by dynamic HTML pages
- Bad use of frames
- Presence of too many irrelevant commands on the page that might hamper the navigation flow by giving the user too many options to click on, which go down navigation paths that lead the user away from the task.

- Poor navigational design, which leads a user away from one page to another but when the user tries to come back to the older page, the state is changed/lost.
- Unclear or unable to navigate to the correct exit path.
- Navigational failures due to over-dependency on browser back button to control data, because novice users "do not know where the browser ends and the application begins." (Shubin and Meehan, 1996)
- Network delay and increased download time due to bad navigation design.

Related articles

Tips for better site navigation
http://www.computer.org/itpro/homepage/Mar_Apr01/lam/lam06.htm

Navigation: - definition, information, sites, articles
http://www.marketingterms.com/dictionary/navigation/

Examples of Related Bugs and Issues:

Neimanmarcus.com's navigation: It was bad ... now it's worse
http://techupdate.zdnet.com/techupdate/stories/main/0,14179,2622380,00.html

Boo.com plays tricks on customers
http://techupdate.zdnet.com/techupdate/stories/main/0,14179,2399916,00.html

Nagged by navigation
http://www.zdnet.com/anchordesk/talkback/talkback_69349.html

7.20 Process Failures

Systems can be defined in terms of their processes. An e-commerce system contains many processes within itself at various stages. These include processes that take care of order taking, payment processing and order fulfillment. Some of these processes are manual and may need no software/hardware implementation. A sales associate, for example, looks at the order database and processes the order manually. But in larger e-commerce sites, where the orders are in the magnitude of tens of thousands, automated processes are put in place. These automated backend processes include inventory control systems, order tracking systems, shipping-address printing systems and are susceptible to failures. The failures could be due to their own internal software/hardware failures; they can affect the e-commerce site and prevent it from fulfilling its transactions completely and on time.

These back-end processes can be expected to fail in the following two ways:

- Process Crash
- Process crash is a complete failure of a process step resulting in the user being unable to go to the next step of the process. This type of failure brings the whole transaction to a complete halt.
- Process Omission
- Process omission is one or more process steps being left out by mistake. If the omitted process is critical, the resulting failures also may be critical.

Listed below are some of the potential risks of failures for these backend processes and systems. I have assumed some of these backend processes to be independent and wholly autonomous systems, ones that work in conjunction with the e-commerce system by processing data for it. Others I assume to be part of the e-commerce system (like search, browse, user-authentication/authorization, etcetera).

Failure Modes

7.20.1 Failures at the Level of Order Taking

Process crash

- Order taking page does not load
- New customer registration process not working
- Login process fails: Registered customer unable to login into his/her account.
- Search process not working: Unable to search the catalog
- Browse process fails: ‘Browse,’ which is a type of pre-defined search is out of sense with the context of the store.
- Order selection fails: Unable to select/add/remove items.
- Cart process loses state: Item list shows no items in the cart.

7.20.2 Partial Process Fulfillment/ Process Omission

- Returning of ‘partial’ search result, with not all items displayed in the search result
- Displaying catalog with omitted product categories
- User may add items, but item list loads partially with some items omitted.
- Some steps in the process may get skipped erroneously; process may misdirect the user to the checkout step when the user is only half done with the shopping (may be due to incorrect page linking, erroneous time-outs and redirect scripts).

7.20.3 Failures at the Level of Payment Processing

- User may be abruptly directed out of a secure state to a non-secure state inside the cart as some steps in the secure connection process might have been omitted

7.20.4 Failures at the Level of Order Fulfillment

- Process partially implemented and last few steps maybe missing. For example order taking goes all the way until the end but shipping and handling process is missing.
- System prints wrong shipping addresses.
- Order tracking system fails.
- E-mail system fails: Delay in sending e-mail confirmation for orders placed.
- Inventory control system error: Order placed for an item not available in the inventory.

Examples of Related Bugs and Issues

Day 16: Best Buy ... bad experience (check Google's cache, page unavailable now)
http://www.usatoday.com/money/columns/lamb/holiday/day16.htm

Delivery disaster costs Toys R Us
http://www.zdnet.com.au/newstech/news/story/0,2000025345,20104173,00.htm

Webvan running out of Thanksgiving goodies
http://news.com.com/2100-1017-248881.html?legacy=cnet

FTC fines e-tailers $1.5 million for shipping delays
http://news.com.com/2100-1017-243684.html?legacy=cnet

(Delay)

Seven Internet retailers settle FTC charges over shipping delays during 1999 holiday season
http://www.ftc.gov/opa/2000/07/toolate.htm

You left out poorly designed order-handling systems
http://www.zdnet.com/anchordesk/talkback/talkback_229505.html

How to click off the customer (no email reply address, stock runs out after order)
http://www.computerworld.com/news/1999/story/0,11280,37299,00.html

How the e-shopping experience could be better
http://www.cnn.com/2000/TECH/computing/02/14/online.buy.idg/

Disappointing & frustrating Macys.com shopping experience
http://www.planetfeedback.com/sharedLetters/viewLetter/0,2936,104161-120-0-0-20-0-fb_date-desc-162081-0,00.html

7.21 Data and Data-handling Risks

Garbage in, garbage out (GIGO). That's been the mantra to remember for developers, testers and data managers for years. Data-related risks can range from just plain bad data to an inability to enter data or the inability to store data. Incompatibilities due to data type mismatch, accepting data without proper input validation are some of the common types of failures we see with respect to data and data handling.

Data related failures could be broadly blamed on three entities: The user, the data validation routines and the physical storage media. The user might crash the system by giving it bad data or induce the system to behave incorrectly by giving incorrect data. The data validation systems might be ineffective in filtering correct data type from incompatible ones. And, finally, the physical media might put restrictions on storing the data due to insufficiency problems. Bennett's et al. (1996) classification of data processing errors was used as the basis for the sub-category "Data I/O errors due to users."

Listed below are some example failure modes for this category.

Failure Modes

7.21.1 Data I/O errors Due to User:

- Invalid variable selection in data input fields
- More variables selected than necessary in the shopping cart input fields
- **Missing Input**: Some required data in the form field missing in the shopping cart

- **Incorrect Input**: incorrect entry of data in form fields such as wrong quantity, wrong item selection, incorrect billing details, and incorrect shipping details
- **Cross-matched or mismatched data entry**; swapping apartment # for age field
- **Invalid data format**: entry of date or e-mail address in a format that does not conform to the standard format of entry for that element
- Inability to accept data through alternate elements such as prompt boxes, dialog boxes, pop-up boxes, etcetera
- **Transposition errors**: user data entry swaps positions for various reasons
- **Copying errors**: number zero (0) being entered as alphabet 'o'!
- **Coding data errors**: wrong selection of entry codes, wrong entry of 'M' for a female in the 'sex' field in a form
- **Consistency errors**: Lack of consistency in entry of default data to support user inputted data, for example incorrectly appending a prefix of "Mr." when the user is identified to be a female!

7.21.2 Data Errors Due to Failure of Validation Routines.

- Audit don't edit. This clearly means to validate the data entry against the type acceptable by the shopping cart software, inform the user of any inconsistencies and ask the user to make changes. Do not let the validation routine edit the entry to some meaningless default.
- "Double entry and validation technique" fails. A user entering new password twice in password set/change field and both the fields are

validated against each other to confirm correctness of an entry, but the string comparison function does not work!.

- Real time validation fails.
- Over-enthusiastic validation script that replaces correct data with incorrect type. For example, a script checks for standard American date format (mm/dd) but swaps correct date inputs with the swapped but incorrect format of dd/ mm.

7.21.3 Data Errors Due to Physical Media Errors/ File I/O/ Data Incompatibilities

- Data entry unable to be processed due to required file renamed, moved, or deleted
- File access permission denied, hence data unable to be written into the file
- Unable to input data due to display of “fatal server error” error message.
- User unable to enter data due to insufficient disk space
- Disk operation error causes user to abort data entry
- Unable to enter data due to mundane but potential issues like server shutdown, maintenance, upgrade or power failure
- Server cannot process form input: POST format not recognized as valid may be due to an incompatibility between the httpd server and the user client.
- Inability to enter data through input devices such as Keyboard; touch pads or select data elements through keyboard short cuts

7.22 Third-party Software Failure

Many key services in an e-commerce site are provided by third-party services. The bill-payment processing is one such key service that is generally provided by a third-party. This implies that the third-party services should be failure-free in order for the e-commerce transactions to be processed to completion. But in reality there have been many instances where e-commerce sites have had to halt transactions because some subscribed third-party service has crashed. One indication of third-party failures could be the loss of one or more specialized functions from the website. For example in an online trading site, the stock values are pushed to the site by one service and the corresponding charts are drawn by a different third party service which specializes only in chart drawing. A missing chart could indicate that the third-party chart provider is down. Listed below are some failure modes, which exemplify this risk category.

Failure Modes

- Third-party bill processing or payment processing system fails due to unreliable batch processing, processor inadequacies or due to excessive load
- Third-party virtual server or dedicated server fails
- Password protection system is down due to a glitch in the third-party that provides the protection service
- Changes to system configuration at the third-party location causes compatibility issues with e-commerce sites where it is deployed
- “Unable to process payment”, probably the browser has javascript turned off or does not support it

- Data and communication hubs crash at the third party provider and the e-commerce site is affected
- The line to the third-party data center fails
- Third-party interfaces shut off some of the site’s e-commerce features
- SSL web server certificate expires at the third-party bill processing unit; as a result the e-commerce site is unable to process payments

Examples of Related Bugs and Issues

PayPal's outages disrupt eBay auctions
http://news.com.com/2100-1017-240760.html?legacy=cnet

Glitch bungles Amazon orders
http://news.com.com/2100-1017-980273.html?tag=lh

Amazon experiences technical glitches

“Amazon said that some third-party web platforms experienced problems…”
http://www.auctionbytes.com/pages/abn/y01/m04/i30/s02

Fast, simple ... and vulnerable
http://www.wired.com/news/technology/0,1282,33972,00.html

eBay Sellers complain of glitch using PayPal logos
http://www.auctionbytes.com/pages/abn/y02/m09/i06/s02

Should online shops accept PayPal payments?
http://www.aboutpaypal.org/paypal_accepts.htm

Third-party disasters--don't be a victim
http://www.zdnet.com.au/newstech/security/story/0,2000024985,20262450,00.htm

Update: Online trading problems caused by upgrade glitch at ADP
http://www.computerworld.com/managementtopics/ebusiness/story/0,10801,53531,00.html

E-chart software glitch plagued Jepp
http://www.ainonline.com/issues/12_01/12_01_echartpg16.html

HP upgrade muddies BigPond email

“Telstra’s BigPond email service across the country is on the blink after a Hewlett Packard software upgrade at 5 am Wednesday…”
http://www.itnews.com.au/story.cfm?ID=7549

Online-billing glitches drive many customers back to checks
http://www.naplesnews.com/02/11/business/d845403a.htm

7.23 ISP and Web Hosting Problems

These are two services that can be called the backbone services to the entire concept of online business. One provides the connectivity to the outside world and the other hosts the site for the outside world to see. Hence any failure in these services leads to the site going totally offline to the users. Sometimes, because the same provider provides both these services, any failure can mean both loss of connectivity and the site being inaccessible.

Listed below are the common types of failures encountered by the e-commerce site due to problems in their ISP or web host.

Failure Modes

- Hosting platform crashes
- Insufficient data storage space in hosting server
- Decreased bandwidth and poor data transfer rate
- Poor site performance due to overburdened network connections
- Reduced simultaneous/concurrent transaction sessions due to limited network capacity
- Web host is unable to accommodate high traffic volume
- One of the network services is disrupted and the web host is unable to redirect requests to alternate services, as the network design is non-redundant
- Failures along the Internet backbone
- DNS entries are not pointing to the right IP address
- All communication ports closed by mistake and system needs reboot
- Growing log files of one site consumes disk space and chokes other sites in a shared server hosting arrangement
- Frequent restoration of files from backups with no notification

- Web host does not work with secure connections but works well with regular http connections
- Significant planned downtime with no advance notification
- Hosting server dies with no errors
- Kernel-level bugs, kernel panic in Linux based host systems
- Bad RAM
- Badly configured configuration tables, IP tables, and random services blocked off
- Server ran out off swap space
- Blacklisted IPs
- IP conflicts
- Nameserver fails

Examples of Related Bugs and Issues

Web-Host Failures

Dell Web hosting site back after unknown glitches
http://news.com.com/2100-1001-248082.html?legacy=cnet

Jumpline Apologizes for glitches related to platform move
http://thewhir.com/marketwatch/jum031802.cfm

Glitch wipes out numerous Tripod sites
http://www.cnn.com/2001/TECH/internet/03/21/tripod.glitch.idg/

What can go wrong.

http://www.quickmba.com/ebiz/webhost/problems.shtml

ISP Failures

What's behind AOL outages?

http://news.com.com/2100-1023-209550.html?legacy=cnet

AOL outage brief but dangerous

http://news.com.com/2100-1023-208445.html?tag=rn

Sprint customers suffer outage

http://news.com.com/2100-1033-208935.html?tag=rn

Net blackout hits some regions

http://news.com.com/2100-1033-279232.html?tag=rn

Four servers for Web fail briefly

http://www.computerworld.com/news/2000/story/0,11280,49040,00.html

Netcom suffers Bay Area outage

http://www.computerworld.com/news/1997/story/0,11280,20385,00.html

Software upgrade sparked AT&T outage

http://www.computerworld.com/news/1998/story/0,11280,18659,00.html

7.24 Browser Failures

Probably this is the most vulnerable piece of software in the entire e-commerce system because it acts as the window for information transaction and management for the entire system. A browser crash can be a failure of the browser software itself or can be a symptom of failure for some other component. Either way, when a browser fails and its window closes, it closes with it all possibilities of a successful e-commerce transaction. A user who finds that his/her browser crashes whenever he/she loads the site or loads the shopping cart, will possibly think twice before shopping again at your site.

Browser failures have been discussed in detail in various component failure categories such as system security, memory leaks, privacy and also various qualitative failure categories, such as Internationalizability or reliability.

Instead of repeating all those failure modes again, I have provided pointers to the failure modes. Also, refer to the bug examples listed under each of these categories as they contain examples of browser-related bugs.

Category: Incompatibility

Sub-category: Content-browsers incompatibility

Browser- plug-ins incompatibility

Category: Error messages category

Sub-category: Browser-incompatibility

Category: System security category

Sub-category: Browser-vulnerabilities

Category: Memory leaks category

Sub-category: Issues due to memory leaks in browsers

Chapter 8: List II: Qualitative Failures

8.1 Introduction

The previous section provides a list of potential failure modes for components. This section focuses mainly on qualitative failures, such as degrading performance, poor reliability or non-compliance to some legal act.

Expectations over the quality of a function are very subjective. For example, one organization’s assumptions of what constitutes a good performance level may be very different from another. E-commerce firm A’s expectations over the reliability of their shopping cart may also be very different from that of company ‘B.’ Hence determining the threshold for failure is very subjective and quantifying a quality attribute is even more difficult.

There are many ways an application may fail the quality criteria in a Web environment. One is that of narrowly focusing on meeting some quality requirements without considering the impact on others. A supporting example comes from Barbacci et al. “Replicating communication and computation to achieve dependability might conflict with performance requirements (e.g., not enough time).” (Barbacci, Klein, Longstaff, Weinstock, 1997).

Boehm’s illustrates this problem in “*Characteristics of Software Quality*” (Boehm 78), “The major problem is that many of the individual characteristics of quality are in conflict; added efficiency is often purchased at the price of portability, accuracy, understandability, and maintainability; added accuracy often conflicts with portability via dependence on word size; conciseness and conflict with legibility.

Users generally find it difficult to quantify their preferences in such conflict situations."

In this taxonomy, the risk categories for quality follow the classification of quality attributes as described under ISO 9126 (ISO 9126:1991). **ISO 9126** is the Software Product Evaluation Standard from the International Organization for Standardization. This international standard defines six characteristics that describe, with minimal overlap, software qua lity.

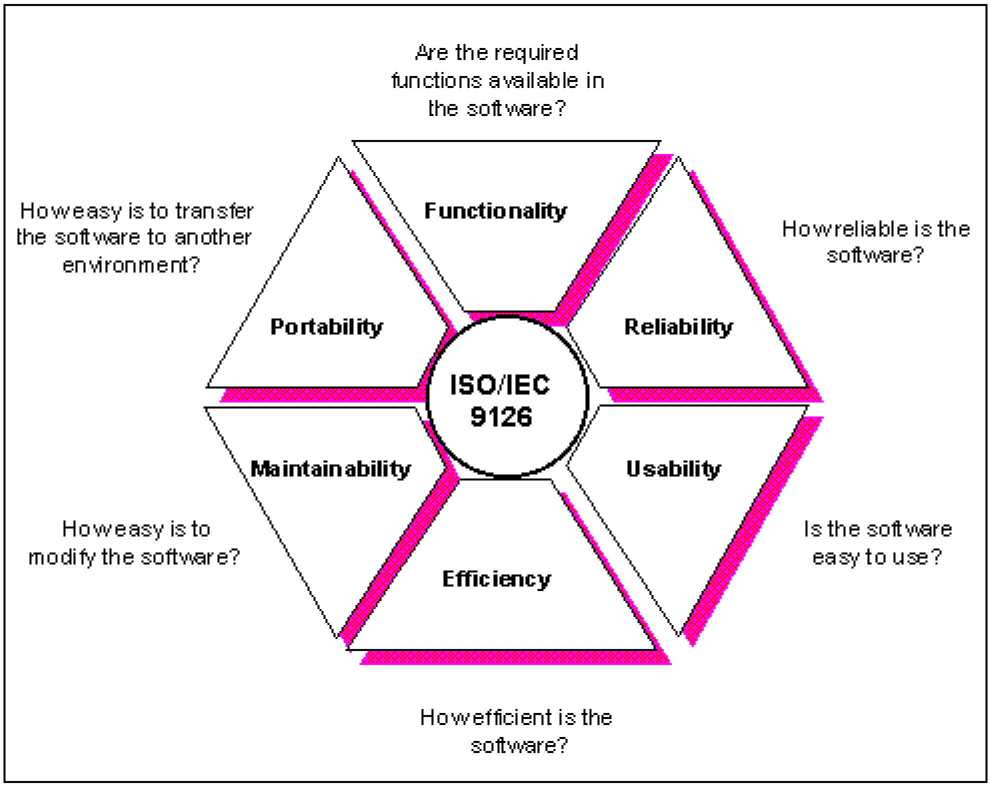

Are the required functions available in the software?
Functionality
How easy is to transfer the software to another environment?
Portability
How reliable is the software?
Reliability
ISO/IEC 9126
Maintainability
Usability
How easy is to modify the software?
Is the software easy to use?
Efficiency
How efficient is the software?

Fig: 8.1: ISO 9126

Source: http://www.cse.dcu.ie/essiscope/sm2/9126ref.html

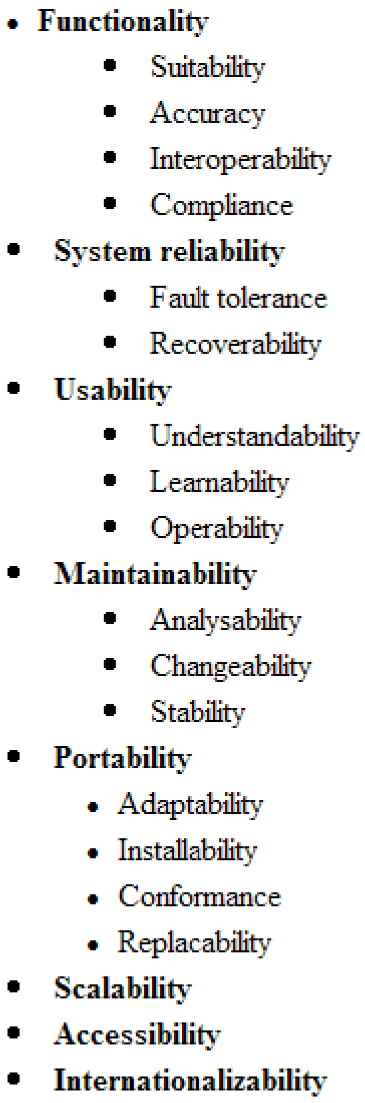
ISO 9126 classifies each of the above six categories into a few more sub-categories. In this taxonomy I have upgraded the sub-categories to top-level categories to give more breadth[6], dropped[7] some categories and added[8] some categories

In this thesis I have classified the qualitative attributes as follows (This list is slightly modified from ISO 9126):

[6] Since the goal of the taxonomy is to help a tester generate more ideas by giving him/her a large number of specific categories, I have tried to give maximum breadth to the classification with as many top-level categories as possible.

[7] Dropped cost, efficiency and broke security into three separate categories, document confidentiality, system security and client privacy.

[8] Added Internationalizability, Accessibility etc



- **Functionality**
  - Suitability
  - Accuracy
  - Interoperability
  - Compliance
- **System reliability**
  - Fault tolerance
  - Recoverability
- **Usability**
  - Understandability
  - Learnability
  - Operability
- **Maintainability**
  - Analysability
  - Changeability
  - Stability
- **Portability**
  - Adaptability
  - Installability
  - Conformance
  - Replacability
- **Scalability**
- **Accessibility**
- **Internationalizability**

Qualitative Categories

Functionality

Definition:

“A set of attributes that bear on the existence of a set of functions and their specified properties. The functions are those that satisfy stated or implied needs” (ISO 9126: 1991). ISO 9126 sub-divides Functionality into Suitability, Accuracy, Interoperability and Compliance.

8.2 Suitability

Definition:

“Attributes of software that bear on the presence and appropriateness of a set of functions for specified tasks.” (ISO 9126: 1991)

Failure Modes

Incomplete Functional Implementation

This sub-category lists all functional failures caused by an incomplete function implementation.

- Incomplete calculations: Failure to include tax costs (sales tax, state-specific taxes, value-added tax) in the total value field.
- Incomplete cart functionality: Update button not present. As a result, you can add/delete items but final order does not reflect the changes.

- Incomplete search: searches some categories but omits others due to incomplete implementation of indexing.
- Incomplete checkout process: no confirmation of the transaction.
- Unable to order again; order process works but re-order does not work since it is interpreted as a redundant order.

Functional Implementation Correctness

This sub-category lists issues that arise out of implementation problems. The function may be correct but the implementation may not be the most appropriate or correct.

- Requiring to enter a domestic zip code for international orders; order form shows error if the field is left empty
- Date format differs depending upon customer location; providing the wrong date format may lead to incorrect shipping date or order date
- Price field implemented as a pull-down option. This is not a suitable implementation of the functionality unless the customer has a choice of prices.
- Price field implemented as a hidden text embedded in the HTML file. Implementation not suitable due to security reasons since hidden files can be manipulated.
- Implementing product catalogs as a collection of large images. Large files may not be the most suitable or correct implementation since they increase load time.

Missing Functions

This sub-category holds failures that are caused due to missing functions, such as last name field missing, missing months in the date pull down field.

- Missing search field
- Search field present but search button missing
- Checkout button missing
- Unable to go back to the shopping catalog after adding an item
- No back button provided, browser's back button causes page expiration

Examples of Related Bugs and Issues

Study: Top firms' Web sites error-ridden
http://news.com.com/2100-1040-232257.html?tag=rn

8.3 Accuracy

Definition:

"Attributes of software that bear on the provision of right or agreed results or effects." (ISO 9126: 1991)

Failure Modes

- Inaccurate order value due to non-execution of update routine
- Inaccurate computations/calculations like shipping costs/taxation etcetera (Check categories on 'computation/calculation failures') and prices (Check category on 'Human errors').

- Inaccurate relevance ranking in search results
- Inaccurate results due to incorrect rounding of numerical values
- Low accuracy due to incorrect treatment of significant digits
- Inaccurate display of time zones across markets possibly due to incorrect configuration of timing functions. In the case of auctions, it may lead to an inaccurate begin/end time.
- Inaccuracies due to data loss over networks
- Inaccurate data due to deliberate data tampering
- Inaccuracies due to increase in transmission delay in time-dependent functions such as electronic trading where a late arrival of stock values (more than the allowed 20 minute delay) can lead to inaccurate display of stock positions.
- Non-execution/suspension of auto-refresh routines may cause inaccuracies in dynamically displayed real time content such as market news in financial web sites.
- Connecting to the wrong backend data feed or publishing from the wrong database can alter the accuracy of online content
- Websites may accidentally publish older data, older prices and closed sale prices that are not current or accurate.
- News websites have accidentally published confidential data such as election results earlier than the release time; shopping websites have published sale prices prior to actual sale day leading to users/shoppers being confused by perceived inaccurate data.
- Failure of regular automated site-wide update routines leaving the site with old and stale data.

Examples of Related Bugs and Issues

Inaccurate cart contents

http://findit.emp.state.or.us/faq/aol-cart.cfm

8.4 Interoperability

Definition:

“Attributes of software that bear on its ability to interact with specified systems.” (ISO 9126: 1991)

Failure Modes

8.4.1 Content-Browsers Incompatibility

- Errors when browsers run scripts that use newer versions of Javascript. An example is the failure in pre-Javascript 1.2 browsers when they try to run event handlers. Event handlers were introduced only from Javascript 1.2.
- Browsers are met with errors when they run scripts that use outdated features of older versions of Web scripts.
- Browser-biased scripting: using Navigator- or IE-specific features of script not supported by both browsers
- IE supports powerful DHTML scripting while Netscape does not.
- Navigator supports Javascript style sheets, while IE does not.
- Differences in browser's event model maybe a cause of errors

- Some browsers produce different results if a script's language attribute is modified, even when the browser supports both values of the language attribute.
- Incompatibility between data content and data type acceptable by functions
- Formatting Issues with Standard HTML Tags
- Blink will work in Netscape but not in IE
- Marquee will work in IE but not in Netscape
- Browser does not support the font; displays only in default font
- Unsupported attributes in supported elements
- Newer browsers recognize color by both its name and code but some older browsers do not recognize all color names
- Internet Explorer 4 introduced the all collection to allow access in script to any element on the page. Navigator has no such object, and trying to refer to it will create an error.
- The event object occurs in both Navigator 4 and IE4, but has different properties in each browser. It doesn’t exist at all in earlier browsers, or browsers from other manufacturers.

8.4.2 Browser- Plug-ins Incompatibility

- Incorrect plug-in
- Incorrect version of plug-in
- Crashes browser/system due to over consumption of RAM
- Plug-in incompatible with the security setting of the browser

- Plug-in incompatible with the data it is supposed to play/display
- Navigator has a plug-in collection for its navigator object that allows you to iterate through all the plug-ins loaded by the browser. Internet Explorer doesn't support these plug-ins, and the plug-in collection will always be empty, so your code will get no usable result.
- Interoperability issues due to corrupted plug-ins

8.4.3 General Incompatibility Issues

- The content does not display correctly in client machine due to resolution conflict and color incompatibility.
- Browser and client machine hardware incompatibilities (e.g., a 32 bit browser and a 16 bit machine)
- Browser is not compatible with underlying operating system (32 bit browser and a 16 bit OS)
- Operational problems due to protocol and the network type mismatch
- Web server is not compatible with host machine/OS or both

Related Examples of Bugs and Issues:

Fashion site thinks differently for Mac
http://news.com.com/2100-1040-232397.html?tag=rn

8.5 Compliance

Definition:

“Attributes of software that make the software adhere to application related standards or conventions or regulations in laws and similar prescriptions.” (ISO 9126: 1991)

Failure Modes

- Non-compliance with usability guidelines
- Non-compliance with federal accessibility guidelines
- Omission of basic information under Distance Selling Regulations
- Lack of compliance with Data Protection Act
- Standard terms of business omitted
- Inadequate site design leading to legal notices not being part of the contract
- Committing to supply goods when item is not in stock
- Misuse of others’ intellectual property rights, like using competitor’s name in meta tags
- Exposure to libel where the public can add text to your site e.g. on bulletin boards
- Non-compliance with restrictions on the type of conducting a particular type of online business such as online gambling, sale of restricted drugs without prescription, pornography, tobacco etcetera
- Non-compliance with security guidelines that lay rules with respect to encryption keys

- Non-compliance with licensing regulations that are needed to sell certain types of products
- Non-compliance with domain name registration rules and country level domain names registration rules
- Scroll bars fail to work in “Clickwrap” agreements, and user is unable to scroll and read all sections of the agreement inside the wrap text.
- Non-compliance towards digital signature authentication guidelines
- Non-compliance with privacy laws
- Failure to post a privacy disclosure statement

Europe: Data Privacy Directive

Malaysia: Personal Data Protection Bill

Japan: comply and seek “Privacy Mark”

Singapore: E-commerce code for the protection of personal information and communications of consumer of E-commerce

- Non-compliance with advertising standards, untruthful, false and misleading ads

Japan: Prevention of unreasonable premiums and misleading representations concerning products and services

Philippines: The Consumer Act

Thailand: Consumer Protection Act 1979

- Violation of consumer protection laws that include anti-fraud, advertising, usury, Installment contracts and rebate standards

Japan: Direct Sales Law, 1976

Japan: Law on Consumer Contracts, 2001

Korea: Basic Law on e-commerce, July 1999

Taiwan: Consumer Protection Law (CPL), 1994

New Zealand: eMarketing Standards Authority

- Violation of content control regulations imposed by many countries

Germany: Article 131 of German Penal code

Germany: Communications Services Act

France: Article R.645-2 of the French Criminal Code

China: “Tentative Provisions”, Feb 1996

China: Computer Information System Internet Security Administration – Jan 2000

Hong Kong: Control of Obscene and Indecent Articles Ordinance (COIAO)

Singapore: Internet Code of Practice

Australia: Broadcasting Services Amendment (Online Services) Act Jan-2000

New Zealand: Voluntary code of practice for Internet service providers

- Non-compliance with electronic interconnection standards for e-commerce/ IPC-2500 series

Examples of Related Bugs and Issues

UK Website Compliance Study 2002

http://www.dataprotection.gov.uk/dpr/dpdoc.nsf/ed1e7ff5aa6def30802566360045bf4d/8c51aabba9a8ad7d80256b45005ce102?OpenDocument

Legal Case: Violation of content control regulation
German CompuServe case
http://www.qlinks.net/comdocs/somm.htm

Legal Case: Germany took action against Amazon.com in 1999 for selling Nazi-related materials.
Germany urges global fight to stop Net hate speech
BN.com halts Hitler book in Germany
http://news.zdnet.co.uk/story/0,,t269-s2075790,00.html

Legal case: Leading Internet bookstores are under fire again for selling hate literature abroad, this time from Canada.
http://news.com.com/2100-1017-239248.html?legacy=cnet

German-owned BOL pulls Mein Kampf worldwide to prevent purchase by Germans
http://news.zdnet.co.uk/story/0,,t269-s2073216,00.html

French judge orders Yahoo!'s Nazi auction 'zoned' so French can't bid
http://www.freedomforum.org/templates/document.asp?documentID=3457

Consumer Prevention?

German shoppers are paying the price for an arcane and archaic collection of restrictions on retailers
http://www.time.com/time/magazine/intl/article/0,9171,1107990927-31666,00.html

System Reliability

8.6 Fault Tolerance

Definition:

"Attributes of software that bear on its ability to maintain a specified level of performance in cases of software faults or of infringement of its specified interface." (ISO 9126: 1991)

Failure Modes

- Loss of transaction when a single application server fails and no fail-over server exists to back up the transaction
- Memory loss resulting in loss of online shopping baskets, credit card information, and personalized page settings and no redundancy available
- Unable to retrieve submitted information from a corrupted client cookie (The Cookie is sometimes used to track submitted data until all the fields are completed and committed and in the case of fault, it is used to retrieve the previous state before fault.)

- Unable to retrieve data from a corrupted database (a database that holds the state of the process, submitted information, transactional details, etcetera)
- Unable to retrieve the Uniform Resource Locators (URLs) used to store the details of the state of the cart, section, and input parameters prior to the occurrence of fault
- Load balancer fails along with the application server. When a user hits the reload button on no response there is no re-routing of request to a new application server or a failover application server.
- Unable to create an instance of a component services object, which is used to replicate the shopping basket logic across a cluster of application servers. When one object is gone, we do not have another to handle the failure.
- Inability to implement clustered services due to disparate hardware and operating system platforms
- Unable to effectively mask single point failures in backend applications from users
- Inefficient methods and non-existent strategies to replicate user's transient state (e.g., the contents of a Web shopping cart or travel itinerary) or data
- Multiple disk failures in disk arrays
- Failures in redundant network interfaces and redundant power supplies
- Unable to identify and respond to all error/fault conditions
- Non-detection of fault conditions by scheduled self-validation tools.

8.7 Maturity

Definition:

“Attributes of software that bear on the frequency of failure by faults in the software.” (ISO 9126: 1991)

Failure Modes

- The shopping cart may have a lower MTBF (Mean time between failure) due to the use of buggy scripts with a large number of unfixable bugs and components with intrinsic faults that are difficult to replace.
- Some shopping carts are essentially made up of separate but inter-dependent functional engines such as a personalization engine, shipping engine and payment engine. A higher failure rate in any of the engines may lower the mean time between failures of the cart.
- Frequent loss of connectivity with backend applications per shopping session might boost the overall failure frequency.
- Unavailability of alternative paths with lesser bugs will continue to make users encounter failures during their shopping session in the same paths keeping the failure rate consistent and high.

Usability

8.8 Understandability

Definition:

“Attributes of software that bear on the users' effort for recognizing the logical concept and its applicability” (ISO 9126: 1991)

Failure Modes

- Look for unnecessary steps between item selection and checkout. The more clicks, the more confusion and the greater the probability that the customer will abandon the transaction.
- Do not link to any external site/page from the shopping cart page as this leads to the shopper getting confused/uninterested, causes shopping cart abandonment.
- Check if thumbnail photos of the items can be added to the shopping carts in addition to a text description. This reassures the customer that the right item has been added to the shopping cart.
- Presence of standard "credit card" images on the UI adds trust psychologically on the site's security. Check the shopping cart for images or text that might cause mistrust in the user.
- Check if the UI provides functionality for discounts and coupons. Provide separate field in the UI to display discounts due to coupons as it helps user note the discounts better.
- Provide separate columns to display "total" bill as the user adds items to the cart.

- Too much information to type into the cart; avoid this common problem

8.9 Learnability

Definition:

“Attributes of software that bear on the users' effort for learning its application (for example, operation control, input, output)” (ISO 9126: 1991)

Failure Modes

- Try not using pop-up window based shopping carts because if the user clicks elsewhere in the main window, the pop-up is sent "behind" the main window.
- Provide "remove item" or "add item" buttons instead of asking the user to change "item quantity number" as it is easier and more error free.
- Check if the “number of items” in the cart is displayed. Users prefer carts that show the current data and state, like how many items are in the cart? What is the total?
- Check if the "Continue Shopping" and "Proceed to Checkout" buttons are visible.
- Do not limit the features of the shopping cart--keep it flexible.
- Cart is too hard to use. Solution: reduce functional complexity.
- Check for Hi-Tech whiz creations like flash display of catalog and constantly flashing blue lights in a shopping cart because it may reduce the usability of the cart. A classic example of a site that got booed away due to its technical gimmickry was www.boo.com.

- Not sticking to known paths in navigability and sequence of shopping decreases the usability of the shopping cart. Check for odd sequencing issues like re-sequencing shipping costs after the user has been billed and charged. This will confuse the customer about whether the purchase was executed or not!
- When new functionality is added to the shopping cart, check if it is user understandable, otherwise provide help.
- Check for odd naming of known metaphors.

8.10 Operability

Definition:

“Attributes of software that bear on the users' effort for operation and operation control” (ISO 9126: 1991)

Failure Modes

- Test shopping carts with pop-up/ad eliminating software turned ON. Pop-up shopping carts may not work if the pop-up eliminator is ON.
- Check if Pop-up shopping carts have sufficient "real-estate" space when the user adds more items.
- Look for items that have not been linked back to the “item”/catalog page.
- Check if the shopper is able to navigate back to shopping process, after "adding" or "removing" items.

- Check if it is possible to add additional items directly from the cart page, instead of going back to browsed pages. This improves functionality and enhances usability.
- If providing detailed info on products to users, test if you are able to return back to the shopping cart from the detailed page and check if the state of the shopping cart is maintained.
- Try enhancing the usability by providing an auto-update cart facility after user has added/removed item.
- Check for appropriate positioning of buttons. Place "Continue Shopping" on the left and "Checkout" button on the right as users perceive it analogous to "back" and "going forward" respectively.
- Check if the user is conveyed the information of order placement. Warn the customer when the transaction becomes final; do not surprise them by abruptly billing their contents.
- Check forms against data requirement. Collect only essential information about the user that is absolutely a must for completing the deal, unnecessary questions and making optional questions compulsory makes the user experience bad.
- Check for plug-ins or media files that are not common in any general browser software, and recommend not using them. Expecting users to download software to shop at your site is high handedness! This may cost you heavily in terms of loss of customers to other competitors.
- Provide the user with the functionality to choose the mode of shipment. Check for fixed default radio buttons, non-flexible shipping options, and erratic placement of multiple selection checkboxes

- Check if shipping can be calculated before checkout. Shoppers prefer getting an idea of the total cost of the item.

Examples of Related Bugs and Issues

I hate waiting

http://www.zdnet.com/anchordesk/talkback/talkback_229493.html

You forgot 'overcomplicated technology'!

http://www.zdnet.com/anchordesk/talkback/talkback_229472.html

Dell Computer's site has got to be the worst for hiding charges

http://www.zdnet.com/anchordesk/talkback/talkback_229509.html

Last minute surprises make me crazy too

http://www.zdnet.com/anchordesk/talkback/talkback_229513.html

Cookie crumbled

http://www.zdnet.com/anchordesk/talkback/talkback_229516.html

Convert browsers to buyers (article)

http://builder.cnet.com/webbuilding/pages/Graphics/Ecommerce/index.html

E-commerce and usability

http://wdvl.internet.com/Authoring/Design/Basics/ecom1.html

Two usability bug examples: Usability testing

http://www.smartisans.com/usability_testing.htm

Maintainability

Definition:

“The ease with which a software system or component can be modified to correct faults, improve performance, or other attributes, or adapt to a changed environment” (IEEE 90)

ISO 9126 sub-divides Maintainability into Analyzability, Stability and Changeability

8.11 Analyzability

Definition:

“Attributes of software that bear on the effort needed for diagnosis of deficiencies or causes of failures, or for identification of parts to be modified” (ISO 9126: 1991)

In the event of a failure in the e-commerce system, the first place that is checked for information regarding the cause is the system log file. In some other cases the vendors may provide diagnostic tools that can be used to analyze and ascertain the cause of a failure. Hence any failure or problems in the system logs or detection tools may affect the analyzability of the e-commerce system.

Failure Modes

- Unable to analyze parts of programs independently due to lack of encapsulation and lack of protection for individual functions
- Incomplete/Insufficient information present in the system logs

- Improperly configured system logs: failure to collect vital information regarding cause of failure
- Loss of information due to corrupted log file/ Loss of log file along with the system failure
- Lack of vendor supplied diagnostic tools and methods for diagnosis of failures and detection of faults.
- Undocumented process/data relationship.

8.12 Changeability

Definition:

“Attributes of software that bear on the risk of unexpected effect of modifications.” (ISO 9126: 1991)

In an e-commerce system “requests can reflect changes in functions, platform, or operating environment. A request could also modify how the system achieves its quality attributes. These changes can involve the source code or data files. They can be made at compile time or run time” (Bass, Klein, Bachmann, 2001). The ability to add, remove and modify existing components and functionality also fall under changeability.

Failure Modes

8.12.1 Unable To Add New Functionality

- Unable to add real-time credit card validation to shopping cart
- Unable to add live shopping assistance

- Unable to add “last viewed items”
- Unable to add a new payment processor
- Unable to add new credit cards
- Unable to add store locator
-

8.12.2 Unable to Modify Existing Functionality

- Unable to add “Advanced Search” to existing basic search.
- Unable to add modify existing database fields/rows/columns.
- Unable to extend the ability of a cart to accept more than one option, say on shipping area conditions.
- Unable to modify validation processes, for example unable to upgrade client side validation of email addresses to server side validation.
- Unable to customize functionalities like Item list, Extra Item, automatic discounts etc.

8.12.3 Unable to Delete Existing Functionality

- Unable to delete a field since the field affects associated references in another table.
- Unable to de-link store administrator functions/tabs from screen modified for general client/customer due to embedded references to common data sources.

8.12.4 Change in OS, Middleware, and Hardware Due to Change in Input Hardware/Output Hardware.

- Touch screen monitor throws invalid device error when swapped with regular keyboard system due to calibration problems, driver incompatibility with existing OS.
- Screens of handheld devices unable to scale down shopping cart images.

8.12.5 General Changeability Issues

- System unable to work in changed environment with new systems that were not previously a part of the network/ congregation.
- Unable to respond to change from static response to new requests to dynamic response to new requests.
- Inability to react to changes in terms of sectional load/ number of users/ peak seasonal fluctuations.
- Unable to respond to needs of change in other quality attributes like change in usability, change in system reliability etc
- Unable to implement strategies that implement changeability
- Unable to introduce "indirection" which is a way of inserting mediators between components to control direct knowledge of each other. Indirection allows you to add additional components without the knowledge of the other.
- Unable to implement "separation", separation of data from function. That way, changing functions is data-independent

8.13 Stability

Definition:

“Attributes of software that bear on the effort needed for validating the modified software.” (ISO 9126: 1991)

Failure Modes

- Opening any web page for modification may unexpectedly reset its access rights, thus making a page go unavailable.
- Addition of new items to form elements or Addition of new form elements to the user interface in an unplanned manner may make the User Interface unstable.
- Modifications that lead you to shopping cart states that do not permit you to exit/back-track.
- Shopping cart loses the session state and it is unable to retrieve data on reload.
- Change in data flow or navigational flow, may result in failure to show activity or fail to load data.
- Shopping cart problems due to unstable software build.
- Incompatibility issues between shopping cart and browser that makes the browser unstable when the cart loads.
- Changes within the script of the shopping cart may cause changes in the shopping cart behavior. Look for unstable code, branching problems, control violations within the code.

- Change in back end connectivity, unstable network/ database configurations.
- In search functions, search mechanisms often change their ranking and optimizing algorithms and hence the results maybe tend to become unstable and irrelevant.
- Change in levels of submission volumes may cause the search function to vary in quality of retrieved results.
- Instability induced into the indexing server or index database due to fast changing dynamic data may result in deteriorating performance in the search functionality.

Related Examples of Bugs and Issues

eBay site crashes again

http://money.cnn.com/1999/08/06/technology/ebay/

Why searches fail

http://www.searchtools.com/info/whysearchesfail.html

External risks to system availability

http://www.oag.state.ny.us/investors/1999_online_brokers/points_reference.html#154

Portability

Definition

"The ease with which a system or component can be transferred from one hardware or software environment to another" [IEEE 1990]

ISO 9126 sub-categorizes Portability into Adaptability, Installability, Conformance and Replacability.

8.14 Adaptability

Definition:

"Attributes of software that bear on the opportunity for its adaptation to different specified environments without applying other actions or means than those provided for this purpose for the software considered." (ISO 9126: 1991)

Sound confusing?

Nary Subramaniam and Lawrence Chung define and provide a very good example for software adaptability in their paper "Metrics for Software Adaptability" (Subramaniam and Chung), "An adaptable software system can tolerate changes in its environment without external intervention. For example, a dual-mode cell phone can find out by itself if any one of the two wireless standards it supports is available at its current location and if so it starts using that standard." This adaption to the available standard happens without the external intervention of any action.

Adaptability of an e-commerce system is determined by the adaptability of its components

Failure Modes

8.14.1 Inadaptability of Components with Other Components/systems

This sub-category lists failures that occur due to the inadaptability of system components with other system components or with other external systems.

- Payment systems fail to adapt to foreign markets and foreign accounts. For example the payment system works well with US market but fails when used by the European market.
- Shopping cart fails to integrate data from disparate sources.
- The site works well with static content but unable to adapt when upgraded to handle dynamic content.
- Shopping cart handles off-line processing well but unable to adapt to a real-time processing system.
- The site is trouble free under normal traffic but fails to adapt to sudden loads (spiked loads) or peak traffic.
- Lack of modularity in code design, rigid style sheets and control flow reduces the degree of adaptability in shopping carts.
- The shopping cart functions works well in the regular mode (HTML mode) but fail to adapt in a special mode (for ex. Flash mode)

8.14.2 Inadaptability of Systems with People (Users)

This sub-category lists failures that occur due to the inability of the system to adapt to the preferences of the user.

- Shopping cart behaves normally under default settings but fails to adapt to custom settings that support personalization or user-customization. Basically the shopping cart engine fails to interface well with the customization engine.
- E-commerce site design is well adapted for younger user groups but fails to adapt to the usability requirements of older user groups.

8.14.3 Process Inadaptability

This sub-category lists failures that occur when the system fails to adapt to new processes

- Shopping cart functions work well under normal checkout process but produces error when using the Express checkout. Basically the shopping cart fails to adapt to new or alternative checkout processes.
- Shopping cart produces calculation errors when discount coupons are added to normal checkout calculations.

8.14.4 Delivery and Operations Inadaptability

This sub-category lists the failures that occur when the cart fails to adapt to new modes of delivery

- The cart delivery system functions in two modes, one that allows the user to download the product (commercial software, documents, ebooks, music, etcetera) directly or specify a shipping address for a physical product (CD-ROM, paper version of the document, etcetera). The cart's internal systems work well in one mode but fail to work in the other.

8.14.5 Business Logic Inadaptability

This sub-category lists the failures that occur when the e-commerce system is unable to adapt to new business rules and logic.

- The shopping cart works well when operated in a horizontal portal but fails to adapt to the specificities when operated on a vertical portal
- The shopping cart fails to adapt to the new changes (functional changes, changes to the system internals, or any overall change in the choice of product inventory) placed on it due to new business policies based on consumer demand

8.15 Installability

Definition:

“Attributes of software that bear on the effort needed to install the software in a specified environment” (ISO 9126: 1991)

In e-commerce sites, the activity of “installation” takes place at three places:

The “installation” of the shopping cart and other necessary supporting components on the server

The “installation” of required plug-ins, media players, browser support, JVM, etc that are required to run the shopping cart on the client machine

In the case of a commercial software-download site, like real.com that sells software online, it is required to download and install some required executables (download wizards) before the actual software is downloaded.

Hence an e-commerce site can fail due to uninstall ability at any of the above three areas. Depending upon the function and design of the e-commerce site, there can be even more areas where software/files must be installed for the site to work.

Failure Modes

8.15.1 Installation Failures Due to CD-ROM Failures

This sub-category lists failures that occur when the installation is aborted or stalled due to a CD-Rom failure.

This occurs at places where the installation takes place from a CD-ROM. This failure may occur on the server-side where a shopping cart can be installed using a CD-ROM. It can also occur on a client machine where any supporting component, such as JVM, Plug-ins or browser service pack, is again being installed from a CD-ROM.

- CD-ROM physically corrupt/ damaged
- CD-ROM autorun.exe file absent
- CD-ROM content incomplete/missing files/incorrect versions of files present
- CD-ROM drive breaks down
- CD-ROM drivers unavailable, corrupted or incorrect

8.15.2 Internet Download Failure

This sub-category contains failures that occur when the installation file is being downloaded from the Website.

- Unable to download and install the plug-in required for viewing the shopping cart, because the install screen for the plug-in breaks the browser
- The download program that manages the file transfers crashes mid-way during the download
- Unable to download the installation file because it is large and not enough bandwidth to download it
- Client is unable to establish connection with the download server
- Incomplete file download
- Unable to uncompress the downloaded file
- The downloaded file is unable to integrate with the browser due to version conflict; downloaded installation file incompatible with the current browser version

8.15.3 Auto-Installation Failure

This sub-category lists failures that affect the installability when the auto-installation routines fail to initialize.

- The absence of a required file triggers the auto-installation routine, which initializes and runs the installation program. The file may be absent and the auto-installation routine fails to detect the absence.
- The auto-installation routine has been set to look for the presence or absence of a file of type A when it is actually supposed to look for a file of type B.
- The auto-installation routine runs and tries to install even when the required file/ plug-in is already present.

- Installs the wrong file
- Looks for the wrong file extension
- Has an incompatibility problem with the browser type or browser version

8.15.4 Installation Wizard Errors

This sub-category holds errors that cause a wizard to fail and cause other installability problems.

- Installation wizard fails to start/initialize
- Installs the shopping cart in the wrong directory
- Looks for other required files in the wrong pre-coded directory and does not allow the user to change directory options either
- Crashes due to irresolvable memory reference (Commonly this error throws a similar error message like what IE throws before it crashes.)
- Causes memory leaks

8.15.5 Installation Help Errors

This sub-category discusses possible issues such as delay in installations that may be caused due to a poor help feature. Though the help feature may not crash the entire installation routine, a poor help feature certainly causes problems in the form of delay due to non-resolution of critical problems.

- A confusing help feature causes installation delay.
- Help screen stalls; while the system continues to indefinitely prepare the help feature for first use

- Help feature contains incomplete details; incorrect details or sometimes-important information may also be missing
- Outdated help feature imported from earlier version of the shopping cart software. It may be irrelevant for the present version
- Help feature not included with product at all

8.15.6 Un-installation Failures

This category addresses un-installation failures that hamper installability.

- Un-installation of some other program removes certain files that are shared by the shopping cart program, causing the shopping cart to fail.
- Un-installation routine accidentally deletes the entire folder or corrupts the entire disk track/sector.
- Even after the un-install routine has been run, some pieces of the removed program still stick around causing problems to other programs.
- Unable to remove all dependencies
- The un-installation of an unwanted program forces the administrator to re-install the entire shopping cart.

8.15.7 General Installation Failures

This category holds some general errors that are commonly faced during installs.

- Unable to re-install over existing version
- Failure to get back to default installation option
- Missing elements in the custom installation option

- Complicated installations: Too many parameters to be looked up, manually locating setup files
- Installation failure due to absence of some OS-dependent files in the installation package

Examples of Related Bugs and Issues

Vulnerability in HP software installation programs.
http://www.cert.org/advisories/CA-1996-27.html

Oracle application server Web cache installation file permission error lets local users obtain elevated privileges.
http://securitytracker.com/alerts/2001/Dec/1003072.html

8.16 Conformance

Definition:

“Attributes of software that make the software adhere to standards or conventions relating to portability” (ISO 9126: 1991)

Failure Modes

- Non-conformance with W3C HTML standards and recommendations for HTML documents and HTML user agents (browsers)
- Non-conformance with XML-e-commerce standards (XML, DOM, XSLT, XSL-FO, Schema)
- Non-conformance to EDI (Electronic Data Interchange) standards

- Non-conformance to standards defined under Health Insurance Portability & Accountability Act (HIPAA) for e-commerce by health care related e-commerce sites
- Non-conformance to software interface standards defined by IEEE, ANSI, ISO-IEC and some military standards

8.17 Replaceability

Definition:

"Attributes of software that bear on the opportunity and effort of using it in the place of specified other software in the environment of that software "
(ISO 9126: 1991)

Failure Modes

- Unable to replace the pricing engine with a dynamic pricing engine within the price system architecture
- Unable to replace fixed models for cost calculations with more real time and dynamic cost calculation models
- Unable to replace shipping cost model, which uses one specific product property of the product object (e.g., "weight" or "distance") with a flexible shipping cost model that considers multiple product properties
- Unable to replace the classical offline payment systems with other new online payment and transaction systems that use encryption and SSL
- Problems when trying to replace services/components/content that were provided by third party providers with In-house services/components/content and vice-versa

- The personalization engine is unable to replace different application objects with equivalent objects to suit the user preferences.
- Unable to replace backend connector engines
- Unable to replace faulty hardware
- Unable to replace modules that have multiple dependencies
- Unable to replace components without pulling the site offline
- Unable to replace component A with component B due to functional incompatibilities, operational incompatibilities or technical unsuitability
- Replacement of component A with component B causes component C to fail

8.18 Scalability

Definition:

"The ease with which a system or component can be modified to fit the problem area" [IEEE 1990]

Failure Modes

8.18.1 Vertical Partitioning

"Vertical partitioning adds an additional layer to an application" (Skinner)

- Additional time delays due to the new processing layer placed ahead of a component
- Additional and unwanted maintenance overheads due to the new additional layer

- Queuing and De-Queuing faults may occur in implementations where a queue is added as a processing layer to scale up for the excessive load on some component
- Increase in design complexity due to addition of new layers
- Any failure in the additional layer may propagate through the base layers causing further failures.

8.18.2 Vertical Scaling

“Vertical scaling throws additional hardware at the application environment” (Skinner)

- Adding more hardware to an existing problematic piece may continue to give problems and add to cost of maintenance.
- An inability to improve performance once maximum scalability limit is achieved. The scalability graph begins to level up when the maximum limit of vertical scaling is reached.
- Performance problems may overrun the capacity available to deal with them.
- E-commerce architectures with poor layering may fail to scale up. Adding additional resources for each layer is easier and less expensive than adding resources to the whole structure.

8.18.3 Horizontal Partitioning

“Horizontal partitioning breaks a single logical component on a single server into several logical components on several servers.” (Skinner)

- Costly overheads due to the division of resources that have dependencies

- Re-architecture of existing applications to create new logical components creates the risk of failure due to logical incompatibility, protocol mismatch, illogical data communication levels etcetera.
- Creation of new physical components may increase performance overheads and cost overheads.
- Possibility of administrative human error in a non-automated environment due to increased handling of monitoring and backup processes

8.18.4 Horizontal Scaling

“Horizontal scaling is the process of moving a single component into a "farm" of identical components” (Skinner)

- Failures in components that hold state information

Examples of Related Bugs and Issues

“A snippet that enunciates a horizontal scaling problem”

(Skinner)

Problem:

“Anyone who placed an order and subsequently went back to update the order could frequently not find that order. Also, when a customer changed an address on the Web site and later returned to place an order, the order was often associated with the old address.

Cause:

“A technician explained that requests made by the Web site were randomly distributed across the six database servers. When a visitor returned to the site to view an order, there was only a one in six chance that the visitor's request would be routed to the correct database server.

The basic problem was that when a single database server was queried about an order, it needed (but did not have) the information held on the other database servers. Because of this, the database was not a good candidate for horizontal scaling.” (Skinner)

Outages plague eBay again
http://news.com.com/2100-1017-227811.html?tag=bplst

Crashing success for the Web?
http://abcnews.go.com/sections/business/TheStreet/onlineretail991202.html

Encyclopaedia Britannica's new Web site crashes

The Chicago-based company apologized but said its computers were unable to handle the 12 million to 15 million "hits" it received Tuesday from computer users trying access www.britannica.com on its inaugural day.
http://www.infowar.com/p_and_s/99/p_n_s_102299e_j.shtml

CBS Web site crashes on "Survivor" debut
http://news.com.com/2100-1023-241307.html?legacy=cnet&tag=st.ne.ron.lthd.ni

8.19 Accessibility Risks in Shopping Carts

According to statistics provided at www.webaim.org “an estimated 20 percent of the population in the United States (40.8 million individuals) have some kind of disability, and 10 percent (27.3 million individuals) have severe disability. The 27.3 million individuals with severe disabilities are limited in the way that they can use the Internet”. W3C’s Web Accessibility Initiative (WAI) has produced the Web Content Accessibility Guidelines 1.0, which explain in detail how to make a Web site accessible for people with a variety of disabilities.

A list of risks that affect shopping carts in terms of accessibility is provided below. They have been categorized with respect to the different severe disabilities that affect Internet users and based on the classification system provided at www.webaim.org

Failure Modes

8.19.1 Visual Impairments

Blindness

(In General)

- If the cart catalog is categorized under headings and captions to denote product categories, shipping options, billing options, then you should test the content with a screen reader and check if the document's text or other text equivalents make sense. Test and verify that the change in categorization is understandable to a person who cannot see the headings or captions.

- If you are using style-sheets to render your shopping cart pages, check if it is possible for screen readers and non-CSS supporting browsers to render the cart page correctly.
- When dynamically updating shopping cart pages, the equivalents to dynamic content may not get updated when the products, rates, or prices are dynamically updated. Thus only non-updated data is available to impaired or disabled users.
- If you are using an image-mapped-shopping cart page, check for redundant links to every navigable section.
- In shopping cart product pages, where size tables, price tables, shipping tables, schedule tables are used, test if row and column headers are identified. Otherwise the screen reader will output a stream of non-distinguishable data.
- Also when testing tables that use two or more logical levels, check if alternative text exists to identify which cell is a header cell and which contains data.
- When frames or layers are used to create the browse catalog in the cart, check each frame for title. This will help identification and navigation when read by screen readers.
- Some of the tools that help disabled or impaired users do not support programmable objects such as applets or scripts. Check for alternative functionality in the shopping cart under a situation when the user has these options turned off
- Use of pop-up shopping carts, pop-up advertisements, pop-up alerts, sales pitches in new windows takes the control off the active window that's being

read. It confuses the reader by making the screen reading software alternate between the different windows.

- Navigation across the shopping cart pages should be consistent and straightforward. Look for looping navigation and random return-backs and other similar issues that cause the screen reader navigation problems.

Visual-Impairment: Text Equivalent

- Check for equivalent test for every non-text element in the product catalog of the shopping cart
- Check for issues where the text equivalent describes the graphic but not the content in the graphic image!
- Look out for an open or broken ALT tag; this disables the screen reader from reading the content inside the tag.
- If tables are used to format the appearance of equivalent text, test using a screen reader whether the content read from the table makes sense. Tables tend to confuse the screen reader and the screen readers tend to read the content in different cells in a haphazard way.
- Check if by error the alternative text equivalent is provided in a language different from the language of content. Apply this test to all sites that have multi-lingual international sites.
- Check for typos, spelling errors and word jumble in the alternative text. A screen reader does not have the ability or the intelligence to notice the errors and it reads out non-tangible words to the user.
- Check for unknown abbreviations, acronyms, and unfamiliar complex words and jargon describing non-text elements in the cart.

- Check if the items that have been added into the shopping cart have alternate text, which the screen reader can read out so that the user can verify that the item just purchased, has been added.
- Check if the images of credit cards that are displayed at checkout have alternative text. Otherwise, the user will not be able to make out what cards are accepted.
- Test shipping cost calculators, gift-wrapping cost calculators and other such user-aid tools for equivalent text support. Test each button and field for alternative text describing the functionality.
- In the billing and shipping section of a shopping cart, test the order of entry and order of tabs with a screen reader. Sometimes screen readers read tabs in the wrong order.
- Test multimedia presentations and alternative text for correctness of data and also check if they appear in sync.

Color-Blindness

Total Color Blindness

- Testers should test if an alternative text exists for text that conveys information by means of color. For example, if all items marked for sale are marked in ‘red’ or new items are marked in ‘blue,’ then equivalent alternatives should exist in regular black text.
- Testers should test for alternatives when color is used as the primary way to indicate an action, such as making a link turn purple from blue, when it is clicked. Similarly some shopping carts mark ‘visited’ categories by changing its color when a user clicks on it. This color change is to aid the shopper in keeping track of what they have seen and what they have not

seen. A person with color blindness may not notice the change in color as he has trouble identifying the colors. An alternative may be using an object like an asterisk or a cursor instead of color to identify change in state.

Color deficiencies

- People with a color deficiency can see some colors but some pairs of colors look the same to them. So, for example, foreground and background colors may appear the same. Look for difficult color combinations in catalog design, link identification, announcements, etcetera. Reds, greens, oranges and yellows are the most likely to cause problems. For color combinations, see http://www.webaim.org/intro/intro

Low Vision

- People with low vision impairment use screen enlarger software to increase readability of small text but the enlarger limits the visible area of the browser screen. Screen designs that communicate well at normal text size may be confusing and hard to use when viewed in an enlarged mode.
- If pages rely on scrolling, screen enlargers will yield pages that require more scrolling. If items on the screen are not appropriately grouped, the user will have to remember and correlate too many details that should be displayed together
- Another risk lies in the use of graphics with embedded text for product catalogs, because of an enlarged screen space, the images may get highly pixilated and embedded text may become hard to read.
- People with age-related visual impairments—such as macular degeneration, glaucoma and cataracts—prefer shopping cart pages and functionality that can be enlarged, scrolled and purely textual, mostly devoid of graphics.

8.19.2 Hearing Impairments

Deafness

Lack of visible textual support

- In sites where sounds signifies a buying process, purchase alerts, error message, instant messaging from a live representative, then testers should look for alternative visible textual support.
- Test for the absence of subtitles or other text.
- Subtitles or alternative synchronized text should supplement video catalogs, virtual demos of products, product information sessions and help videos for shopping.

Deaf-Blindness

- Test for special cases of double disability.
- Test for all risks that are applicable to blindness. Since the deaf-blind use screen readers that convert text into Braille (through a refreshable Braille device attached to the computer), you must test to confirm that there is text that is readable by the Braille reader. Sound as an alternative to visible text is workable for the blind but will not work for people with double disabilities, such as deafness and blindness.

8.19.3 Mobility Impairments

- Mobility impairments range from minor to major problems restricting voluntary muscle movements. Because of a lack of dexterity, the user may find it difficult to click on small (single letters and alphabets), product links and catalog navigation links. Similarly, look for small functional buttons or other tiny targets that must be hit precisely.

- Others using devices to access the keyboard, such as a mouth stick or a head wand have lower dexterity than regular users. Test auto refresh, time outs and auto-exits from secure billing areas and verify that the time available is sufficient for them.
- Users using devices such as a head wand have to shake their head 20 times to browse through 20 links. Test if the unnecessary links can be minimized, or offer short cuts to skim past groups of related links.
- Keyboard shortcuts and keyboard functionality is vital to users with mobility impairments; the risk is high that they will be neglected if too much mouse-centric functionality is added.

8.19.4 Cognitive impairments

Some of the better-known cognitive impairments are Downs Syndrome and Alzheimer's disease. The lesser-known cognitive impairments include reading and learning disorders. Individuals with cognitive impairments often benefit from graphics or icons that supplement the text, providing a monotonous small-sized ‘text only’ interface devoid of any meaningful graphics and animations may not benefit such users.

Seizure disorders

- Beware of flickering sales tickers, ad banners, notifications, alerts, or interactive messages that are provided in some shopping carts. These seemingly harmless gimmicks may be a serious risk to people with seizure disorders if their frequency of flickering is between 2 Hz to 55 Hz.

Related Internet Links

Western Australian Electronic Commerce Center, information on accessibility and usability
(http://www.ecommercecentre.online.wa.gov.au/matrix/acc.htm)

Section 508: The amended Rehabilitation Act that requires federal agencies to make their electronic and information technology accessible to people with disabilities.
(http://www.section508.gov)

8.20 Internationalizability

Definitions (from Guidelines for building multilingual Web sites):

Internationalization

"Internationalization is the preparation of a product so that it can be customized for particular locales efficiently."

Globalization

"Globalization is the design of a product so that it remains the same for all locales."

Localization

"Localization is the customization of a product for a particular locale."

Failure Modes

8.20.1 Issues due to insufficient support for international character encoding methods and a few other source code related issues at the design level.

- Localization problems due to the use of different code bases for different regions and languages
- Decreased source code reusability due to the use of conditional compilation for specific languages within the source code. Also test for errors that may surface when source code, which contains language specific compilation directives for one language, is reused for a different language. The code must be properly cleaned up before reuse.
- Errors due conflicts between programmer assignments and character encoding systems over the 8th bit which is used by some character encoding system for representing non-ascii characters.
- Some Asian languages pose certain challenges due to their multi-byte character size. It is common to see errors such as incorrectly incrementing a pointer in a character array having multi-byte characters and assuming that the new position points to a new character. Jumping to different positions in a byte array without checking for boundaries also causes confusion.
- Errors can arise due to the occurrence of partial characters where whole characters are expected. Partial characters are created when multi-byte characters are read into a fixed buffer and, due to insufficient space, half of a multi-byte character is left behind.

8.20.2 Issues Related to Locale Dependencies

- Failure to implement the correct time/date format
- Failure to sort the textual elements in accordance with the rules of the local language
- Improper representation/use of special symbols, bit streams specific to the language
- Incorrect usage of measurement systems or errors while converting from one system to another
- Incorrect tags and fonts in the web page body
- Failure to display the correct delivery dates or processing time with respect to the local date/time settings
- Insufficient space for translations in multi-lingual web pages
- Incorrect and Inconsistent text formatting for different languages
- Inconsistent navigation and website layout across websites
- Incorrect legal disclaimers across international websites
- URLs and links are inconsistent/ absent across other local (other language/country equivalent) sites
- Inability to customize active client components such as applets or java scripts that contain/display locale specific data

Examples of Related Bugs and Issues

I hate to be specifically bad-mouthing a particular name
http://www.zdnet.com/anchordesk/talkback/talkback_229490.html

Withholding information

http://www.zdnet.com/anchordesk/talkback/talkback_229524.html

References

Amland, S., 1999. *Risk Based Testing and Metrics.* Paper presented at the 16th International Conference on Testing Computer Software, Washington, D.C., USA.

Ammar, H.H., Nikzadeh, T. and Dugan, J.B., 1997. *Petri nets.* A methodology for risk assessment of functional specification of software systems using colored. Proceedings, Fourth International Software Metrics Symposium, pp. 108–117.

Anderson, James P. ,April 1980. *Computer Security Threat Monitoring and Surveillance.Technical Report Contract 79F296400*. James P. Anderson Co., Box 42 Fort Washington, PA 19034 (215) 646-4706,.

Aslam, Taimur, August 1995. *A Taxonomy of Security Faults in the UNIX Operating System*. Master's thesis, Purdue University.

Attanasio, C. R., Markstein, P. W., and Phillips, R. J. , 1976. *Penetrating an operating system: a study of VM/370 integrity*. *IBM System Journal*, 15(1):102-116.

Bach, J. (1999). *Heuristic Risk-Based Testing. Software Testing and Quality Engineering Magazine.*

Banerjee, N., 1995. *Utilization of FMEA concept in software lifecycle management*. Proceedings of Conference on Software Quality Management, pp. 219–230

Barbacci, M., Carriere, J., Kazman, R., Klein, M., Lipson, H., Longstaff, T., and Weinstock, C., 1997. *Steps in an Architecture Tradeoff Analysis Method: Quality Attribute Models and Analysis*, CMU/SEI-97-TR-29. Software Engineering Institute, Carnegie Mellon University.

Bass, Leonard J., Klein, Mark, and Bachmann, Felix. *Quality Attribute Design Primitives and the Attribute Driven Design Method*. PFE 2001: 169-186

Becker, J.C. and Flick, G., 1996. A practical approach to failure mode, effects and criticality analysis (FMECA) for computing systems. High–Assurance Systems Engineering Workshop, pp. 228–236.

Beizer, Boris, 1990. *Software Testing Techniques*. Van Nostrand Reinhold, 115 Fifth Avenue, New York, New York 10003, second edition.

Bennett, S., Myatt, M., Jolley, D., and Radalowicz. A., 1996. *Data Management for Surveys and Trials. A Practical Primer Using Epi Info.* Brixton Books.

Bishop, Matt, May 1995. *A Taxonomy of UNIX System and Network Vulnerabilities*, TechnicalReport CSE-95-10. The University of California, Davis.

Brinkley, Donald L. and Schell, Roger R, 1995. *What is there to Worry About? An Introduction to the Computer Security Problem. Information Security: An Integrated Collection of Essays*, chapter 1, pages 11-39. IEEE Computer Society Press

Boehm, B., 1976. Software Engineering: IEEE transactions on Computers, C-25. Reprinted in Bergland & Gordon (1979)

Boehm, B., Brown, J.R., Kaspar, H., Lipow, M., McLeod, G., and Merrit, M., 1978. *Characteristics of Software Quality*, North Holland.

Bouti, A., Kadi, D.A. and Lefrançois, P., 1998. An integrative functional approach for automated manufacturing systems modeling. Integrated Computer-Aided Engineering, 5(4), pp. 333–348.

Chemij, Wasel, 1994. *Parallel Computer Taxonomy,* MPhil, Aberystwyth University http://www.gigaflop.demon.co.uk/comp/chapt7.htm.

Chillarege, Ram, 1989. http://www.chillarege.com/odc/home.html

Coutinho, J.S., 1964. *Failure-Effect Analysis*. Trans. New York Academy of Sciences, pp.564–585.

de Klerk, L. Louis (Inobits Consulting Pty., Bender, Jason (MSNBC)), April 2000 version 1.0. Microsoft Enterprise Services White Paper *E-Commerce Technical Readiness*, Microsoft TechNet. http://www.microsoft.com/technet/treeview/default.asp?url=/technet/itsolutions/ecommerce/maintain/optimize/capplan.asp

DeLemos, R., Saeed, A. and Anderson, T., 1995. *Analyzing Safety Requirements for Process Control Systems.* IEEE Software, Vol. 12, No. 3, May 1995, pp. 42–52.

Dilley, C., May 14, 2001. *Known HTTP Proxy/Caching Problems*. http://www.wrec.org/Drafts/draft-ietf-wrec-known-prob-03.txt

Du, Wenliang and Mathur, A. P.
1997 *Categorization of Software Errors that led to Security Breaches*
1997 *21ST NATIONAL INFORMATION SYSTEMS SECURITY CONFERENCE, CRYSTAL CITY, VA, 1998*; Coast TR 97-09.

Edols, Liz. Taxonomies are what? http://www.freepint.com/issues/041001.htm#feature

Extreme Tech, cited Aug 2002. www.extremetech.com. Available from http://www.extremetech.com/article2/0,3973,12087,00.asp.

Fenelon, P. and McDermid, J.A., 1993. *An integrated tool set for software safety analysis.* The Journal of Systems and Software, 21, pp. 279–290

Flynnm M.J., December 1966. *Very High-Speed Computing Systems*, Proceeding of the IEEE, 54(12), p1901-1909.

Gerrard, Paul, Februry 2002. Risk Based Test Reporting, Stickyminds.com.

Goddard, P.L.

1993 *Validating the safety of embedded real-time control systems using FMEA*. 1993 Proceedings Annual Reliability and Maintainability Symposium, pp. 227-230.

1996 A combined analysis approach to assessing requirements for safety critical real-time control systems. 1996 Proceedings Annual Reliability and Maintain-ability Symposium, pp. 110–115.

2000 *Software FMEA techniques*. 2000 Proceedings Annual Reliability and Maintainability Symposium, pp. 118–123.

Guidelines for building multilingual Web sites, September 2000. P923, E. P. Multilingual Web sites: Best practice, guidelines and architectures.

Handler, W., 1982. *Innovative computer architecture - how to increase parallelism but not complexity*, p1-41, in *Parallel Processing Systems, An Advanced course*, Evans, D.J., ed, Cambridge University Press, Cambridge, 0-521-24366-1.

Hockney, R.W. and Jesshope, C.R., 1988. *Parallel Computers 2,* Adam Hilger/IOP Publishing, Bristol, 0-85274-812-4.

Howard, John D., April 1997. An Analysis of Security Incidents on the Internet. Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213 USA.

Institute of Electrical and Electronics Engineers, 1990. IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries. New York, NY.

ISO/IEC 9126-Information Technology, Software Product Evaluation, Quality, Characteristics and Guidelines for their Use, 1991.

Jayaram, N.D. and Morse, P.L.R., April 1997. *Network Security — A Taxonomic View*. In European Conference on Security and Detection. School of Computer Science, University of Westmister, UK, IEE, 28-30. Conference Publication No. 437.

Java Glossary, java.about.com [cited Jan 2003] Available from (http://java.about.com/library/glossary/bldef-webserver.htm)

Kaner, Falk, Nguyen. *Testing Computer Software* John Wiley 2nd edition, 1999

Karolak, Dale Walter, 1996. S*oftware Engineering Risk Management*, IEEE Computer Society Press.

Knight, Eric, March 2000. Computer Vulnerabilities. http://bitzi.com, eBook.

Knuth, D.E. *The Errors of T E X*. Software Practice and Experience, 19(7):607-685, 1989.

Krsul, Ivan Victor, May 1998. *Software Vulnerability Analysis*. Ph.D. dissertation, Purdue University. http://www.krsul.org.

Kumar, Sandeep and Spa, Eugene H., ord. 1995. *A Pattern Matching Model for Misuse Intrusion Detection*. *National Computer Security Conference*, 17:11–21, October 11–14 1994. Proceedings: Communicating our Discipline, Strategies for the Emerging Information Infrastructures; Baltimore, Maryland.

Landwehr, Carl E., Bull, Alan R., McDermott, John P., and Choi, William S, September 1994. *A Taxonomy of Computer Program Security Flaws*. *ACM Computing Surveys*, 26(3):211–254.

Linde, Richard R., 1975. *Operating System Penetration*. In *National Computer Conference*, pages 361-368, Santa Monica, California, May 19-22 1975. System Development Corporation.

Lindqvist, Ulf and Jonsson, Erland, 1997. How to Systematically Classify Computer Security Intrusions. In *IEEE Security and Privacy*, pages 154–163, Department of Computer Engineering, Chalmers University of Technology, SE-412 96. G¨otegorg, Sweden.

Lough, D. L., 2001. A Taxonomy of Computer Attacks with Applications to *Wireless Networks.* Virginia Polytechnic Institute and State University.

Luke, S.R., October 1995. *Failure mode, effects and criticality analysis (FMECA) for software*. 5th Fleet Maintenance Symposium, Virginia Beach, VA (USA), 24-25, pp. 731–735.

Lutz, R.R. and Woodhouse, R.M., 1999, *Bi-directional Analysis for Certification of Safety-Critical Software*. Proceedings, ISACC’99, International Software Assurance Certification Conference, Chantilly, VA, Feb. 28–Mar 2, 1999.

Maier, T., 1997. FMEA and FTA to support safe design of embedded software in safety–critical systems. Safety and Reliability of Software Based Systems. Twelfth Annual CSR Workshop, pp. 351-367.

Martin, P., 1999. *Managing Database Server Performance within an Electronic Commerce Framework*. http://www.cs.queensu.ca/home/cords/publications/m99.pdf

Marick, Brian, December 1990. A survey of software fault surveys. *Technical Report UIUCDCS-R-90-1651*, University of Illinois at Urbana-Champaign.

Meehan, Shubin, 1996. Navigation in Web Applications. http://www.user.com/webapps/webapps.htm

Neumann, Peter G. and Parker, Donn B., 1989. A Summary of Computer Misuse Techniques. In 12th National Computer Security Conference, pages 396-407, 1989.

Oracle Corporation, ch.32, Oracle9i Database Administration: Recover Databases.

Bibsey, R. and Hollingworth, D., 1978. *Protection analysis project final report*. Technical Report ISI-RR-78-13, Institue of Information Sciences, University of Southern California.

Parker, Donn B.

1989. COMPUTER CRIME Criminal Justice Resource Manual. U.S. Department of Justice National Institute of Justice *O.ce of Justice Programs*, August 1989. Prepared by SRI International under contract to Abt Associates for National Institute of Justice, U.S. Department of Justice.

1992 *Computer Security Reference Book*, chapter 34, Computer Crime, pages 437-476. CRC Press, K.M. Jackson and J. Hruskh, U.S. Associate.

Perry, Tekla S. and Wallich, Paul, May 1984. Can Computer Crime Be Stopped? *IEEE Spectrum*, 21(5):34-45.

Pries, K.H., 1998, *Failure mode & effects analysis in software development.* SAE Technical Paper Series No. 982816. Warren-dale, PA: Society of Automotive Engineers.

Richardson, Thomas Winfred, 2001. *The Development of a Database Taxonomy of Vulnerabilities to Support the Study of Denial of Service Attacks*. PhD thesis, Iowa State University.

Reifer, D.J., 1979. *Software Failure Modes and Effects Analysis.* IEEE Transactions on Reliability, R-28, 3, pp. 247-249.

Ristenbatt, Marlin P., 1988. Methodology for Network Communication Vulnerability Analysis. In MILCOM: 21st Century Military Communications — *What's Possible*, pages 0493-0499.

Ristord, L. and Esmenjaud, C., 2001. FMEA Performed on the SPINLINE3 Operational System Software as part of the TIHANGE 1 NIS Refurbishment *Safety Case*. CNRA/CNSI Workshop 2001–Licensing and Operating Experience of Computer Based I&C Systems. Ceské Budejovice–September 25–27, 2001.

Roberts-Witt, S. L., 1999. Practical taxonomies: hard-won wisdom for creating a workable knowledge classification system. http://www.phys.uni.torun.pl/~duch/ref/s-search/taxonomy/featureb1.htm

Saeed, A., de Lemos, R. and Anderson, T., 1995. On the safety analysis of requirements specifications for safety-critical software. ISA Transactions, 34(3), pp. 283–295.

Searchtools, www.searchtools.com, cited January 2003. Available from, http://www.searchtools.com/info/classifiers.html

Signor, M. C., 2000. The failure analysis matrix: A usable model for ranking solutions to failures in information systems. Dissertation Abstracts International, 61(11), 5971B. (UMI No. 9994475).

Skinner, J. *Extended Scalability on the Data Tier*. Retrieved, from the World Wide Web: http://www.microsoft.com/technet/itsolutions/ecommerce/maintain/optimize/scaledat.asp

Stålhane, T. and Wedde, K.J., 1998. *Modification of safety critical systems: An assessment of three approached.* Microprocessors and Microsystems, 21(10), pp. 611-619.

Stein, L.D., May 2000. Web Security: A step-by-step reference guide.

Straub, Jr., Detmar W. and Widom, Cathy Spatz, 1984. Deviancy by Bits and Bytes: Computer Abusers and Control Measures. In Proceedings of the 2nd IFIP International Conference on Computer Security.

Subramaniam, A.L. and Chung, N. *Metrics for Software Adaptability*. Dallas: University of Texas. http://www.utdallas.edu/~chung/ftp/sqm.pdf

WAI, W3C-WAI, May 5, 1999. Web Content Accessibility Guidelines 1.0.

Whatis, www.whatis.com, cited January 2003. Available from, http://whatis.techtarget.com/definition/0,,sid9_gci331416,00.html.