Failure Patterns: A New Way to Analyze Failures

Ingrid A. Buckley
Florida Atlantic University, Boca Raton, FL, USA, Buckley.Ingrid@gmail.com

Eduardo B. Fernandez
Florida Atlantic University, Boca Raton, FL, USA, ed@cse.fau.edu

ABSTRACT

We propose the concept of failure patterns as a way to describe how faults lead to system failures. We use this concept to describe a failure scenario which illustrates an incorrect status message in a system. A failure pattern describes how a fault becomes a failure indicating how the fault propagates through the system units until it produces a failure. The pattern explicitly shows how flaws in the system allow the propagation of faults. The information in failure patterns is useful to evaluate and design reliable systems. The pattern also shows how to stop the failure or mitigate its effects. The patterns are also useful to reconstruct how the failure happened, which may have forensic value.

Keywords: Failure, Reliability Patterns, Patterns, Reliability, Security Patterns

1. INTRODUCTION

Due to our increased dependence on critical services to conduct basic daily functions we need to avoid failures in systems. In the past, several incidents have occurred which have brought an increased awareness of the need to improve the reliability of critical services, applications, and systems. Between 1985 and 1987 six patients received too much radiation from the Therac-25 machine that resulted in serious health damage (Leveson and Turner 1993). Boeing aircraft have experienced failures which resulted in fatalities on several separate occasions (Boeing Commercial Airplanes, 2011). The Northeast and Midwestern United States and Ontario, Canada experienced a massive power blackout in 2003 due to a power surge in the electricity power grid system (NYSO, 2004). There is also the possibility of intentionally provoked failures. A promising approach to build reliable systems involves the use of patterns (Buckley and Fernandez, 2011b). A pattern is an encapsulated solution to a recurrent problem in a given context. Design patterns (Gamma, 1994) embody the experience and knowledge of many designers, and when properly catalogued, they provide a repository of solutions for useful problems. Initially used for improving code, patterns are starting to be used to build reliable and critical systems (Buckley and Fernandez, 2009), (Buckley and Fernandez, 2011), (Saridakis, 2002).

We introduce here the concept of failure patterns. Their function is to describe how a fault propagates throughout the system until it produces a failure, which system components are affected, and how to mitigate or stop the effects of the failure. Failure patterns complement reliability patterns (Buckley and Fernandez, 2012) by providing a way to show how failures occur, and how they can be avoided. They also help to classify failures, their affected components, and allow for the reconstruction of failures in the system. We adapted the template used by misuse patterns for security attacks to describe failure patterns (Fernandez, 2007).

Section 2 discusses failure patterns and their template. Section 3 describes a failure pattern that produces incorrect status messages in systems. Section 4 discusses related work, while section 5 presents some conclusions.

2. FAILURE PATTERNS

Faults are manifested as errors, which in turn are manifested as failures (Avizienis, 2004). A failure pattern describes how a failure is produced from a fault, identifies the components which are involved in the failure, the specific errors which allowed the failure to occur, and the effect of the failure on the system. If the failure was intentional (instead of produced by an accidental fault), it can also be described. The failure pattern also provides a solution to avoid this failure in the form of reliability and security patterns, as well as a way to store and analyze the information collected at each stage of the failure. Due to their dynamic descriptions, failure patterns allow countermeasures to be included to mitigate the identified failures.

The primary characteristics of a failure pattern can be summarized as:

- It shows how a fault is manifested in a given system until it produces a failure.
- It helps to identify countermeasures to avoid or mitigate failures.
- It allows the reconstruction of a failure scenario by seeing its effect in specific units.
- It provides a guide of application for software development to follow that helps to reduce failures.

We describe failure patterns using the POSA template, which has been used to describe security (Schumacher, 2006) and reliability patterns (Buckley and Fernandez, 2011). We have tailored this template to fit the purpose of failure patterns. We utilize a failure scenario to illustrate how failures can manifest and affect the system. The failure pattern builds on the template used for reliability patterns by adding two additional sections, namely forensics and countermeasures. The forensics section identifies failure effects to reconstruct how the failure happened. This is especially important for intentional failures, but it can also be useful for accidental failures.

We now enumerate each section of the failure pattern's template:

Name - The name of the pattern should correspond to the generic name given to the specific type of failure in standard failure repositories such as CFDR (USENIX, 2009).

Intent or thumbnail description - A short description of the intended purpose of the pattern, including which problem produced a given failure or how an attacker would induce a given failure.

Example of a specific problem where a fault produced a failure in an unprotected system.

Context -This section describes the general environment, including the conditions under which the failure may occur. These may include minimal countermeasures usually present in the system as well as typical errors of this kind of system.

Problem - Defines how the fault successfully propagates and produces the failure in an unprotected system. From an accidental or intentional perspective, the problem is how to find a way to produce a failure in the system. An additional problem occurs whenever a system is protected by some anti-failure mechanisms. The **forces** indicate what factors may be required in order to produce the failure and in what way; for example, which errors can allow its propagation. Also, which factors may obstruct or delay producing the failure.

Solution - This section describes how the fault propagation or error can cause a failure. UML class diagrams show the system before and during the failure. Sequence diagrams show how the fault propagates through the units of the system and becomes a failure. State or activity diagrams may add further detail.

Known Incidents - Specific incidents where this failure occurred. Details of past failures are useful to decide where to look for evidence and how to stop the failure.

Consequences - The benefits and drawbacks of a failure pattern from a reliability viewpoint. Is the effort and cost for stopping the failure commensurate with the results obtained? Which are the possible sources of failure?

Countermeasures - This is a new section compared to the templates for reliability and security patterns. It describes the dependability measures necessary in order to stop, mitigate, or trace this type of failure. This implies an enumeration of which reliability patterns are effective against this failure.

Where to look for causes - This section may includes the identification of which units allowed the propagation of the fault, and may contain relevant information to trace back the failure.

Related Patterns - Discusses other failure patterns with different objectives but performed in a similar way or with similar objectives but performed in a different way.

3. FAILURE PATTERN: INCORRECT STATUS MESSAGE IN SAFETY-CRITICAL SYSTEMS

Intent – Many systems can endanger humans if their indicators fail, e.g., a radiation treatment machine or an airplane inflight. The operator may not realize that some operation occurred or not and can act on a given indication thus provoking a failure.

Example

The Therac-25 machine was used at the Ontario Cancer Foundation to perform radiation treatment on a patient for carcinoma of the cervix (Leveson and Turner 1993). Figure 1 shows the general layout of the radiation room and the different units of the machine. The operator activated the machine but it shut down after five seconds displaying a "no dose" message indicating a treatment pause. The operator reactivated the machine and the same thing occurred. Since the machine did not show that a dose was delivered, the operator tried again to deliver the correct dose. However, the same thing occurred as in the first attempt. The operator attempted the same procedure four more times. After the treatment ended, the patient complained of a burning sensation in the treatment area. Several weeks after the treatment, the treatment area was excessively swollen, and the patient complained of hip pain and burning. The patient died from cancer; however, the overdose resulted in hip damage which could only have been corrected with a hip replacement.

Context - Health care facilities that use systems to perform critical procedures on patients. These systems have status indicators that show if a operation has been applied. An example was described above. Another example are aircraft that display indications to operators to make critical decisions during a flight. In general, this context includes any situation where an operator needs to make critical decisions in real time based on reading some indicators.

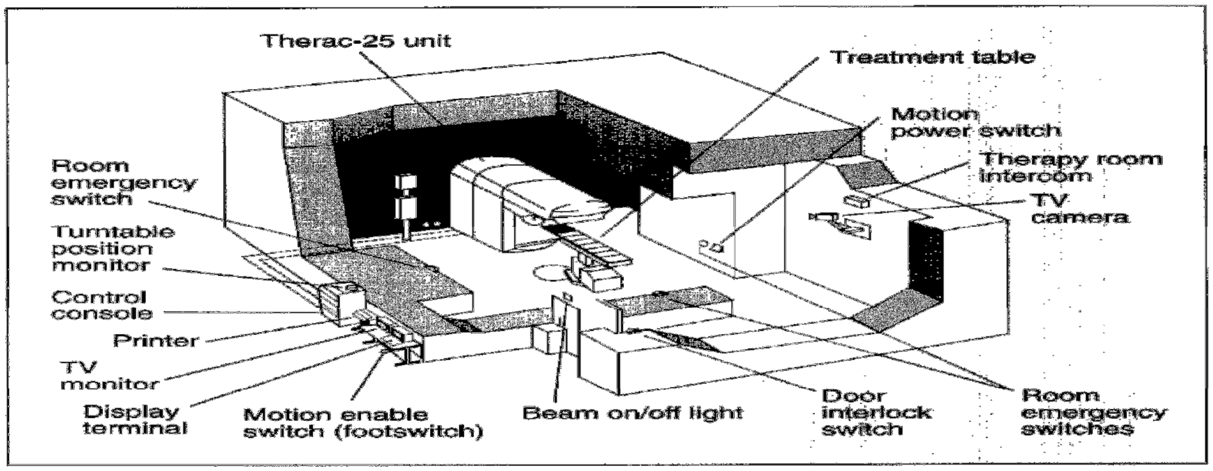

Figure1: Radiation Treatment Machine (Leveson and Turner 1993)

Problem - How erroneous operator message failures originate and propagate? They can result because of the following susceptibilities in typical machines:

- Incorrect design of the hardware and software components (Leveson and Turner 1993).
- Incorrect user manual, which does not describe in detail error codes (Leveson and Turner 1993),
- Faults in any component, e.g., an electronic display may have a broken light bulb (indicator).
- The user interface receives or produces an erroneous value.

Solution - A hardware or software fault in the functional or control unit may send erroneous status indications to the display. A fault in the display, in the interface, or in the functional unit, may produce a similar effect.

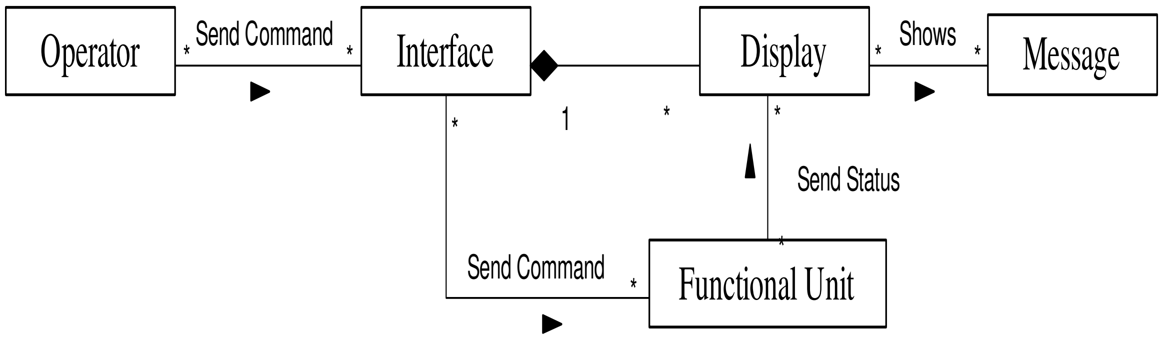

Figure 2: Class Diagram for Failure Pattern

Figure 2 illustrates the structure of the system. The **Operator** utilizes an **Interface** to send commands to the **Functional Unit**. The **Display** shows **Messages** that are returned when a particular operation is selected by the operator and also the status of the units.

Figure 3 shows how a fault propagates until it shows an erroneous indication because either the functional unit is faulty or the display is faulty.

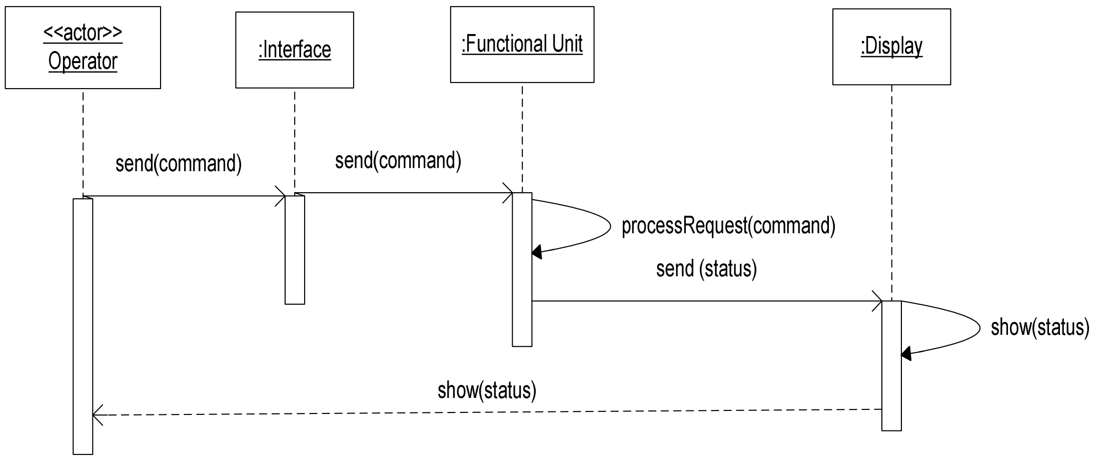

Figure 3: Sequence Diagram – Showing an Erroneous Status Display

Consequences - The advantages of this pattern are:

- Incorrect design or malfunction of the functional unit that carries out patient operations may produce a wrong status message in the system.
- Malfunction of the display could produce an erroneous indication.
- A wrong user manual could confuse and mislead the operator.

Possible sources to produce a failure:

- The functional unit, interface, and display can be single points of failure, and are the places to check to detect faults.

Countermeasures and Forensics - Figure 4 shows the countermeasures that were added to Figure 2, to stop or mitigate a failure. Stereotypes refer to the patterns applied to some units.

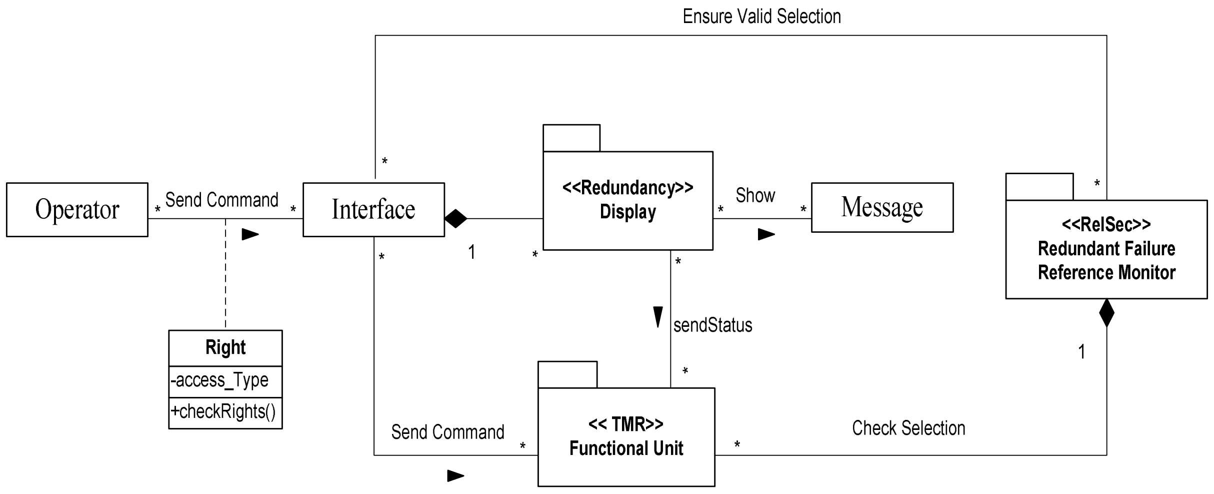

Figure 4. Class Diagram for Failure Pattern with failure

The **Operator** utilizes an **Interface** to manipulate the health care machine. The operator must be authorized before using the Interface; this authorization is defined by the **Right** association class. Through the **Interface**, the operator can select different operations. The display shows the **Message** that is produced when a particular operation is selected or from status indications. The **Display** has Redundant Display Indicators**.** The **Functional Unit** includes three functional units that conduct the same processing and vote on the correct output. The **Redundant Failure Reference Monitor** ensures that a selected operation matches the messages produced as output to the operator. This helps to ensure that the correct status is given for each operation selected by the operator.

Additionally, failures can be mitigated or stopped using the following suggestions:

- A failure which results in absence of service can be identified with an acknowledgement pattern to signal that some error has occurred in the system.
- A loss of data failure can be detected by the reference monitor, which validates the output from the components before it is exhibited on the display.
- The display indicators should be redundant so that messages can always be displayed at all times.
- Error messages should be simple and describe clearly what failure has occurred.
- Including fail safe states can help handle failures and warn operators so that some predefined course of action is always taken (Princeton University, 2007).
- Care should be taken to have clear user manuals.

Additionally the following forensic mechanisms are possible:

- Logs in the functional unit and other components, can provide a trail of activities that were completed and those that were in progress when a failure occurred.
- A failure analysis tool for hardware, and software related faults and errors can also provide additional information to identify flawed components (Wedyan et al., 2009).
- Fault injection techniques can be used to trace and reconstruct a particular failure (Hsueh et al., 1997).

Where to look for Evidence - Figure 2 shows the classes which are most susceptible to failures: the interface, functional unit, and display classes. We can check what information was sent to the interface and functional units if these classes keep a record of their message history.

Known Incidents

- The Therac-25 incident of 1993 (Leveson and Turner 1993), described in the example.
- A Spanair MD-82 jetliner crashed on take-off because the flaps had not been extended and the alarm for that condition had not sounded (CBS News, 2009).
- OREX is a radiology reader with built-in redundancy to counter failure (OREX, 2003).

Related Patterns - The solutions mentioned to mitigate the problem of erroneous messages have been used in other safety-critical systems to reduce failures.

- **Triple Modular Redundancy (TMR) Pattern** - An approach which uses three systems to perform a process and the result is processed by a voting system to produce a single output. If any one of the three systems fails, the other two systems can correct and mask the fault. If the voter fails then the system will fail (Buckley and Fernandez, 2012).

- **Reliable Security (RelSec) Pattern** - Applies reliability mechanisms to the functions that provide security in a system (Buckley et al., 2011).

- **Acknowledgment Pattern** – As indicated, it can check for an absence of events that may indicate a failure (Buckley and Fernandez, 2012).

4. RELATED WORK

Fault trees are used to make decisions based on the knowledge and experience of designers about a particular situation or from related situations. A fault model is created by looking at an undesired state which is analyzed in the context of its environment, looking at all reasonable ways in which the undesired state can occur (U.S Nuclear Regulatory Commission, 1981). A fault tree provides a graphic model of various parallel and sequential combinations of faults that will result in the occurrence of the predefined undesired events. Fault trees can become qualitative models if they use failure probabilities. Fault trees differ from failure patterns because they do not show the relationship of faults and errors to the system structure and components. They also do not show how faults propagate throughout the system.

Wu presented an approach using a compositional method for failure analysis of a system based on the software architecture of the system. The method used is based upon the use of Communicating Sequential Processes (CSP) as the failure modeling language and its associated tools for failure analysis. The failure behavior of each component is modeled in terms of failure propagation and generation. The result of such modeling is the combination of different possible failure flows from external failures or component internal failures to system-level failure (Wu and Kelly, 2005). This approach bears some similarity to failure patterns; however, it focuses on modeling failure in components to see their effects. Countermeasures are not provided to handle failures in the system; nor does it provide a guide to recreate a failure based on a particular scenario.

Fenelon presented the Failure Propagation and Transformation Notation (FPTN) approach which is used for safety analysis. Safety analysis is structured in response to system structure modules and failure behavior of each module is described in terms of failure flows, which can be classified into four forms of failure: propagation, transformation, handling, and generation (Fenelon, and McDermid, 1993). This approach provides generalized failure flows for each module, and differs from failure patterns in that, it does not provide countermeasures to handle failures in specific components. Also, it does not provide a guide to recreate a failure based on a particular scenario.

Bozzano et al. proposed the ESACS project which provides two alternatives of failure modeling: system model prototyping for safety – using predefined components enriched with failure modes to perform fast prototypes, or failure mode injection. Their method extends the system model by injecting failure modes (Bozzano et al., 2003). Leveson's STAMP has a number of causal analysis techniques proposed for accident modeling and analysis (Leveson, 2002).

The work presented here is analogous to the concept of misuse patterns proposed in (Fernandez et al., 2007), (Fernandez et al., 2009) to describe security attacks. Misuse patterns are described from the point a view of the attacker, including how the attack is performed, the context in which it is performed, and ways to stop the attack. The name "failure pattern" has been used in the context of "patterns of failure", analyzing how system units may fail but using ad hoc notations and methods; for example (Wolforth et al.) uses a special language to describe failures.

5. CONCLUSIONS

We have introduced the concept of failure patterns as a systematic description of the steps which lead to a failure as well as an indication of the countermeasures needed to avoid or mitigate those failures. We illustrated the use of the failure pattern with an example of a radiation machine to show failures, which is similar to those that appeared in the Therac-25. Failure patterns are complementary to dependability patterns which prevent failures. They are effective when used for building machines that carry out crucial functions. In future work, we intend to integrate this approach with our methodology to build reliable and critical systems (Buckley and Fernandez, 2011b). Additionally, we are building a catalog of failure patterns that can be used to analyze failures.

REFERENCES

Avizienis, A., Laprie, J. C., Randell B., and Landwehr, C. (2004). "Basic Concepts and Taxonomy of Dependable and Secure Computing", *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 1, pp. 11-33, Jan.-Mar. 2004, doi:10.1109/TDSC.2004.2.

Boeing Commercial Airplanes. (2011). "Statistical Summary of Commercial Jet Airplane Accidents Worldwide Operations 1959 - 2010", June 2011, http://www.boeing.com/news/techissues/pdf/statsum.pdf, (Last accessed: February 23, 2011).

Bozzano, M. et al.(2003)."ESACS: an integrated methodology for design and safety analysis of complex systems", Procs. of *ESREL 2003 - European Safety and Reliability conference* (ESREL 2003), Taylor & Francis Publisher.

Buckley, I. A, Fernandez, E.B., (2009) I.Buckley, E.B.Fernandez, Rossi,G. Rossi, and Sadjadi. M. (2009). "Web services reliability patterns", Proc. of the *21st International Conference on Software Engineering and Knowledge Engineering (SEKE'2009)*, pp. 4-9, Boston, July 1-3, 2009.

Buckley, I. A. and Fernandez, E.B.(2009)."Three patterns for fault tolerance", Procs. of the *OOPSLA MiniPLoP*, October 26, 2009. http://www.refactory.com/miniploppapers/FTPatts.pdf. (Last accessed: August 29, 2011)

Buckley, I. A. and Fernandez, E.B.(2011). "Enumerating software failures to build dependable distributed applications", Procs. of the *13th IEEE International High Assurance Systems Engineering Symposium (HASE 2011)*, Boca Raton, Nov. 10-12, 2011.

Buckley, I.A., Fernandez, E.B. and Larrondo-Petrie M. M. (2011). "Patterns Combining Reliability and Security", Procs. of *PATTERNS 2011: The Third International Conferences on Pervasive Patterns and Applications*, pp. 144-150, September 25-30, 2011, Rome, Italy.

Buckley, I. A. and Fernandez, E.B. (2012). "Dependability Patterns: A Survey", unpublished manuscript, Department of Computer & Electrical Engineering and Computer Science, Florida Atlantic University, 2012.

Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., and Stal, M.(1996). *Pattern-Oriented Software Architecture: A System of Patterns*, vol. 1, Wiley, 1996.

CBS News. (2009). "153 Killed In Madrid Plane Crash", http://www.cbsnews.com/stories/2008/08/20/world/main4365961.shtml, 03/23/ 2009. (Last Accessed: February 23, 2012).

CERT Coordination Center,Carnegie Mellon University. (2006). www.cert.org.

Fenelon, P,. and McDermid, J. (1993). "An integrated toolset for software safety analysis", *Journal of Systems and Software*, *21* (3), pp. 279-290, 1993.

Fernandez, E.B., Pelaez, J.C., and Larrondo-Petrie, M.M. (2007). "Attack patterns: A new forensic and design tool". Procs. *of the Third Annual IFIP WG 11.9 Int. Conf. on Digital Forensics*, Orlando, FL, Jan. 29-31, 2007, Chapter 24 in Advances in Digital Forensics III, P. Craiger and S. Shenoi (Eds.), Springer/IFIP, pp. 345-357.

Fernandez, E.B., Yoshioka, N., and Washizaki, H. (2009). "Modeling misuse patterns", Procs. of *the 4th Int. Workshop on Dependability Aspects of Data Warehousing and Mining*

Applications (DAWAM 2009), in conjunction with the 4th International Conference on Availability, Reliability, and Security (ARES 2009), March 16-19, 2009, Fukuoka, Japan.

Gamma, E., Helm, R., Johnson, R. and Vlissides, J. (1994). *Design patterns: elements of reusable object-oriented software*, Boston, Mass:Addison-Wesley, 1994.

Hsueh, M., Tsai, T.K., and Iyer R.K.(1997). "Fault Injection Techniques and Tools". Procs. of *IEEE Computer*, vol.30, no.4, 75-82, April, 1997.

Leveson N.G., and Turner C. S.(1993). "An Investigation of the Therac-25 Accidents", Computer. vol. 26, no.7, 18-41, July 1993, doi:10.1109/MC.1993.274940.

Leveson, N. (2002)."A Systems Model of Accidents". Procs. *of the 20th International Conference of the System Safety Society*, Unionville, U.S.A., 2002, International Systems Safety Society, pp. 476-486.

NYSO. (2004). "NYISO Interim Report August 14, 2003Blackout", January 2004 http://www.hks.harvard.edu/hepg/Papers/NYISO.blackout.report.8.Jan.04.pdf, (Last accessed: November 27, 2011).

OREX.(2003). "The OREX PcCR 1417". 2003. http://www.activexray.com/pdf/ OREX1417.PDF. (Last accessed: November 27, 2011).

Princeton University. (2007). Radiation Safety Guide SECTION 16: Requirements and Precautions for the Use of "Radiation Producing Machines and Devices", 2007, http://web.princeton.edu/sites/ehs/radsafeguide/rsg_sec_16.htm, (Last accessed on January 25, 2012).

Saridakis,T. (2002). "A system of patterns for fault tolerance". Procs. of *EuroPLoP* 2002, pp. 1- 30, http://hillside.net/europlop/HillsideEurope/Papers/EuroPLoP2002/2002_Saridakis_ASystem OfPatternsForFaultTolerance.pdf. (Last accessed: September 4, 2011).

Schumacher, M., Fernandez, E. B., Hybertson, D., Buschmann, F. and Sommerlad, P.(2006). *Security Patterns: Integrating security and systems engineering*, Wiley, 2006.

USENIX. (2009). "The computer failure data repository (CFDR)", February 15, 2009 http://cfdr.usenix.org/, (Last accessed: November 27, 2011).

U.S Nuclear Regulatory Commission. (1981). "Fault Tree Handbook", January 1981, http://www.nrc.gov/reading-rm/doc-collections/nuregs/staff/sr0492/sr0492.pdf (Last accessed: November 27, 2011).

Wedyan, F. Alrmuny, D and Bieman, J. M. (2009) ."The Effectiveness of Automated Static Analysis Tools for Fault Detection and Refactoring Prediction", Preprint of paper published in *International Conference on Software Testing, Verification, and Validation (ICST 2009),* pp 141-150, 2009.

Wolforth, I., Walker M., Grunske L., and Papadopoulos Y., "Generalizable safety annotations for specification of failure Patterns, *Softw. Pract. Exper.* 2010; vol. 40, pp.453–483

Wu, W. and Kelly, T. (2005). "Failure Modeling in Software Architecture Design for Safety", Procs. of *WADS '05 Proceedings of the 2005 workshop on Architecting dependable systems*, ACM, May 2005, St. Louis, MO, 17, doi:10.1145/1083217.1083222.