A Survey of Safety Analysis Techniques for Safety Critical Systems

Aftab Ali Haider and Aamer Nadeem

Abstract—**This paper is mainly focused on the study of the techniques available for the safety analysis of critical systems. It is never possible to build a completely safe system. There is a possibility to bring the behavior of these systems within acceptable limits. For safety evaluation of such systems both formal and informal techniques are available. Both techniques have their own prospects and consequences. Informal techniques are simpler to learn and easier to interpret and have more space for creativity and imagination of the analyst. Formal techniques due to their rigorousness ensure completeness. In this paper, we have analyzed both techniques after defining few parameters. Our study found it that formal techniques are better but usage of informal techniques can never be overlooked. Some approaches combine formal and informal techniques to reap the benefits of both. In some cases, informal techniques can be used as pre-requisite to narrow down the input of minimal critical set for formal techniques and reduce the effort required for formalization of the entire system.**

Index Terms—**Formal techniques, informal techniques, safety analysis, safety critical systems, fault trees.**

I. Introduction

A safety critical system is one whose malfunctioning may result in loss of human lives or some serious injury, severe damage or loss to some expensive and sensitive equipment or leakage of pollutants or nuclear radiations and wastes which may harm the environment badly [5].

Safety is internal property of a system but safe system can never be guaranteed. However, if in some system risk of damage to life, environment or property can be controlled and brought within the acceptable limits, then such a system can be called safe. With continually increasing penetration of IT into industry and service sectors, numbers of critical systems are increasing and there is more demand for safer systems [3], [11].

This paper critically analysis all such techniques (formal or informal). A wide variety of techniques for safety analysis is available. These techniques are both formal and informal such as Fault Hazard Assessment (FHA) [12], Fault Tree Analysis (FTA) [15], Failure Mode Effect Analysis (FMEA) [21] and FSSA [8] and DCCA [19]. Formal techniques are currently focus of the academic research because these techniques keenly consider the system under analysis and have bright prospects to ensure safe systems. Many newer formal approaches for safety analysis are developed [4]. Formal methods have higher probability of providing better safety analysis for critical systems [4], [8]. If formal and informal approaches are used in combination, it is definitely an important step towards safer software systems. If Formal methods stress thorough and in-depth analysis then informal methods have greater scope for intuition and imagination by various stakeholders [9]-[11], e.g., possible hazards or errors imagination. Combination of formal and informal techniques can be constructively used for safety analysis.

Manuscript received September 19, 2012; revised October 14, 2012.

The authors are with Center for Software Dependability, Muhammad Ali Jinnah University (MAJU), Islamabad, Pakistan (email: aftab775@yahoo.com, anadeem@jinnah.edu.pk).

II. Analysis of Safety Approaches

In this paper, we have analyzed both formal and informal techniques as under;

A. *Informal (Traditional Techniques)*

FMEA or FMECA is a bottom up approach where all possible errors are enlisted and then classified to target those errors that may cause hazards or mishaps. FMEA is definitely useful to propose changes in a system during its development, thereby reducing the cost to be incurred if these errors and there consequences were overlooked. All errors are presented in tabular form and are helpful to study the system, its event and safety concerns. Finding out all such errors is a hectic effort and can never be ensured that all the errors are found. Success of FMEA depends upon thorough understating of the system. Alternatively, Fault trees can be automated to reduce the time required to carry out the safety analysis and eventually reduce the error or manual shortcomings of conventional FMEA. The tool that uses model for generation of fault trees, are discussed in [27], can be refined to improve the generation of quality trees. Formalization of FMEA is possible and it should be made a necessary part of comprehensive and detailed safety analysis to improve its quality aspects. FFMEA can be used standalone without combination of other safety techniques for successful and complete analysis of safety critical systems [21], [18].

Despite its several advantages FMEA has a vital drawback of intuitively finding out errors. Imagination of analysts in carrying out safety analysis is quite important. Secondly there might be many errors that are harmless and do not lead to mishaps. Such errors are given undue attention in this technique. This effort can be seen as an overhead and has no role in value addition to the safety of the system. Then FMEA is of course an error prone technique as “human is err” and has limitations to analyze the safety concerns. Out of existing traditional techniques, FMEA is considered better [15] [17].

FTA is a top down approach, in which all possible hazards are enlisted, for a safety critical system and then conventional method of fault trees is used to found all those errors that are root cause of these hazards. Since hazards that may occur are definitely few as compared to the errors in the system, FTA has more probability of success then FMEA but it does not happen for the reasons to be discussed later. If all such hazards are carefully enlisted, then all possible errors can be found and effort can be made to either remove these errors or counter the effect of these errors through error handling routines [15].

TABLE I: COMPARISON OF SAFETY ANALYSIS TECHNIQUE

Parameters	HAZOP	FTA	FMEA	FMECA	CORAS	DCCA	MUC	FSSA
Risk Identificatio n	Security aspects are focused	Top Down Approach	Bottom Up Approach	FMEA of critical part	Integrated informal approach	Empty, Single and Multi failure Mode, Minimal Critical set of failure modes	Misuse cases to analyze behavior	Addresses primary failure and rule out hidden failures
Risk Analysis	Used as input for other techniques	Major events are analysed	Minor errors that may occur are evaluated	Critical parts are analysed and mitigated	A combinatorial effort for informal techniques	Minimal Critical set of failure modes	Addresses misbehavior	Hidden and complete set of failure modes is discovered through formal check
Risk Evaluation	Used to evaluate other techniques	Can be evaluated with some criteria	Can be evaluated with some criteria	Can be evaluated with some criteria	A combinatorial effort for informal techniques	Compare with Criteria	Unitended Behavior to find flaws	Compare with non failure sensitive specification
Risk Treatment	Multiple options are identified	Can be priorotized	Can be priorotized	Alternativ e to critical parts are identified	A combinatorial effort for informal techniques	Addresses Criticality Set	Addresses unitended behavior	Completeness of the system
Automation	No	Yes	Yes	Yes	No	No	Semi automated	No
Formalizati on of Approach	Not Possible	Possible	Possible	Possible	Semi Formal	Already Formal Method	Semi formal	Already Formal Method
Skills of Analyst	Software Skills	Creative and Imagination	Creative and Imagination	Creative and Imaginati on	UML , Software Skills, Creativity	Creativity , Formal Methods	UML, Creativity and Imagination	Formal Methods and Informal Approaches
Component s, Languages or Artifacts Used	Functional and Operational Specficatio n	Fault Trees	Fault Trees	Fault Trees	Fault Trees , Use Cases	CTL and Automata	Misuse Cases, Fault Trees	Fault Trees, State Charts and Automata
Risk Identificatio n	Security aspects are focused	Top Down Approach	Bottom Up Approach	FMEA of critical part	Integrated informal approach	Empty, Single and Multi failure Mode, Minimal Critical set of failure modes	Misuse cases to analyze behavior	Addresses primary failure and rule out hidden failures

Possibility to enlist all the errors that are reason for some hazard or fault is quite implausible. Such errors can be overlooked and may result in mishaps. Again the success of FTA is strongly dependent on the imagination and creativity of the analysts. Since human errors are always present, this technique is also error prone. It is also important to discuss that generation of all the possible fault trees for FTA is not a simple task and only a partial portion of the fault trees can be generated and studied which further limits the FTA [1], [2], [4].

Understanding of system and relationship between events is possible through graphical format of FTA. Concentration of FTA on failures is a redundant work because there are many failures that don’t have any roots in the system and then many failures might converge on same type of errors and considering these errors again and again is definitely redundancy of work, and FTA should not do extra effort on this exercise. Qualitative evaluation along with quantitative evaluation can be helpful to avoid accidents because quantitative analysis may mislead [2], [4].

FHA considers the functionalities of the software system for hazard assessment. FHA uses the basic concepts of HAZOP (Hazard Analysis and Operability). HAZOP has focus on the deviations of a system from its design intentions. It is assumed that all the errors are mainly due to failure to meet design considerations and if these considerations are carefully met in the development of the system then these errors can be minimized. This technique mainly assumes that system has been carefully studied and all the possible hazards, their effects or consequences and remedies are incorporated in the system. Once all such effort has been made, then any failure is due to improper development of the system. But thorough and in-depth study of the system and incorporating every possible event in the design is impossible. Therefore, this technique has its limitations and cannot be considered better than FTA and FMEA [12].

Petri net models which use backward analysis by using faults and failures to study the hazards and determine critical areas of the system. This technique helps to use fault tolerant or fail-safe safety mechanisms to address critical areas of the system. One of the major draw-back which kicks this technique out of the competition is complexities involved in the buildup of Petri net models. Secondly, Petri net models are not applied on the realistic systems, hence there efficacy and success is still an interrogation mark [22].

CORAS project [23] carefully integrates all the traditional techniques. This integration helps to present combinatorial results of these techniques.

B. Formal Techniques

Through analysis and evaluation of the informal techniques it is very clear that these techniques are error prone and these techniques do not ensure complete analysis of the system. Formal methods are more rigorous in nature and thoroughly study the safety concerns of a critical system [15]. In order to deal with incomplete or insufficient functional requirements one of the oldest semi-formal methods is verification of functional correctness. Graphical Requirement Analysis (GRA) converts functional requirements into logic bases graphical representation. Functional requirements are traced and can be verified but this identification of functional requirement is not in full detail but present these requirements up to certain level. Thus functional correctness is not the best solution to safety analysis. DCCA uses mathematical proofs to eliminate the error occurrence of informal methods [30].

Fault Trees can be formalized to verify the completeness of the analysis and flaws of analysis that may occur due to informal fault trees can be removed. FTA approach can declare a system either less safe than the actual safeness of the system or it can over emphasize the safeness of the system [47]. A formalized fault tree ensures that all the possible causes for a consequence are listed. Any failure can't occur without a cause and minimal cut set through completeness condition is guaranteed. But there might be certain conditions where it can be proved that an FTA may consider a safer system unsafe. Suppose there are two events, if any of these events fail, system fails. FFTA uses AND-gate to describe the system but OR-gate must be in the fault tree. This factor shows that a formalized FTA may not result in minimal cut set [6].

This drawback has been considered in DCCA. The reason is that DCCA uses computational tree logic (CTL) the logic behind CTL is branching time logic [27] whereas FFTA rely on linear time logic. DCCA considers "existential properties of concurrent program in addition to its universal properties" [28] and secondly inner nodes of a FTA are formalized in FTA which is quite complicated task and some time too hard to incorporate but DCCA doesn't require formalization of the inner nodes of a tree. Thirdly, another concern is order of growth required to fulfill the completeness condition in FFTA or DCCA, which is exponential to find the minimal cut sets. This factor is of less criticality in DCCA. FTA or FMEA can be used for pre analysis of the minimal criticality sets for DCCA. Last but not the least is the monotony of criticality which minimizes proof effort required to find minimal critical set by taking an empty set, a singleton set and then multiple element set. Existence of an empty set as a critical set proves that the system design is improper and has functional flaws. Existence of a singleton set as minimal critical sets emphases functional correctness of the system and this principle is similar to traditional FTA then there is multiple elements set (an improved FMEA) which is already described in the first mode drawback of FFTA [19][25][26].

Functional correctness and fault tolerance is the two main or say vital safety concerns of a system. DCCA focuses on the functional correctness of the system. The main reason is that functional properties can be more rigorously verified through formal methods. This problem is present in almost all the formal methods used to address the safety concerns of the system. However, FSSA technique considers functional correctness as well as fault tolerant behavior of the system to counter this flaw of the formal methods. Basic concept of FSSA is derived from ForMoSa methodology [7]. FSSA is completely distinct from all other safety techniques whether these are formal or informal. Other techniques consider the intended behavior of the system as an input and then find out errors in it, but FSSA takes chaotic model as an input which comprises both intended and un-intended behavior and is a "set of all possible inputs and output" [8]. This model can be represented in a tabular form. Then elimination of failure behavior is then carried out by considering the entire chaotic model. Thus FSSA ensures that all the failure modes are discovered. This factor shows superiority of FSSA over DCCA and FFTA. But FSSA should not be taken as alternative to formal techniques due to its complexities involved in it. But it must be used as a complementary for highly critical systems for better safety analysis. FSSA has two limitations which are i) sets that are used for analysis are of exponential size ii) and lack of creativity, or intuition in formalism. But these limitations are addressed by FSSA by taking the failure modes at component level rather than at system level. Another concept about FSSA is that it is very difficult and complex approach. Then states are missing in the formalism with the objective to avoid mutual exclusiveness because mutual exclusiveness is most of the time source of errors. Another important advantage of FSSA is that it can be used for traditional techniques of safety analysis [8].

UML and its artifacts can be used for safety analysis and can be integrated with other safety techniques for both formal and informal analysis. Use cases can be used either directly for early analysis through some of the adaptations or extensions in the diagram or textual descriptions or can be integrated in formal or textual form with some other safety technique. Fault trees can be generated from the use cases. Misuse cases (MUC) are another application of UML in deriving the failure modes from a system. MUC if compared with traditional techniques is that it relates threats to the actions of a system. MUC as compared to HAZop identifies potential hazards. The major advantage of this step is relation of an activity to possible hazard. MUC is less confusing as compared with FMEA. Unintended behavior of the system can be viewed with the help of misuse cases. Water tank example explained in [11] a misuse case can "aggravate" or "mitigate" the effect some other use case or misuse case. MUC is far better than FMEA in identifying the failure modes of the system where user interacts but is less effective than FMEA as far as "inner working" is concerned. Secondly, MUC approach is easier to use and learn than FMEA and can be combined with FMEA for better results. The main disadvantage of MUC is that it is less descriptive than fault trees and requires necessary changes to minimize this gap, which make MUC quite complicated. The discussion on performance of MUC as compared to traditional techniques shows that misuse cases can be used in combination with traditional techniques to reap the benefits of UML. MUC as compared to DCCA is its ability to represent what an actor can do and what it can not do. MUC can introduce indirect negative actions, which are not introduced in DCCA because DCCA considers only those "events or conditions that are directly affecting the critical events". However, DCCA has main advantage to represent "explicit timeline and causal relationship between events and also has explicit yes no choices" [9], [10].

Formal techniques are rigorous and ensure completeness but informal techniques have valid advantage of utilizing the skills and creativity of the analyst. It will be more advantageous if both types of techniques are combined for better depiction of safety concerns of the system.

III. Conclusion

Traditional techniques can perceive a safe system unsafe and vice versa. Thus safety concerns of a system are ambiguous in traditional techniques and require formalism to cater for this problem. These techniques if used as an input to formal techniques by narrowing the scope of minimal critical set to be further addressed in formal techniques shows better results. There are many formal techniques which either take informal techniques as input or integrate them.

This study is a survey of the existing safety analysis techniques. Although DCCA has high success rate out of all safety analysis techniques, but its major drawback is its inability to consider unintended behavior and thus fails to incorporate fault tolerance of the system. In future we will compare DCCA and FSSA and efforts will be made to incorporate unintended behavior in DCCA. Another issue is about the reusability of the models used for safety analysis. These models can be stored in some database and reused for same domain if required.

References

[1] W. Vesley, J. Dugan, J. Fragole, J. Minarik, and J. Railsback, "Fault Tree Handbook with Aerospace Applications," *NASA Office of Safety and Mission Assurance*, *NASA Headquarters*, Washington DC 20546, August 2002.

[2] G. Bruns and S. Anderson, "Validating safety models with fault trees," in *Proc. of 12th International Conference on Computer Safety, Reliability, and Security*, pp. 21 – 30, Springer-Verlag, 1993.

[3] N. G. Leveson, "Software Safety: Why, What and How," *Computing Survey*, vol. 18, no. 2, June 1986.

[4] R. M. Sinammon and J. D. Andrews, "New approaches to evaluating fault trees," *Reliability Engineering and System Safety*, 1997, vol. 58, pp. 89-96.

[5] O. Bowen and V. Stavridou, "Safety Critical Systems, Formal Methods and standards," *Software Engineering Journal*, 1982

[6] J. Gorski and A. Wardzinski, "Formalizing fault trees," in *Proc. of Achievement and Assurance of Safety*, Springer-Verlag Berlin Heidelberg, 1995

[7] F. Ortmeier, A. Thums, G. Schellhorn, and W. Reif, "Combining formal methods and safety analysis – the ForMoSA approach," in *Proc. of Integration of Software Specification Techniques for Applications in Engineering. Springer LNCS 3147*, 2004

[8] F. Ortmeier and W. Reif, "Failure-sensitive specification: A formal method for finding failure modes," *Technical Report 3, Institut fur Informatik*, Universit-at Augsburg, 2004

[9] T. Stålhane and G. Sindre, "A Comparison of Two Approaches to Safety Analysis Based on Use Cases," *ER 2007*, LNCS 4801, pp. 423–437, 2007.

[10] C. Parent and K. Dieter, "Conceptual modeling - ER 2007," Published by Springer 2007

[11] G. Sindre, "A Look at Misuse Cases for Security Concerns," in *Proc. of Henderson Sellers, IFIP WG8.1 Working Conference on Situational Method Engineering: Fundamental of Experiences (ME 07) Geneva*, Switzerland, IFIP Series, Heidelberg –Springer 2007

[12] T. A. Kletz, "Hazop and HAZAN notes on the identification and assessment of hazards," *Technical report, The Institution of Chemical Engineers, Rugby, England*, 1986

[13] J. M. Dermid, "Software Hazard and Safety Analysis."

[14] M. JChudleigh, "Hazard Analysis Using Hazop: A Case Study," in *Proceedings of the 12th International Conference on Computer Safety, Reliability and Security,* ed. J. Gorski, pp. 99-108, 1993.

[15] R. E. McDermott, R. J. Mikulak, and M. R. Beauregard, "The Basics of FMEA," *Quality Resources*, 1996

[16] Y. Papadopoulos, D. Parker, and C. Grante, "Automating the Failure Modes and Effects Analysis of Safety Critical Systems," in *Proceedings of the Eighth IEEE International Symposium on High Assurance Systems Engineering (HASE'04)*, 2004

[17] R. Dev, "Software System Failure Mode And Effects Analysis (SSFMEA) - A Tool For Reliability Growth," in *Proceedings of the International Symposium on Reliability and Maintainability*, 1990.

[18] C. J. Price and N. Taylor, "Automated multiple failure FMEA," *Reliability Engineering and System Safety*, 2002, vol. 76, pp. 1-10

[19] G. Schellhorn, F. Ortmeier, and W. Reif, "Deductive cause-consequence analysis (dcca)," in *Proceedings of IFAC World Congress*, 2005

[20] J. G. Griggs, "A method of software safety analysis," in *Proceedings of the Safety Conference (Denver, Colo.)*, vol. 1.

[21] G. Bruns and S. Anderson, "Validating safety models with fault trees," in *Proc. of 2th International Conference on Computer Safety, Reliability, and Security*, pp. 21 – 30, Springer-Verlag, 1993

[22] N. G. Leveson and J. L. Stolzy, "Safety analysis using petri nets," *IEEE Transactions on Software Engineering*, vol. 13, no. 3, 1987

[23] K. Stølen, F. Braber, T. Dimitrakos, R. Fredriksen, B. A. Gran, S. Houmb, M. S. Lund, Y. C. Stamatiou, and J. O. Aageda, "Model-based risk assessment – the CORAS approach," 2002

[24] D. H. Stamatis, *Failure Mode and Effect Analysis: FEMA from theory to Execution*, American Society for Qu

[25] *ality (ASQ)*, Milwaukee, Wisconsin, 1995. P. Bieber, C. Castel, and C. Seguin, "Combination of fault tree analysis and model checking for safety assessment of complex system," in *Proc. of Dependable Computing EDCC-4: 4th European Dependable Computing Conference*, vol. 2485.

[26] G. Schellhorn, A. Thums, and W. Reif, "Formal fault tree semantics," in *Proceedings of The Sixth World Conference on Integrated Design and Process Technology*, Pasadena, CA, 2002.

[27] E. A. Emerson and C. L. Lei, "Modalities for Model Checking: Branching Time Logic Strikes Back," in *Proc. of 12th Annual Symposium on Principles of Programming Languages*, 1985.