Safety-Critical Software

200714175 이정현*, 200312461 김계성 ,
200310405 류규현, 200412302 김무진

Contents

Introduction

Introduction

=

Safety-Critical System

- **Purpose of this Report**

  to bring together concepts necessary for the development of software in safety-critical Systems.

Comments on Software Safety

Comments on Software Safety

- **the concepts relating the safety-Critical Software**

1) Safety is a System Issue
2) Safety is Measured as Risk
3) Reliability is Not Safety
4) Software Need not be Perfect
5) Safe Software is Secure and Reliable
6) Software Should Not Replace Hardware
7) Development Software is Also Safety-Critical

Comments on Software Safety

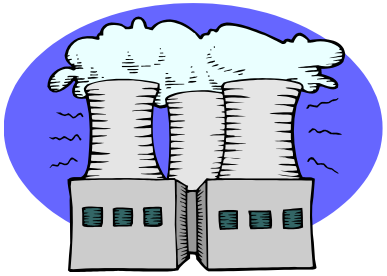
1) Safety is a System Issue

- Safety is not a software issue, rather, it is a system issue.
- It is the control of System with hazardous components, or the providing of information to people who make decisions that have potentially hazardous consequences, that leads to hazardous system. Thus, software can be considered unsafe only in the context of a particular system.
- At the system Level, software may be treated as one or more components whose failure may lead to a hazardous system condition. Such a condition may result in the occurrence of an accident.

Comments on Software Safety

2) Safety is Measured as Risk

- Safety is an abstract concept.
- **"This system is safe."**
  - Not cause harm either to people or property
- There are many system that can be made completely safe, But, making system that safe may interfere with their ability to perform their intended function.
- Ex)



The nuclear reactor

Comments on Software Safety

2) Safety is Measured as Risk

- Thus, the definition of safety becomes related to risk.

$$Risk = \sum_{hazard} E_{hazard} \times P_{hazard}$$

  - E_{hazard} : a measure of the effects that may be caused by a particular mishap
  - P_{hazard} : probability that the mishap will occur

- This paper will not further define how risk may be measure.

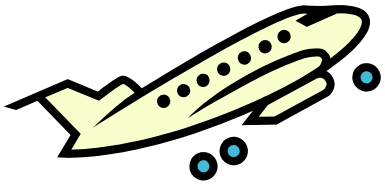
- However, **no System will be wholly safe.**
- We must attempt to minimize the risk by either containing the hazard or reducing the probability that the hazard will occur.

Comments on Software Safety

3) Reliability is Not Safety

- **Reliability**
  - How well the system performs its function.
  - The System's correctness with regard to its specification.
  - The ability of the system to **tolerate faults**.

- **Safety**
  - In terms of the absence of hazardous behaviors in the system.
  - The system functions do not lead to an accident.



Reliability $\neq$ **Safety**

Comments on Software Safety

4) Software Need not be Perfect

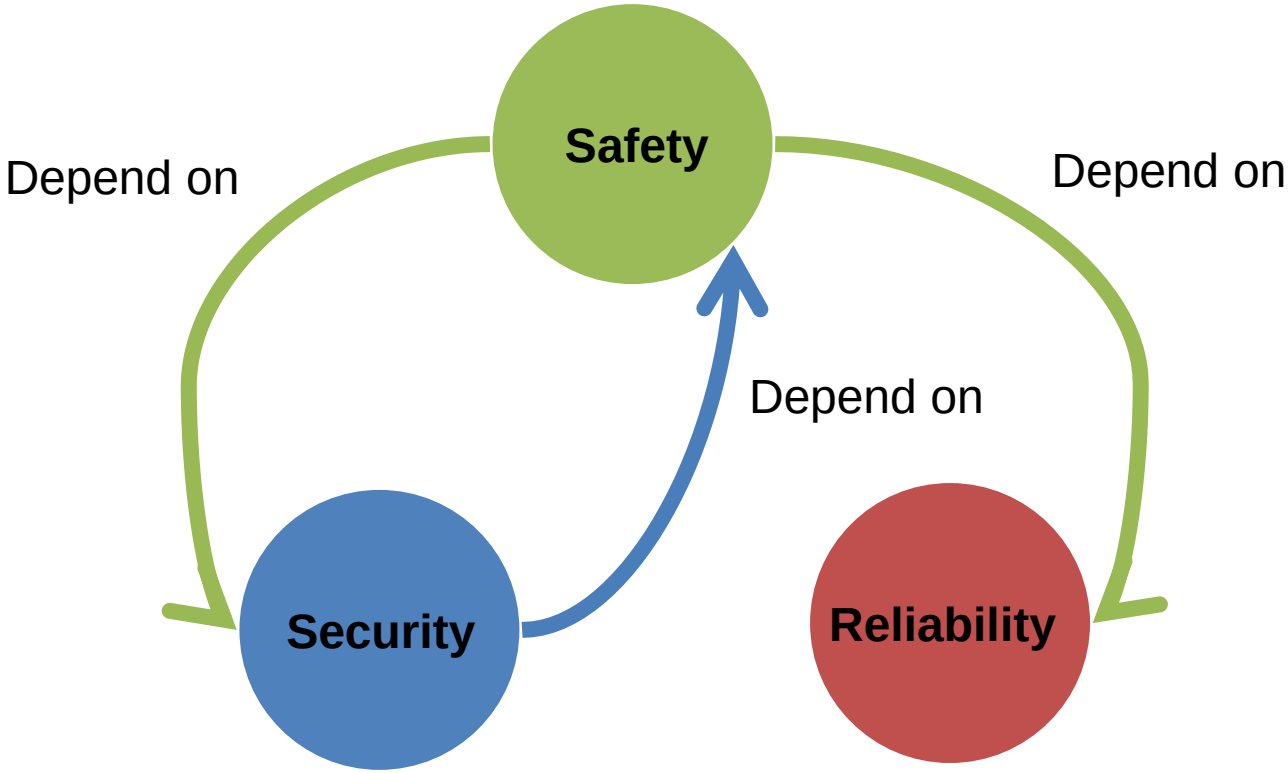
User

No Error = Perfect

- Only error that cause the system to participate in an accident are of importance. But the system could still be safe.
- Developers and analysts of safe software can concentrate their most detailed scrutiny on the safety condition and not on the operational requirements.
- The system are imperfect and may not behave as expected

Comments on Software Safety

5) Safe Software is Secure and Reliable



Comments on Software Safety

5) Safe Software is Secure and Reliable

- **Secure**
  - A secure system may need to be reliable.
  - If, unreliable : It is possible that a failure could occur such that the system's security is compromised.
  -
- **Safety**
  - In terms of the absence of hazardous behaviors in the system.
  - The system functions do not lead to an accident.

Security ≠ **Safety**

Comments on Software Safety

6) Software Should Not Replace Hardware

Software	Hardware
• flexible • easy to modify • the reproduction costs are very low	• quite expensive to reproduce

- Hardware obeys certain physical laws that may make certain unsafe behaviors impossible.
- Hardware fails in more predictable ways than software, and a failure may be foreseen by examining the hardware a bar may bend or show cracks before it fails.
- Software does not exhibit physical characteristics. making the failures unexpected and immediate

Comments on Software Safety

7) Development Software Is Also Safety-Critical

- Safety analysis of a system is performed on a number of artifacts created during the development of the system.

- First, The analysis of the current stage of the development shows that a system performing according to the current description is safe
- Second, There is certainty that any artifacts created in subsequent development stages precisely conform to the current description

- The earlier a system can be analyzed for safety with a guarantee that the second condition will be met, the more cost effective will be the overall development, as less work will need to be redone if the current system description is shown to be unsafe.

Hazard Analysis Techniques

Hazard Analysis Techniques

- **The effort of performing a hazard check of a system**

- The general approach to hazard analysis is first to perform a preliminary hazard analysis to identify the possible hazards.

- Subsequently, subsystem and system hazard analyses are performed to determine contributors to the preliminary hazard analysis.

Hazard Identification	Hazard Analysis
• The Delphi Technique • Joint Application Design • Hazard and Operability Anaysis	• Fault Tree Analysis • Event Tree Analysis • Failure Modes and Effects Analysis

Hazard Analysis Techniques

1. Hazard Identification

- The only acceptable approach for hazard identification is to attempt to develop a list of possible system hazards before the system is built
- The obvious approach is to use “brainstorming,” whereby the experts list all of the possible hazards that they envision for the system.

The Delphi Technique	Joint Application Design	Hazard and Operability Analysis

Hazard Analysis Techniques

1) The Delphi Technique

- The key idea behind the Delphi Technique is that the opinions are presented anonymously and that the only interaction between the experts is through the questionnaires.

Hazard Analysis Techniques

2) Joint Application Design

- Joint Application Design (JAD)'s purpose is to help a group reach decisions about a particular topic.
  - 사용자가 개발 과정에 참여함으로써 개발 속도가 빠르고 사용자 만족도가 크다.
  - 사용자 의견을 시스템 요구 사항에 즉각 반영 가능.

Hazard Analysis Techniques

3) Hazard and Operability Analysis

- First, the designers identify their concepts of how the system should be operated.
- The second step is to determine when the identified conditions can become safety-critical.
  - 몇 개의 단계로 나누어 오류가 발생할 만한 요소를 검사한다.
- Hazard and operability analysis is an iterative process that should be started before any detailed design.
- It should be continually updated as system design progresses.

Hazard Analysis Techniques

2. Hazard Analysis

- The purpose of hazard analysis is to examine the system and determine which components may lead to a mishap.
- There are two basic strategies to such analysis, which have been termed inductive and deductive

Fault Tree Analysis	Event Tree Analysis	Failure Modes and Effects Analysis
• The deductive techniques	• The inductive techniques	• The inductive techniques

Hazard Analysis Techniques

1) Fault Tree Analysis

- FTA is Deductive hazard analysis techniques.
  - 원하지 않는 특정한 사건들로 시작되며 이 사건의 원인 분석을 위한 접근법을 제공

- It is important to choose undesirable event carefully.
  - 너무 일반적이라면 FTA는 크고 다루기 힘들게 된다.
  - 너무 구체적이라면 시스템 분석에 대한 충분한 시야 제공 못함.
  - Undesirable event이 선택되면, FTA의 상위 사건으로 사용.
  - 모든 가능성 있는 방법을 모색하기 위해 분석된다.

- Fault tree analysis can be an expensive and time-consuming process.

Hazard Analysis Techniques

1) Fault Tree Analysis

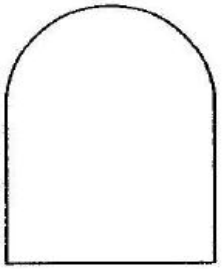


Figure 1. *And* gate.

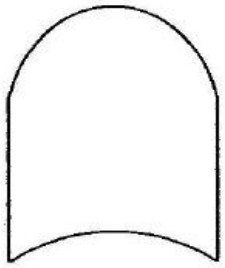


Figure 2. *Or* gate.

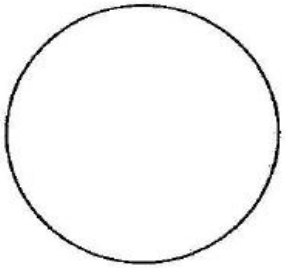


Figure 3. *Basic* event.

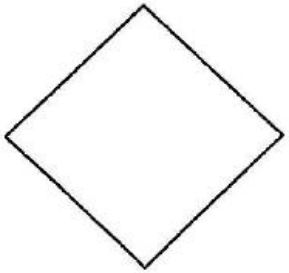
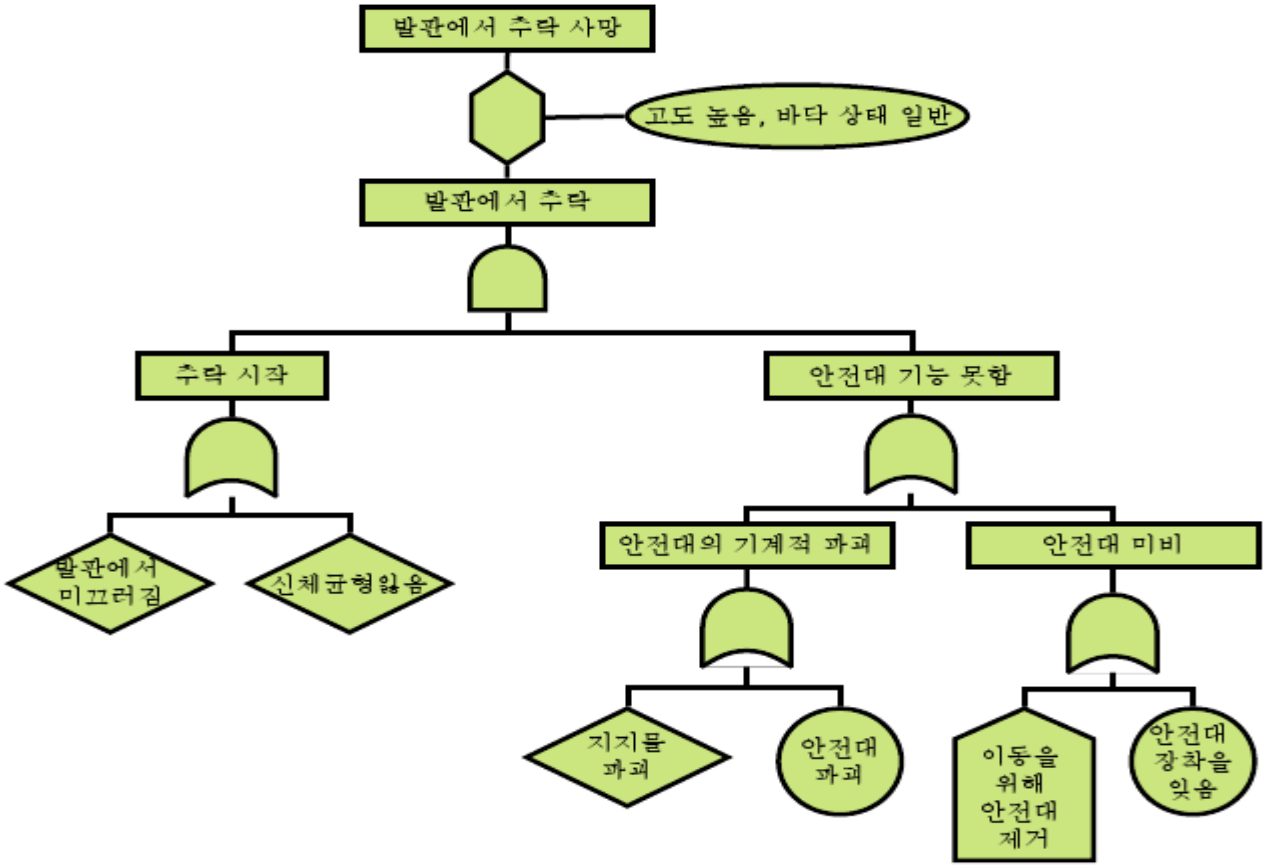


Figure 4. *Undeveloped* event.

Hazard Analysis Techniques

1) Fault Tree Analysis



Hazard Analysis Techniques

2) Event Tree Analysis

- Its approach is to consider an initialing event and its possible consequences, then for each of these consequential events in turn, the potential consequences are considered, thus drawing the tree.

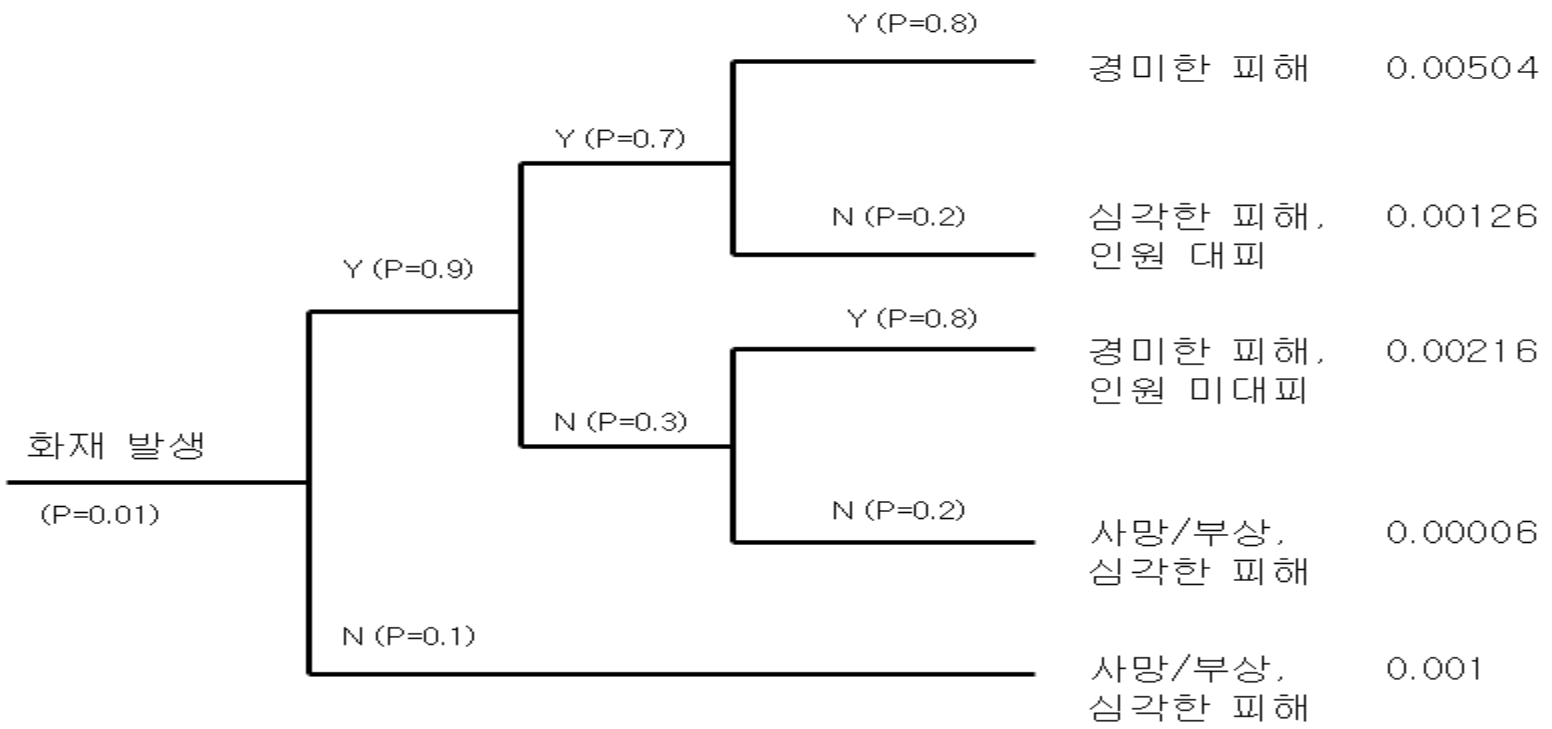
- The purpose of event tree analysis is to consider an initiating event in the system and consider all the consequences of the occurrence of that event, particularly those that lead to a mishap.

- The initialing events for event tree analysis may be both desirable and undesirable.
  - the choice of initiating events is the range of events that may occur in the system.

- It is forward-looking and considers potential future problems

2) Event Tree Analysis

화재 발생	중간 사상			결과	확률
	화재 감지기 작동	화재 경보 작동	스프링클러 작동		



Hazard Analysis Techniques

3) Failure Modes and Effects Analysis

- FMEA consists of constructing a table based on the components of the system and the possible failure modes of each component.

- The approach used is to create a table with the following columns: component, failure mode, effect of failure, cause of failure, occurrence, severity, probability of detection, risk priority number, and corrective action.

- For each component, a list of the possible failure modes is created.
  - The engineer is then required to enter a value indicating the frequency of occurrence of the particular cause of the failure mode.

Hazard Analysis Techniques

3) Failure Modes and Effects Analysis

Table 1. Example failure modes and effects analysis table

Component	Failure mode	Effect of failure	Cause of failure	Occurrence	Severity	Probability of detection	Risk priority number	Corrective action
Tie bar bracket	Bracket fractures	Stabilizing function of tie bar removed. All engine motion transferred to mountings.	Inadequate specification of hole-to-edge distance	1	7	10	70	Test suitability of specification
	Bracket corrodes	As above	Inadequate specification for preparation of bracket	1	5	10	50	Test suitability of specification
	Fixing bolts loosen	As above	Bolt torque inadequately specified	5	5	8	200	Test for loosening
			Bolt material or thread type inadequate	1	5	10	50	Test suitability of specification

- The End -