http://dx.doi.org/10.3745/KTSDE.2013.2.10.661

A Collision Analysis Technique for Prevention Actions of Accident in Safety Critical System

Jang-Jin Kwon† · Jang-Eui Hong††

ABSTRACT

A safety critical system is a system that leads to injury of people, damage of property and environment due to functional failures or occurrence of undesired condition. Therefore, to ensure the safety of a system, system engineers should consider the inherent hazards of the system at design phase of the system development, and also should design the prevention actions to minimize damage when an accident occurred. The objective of these actions is preventing the serious damage from accidents that can occur due to unforeseen circumstance. Recently, many studies have been performed to identify and analyze their hazards at design phase of safety critical systems. This paper suggests a safety analysis technique for analyzing the collision among those prevention actions to reduce accident and its effect by the collision of these actions that did not mentioned in previous studies. Through the proposed technique, it would improve robustness of safety and would help the design of prevention actions into system for the occurrence of accidents.

Keywords : Safety, Safety Critical System, Collision Analysis, Prevention Action

Safety Critical 시스템에서 사고의 예방동작간 충돌 분석 기법

권 장 진† · 홍 장 의††

요 약

Safety Critical 시스템은 시스템의 기능적인 실패 또는 예기치 못한 상황의 발생으로 인해 인명피해, 재산피해, 환경 피해와 같은 치명적인 사고를 초래할 수 있는 시스템을 의미한다. 그러므로 Safety Critical 시스템의 안전을 보장하기 위해서는 시스템 설계 단계에서 시스템에 존재할 수 있는 위험성들이 충분히 고려되어야 하며, 사고가 발생했을 시 피해를 최소화시키기 위한 일련의 예방 동작들이 설계되어야 한다. 현재에는 Safety Critical 시스템의 설계 단계에서 위험성을 식별하고 분석하기 위한 많은 방법들이 연구되었으며, 이 중에는 예기치 못한 사건으로 인한 피해를 예방하는 동작들의 성공 여부를 분석하는 기법도 존재한다. 본 연구에서는 위의 예방 동작들의 성공 여부에 대한 분석뿐만 아니라 기존 연구들에서 언급하지 못한 예방 동작들 간의 충돌과 이로 인해 발생할 수 있는 피해를 분석하는 방법을 제시하고자 한다. 제안한 방법을 통해 Safety Critical 시스템의 안전성이 견고해지고 피해 예방을 위한 동작들의 올바른 설계에 도움이 될 수 있을 것이다.

키워드 : 안전성, Safety Critical 시스템, 충돌 분석, 예방 활동

1. 서 론

오늘날에는 컴퓨터, 각종 전자 기기 및 소프트웨어의 발전으로 인해 생활 곳곳에 수많은 시스템들이 스며들어 있다. 시스템들 중에는 생활의 소소한 부분을 제어하기 위한 간단한 시스템들이 존재하는 반면, 기능적인 고장 또는 실패와 같이 예기치 못한 상황이 원인이 되어 치명적인 피해를 초래할 수 있는 시스템도 존재한다. 이와 같이 안전에 직접적인 영향을 미칠 수 있는 시스템들을 Safety Critical 시스템이라고 부른다[1].

Safety Critical 시스템에서 안전성을 보장하기 위해서는 시스템 설계 단계에서의 위험성 분석 활동이 매우 중요하다. 위험성 분석 활동을 통해 여러 위험성들과 근본적인 원인들이 식별되면 이 정보들을 토대로 설계의 수정, 기능의 제약 등 안전성 보장을 위한 엄격한 규제가 이루어져야 한다. 또한 피해 예방 동작들을 구성함으로써 시스템에서 발생할 수 있는 사고들이 치명적인 피해로 이어지는 것을 예방할 수 있어야 한다[2-4].

Safety Critical 시스템의 위험성 분석 방법은 기존에 많은 연구가 진행되었다. 이러한 연구들은 위험의 분석 단계,

※ 이 논문은 2012년도 충북대학교 학술연구지원사업의 연구비 지원에 의하여 수행되었음.
† 준 회 원 : 충북대학교 컴퓨터과학과 석사과정
†† 종신회원 : 충북대학교 소프트웨어학과 교수
논문접수 : 2013년 5월 28일
수 정 일 : 1차 2013년 7월 8일
심사완료 : 2013년 7월 9일
* Corresponding Author : Jang-Eui Hong(jehong@chungbuk.ac.kr)

분석 수준 및 범위 등에 따라 다양한 위험성 분석 기법들을 제안하고 있다[2-5]. 대표적으로 잘 알려진 위험성 분석 기법으로는 예기치 못한 상황에 대하여 그 원인을 시각적으로 모델링하는 Fault Tree Analysis (FTA) [6]와 시스템에 존재하는 고장 모드를 식별하고 분석하는 Failure Mode and Effect Analysis (FMEA)[7] 등이 존재한다. 이외에도 예기치 못한 사건들이 치명적인 피해로 이어질 수 있는지를 분석하는 Event Tree Analysis (ETA)[8]와 Cause-Consequence Analysis (CCA) [9]가 제안되었다.

본 연구에서는 위의 많은 분석 기법들 중 ETA와 CCA의 분석 목적에 관심사를 두었다. 두 기법은 Safety Critical 시스템의 예기치 못한 사건으로부터 피해를 예방하는 동작들의 성공 여부를 분석한다. 이를 통해 해당 사건이 심각한 피해로 이어질 수 있는 지를 밝혀내는 기법들이다. 하지만 이 두 기법은 예방을 위한 동작들의 성공적인 수행 여부에 대해서는 고려하였지만, 해당 동작 간에 발생할 수 있는 상호 충돌에 대해서는 고려하지 않았다. 예를 들어 상호 배제 관계의 두 피해 예방 동작들이 서로 영향을 미칠 수 있는 관계로 설계 된다면, 해당 동작들이 동시 발생했을 경우 각 예방 동작들의 목적을 달성하지 못할 수 있다. 따라서 본 연구에서는 시스템에 설계된 예방 동작들의 성공 여부를 분석하는 것뿐만 아니라 충돌 가능성이 존재하는 동작들을 식별 및 분석하는 방법을 제안하고자 한다.

논문의 구성은 다음과 같다. 2장은 관련 연구를 분석하였고, 3장에서는 피해 예방 동작 간의 충돌에 대하여 설명한다. 4장에서는 충돌분석 기법에 대하여 제안하고, 5장에서는 제안 기법을 예제 시스템에 적용하였다. 마지막으로 6장에서는 결론 및 향후 연구에 대하여 기술하였다.

2. 관련 연구

2.1 Cause-Consequence Analysis

CCA는 Safety Critical 시스템에서 예기치 못한 사건의 발생이 심각한 피해로 이어질 수 있는지를 판단하는 분석 기법이다[10]. 시스템에서 발생할 수 있는 사고 시나리오를 구성하고, 사고의 피해를 예방하는 동작들의 실패 유무를 분석한다.

Andrews와 Ridley[11]의 연구에서는 Cause-Consequence Diagram을 사용하여 위험성을 분석하는 연구를 진행하였다. 기존의 FTA를 사용한 위험성 분석은 피해 예방 동작들의 순차적인 실패를 고려하는데 어려움이 따른다. 따라서 이 연구에서는 피해 예방 동작들의 순차적인 실패를 고려할 수 있고, FTA처럼 기능 실패의 논리적인 구조를 표현하는 Cause-Consequence Diagram을 제안하였다. 그러나 이들은 예방 동작들의 충돌까지는 고려하지 못했다.

2.2 Event Tree Analysis

ETA는 CCA와 매우 유사한 기법이다. 사고 시나리오를 이용하여 피해를 예방하기 위한 동작들이 실행될 수 있는지를 분석하는 기법이다. ETA는 이벤트의 순차적인 흐름을 가시적으로 표현하는 Event Tree를 사용한다[8, 10].

Andrews[8]의 연구에서는 사고 시나리오를 중심으로 이진트리를 사용한 Event Tree를 구성하였다. 예방 동작들의 성공 또는 실패를 Event Tree로 구성하고, 트리의 모든 패스에 대한 실행 결과를 분석하였다.

Xingang Song[12]은 ETA를 이용하여 시스템의 핵심 기능의 동작 실패에 따라 연쇄적으로 발생할 수 있는 사고를 분석하였다. 이를 통해 핵심 기능의 동작에 대한 안전성을 향상시키기 위한 연구를 수행하였다.

David Huang[13]의 연구에서는 ETA가 각 이벤트의 단일 확률만을 고려한 분석을 수행하기 때문에 불확실하거나 부정확한 상태에 대한 평가는 비현실적이라고 주장하였다. 따라서 Fuzzy 이론을 사용하여 ETA의 단점을 보완하는 연구를 진행하였다. 하지만 위의 대부분의 연구들 역시 피해를 예방하기 위한 동작들 간의 충돌 가능성은 고려하지 않았다.

3. 피해 예방 동작 간 충돌

3.1 예방 동작

Safety Critical 시스템에는 사고의 피해를 예방하기 위한 서브시스템들이 존재한다. 각 서브시스템들은 발생 가능한 사고의 피해를 예방하고 최소화 하는 것을 목적으로 한다. 예방 동작은 이러한 서브시스템의 구성요소로써 피해 예방을 위해 수행되어야 하는 단위 행동을 의미한다. 즉, 동일한 피해를 예방하기 위한 목적으로 다수의 동작들의 집합이 존재할 수 있으며, 이 집합이 하나의 서브시스템을 구성한다. 예방 동작은 다음과 같이 정의할 수 있다.

[정의 1] 예방 동작(Prevention Action) $PA = <O, R, C>$으로 구성된다. 여기에서,
- O : 예방 동작에 의해 수행되는 실제 행위
- R : 예방 동작에 관련 있는 리소스
- C : 예방 동작의 성격을 구분하는 식별자

정의 1에서 언급한 예방동작 PA는 피해 예방을 위한 서브시스템을 구성하는 단위 수행 동작을 의미한다. PA의 구성요소 O는 Lock, Open 등과 같은 구체적인 의미에서 예방 동작을 기술하기 위해 사용한다. 또한 R은 예방 동작과 관련성을 갖는 특정 디바이스, 시스템 또는 물질을 의미하며, O의 대상이 된다. 예를 들면 화재 감지 센서, 결함복구 시스템, 또는 빛, 물과 같은 물질 등이 R에 해당될 수 있다. C는 예방 동작의 성격이 개방적인지, 폐쇄적인지를 구분하는 식별자로써 참과 거짓으로 표현된다. 만약 예방 동작이 특정 시스템을 가동시키거나, 밸브를 여는 것과 같은 개방적인 동작일 경우, TRUE에 해당하며, 특정 시스템 셧다운, 밸브의 잠금 등과 같이 폐쇄적인 성격일 경우 FALSE로 정의된다.

3.2 충돌의 개념

예방 동작 간의 충돌은 서로 다른 피해를 예방하기 위한 목적을 가진 서브시스템들이 동시에 작동하면서 발생할 수 있다. 작동 중인 서브시스템이 수행하는 임의의 예방 동작이 다른 서브시스템을 가동시키는 역할을 한다면, 두 서브시스템이 동시에 작동될 것이다. 이 경우에 동일하거나 관련 있는 리소스에 접근하는 두 서브시스템의 예방 동작이 서로 다른 성격을 갖는다면, 다시 말해서 PA의 C 값이 각각 TRUE와 FALSE 값을 갖는다면 두 동작이 충돌할 가능성이 높아진다. Fig. 1은 예방 동작 간 충돌의 개념을 간략하게 보여주고 있다.

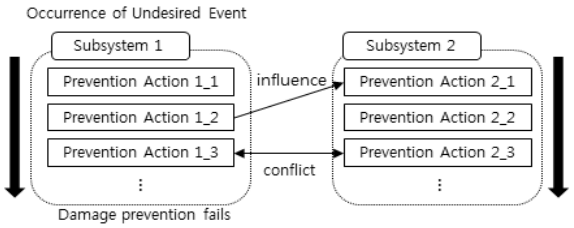

Fig. 1. Concept of Collision among Prevention Actions

Fig. 1을 살펴보면 예기치 못한 사건이 발생했을 때, 해당 사건의 피해를 예방하기 위해 Subsystem 1이 작동한다. Subsystem 1은 여러 예방 동작들로 구성되어 있다. 이 중 Prevention Action 1_2에 의해서 Subsystem 2의 Prevention Action 2_1이 영향을 받을 수 있다. 여기에서, Prevention Action 2_1은 Subsystem 2를 구성하는 초기 예방 동작이며, 이는 Subsystem 2를 트리거 시킬 수 있는 동작임을 의미한다. 만약 Prevention Action 1_2로 인한 결과가 Prevention Action 2_1을 활성화 시키면 Subsystem 2가 Subsystem 1과 동시에 작동할 수 있다. 이와 같은 상황에서 Prevention Action 1_3과 Prevention Action 2_3이 서로 같은 리소스나 연관성이 있는 리소스에 접근하고, 각 예방 동작의 성격이 서로 다르다면 충돌이 발생할 수 있다.

4. 충돌 분석 방법

피해 예방 동작간의 충돌 분석 방법은 다음과 같은 절차로 정의된다.

- Fault Prevention Tree 구성을 통해 Safety Critical 시스템의 사고/피해 예방 동작 및 리소스 식별(4.1절)
- 리소스 맵 구성을 통해 리소스 간의 상관관계 분석(4.2절)
- Fault Prevention Tree의 Cut Set 식별(4.3절)
- 예방 동작 간 충돌 후보 선정(4.4절)
- 실제 충돌 가능성이 존재하는 예방 동작 분석(4.5절)

4.1 사고 예방 동작 및 리소스 식별

Safety Critical 시스템의 예방 동작을 식별하기 위해 본 연구에서는 결함 예방 트리 (Fault Prevention Tree, FPT)

Table 1. Symbols of Fault Prevention Tree

Symbol	Type	Description
	System	represents the whole system, including sub- systems
	Top Node	describes the goal of each subsystem and represents the top node of each tree.
	Basic Node	describes the prevention action required to achieve the top node (i.e., goal)
	Resource	represents an object that is involved in each prevention action

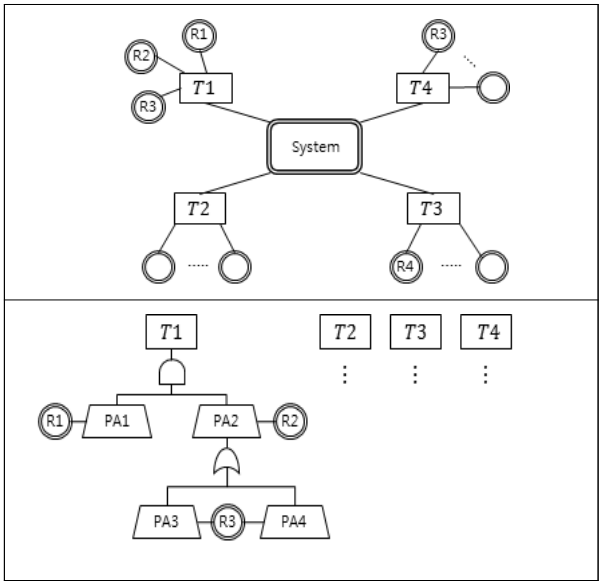

Fig. 2. Example of Fault Prevention Tree

를 정의하였다. FPT는 기존의 FTA[14]에서 사용되는 Fault Tree(FT)를 확장한 것으로 jianwen XIANG[15]이 제시한 Formal Fault Tree구축 방식을 사용하여 구축한다. 논리 연산자를 사용하여 구축하는 트리의 기본 개념은 FT와 동일하지만 예방 동작들과 리소스를 표현하기 위한 몇 가지 심볼들이 추가되었다. 또한 FT와는 다르게 FPT에서는 각 서브시스템의 최종 예방 목표가 트리의 탑 노드에 위치한다. 따라서 탑 노드가 달성되기 위한 예방 동작들의 조합이 하위 노드로 구성된다. Table 1은 FPT에서 사용되는 심볼들을 정의한 것이다. Table 1의 심볼들을 사용한 전체 FPT의 모습은 Fig. 2를 통해 확인할 수 있다. Fig. 2 상단에 나타난 트리는 전체 시스템을 표현한 최상위 트리이다.

피해 예방을 위한 서브시스템의 최종 목표를 나타내는 탑 노드들과 해당 서브시스템에 포함되는 리소스들이 최상위 트리에 표현된다. 예방 동작 간 충돌을 분석하기 위해서는 먼저 충돌이 발생할 수 있는 서브시스템을 선정해야 한다.

따라서 전체 리소스가 표현된 최상위 트리를 통해 동일 리소스가 포함된 서브시스템을 분석 대상으로 선정할 수 있다. Fig. 2의 하단은 특정 서브시스템의 최종 예방 목표인

$T1$의 FPT이다. 특정 서브시스템의 예방 목표 $T1$을 달성하기 위한 예방 동작들의 조합과 각 예방 동작에 포함되는 리소스들로 구성된다.

4.2 리소스 맵 구성

3.2절에서 언급한 바와 같이 예방 동작 간 충돌은 두 예방 동작이 동일한 리소스 또는 서로 관련 있는 리소스에 동시에 접근하면서 발생할 수 있다. 따라서 예방 동작들에 포함된 전체 리소스들의 상관관계를 파악하는 것이 매우 중요하다. 이를 위해 본 연구에서는 리소스들 간의 관계를 모델링하기 위한 리소스 맵을 제안한다.

리소스 맵은 각 리소스가 독립적으로 존재하는지 또는 다른 리소스와 제어관계로 존재하는지를 한 눈에 알아볼 수 있도록 도와준다. 여기에서 제어 관계란 하나의 리소스가 다른 리소스에 제어되는 것을 의미한다. 간단한 예로 물과 스프링클러라는 리소스가 존재한다면 물은 스프링클러에 제어당하는 리소스가 된다. Table 2는 리소스 맵에서 사용되는 심볼들을 보여준다.

Table 2. Symbols of Resource Map

Symbol	Type	Description
	Resource	represents an object that is involved in each PA.
	Control Relationship	applies when a resource controls the other resource.

Table 2의 심볼들을 사용한 리소스 맵의 예는 Fig. 3과 같과 같다. Fig. 3에서 리소스 R1은 R2로부터 제어 당하는 리소스를 나타낸다. R6는 다른 리소스와의 관계가 이어지지 않았으므로 독립적으로 존재하는 리소스임을 알 수 있다.

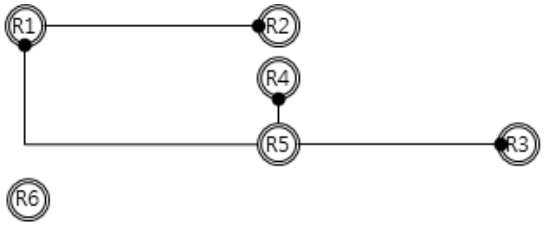

Fig. 3. Example of Resource Map

이와 같이 리소스 맵이 구성되면 각 리소스의 제어 관계를 Table 3과 같이 매트릭스로 나타낼 수 있다.

Table 3. Resource Map Matrix

	R1	R2	R3	R4	R5	R6	sum
R1	-	1	0	0	1	0	2
R2	0	-	0	0	0	0	0
R3	0	0	-	0	0	0	0
R4	0	0	0	-	0	0	0
R5	0	0	1	1	-	0	2
R6	0	0	0	0	0	-	0

매트릭스의 열은 제어 당하는 리소스들을 나타내며 행에 해당하는 리소스에 제어 당할 시 1의 값을 갖는다. Table 3을 살펴보면 R1과 R5의 sum 값이 2인 것을 알 수 있다. 이는 각기 다른 두 개 이상의 리소스로부터 제어 당하는 것을 의미하며 잠재적인 충돌 가능성이 존재하는 리소스임을 알 수 있다.

4.3 Fault Prevention Tree의 Cut Set 식별

FTA에서 Cut Set이란 트리의 탑 노드에 정의한 이벤트를 발생시킬 수 있는 리프노드들의 집합을 의미한다[10]. 본 연구에서 제안한 FPT의 Cut Set도 이와 동일한 의미를 갖는다. 또한 Cut Set은 트리의 구조적 특성만을 기반으로 하여 식별되기 때문에 본 연구에서는 기존의 Cut Set 알고리즘을 변경 없이 사용하였다. FPT에서의 Cut Set은 특정 피해를 예방하기 위해 발생하는 서브시스템의 초기 동작들의 조합을 의미하게 된다.

FPT의 Cut Set을 구하는 이유는 3.2절 충돌 개념에서 설명한 바와 같이 특정 예방동작이 다른 서브시스템의 작동에 영향을 미치는지를 알아보기 위함이다. 만약 특정 FPT의 Cut Set에 해당하는 예방 동작 PA_i가 다른 FPT의 예방 동작 PA_j와 동일 리소스에 접근한다면 PA_i는 PA_j로부터 영향을 받을 가능성이 존재한다. 따라서 모든 FPT의 Cut Set을 구함으로써 해당 Cut Set에 포함되는 예방 동작이 다른 FPT의 예방동작으로부터 영향을 받을 수 있는지 분석한다.

Cut Set을 구하는 방법은 기존에 알려진 방법인 MOCUS 알고리즘을 사용한다. 정확한 방법은 [16]에서 확인할 수 있으며, Fig. 4를 통해 간단한 예를 확인할 수 있다.

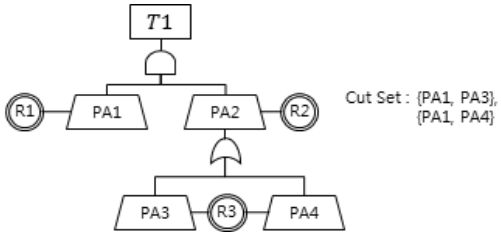

Fig. 4. Example of Cut Set in the FTA

즉, Fig. 4에서 Basic Node인 $PA1$, $PA3$, 그리고 $PA4$는 초기에 발생하는 예방 동작들이며, 이들의 조합이 Cut Set을 구성한다.

4.4 예방 동작 간 충돌 후보 선정

각 서브시스템의 FPT와 리소스 맵이 구성되면 이를 통해 예방 동작 간 충돌 후보들을 선정할 수 있다. 예방 동작들 간의 충돌 후보는 다음과 같이 정의한다.

[정의 2] 예방 동작 간 충돌 후보 (Collision Candidate of Prevention Actions), CCPA : 동일 리소스를 포함하는 두 서브시스템 S_i, S_j에 대하여 $PA_i \in S_i$이고 $PA_j \in$ Cut Set(S_j) 일 경우, 정의 1에 의하여 $PA_i \Rightarrow <O_i, R_i,$

$C_i>$, $PA_j \Rightarrow <O_j, R_j, C_j>$이다. 이때, CCPA = $ASR \cup ACR$로 정의한다. 여기에서

- ASR : {$<O_i, R_i, C_i>, <O_j, R_j, C_j> \mid R_i = R_j$}을 만족하는 예방 동작들의 집합
- ACR : $<O_i, R_i, C_i> \in ASR$에 대하여 R_i를 제어하는 리소스를 포함하는 예방 동작들의 집합

정의 2에서 언급한 ASR은 동일한 리소스에 접근하는 예방 동작들의 집합을 의미한다. 이는 서브시스템 S_i의 특정 동작이 S_j의 작동을 트리거 시킬 수 있는 초기 동작에 영향을 미칠 수 있는지 찾아내기 위한 것이다. 이를 위해 S_i의 모든 예방 동작과 S_j의 Cut Set에 포함되는 예방 동작들을 비교한다. 또한 ASR로 선정된 예방 동작들과 관련 있는 예방 동작들도 식별해야 한다. 따라서 리소스 맵을 통해 ASR에 포함된 예방 동작들의 리소스와 제어 관계를 갖는 리소스들을 식별한다. ACR은 이러한 리소스를 포함하는 예방 동작들의 집합을 의미한다.

4.5 실제 충돌 가능성이 존재한 예방 동작 분석

CCPA가 식별되면 해당 동작들이 실제로 충돌 가능성이 있는지를 판단해야 한다. CCPA로 선정되었더라도 무조건 충돌이 발생하는 것이 아니기 때문이다. 간단한 예로 서로 다른 목적을 가진 두 예방 동작이 하나는 피해 예방을 위해 특정 시스템을 가동시켜야 하고, 다른 하나는 동일 시스템을 셧다운 시켜야 하는 동작일 경우 두 예방 동작은 서로 충돌이 발생할 것이다. 충돌 가능한 예방 동작을 식별하기 위하여 우리는 정의 1에서 언급한 구성 요소 C 값을 이용한다. 식별 방법은 다음과 같다.

- CCPA에 포함하는 동작 PA_i, PA_j의 구성 요소 C_i, C_j를 확인한다.
- $C_i \in PA_i$와 $C_j \in PA_j$에 대한 Exclusive -OR 연산을 수행한다.
- 만약 연산 결과가 FALSE라면 두 예방 동작의 성격이 같음을 의미하므로 충돌이 일어날 가능성이 매우 적다.
- 만약 연산 결과가 TRUE라면 두 예방 동작의 성격이 서로 정반대라는 것을 의미하므로 충돌이 일어날 가능성이 매우 크다.

5. 적용 및 분석

5.1 예제 시스템 정의

본 연구에서는 Intelligent Building Management System (IBMS)를 예제 시스템으로 사용하였다. 이 중 화재 발생과 누수 발생 시의 피해 예방 시스템을 대상으로 하여 분석을 진행하였다. 각 시스템은 화재나 누수 발생 시 이를 감지하고, 치명적인 사고로 이어지지 않도록 예방하기 위한 동작으로 구성되어 있다.

5.2 Fault Prevention Tree 구성

화재 발생과 누수 발생에 따른 피해 예방 동작들이 식별되면, 이를 토대로 FPT를 구성한다. 해당 예제 시스템에서는 두 가지의 피해 예방 서브시스템을 대상으로 하였으므로 두 서브시스템에 대한 하위 FPT가 구성된다. 화재 발생에 대한 서브시스템 $S1$의 최종 예방 목표는 화재를 진압하는 것이고, 누수 발생에 대한 서브시스템 $S2$의 최종 예방 목표는 누수를 차단하는 것이다. 각 최종 예방 목표를 트리의 탑 노드로 하는 FPT는 다음과 같이 구성되며 First-order logic을 사용하여 기술하였다.

<화재 발생에 대한 예방 동작들의 FPT 구성>

- Top Node $T1$ 정의 : 서브시스템 $S1$의 최종 예방 목표. ($T1$) $\forall$ fire : Suppress(fire)
- Top Node $T1$의 게이트 구성 : $T1$이 달성되기 위한 하위 예방 동작들의 조합. ($G1$) (Emit(smoke) $\wedge$ Sprinkle(water)) $\Rightarrow$ Suppress(fire)
- 하위 예방 동작 $PA1_{T1}$, $PA2_{T1}$ 구성 : $G1$을 구성한 예방 동작들의 조합을 논리 연산자 기준으로 나누어 구성. $PA1_{T1}$ = $\forall$ smoke : Emit(smoke) $PA2_{T1}$ = Sprinkle(water)
- $PA1_{T1}$과 $PA2_{T1}$가 달성되기 위한 하위 예방 동작의 조합 구성.
- 최종 리프노드까지 재귀적으로 반복하여 수행.
- 각 서브시스템을 포함한 전체 시스템을 표현하는 최상위 트리 구성.

위와 같은 방법으로 Fig. 5A, 5B, 5C와 같이 FPT를 구성할 수 있다. Fig. 5A는 전체 시스템을 표현하는 최상위 트리이며, Fig. 5B와 5C는 각각 화재 발생과 누수 발생에 대한 FPT이다. 또한 구성된 FPT를 통해 각 서브시스템의 예방 동작과 전체 리소스를 Table 4와 같이 식별 할 수 있다.

Table 4. Identified Prevention Actions from IBMS

IE	PA	Resource	Type
Fire Start	Detects Fire Occurrence	Fire	TRUE
	Operate the Sprinkler System	Sprinkler	TRUE
	Operates the Ventilator System	Ventilator	TRUE
	Sprinkles Water	Water	TRUE
	Emits Smoke	Smoke	TRUE
Water Leak	Detects Water Leakage	Water	TRUE
	Sends a Close-Signal to the Valve	Valve	TRUE
	Close the Valve	Valve	FALSE
	Operates the Alarm System	Alarm	TRUE

Fig. 5A. Top-level Structure of FPT

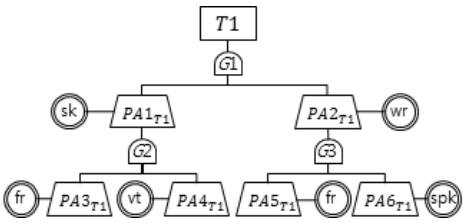

Fig. 5B. FPT of Fire Occurrence

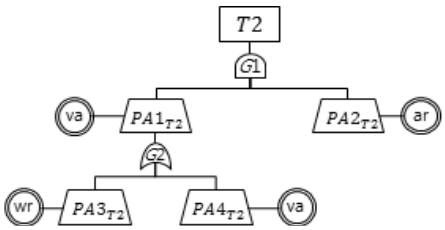

Fig. 5C. FPT of Water Leakage

<Fig. 5B의 정형적 명세>
fr = fire, sk = smoke, wr = water
spk = sprinkler system, vt = ventilator
($T1$) ∀fr : Suppress(fr)
($G1$) (Emit(sk) ∧ Sprinkle(wr)) ⇒ Suppress(fr)
($PA1_{T1}$) ∀sk : Emit(sk)
($PA2_{T1}$) Sprinkle(wr)
($G2$) (Detect(fr) ∧ Operate(vt)) ⇒ ∀sk(Emit(sk)
($G3$) (Detect(fr) ∧ Operate(spk)) ⇒ Sprinkle(wr)
($PA3_{T1}$) ∀fr : Detect(fr)
($PA4_{T1}$) Operate(vt)
($PA5_{T1}$) ∀fr : Detect(fr)
($PA6_{T1}$) Operate(spk)

<Fig. 5C의 정형적 명세>
wr = water, va = valve, ar = alarm system
($T2$) ∃wr : Disrupt(wr) ∧ NotifyDisruption(wr)
(G_{T2}) Close(va) ⇒ Disrupt(wr)
OperateSystem(ar) ⇒ NotifyDisruption(wr)
(Close(va) ∧ OperateSystem(ar)) ⇒
(Disrupt(wr) ∧ NotifyDisruption(wr))
($PA1_{T2}$) ∃va : Close(va)
($PA2_{T2}$) OperateSystem(ar)
(G_{SE1}) Detect(wr) ∧ SendCloseSignal(va) ⇒
Close(va)
($PA3_{T2}$) Detect(wr)
($PA4_{T2}$) ∃va : SendCloseSignal(va)

5.3 리소스 맵 구축

화재 발생과 누수 발생에 대한 FPT를 통해 각 서브시스템에 포함되는 모든 리소스들을 리소스 맵으로 구축한다. 리소스 맵은 Fig. 6과 같다. 또한 리소스 맵이 구축되면 각 리소스의 제어 관계를 Table 5와 같이 리소스 맵 매트릭스로 구성한다.

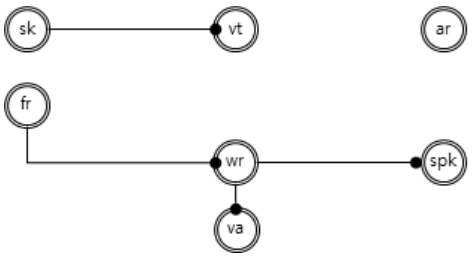

Fig. 6. Resource Map of IBMS

Table 5. Resource Map Matrix

	sk	vt	fr	wr	spk	va	ar	sum
sk	-	1	0	0	0	0	0	1
vt	0	-	0	0	0	0	0	0
fr	0	0	-	1	0	0	0	1
wr	0	0	0	-	1	1	0	2
spk	0	0	0	0	-	0	0	0
va	0	0	0	0	0	-	0	0
ar	0	0	0	0	0	0	-	0

5.4 Cut Set 식별 및 CCPA 선정

Table 6은 5.2절에서 구성한 FPT의 Cut Set을 식별한 것이다. 탑 노드 $T1$의 하위 노드들은 모두 AND게이트로 연결되어 있으므로 Cut Set은 하위 예방 동작 $PA3_{T1}$, $PA4_{T1}$, $PA5_{T1}$, $PA6_{T1}$의 조합이다. Table 6을 통해 Cut Set을 식별하는 과정을 확인할 수 있다.

Table 7의 식별된 Cut Set에 해당하는 예방 동작들과 4.4절에서 설명한 CCPA 식별 방법을 토대로 예제 시스템의 피해 예방 동작들 간 충돌 후보들을 식별한다. 먼저 누수 발생에 대한 모든 예방 동작들과 화재 발생에 해당하는 Cut Set들을 비교한 결과 동일 리소스를 사용하는 동작이 발견되지 않았다. 반대로 화재 발생에 대한 모든 예방 동작과 누수 발생에 대한 Cut Set을 비교한 결과 동일 리소스를 사용하는 예방 동작 Sprinkle(wr), Detect(wr)이 식별되었다. 이는 Sprinkle(wr)로 인해 Detect(wr)이 영향을 받아 누수 감지에 대한 서브시스템이 작동될 수 있다는 것을 의미한다. 다음으로 Table 5의 리소스 매트릭스를 이용해 wr의 sum 값을 확인한다. wr의 sum 값이 2이므로 wr을 제어하

는 리소스가 2개 이상이며, 이로 인해 잠재적인 충돌 가능성이 존재한다는 것을 알 수 있다. 마지막으로 매트릭스에서 wr과 1의 값으로 매핑되는 리소스 spk와 va를 식별할 수 있다. 따라서 CCPA로 화재 발생 시의 Operate(spk), Sprinkle(wr)과 누수 발생 시의 Detect(wr), Close(va)를 선정할 수 있다.

Table 6. Identification procedure of Cut Sets

<table>
<tr><td colspan="4">Identifying cut sets of fault tree about fire occurrence
(by order of numbers)</td></tr>
<tr><td colspan="4">1 → 2 → Cut Set</td></tr>
<tr><td>$T1$</td><td colspan="2">$PA1_{T1}$, $PA2_{T1}$</td><td>$PA3_{T1}$, $PA4_{T1}$,
$PA5_{T1}$, $PA6_{T1}$</td></tr>
<tr><td colspan="4">Identifying cut sets of fault tree about water leakage
(by order of numbers)</td></tr>
<tr><td colspan="4">1 → 2 → 3 → Cut Set</td></tr>
<tr><td rowspan="2">$T2$</td><td rowspan="2">$PA1_{T2}$,
$PA2_{T2}$</td><td>$PA3_{T2}$,
$PA2_{T2}$</td><td>$PA2_{T2}$,
$PA3_{T2}$</td></tr>
<tr><td>$PA4_{T2}$,
$PA2_{T2}$</td><td>$PA2_{T2}$,
$PA4_{T2}$</td></tr>
</table>

Table 7. Included Actions in the Cut Sets

	Cut Sets
Fire Occurrence	Detect(fr), Operate(vt), Operate(spk)
Water Leakage	OperateSystem(ar), Detect(wr), SendCloseSignal(va)

5.5 실제 충돌 가능성 분석

식별된 CCPA들의 구성요소 C에 대하여 서로 다른 서브시스템에 포함되는 동작들끼리 Exclusive -OR 연산을 수행한다. 각 예방 동작의 구성요소 C의 값은 Table 4에서 확인할 수 있으며 연산 결과는 Table 8과 같다.

연산 결과가 TRUE라는 것은 각 예방 동작이 동일하거나 서로 영향을 미칠 수 있는 리소스들을 정 반대의 유형으로 다루는 것을 의미한다.

Table 8. Results of Exclusive-OR operations

Operate(spk) ⊕ Detect(wr) ⟹ TRUE ⊕ TRUE	FALSE
Operate(spk) ⊕ Close(va) ⟹ TRUE ⊕ FALSE	TRUE
Sprinkle(wr) ⊕ Detect(wr) ⟹ TRUE ⊕ TRUE	FALSE
Sprinkle(wr) ⊕ Close(va) ⟹ TRUE ⊕ FALSE	TRUE

Table 8에서 TRUE값을 갖는 Operate(spk)와 Close(va), Sprinkle(wr)과 Close(va)를 살펴보자. Close(va)는 누수를 차단하기 위해 배관을 잠그는 동작이다. Operate(spk)와 Sprinkle(wr)은 화재를 진압하기 위해 스프링클러는 작동시켜 물을 뿌리는 예방 동작이므로 Close(va)와 동시에 작동된다면 예방 동작 간의 충돌이 발생하는 것을 확인할 수 있다. 해당 예방 동작들의 충돌 시나리오는 Fig. 7과 같다. 이와 같이 시스템 설계 단계에서 충돌 가능성이 존재하는 예방 동작들이 식별되면, 각 예방 동작의 임계치를 부여하여 동작의 위험 수위를 조정하거나, 혹은 우선순위 기반의 스케줄링을 활용하여 해당 동작들의 충돌을 피하도록 시스템을 설계할 수 있을 것이다.

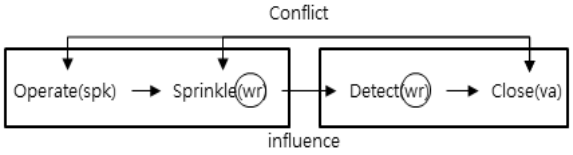

Fig. 7. Collision Scenario among PAs

6. 결론 및 향후 연구

본 연구에서는 Safety Critical 시스템에 설계된 예기치 못한 사고의 피해를 예방하는 동작들이 서로 충돌 가능하다는 것을 보였다. 또한 본 연구에서 제안한 리소스 맵과 Fault Prevention Tree을 사용하여 충돌 가능성이 존재하는 예방 동작들을 식별하는 방법을 제안하였다. 이를 통해 Safety Critical 시스템에서 피해를 예방하기 위한 서브시스템들을 더욱 견고하게 설계할 수 있을 것이며, 시스템의 안전성이 향상하는 것을 기대할 수 있다. 하지만 제안한 방법을 통해 식별된 예방 동작들의 충돌 해결 방안에 대해서는 완벽한 연구가 이루어지지 못했다. 따라서 향후 연구에서는 충돌 해결 방안에 대한 심도 있는 연구를 진행할 것이며, 예방 동작 간의 충돌을 분석하는 도구를 개방하여 안전성을 보장할 수 있는 시스템 설계를 지원할 수 있도록 하고자 한다.

참 고 문 헌

[1] John C. Knight, “Safety Critical System: Challenges and Directions,” Software Engineering, 2002, ICSE 2002. Proceeding of the 24rd International Conference on, pp.547-550.

[2] M. Ben Swarup, et al., “A Software Safety Model for Safety Critical Applications,” *International Journal of Software Engineering and Its Applications*, Vol.3, No.4, pp.21-32, 2009.

[3] Lee,Eun-Seo and Lee,Kyung-Whan, “Trigger design to software defect analysis,” *The KIPS Transactions. Part D*. Vol.10, No.4, pp.707-718, 2003.

[4] Lee,Woo-Jin, “Compositional Safety Analysis for Embedded Systems using the FSM Behavioral Equivalence Algorithm,”

The KIPS Transactions: Part D, Vol.14, No.6, pp.633-649, 2007.

[5] Lee T. Ostrom, et al., *Risk Assessment : Tools, Techniques and their Applications*, Wiley, 2012.

[6] Andrija Volkanovski, et al., "Application of the fault tree analysis for assessment of power system reliability," *Reliability Engineering & System Safety*, Vol.94, Issue.6, pp.1116-1127, 2009.

[7] Snooke, N. et al., "Model-driven automated software FMEA," in Proceedings of the *Reliability and Maintainability Symposium (RAMS)*, 2011, pp.1 - 6.

[8] Andrews, J.D. et al., "Event Tree Analysis Using Binary Decision Diagrams," the *IEEE Transactions on Reliability*, Vol.29, Issue.2, pp.230-238, 2000.

[9] Pauperas, J. et al., "Cause-consequence analysis of a generic space station computer system," in Proceedings of the *Reliability and Maintainability Symposium*, 1991. pp.196-201.

[10] C. A. Ericson, *Hazard Analysis Technique for System Safety*, Wiley Interscience, 2005.

[11] Andrews, J.D. et al., "Reliability of sequential systems using the cause-consequence diagram method," The *Institution of Mechanical Engineers: Part E*, Vol.215, No.3, pp.207-220, 2001.

[12] Xingang Song, et al., "Analysis of Management factors of Main Engine Failure Based on Event Tree Analysis," in Proceedings of the *7th International Conference on System of Systems Engineering (SoSE)*, 2012, pp.8-10.

[13] David Huang, et al., "A Fuzzy Set approach for event tree analysis," *Fuzzy Sets and Systems*, Vol.118, Issue.1, pp.153-165, 2001.

[14] Lee, W. S. et al., "Fault Tree Analysis, Methods, and Applications : A Review," The *IEEE Transactions on Reliability*, Vol.R-34, Issue.3, pp.194-203, 1985.

[15] Jianwen XIANG, et al., "Fault Tree and Formal Methods in System Safety Analysis," in Proceedings of the *CIT '04. The Fourth International Conference on Computer and Information Technology*, 2004, pp.1108-1115.

[16] Fussell, J. B. et al., "MOCUS: A computer program to obtain minimal sets from fault trees," in Proceedings of the *Aerojet Nuclear ANCR-1156*, 1974.

권 장 진

e-mail : jjkwon@selab.cbnu.ac.kr

2012년 충북대학교 컴퓨터공학부(학사)

2013년~현 재 충북대학교 컴퓨터과학과 석사과정

관심분야 : 소프트웨어 품질, 모델 기반 개발, 소프트웨어 안전성 등

홍 장 의

e-mail : jehong@chungbuk.ac.kr

2001년 KAIST 전산학과(박사)

2002년 국방과학연구소 선임연구원

2004년 (주)솔루션링크 기술연구소장

2004년~현 재 충북대학교 소프트웨어학과 교수

관심분야 : 소프트웨어 공학, 소프트웨어 품질, 저전력 소프트웨어, 소프트웨어 프로세스 개선