Hazard analysis techniques

DSLab

서영주

Table of Contents

1 개요

안전 필수 시스템 (safety-critical system)이란 시스템의 동작에서 사고가 발생할 경우 주위에 큰 피해를 줄 수 있는 시스템을 말한다. 여기에는 원자력 발전소나 자동차, 비행기 같은 여러 산업 분야들이 포함되는데 이러한 시스템들로부터 발생할 수 있는 사고의 원인을 찾아내고 최종적으로 여기서 발생하는 피해를 줄이기 위해 여러 가지 hazard analysis 기법들이 개발되어 왔다. 이러한 hazard analysis 기법들을 시스템에 효과적으로 적용하기 위해서는 어떠한 기법들이 있는지, 기법들의 적용 방법이나 특징들에 대해 알아야 한다.

또한 시스템의 safety, hazard와 관련된 표준들이 산업 분야마다 차이가 있기 때문에 hazard analysis와 safety의 개념에 대해 이해하려면 대상 분야에서 사용하는 용어의 정의, 그리고 관련 표준들에 대해 알아야 할 필요가 있으므로 이에 대한 내용들도 알아야 할 필요가 있다.

1.1 용어 정의

[1]에서는 Hazard analysis를 시스템의 hazard에 대해 평가와 확인을 수행하는 프로세스를 말하며, hazard를 제거하거나 시스템의 risk를 받아들일 수 있는 수준까지 낮추는 것 까지도 이에 포함된다고 정의하고 있다. 그리고 hazard란 accident가 발생하기 위한 전제조건이 되는 상태를 말한다. 여기에는 외부의 이벤트나 내부의 하드웨어/소프트웨어의 상태가 다 포함된다.

시스템은 갑자기 위험한 상태 (hazardous condition)에 들어가게 되는 것이 아니라 시스템의 하위에서 발생할 수 있는 fault, error, failure의 발생에 의해 위험한 상태에 들어가게 되는 것이고, 이것이 최종적으로 시스템의 accident를 발생시킬 수 있다.

Fault란 하드웨어 장치의 결함 또는 컴퓨터 프로그램의 잘못된 동작을 말하며, error란 연산 시 나와야 할 올바른 값과 계산 결과의 차이를 말하고 failure는 요구사항에 정의된 동작을 시스템이나 컴포넌트가 정상적으로 이행하지 못하는 것을 말한다. 이렇게 fault-error-failure chain에 의해 시스템이 위험한 상태로 갈 수 있으며, 여기서 사람의 죽음이나 장비 등의 피해를 가져오는 계획되지 않은 이벤트가 발생하는 것을 accident라고 한다. 그림 1과 그림 2가 fault-error-failure chain과 hazard 및 accident간의 관계를 잘 보여준다.

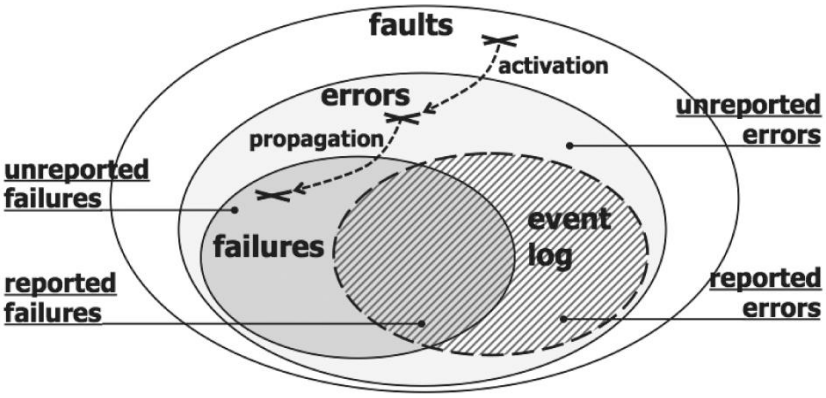


그림 1 fault-error-failure chain

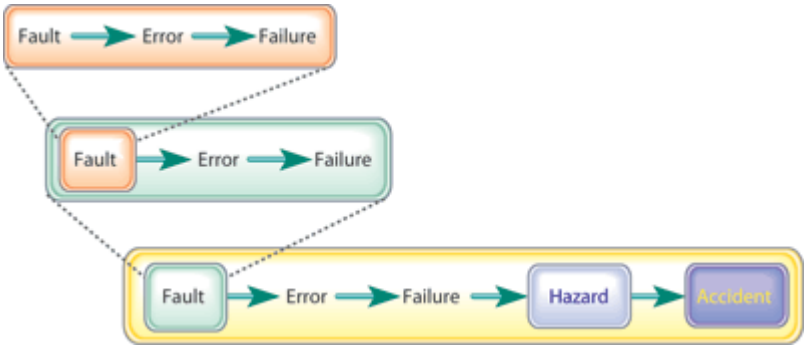


그림 2 fault-error-failure와 hazard 및 accident의 관계

Hazard analysis를 통해 시스템은 safety한 상태가 될 수 있으며, 이것은 시스템이 hazard가 없는 상태를 말한다. 또한 hazard analysis는 risk를 낮추는 것도 목적으로 하고 있으며, 이 때 risk란 system의 hazard가 accident를 일으킬 확률과 accident의 중요성을 통합하여 측정한 결과를 의미한다.

1.2 관련 표준들

소프트웨어의 safety와 관련된 표준으로는 앞서 용어 정의 부분에서 인용한 것과 같이 소프트웨어 엔지니어링의 용어 정리와 관련된 표준으로 [2]가 있고, 소프트웨어 개발 생명 주기[3], 유지보수 및 폐기 계획[4], 소프트웨어의 테스트 문서[5], 유닛 테스팅[6], 검증[7] 등 그 외 많은 항목들에 관하여 IEEE에서 각각의 표준들을 정의하고 있다.

소프트웨어보다 상위 수준의 시스템의 safety와 관련된 산업 표준들은 모든 종류의 산업에 적용

가능한 기본적인 기능 안전 표준이 되는 ISO 61508[8]을 기본으로 하며, 산업별로 특성들을 고려한 다양한 추가사항들을 정의한 표준들이 존재한다. 그림 3은 산업 안전 표준들의 관계도를 보여준다. 이 중에서 산업의 적용이 많은 분야는 자동차 (ISO-26262[9]), 항공 (DO-178B[10], DO-178C[11]), 원자력 (IEC-61513[12]) 등이 있다.

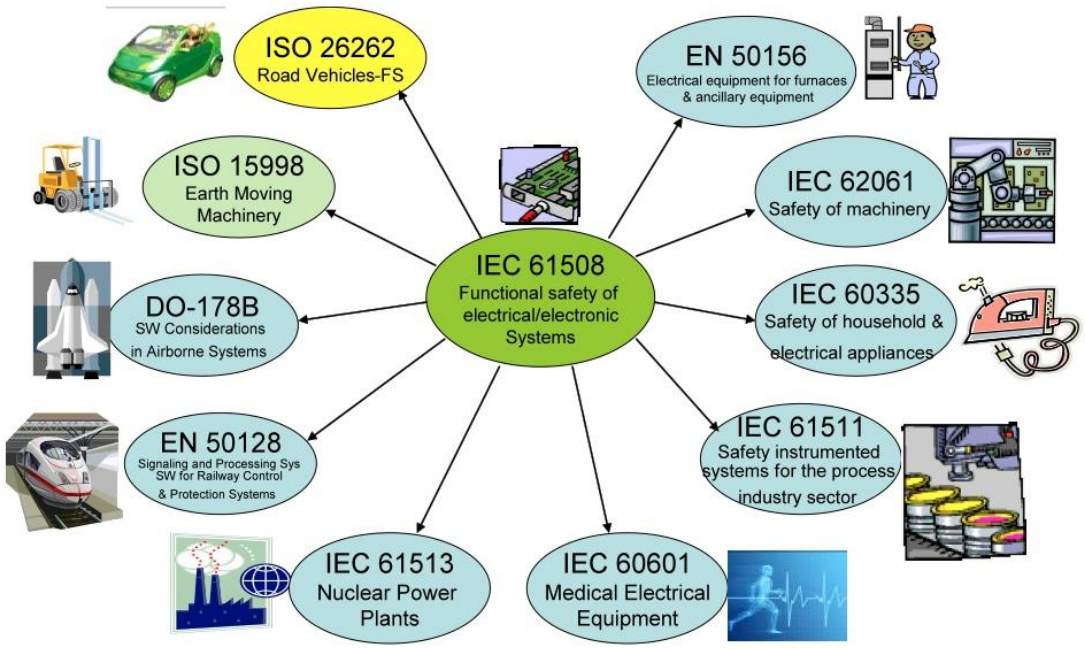


그림 3 산업 안전 표준 관계도

2 Development process와 hazard analysis

Hazard analysis를 수행할 때 이를 단독으로 개발된 시스템에 사용하여 시스템에서 발생할 수 있는 hazard의 원인을 찾는데 사용할 수도 있지만, 시스템의 개발 단계에서 development life cycle과 함께 사용할 수도 있고, 여러 hazard analysis 기법들을 함께 적용하여 engineering process를 만들어서 시스템의 safety를 높이기 위해 적용할 수도 있다.

2.1 Software development process 단계 별 hazard analysis 기법

미국 국방부에서는 software system의 safety를 위한 software system safety engineering process[]에 대한 handbook을 제공하고 있으며, 여기서는 소프트웨어를 포함하는 시스템의 safety를 위한 development process를 제시한다. 해당 프로세스는 그림 4 Software system safety engineering process에 나타나 있으며, System safety task implementation 단계의 하위 분석 단계에서 여러 hazard analysis 기법들을 이용해 시스템 내에서 발생할 수 있는 hazard와 그 원인에 대한 분석을

수행하는 형태로 되어있다. 예를 들어 그림 5 처럼 System hazard analysis 단계에서 발생할 수 있는 hazard의 원인을 분석하기 위해 fault tree analysis 기법을 사용할 수 있음을 제안한다.

그림 4 Software system safety engineering process

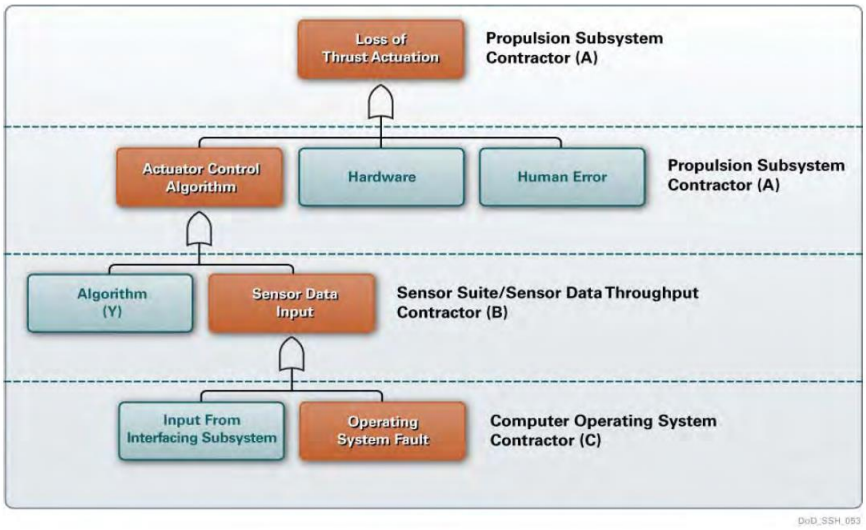


그림 5 Fault tree analysis를 이용한 System hazard analysis 단계의 예

2.2 Software life cycle에서 사용

IEC 61508같은 경우에는 development process 뿐만 아니라 유지보수, 시운전까지 포함하는 software life cycle을 제시한다. 그림 6에서 3번 단계에 해당하는 것이 hazard and risk analysis 과정이다.

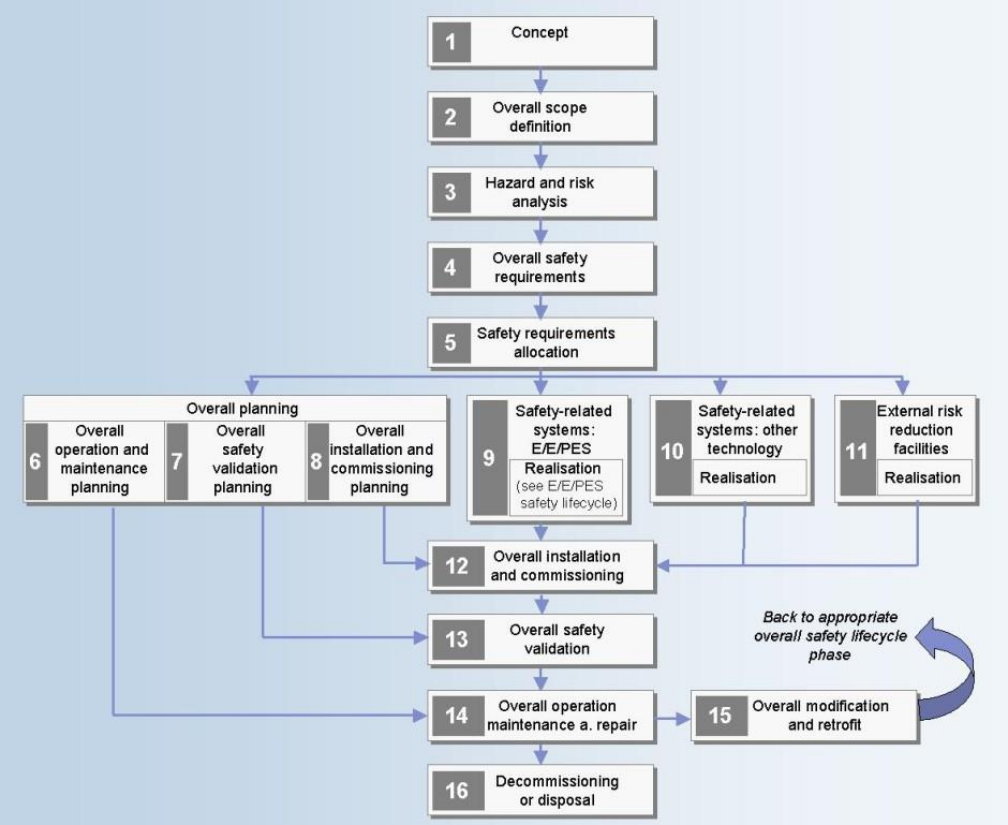


그림 6 IEC 61508의 safety life cycle

3 Hazard analysis 기법들

Hazard analysis 기법들은 여러 가지가 있으며 크게 이벤트의 발생을 유도했을 때 failure의 발생 여부를 확인하는 inductive 방식과 failure의 발생 원인을 찾는 deductive 방식으로 나뉜다. 또한 기법들 별로 표기법이나 특성들이 다르기 때문에 분석하고자 하는 대상의 특징을 고려하여 여러 hazard analysis 기법들 중에서 적절한 기법을 선택해야 한다.

3.1 Hazard analysis 기법 유형

그림 7이 hazard analysis 기법 유형의 분류를 보여준다. hazard analysis 기법의 주요 모델링 방식은 inductive/forward와 deductive/backward로 나뉜다. 이 두 가지를 나누는 기준은 특정 이벤트로부터 찾고자 하는 것으로 inductive 방식의 hazard analysis 기법은 초기 이벤트로부터 failure의 발생으로 이어질 수 있는 시나리오를 찾는 것이 주 목적이고 Deductive 방식은 failure로부터 어떤 이벤트가 해당 failure의 발생 원인인지를 찾는 것이 주 목적이 된다. 따라서 hazard analysis를 수행할 때 failure가 발생하는 시나리오를 찾는 것이 분석의 주 목적이라면 inductive/forward 방

식의 분석 기법을 사용하여야 하며, 반대로 특정 failure로부터 발생 원인을 찾고자 한다면 deductive/backward 방식의 분석 기법을 사용하여야 한다.

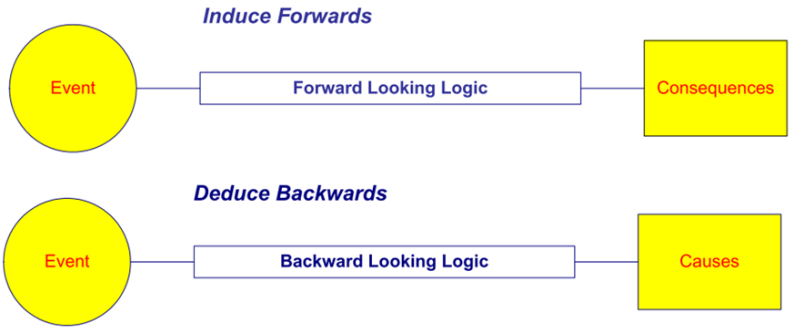


그림 7 Hazard analysis의 기본적인 두 가지 모델

대표적인 deductive 방식의 분석 기법으로는 Fault tree analysis가 있고, inductive 방식의 분석 기법으로는 Failure Mode and Effects Analysis가 있다.

3.2 Fault Tree Analysis (FTA)

3.2.1 개요

Fault tree analysis (FTA)[13]는 top-down 방식의 분석 방법으로 전자, 전기, 원자력 산업등의 분야에서 많이 사용되는 고장 해석 및 신뢰성 평가 기법이다. FTA는 top event 로부터 top-down 방식으로 basic event 까지의 순차적 원인 분석을 통해 tree 형태의 모델을 작성하는 방법으로 top event는 대상 시스템/소프트웨어의 failure 또는 accident가 해당된다. Tree 모델은 시스템의 고장을 정의하고, 고장을 발생시킨 원인을 연역적 방법을 통해 논리적으로 분석하는 형태로 수행되며 check list, common cause failure analysis 와 같은 기법의 분석 방법으로 사용되기도 하는 기법이다.

3.2.2 표기법

FTA에서는 분석의 대상이 되는 event들을 node로 나타내고 event간의 관계를 gate로 나타내어 node와 gate의 연결을 통해 tree를 생성하는 방법으로 모델을 나타낸다. 주요 event들을 나타내기 위한 node의 기호들로는 그림 8에 나온 것과 같은 것들이 있고, 주요 event들의 연결로 인한 중간 결과는 그림 9의 기호를 통해 나타낸다.

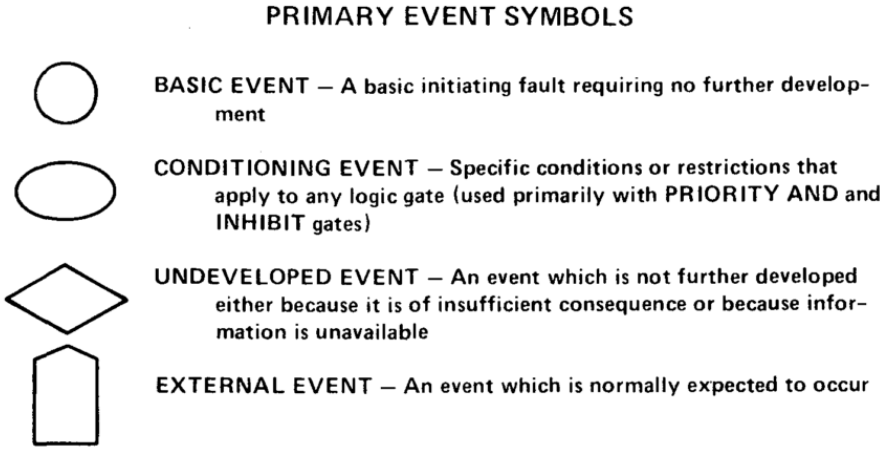


그림 8 Primary event symbol

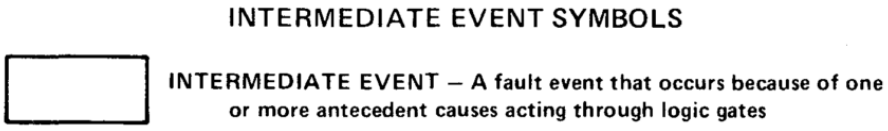


그림 9 Intermediate event symbol

상위의 event와 하위 event 간의 관계는 관계는 그림 10에 있는 gate들을 이용해 나타내게 되며 큰 규모의 tree를 나타낼 때는 그림 11에 있는 transfer symbol들을 사용한다.

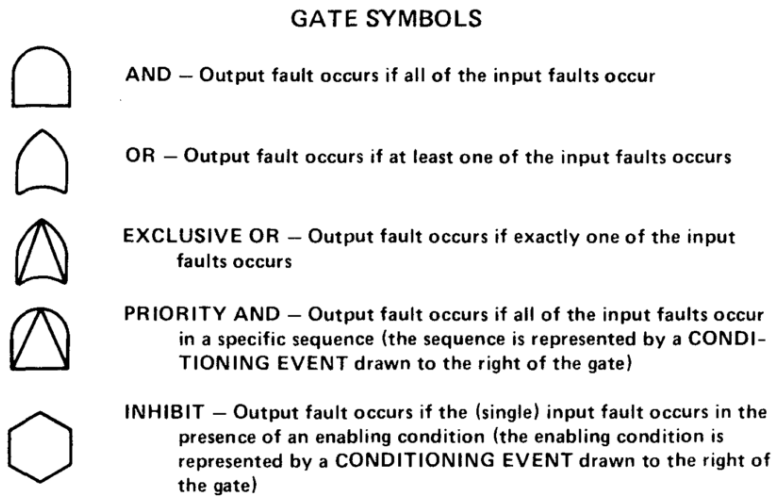


그림 10 Gate symbols

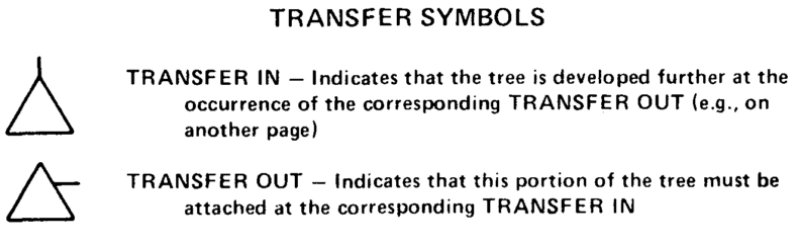


그림 11 Transfer symbols

이를 이용해 실제로 fault tree를 나타낼 경우 그림 12와 같은 형태가 된다.

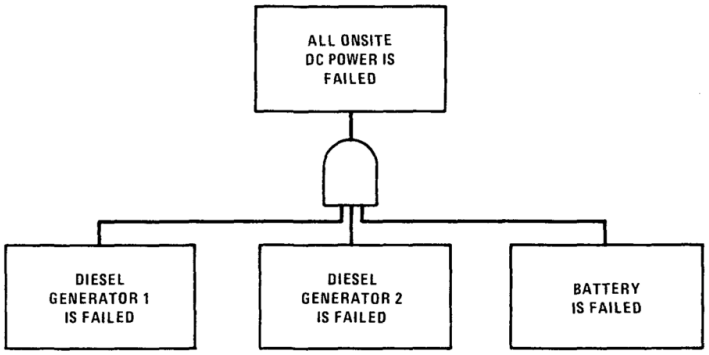


그림 12 Fault tree 예시

3.2.3 분석 절차

FTA를 이용하여 시스템의 분석을 수행할 때에는 아래와 같은 절차를 따라 분석을 수행하게 된다.

1. 대상 시스템/소프트웨어 정의
2. Top event 설정
   2.1. 분석할 대상의 사건, 고장을 선택하여 tree의 root node로 설정하는 작업
3. Fault tree 작성
   3.1. 앞에서 설명한 기호들을 이용하여 작성하며,
      3.1.1. root의 발생 원인 분석
      3.1.2. root와 원인을 게이트로 연결,
      3.1.3. 분석한 원인의 원인 분석 후 연결,

3.1.4. 원인이 분할 할 수 없는 basic event 가 될 때까지 반복하여 작성

4. Fault tree 구조 해석
5. 정량화를 통한 확률 계산
6. 해석 결과의 평가
7. 분석 결과 문서화

이 때 fault tree의 구조를 효과적으로 분석하기 위해 minimal cut-set이라는 것을 작성하며 이는 최상위 event가 발생하기 위한 최소한의 basic node들의 조합을 말한다. Fault tree의 규모가 커질수록 tree의 분석이 어려워지기 때문에 꼭 필요한 작업이 된다.

3.2.4 장•단점

- 장점
  - 사고 원인을 일반화, 간편화하여 분석할 수 있다.
  - 확률 계산을 통해 사고를 예측할 수 있다.
  - 분석 결과를 checklist로 사용할 수 있다.
  - Minimal cut-set을 이용해 사고 발생 확률이 높은 basic event를 파악함으로써 사고 예방을 위한 비용을 줄일 수 있다.
- 단점
  - Fault tree 분석을 수행하기 위한 전문가가 필요하다.
  - Fault tree의 규모가 커질수록 비용의 소모가 커진다.
  - 확률 계산을 위해서는 각각의 basic event의 발생 확률을 알아야 한다.
  - AND, OR등과 같은 논리 gate 만으로는 시간의 흐름과 관련된 표현이 어렵다.

3.2.5 적용 사례

FTA를 소프트웨어에 적용된 사례는 몇몇 가지가 존재 한다. 특히 원자력 계측제어 소프트웨어의 정형 요구사항과 설계 프로그램을 대상으로 적용된 사례가 있다. [14]는 digital nuclear plants

protection system을 위한 정형 요구사항 명세 언어를 대상으로 하여 FTA를 적용한 예제로, 요구사항 명세 언어를 자동으로 변환한다. [15][16] 은 원자력 계측제어 시스템 디자인에 사용되는 FBD 언어를 FTA로 자동으로 변환하여 분석한 사례 이다.

3.3 Failure Mode and Effects Analysis (FMEA)

3.3.1 개요

Failure mode and effects analysis는 subsystem, assemblies, components, functions 의 잠재적인 failure mode가 시스템에 미치는 영향을 상위 수준으로 분석하는 귀납적 분석 방법으로 하위의 작은 모듈/컴포넌트 의 문제가 전체 구조에 어떤 영향을 미치는 영향에 대해 분석하는데 적합한 방법이다. FMEA를 통해 분석을 수행할 경우 failure의 발생 확률을 계산 하는 것 보다 각각의 failure의 발생 빈도에 대해서 분석을 수행하게 된다. 이를 통해 System hazard와 같은 의도하지 않은 시스템 상태에 대해 확인 할 수 있다.

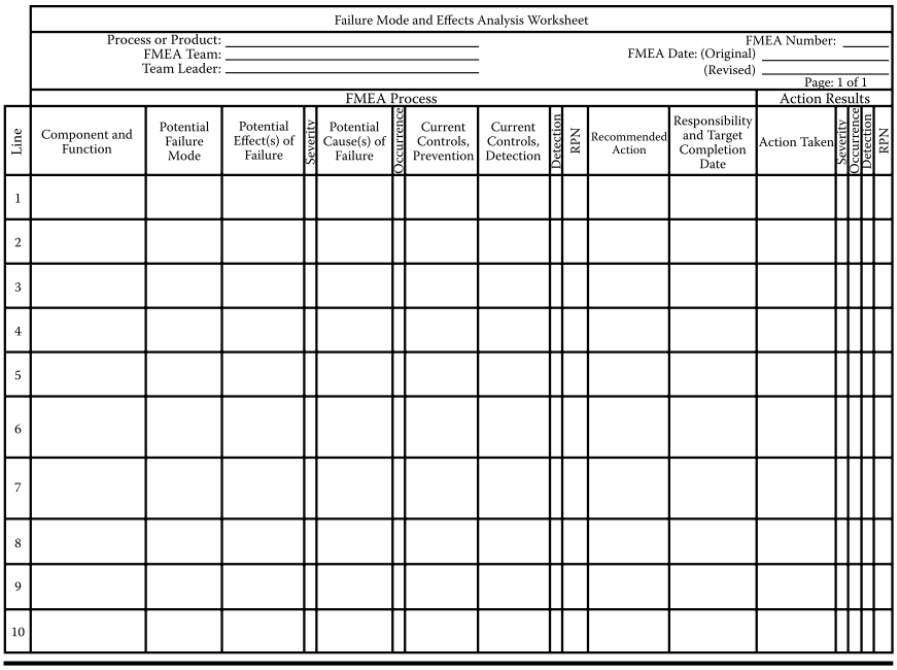
FMEA 수행에는 functional, structural, hybrid 3 가지의 접근 방법이 존재 한다. Functional 접근 방법은 system, subsystem, unit, assembly등 어떤 레벨에서의 기능을 대상으로 한 분석 방법이다. 이 접근 방법은 의도한 기능을 수행하는가에 대해 초점을 두며 시스템을 기능 단위로 모델링하여 분석하는 방법이다. 소프트웨어에 대해서 functional approach를 취하는 것으로 소프트웨어의 function을 대상으로 하여 FMEA를 수행 할 수 있다. Structural 접근 방법은 hardware와 hardware 의 잠재적인 고장 모드에 대해 초점을 맞춘 접근 방법으로 시스템을 subsystem, unit, component 단위로 모델링 하여 분석하는 방법이다. Hybrid 접근 방법은 두 가지 방식을 결합한 것이다.

FMEA 수행에 필요한 데이터는 시스템/소프트웨어 에 대한 디자인 정보이다. 이 디자인 정보는 design concept, the operational concept, major components planned 와 같은 시스템 정보로 design spec, sketches, drawings, schematics, function lists, functional block diagrams 등이 포함된다.

Software 모듈은 failure 가 아닌 incorrect behavior를 보이기 때문에 일반적으로 mechanical, electrical 시스템을 대상으로 FMEA를 수행하는 것이 software를 대상으로 하는 것에 비해 간단하다. 그럼에도 불구하고 software를 대상으로 FMEA를 수행할 경우에는 보통 software functions을 대상으로 한다.

3.3.2 워크 시트

FMEA에는 적용 분야의 분석 프로세스와 이에 맞는 worksheet가 존재하며, 분석 프로세스를 따라 worksheet에서 요구하는 내용을 채워나가며 분석을 수행한다. 그림 13은 system에서 발생할 수 있는 failure mode에 대해 분석하기 위한 worksheet의 예를 보여준다. worksheet에는 위험우선순위 (RPM, Risk Priority Number)가 있으며, 이는 대상 component/function에서 발생하는 failure model의 심각도 (Severity), 발생도 (Occurrence), 검출도 (Detection)을 곱한 값으로 FMEA 분석 후 위험우선순위가 높은 failure mode를 우선적으로 처리하게 된다.



Failure Mode and Effects Analysis Worksheet

Process or Product: _____ FMEA Number: _____
FMEA Team: _____ FMEA Date: (Original) _____
Team Leader: _____ (Revised) _____
Page: 1 of 1

	FMEA Process											Action Results				
Line	Component and Function	Potential Failure Mode	Potential Effect(s) of Failure	Severity	Potential Cause(s) of Failure	Occurrence	Current Controls, Prevention	Current Controls, Detection	Detection	RPN	Recommended Action	Responsibility and Target Completion Date	Action Taken	Severity	Occurrence	Detection	RPN
1																	
2																	
3																	
4																	
5																	
6																	
7																	
8																	
9																	
10																	

그림 13 FMEA worksheet

3.3.3 분석 절차

FMEA를 이용하여 시스템의 분석을 수행할 때에는 분석을 수행하고자 하는 시스템의 정의로 시작하여 plan의 작성 및 데이터 수집을 수항하고 worksheet를 작성하며 분석을 수행하고 최종적으로 분석 결과에 대한 적절성 평가를 수행하게 된다. 이를 정리하면 아래와 같은 순서가 된다.

1 시스템 정의
2 FMEA plan 작성
3 수행 팀 결정
4 데이터 수집
5 FMEA 수행

5.1 FMEA 수행은 그림 13과 같은 worksheet를 작성하는 방식으로 수행

6 corrective action 추천 및 평가

7 위해도 추적 및 문서화

3.3.4 장•단점

- 장점
  - 체계적으로 failure mode로부터 인과관계를 규명함으로써 failure mode에 대한 파악 및 확인이 쉽다.
  - 기법 자체의 이해 및 적용이 쉽다.
- 단점
  - 구성 요소간의 상세한 연관관계나 종속성에 대한 정보가 없으므로 분석을 수행하기 위해 해당 분야의 전문가가 필요하다.
  - Failure mode와 관련되지 않은 hazard analysis에 대한 내용이 부족하다.

3.3.5 적용 사례

FMEA를 소프트웨어에 적용한 사례로는 [17][18]이 있다. [17]은 software FMEA를 이용하여 safety-related application software를 분석한 사례이다. [17]의 대상 소프트웨어 시스템은 `software code installed at an Automatic Test and Interface Processor (ATIP) in a digital reactor protection system (DRPS)' 이다. 수행은 전체 시스템을 대상으로 hazard analysis를 수행하고, software FMEA를 각각의 program에 적용 하였다. 그림 14는 대상 소프트웨어 시스템에 대해 FMEA를 수행한 worksheet 예제 일부이다. [18] 는 작은 embedded system에 사용되는 소프트웨어를 대상으로 software FMEA를 이용해 분석한 사례로 functional approach를 취해 기능별로 software FMEA를 수행 한 사례이다. [18]는 컴포넌트를 기능별로 분리하고, 각 컴포넌트 별 failure mode 및 영향을 분석 하였다.

Software FMEA analysis results for "Channel 3 Equipment Check" sub-module.

Sub-module	Failure mode	Failure cause	Failure effect	Failure detection/comments
Other Channel 3 Equipment Check	Omission	Omission in equipment diagnosis or heartbeat monitoring functions	Periodic automatic test is affected adversely	There is no update procedure for a variable AL_1_ATIP_ATIPD_PHBC.→ Channel A, B, and C cannot recognize when the channel D HB is malfunctioning
	Incorrect realization	There is an error in equipment diagnosis or heartbeat monitoring functions	Malfunction in Channel 3 equipment diagnosis	No failure detected
	Unintended addition	Additional functions in equipment diagnosis or heartbeat monitoring functions	There is no effect on equipment diagnosis	No failure detected
	Function interaction	Inappropriate location of implementation of a function	There is an degradation in this module	No failure detected
	Input definition	Wrong input definition, assignment, address, or type definition	Malfunction in Channel 3 equipment diagnosis	No failure detected
	Input value	Wrong min/max limits or initialization value in heartbeat coefficients	Malfunction in monitoring of channel D ATIP heartbeat signal	No failure detected
	Input timing	An input variable is used before updating	Delay in processing the input signal	No failure detected
	Input format	Omission/addition of an interver, wrong input number, or input disorder	Malfunction in Channel 3 equipment diagnosis	No failure detected
	Output definition	Wrong output definition, assignment, address, or type definition	Malfunction in Channel 3 equipment diagnosis	No failure detected
	Output value	Wrong assignment of an output value	Malfunction in Channel 3 equipment diagnosis	No failure detected
	Output timing	Wrong timing for generating an output value	Malfunction in Channel 3 equipment diagnosis	No failure detected. (Output from this submodule is correctly used in the next step.)
	Output format	Omission/addition of an interver, wrong output number, or input disorder	Malfunction in Channel 3 equipment diagnosis	No failure detected

그림 14 Software FMEA 사례 worksheet 일부

3.4 Hierarchically Performed Hazard Origin and Propagation studies (HiP-HOPS)

3.4.1 개요

Hierarchically Performed Hazard Origin and Propagation studies (HiP-HOPS)는 Functional Failure Analysis (FFA), Failure Mode and Effects Analysis (FMEA)와 Fault Tree Analysis (FTA)와 같은 전통적인 hazard analysis 기법들로부터 나온 기법으로 이들을 확장, 자동화 및 통합하여 현대의 복잡한 safety problem들을 다룬다.

현대에는 전자 시스템의 복잡성이 증가하고 있지만 기존의 분석 기법들을 이에 적용하는 데는 어려움이 있으며, 크게 두 가지의 문제점이 제기되고 있다. 첫 번째는 한 시스템의 분석에 대해 서로 다른 safety analysis 기법을 적용함으로 인해 생기는 분석 결과의 불일치로 design lifecycle의 서로 다른 단계들에 여러 기법들을 적용하기 때문에 결과에 대한 파편화가 발생한다. 예를 들어, FFA는 abstract functional description을 필요로 하는 분석 기법이지만 HAZOP이나 FMEA같은 경우에는 architectural design을 필요로 한다. 두 번째 문제는 다양한 safety analysis 기법들의 분석 결과를 다시 high-level의 FFA 분석과 연관 짓기가 어렵다는 것이다. 여러 분석 기법을 사용하여 도출해낸 low-level의 component failure를 도출해내더라도 이것이 system의 design에 유용한 정보를 줄 수 없다면 분석의 의의가 떨어질 수 있다는 것이다.

HiP-HOPS는 이러한 문제들을 해결하기 위해 개발된 hazard analysis 기법으로 계층적으로 나타나는 복잡한 시스템의 통합적인 평가를 가능하게 하는 것을 목적으로 하며, 이를 사용하는 것으로 시스템의 functional level부터 low-level의 component failure mode까지 분석을 수행할 수 있다. 그림 15와 그림 16이 각각 FFA와 IF-FMEA의 적용 예시를 보여준다.

Failure	ID	Effects on System	Severity	Detection	Recovery	Recommendation
OYB: Loss of Braking (omission) when there is braking intention	FL3	The car tends to drift to the side -30% stability -18/-32% braking -15% steerability In the worst case the drift is opposite to the drivers intention	Critical	Locally, using feedback from pressure sensor Remotely, by the status reporter and monitor tasks	Not Possible	In addition, the failure can be detected by a global rotational acceleration sensor. An Electronic Stability Program device may handle the problem (this is out of the scope of this BBW system)
CNB: Unintended Braking (Commission) when there is no braking intention	FL4	The car tends to drift to the side	Critical	It is possible in certain cases by comparing the state of the pedal with the pressure sensor feedback from the wheel	Release actuator	Detection algorithm should be sufficient to detect pedal sensor failures and internal corruption of the pedal messages. There should be provisions to keep commission failures temporally limited

그림 15 Functional failure analysis 예시

Output Failure Mode	Description of output failure	Input Deviation Logic	Component Malfunction Logic	λ (f/h)
O-PEDAL1. Driver_msg	Omission of PEDAL1 output (braking demand). It can be caused by task malfunction or out of range failures of both pedal sensors	(V>max-PS1.value | V<min-PS1.value) & (V>max-PS2.value | V<min-PS2.value)	PEDAL1.task_ malfunction	1.00E-07
Vs_0-PEDAL1. Driver_msg	PEDAL1 output (braking demand) stuck at 0. It can be caused by memory stuck at 0 failures, or by stuck at minimum failures of both pedal sensors.	Vs_min-PS1.value & Vs_min-PS2.value	PEDAL1.memory_ stuck_at_0	2.00E-07

그림 16 IF-FMEA 예시

3.4.2 분석 절차

HiP-HOPS는 시스템을 Functional block diagram과 hierarchical model로 나타내며, system design의 개념 분석에 FFA를 사용하고 하드웨어나 소프트웨어 component의 분석에 FMEA를 확장한 IF-FMEA를 이용하여 분석을 수행하고 보조 프로그램을 이용한 자동화된 fault tree의 생성을 통해 FFA의 분석 결과에서 나온 functional failure가 IF-FMEA의 분석 결과로 나온 component failure mode와 연결되는지를 보여준다.

이를 정리하면 아래와 같은 단계로 나뉜다. 그림 17가 이 과정의 개요를 보여준다.

1 시스템 레벨

1.1 시스템 디자인을 functional block diagram으로 나타냄.

1.2 Functional block diagram에서 발생할 수 있는 문제들을 찾기 위해 FFA를 수행함.

1.3 FFA에서 찾아낸 single functional failure들간의 조합으로 발생할 수 있는 문제들을 분석함.

2 컴포넌트 레벨

2.1 시스템 디자인으로부터 hierarchical model을 작성함.

2.2 Hierarchical model을 대상으로 IF-FMEA를 수행함.

2.3 IF-FMEA를 이용해 모든 컴포넌트의 failure behavior를 찾음.

3 통합

3.1 FFA의 분석 결과와 IF-FMEA의 분석 결과를 연결하기 위해 fault tree의 자동 생성을 사용함.

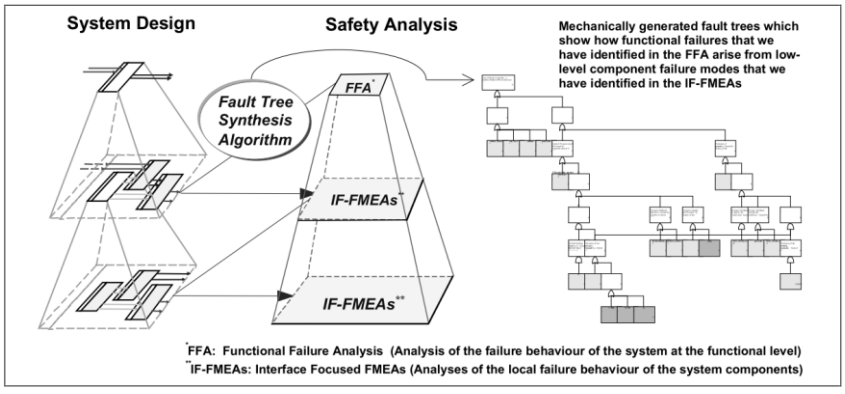


그림 17 HiP-HOPS를 이용한 design 및 safety analysis의 개요

3.4.3 장•단점

- 장점
  - ■ 큰 규모의 분석에 사용할 수 있고 합성 속도가 빠르다.
  - ■ Safety에 대한 평가에 사용하기 적절함.
- 단점
  - ■ 리얼타임과 관련된 제약사항들을 모델에 나타내기가 어렵다.
  - ■ 표준화된 입출력 포맷과의 상호 변환이 안됨. (FTA 등)

3.4.4 적용 사례

HiP-HOPS의 적용 사례로는 [19]가 있다. 여기서는 HiP-HOPS를 이용하여 Software product line (SPL)의 산출물에 대하여 평가를 수행하였다. 대상 SPL은 ISO 26262를 따르는 자동차의 하이브리드 브레이킹 시스템을 개발하기 위한 것이며 여기서 제작된 소프트웨어의 분석에 HiP-HOPS를 사용하였다. 또한 이 외에도 HiP-HOPS는 ITI GmbH사와 연계하여 해당 회사의 상용 소프트웨어인 SimulatioX에 HiP-HOPS 적용을 위한 기능을 구현하였다.

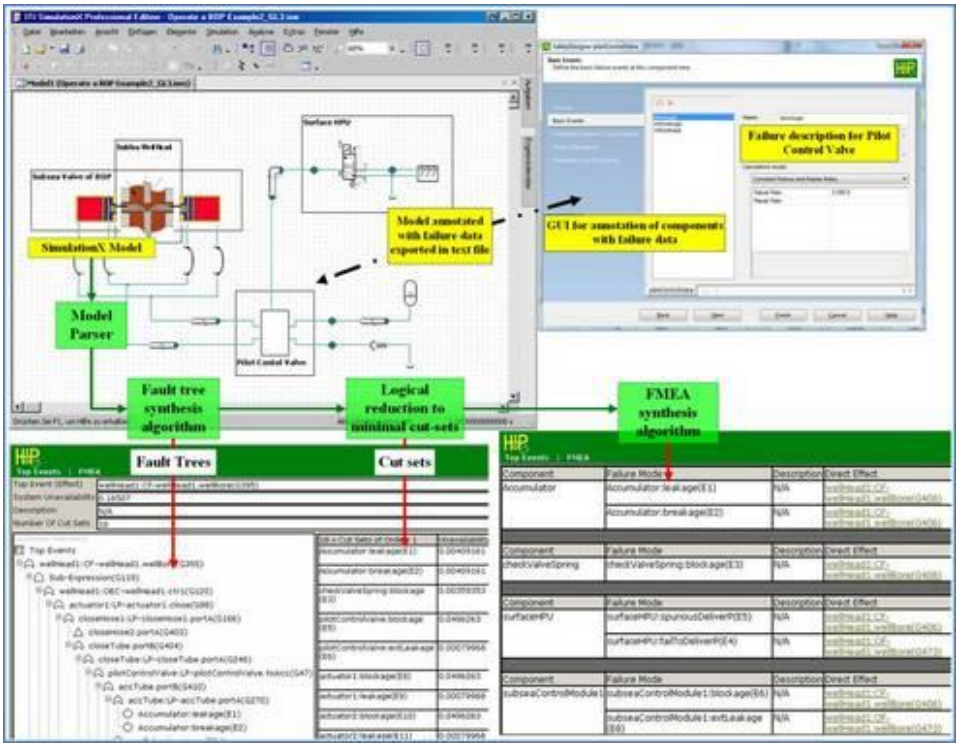


그림 18 SimulatioX를 이용한 HiP-HOPS 분석의 개요

3.5 Systems-Theoretic Process Analysis (STPA)

3.5.1 개요

STPA는 Systems theory에 기반한 System-Theoretic Accident Model and Processes (STAMP)라고 하는 accident model을 사용하는 hazard analysis 기법으로 Systems theory에서는 safety를 emergent property (창발성)라고 보아서 시스템의 일부만을 봐서는 시스템 전체에 대한 safety를 확인할 수 없다는 생각을 기본으로 하고 있기 때문에 시스템을 구성하는 컴포넌트에 문제가 없더라도 컴포넌트간의 상호작용에 의해 accident가 발생할 수 있다고 본다.

따라서 STAMP/STPA는 component failure accidents (하나나 그 이상의 컴포넌트에서 발생하는 fail로 인해 accident가 발생하는 것) 뿐만 아니라 잘못된 디자인이나 non-failing 컴포넌트간의 상호작용 등의 component interaction accidents에 대해서도 분석을 할 수 있게 디자인 되었다. STPA는 시스템을 STAMP 모델로 나타낸 후 여기서 accident의 원인이 될 수 있는 unsafe한 control action을 찾고 그 원인에 대해 분석하는 것을 주요 목적으로 한다.

3.5.2 Systems-Theoretic Accident Model and Processes (STAMP)

STAMP는 systems theory에 기반한 accident model로 기존의 event chain model의 한계로 인해 만들어졌다. STAMP의 기본 개념으로는 safety constraints, hierarchical safety control structure, 그리고 process models이다. STAMP는 이 개념들을 이용해 시스템의 모델을 나타낸다.

STAMP 모델의 제일 기본이 되는 개념이 constraints로 systems theory에서 accident는 시스템의 디자인이나 동작에 대한 constraints의 부족이 원인이라고 생각되기 때문이다. 따라서 시스템에는 각 레벨에 맞는 적절한 constraints가 필요하다. STAMP는 시스템 행동에 따라 적절한 safety constraints를 강제하게 되어있다.

Hierarchical safety control structure란 시스템의 구조를 계층적으로 나타낸 것으로 상위 레벨의 컨트롤러는 전반적인 safety 정책, 표준, 절차 등을 결정하고 이에 대한 피드백을 받는다. 하위 레벨에서는 이러한 정책이나 절차 등을 실제로 수행하는 역할을 한다. 그림 19가 이러한 예를 보여준다. 여기서는 실제 동작하는 Operating Process의 control structure 뿐만 아니라 이 시스템의 개발, 동작과 관련된 규제 기관들 간의 관계도 보여준다.

Process models는 시스템의 모델로 STAMP가 시스템과 accident에 대한 계층적인 control 구조를 나타내기 위해 사용된다. Controller는 컨트롤 되어야 하는 시스템의 모델을 포함하고 있어야 하며, 여기에는 시스템 변수간의 관계, 시스템의 현재 상태, 프로세스의 상태가 변할 수 있는 방식이 나타나야 한다. 이 정보를 이용해 Controller는 Controlled Process에 특정 동작을 수행하게 하는

Control action을 내보내고 이에 대한 Feedback을 받는다. 그림 20이 Process model과 control loop의 관계를 보여준다.

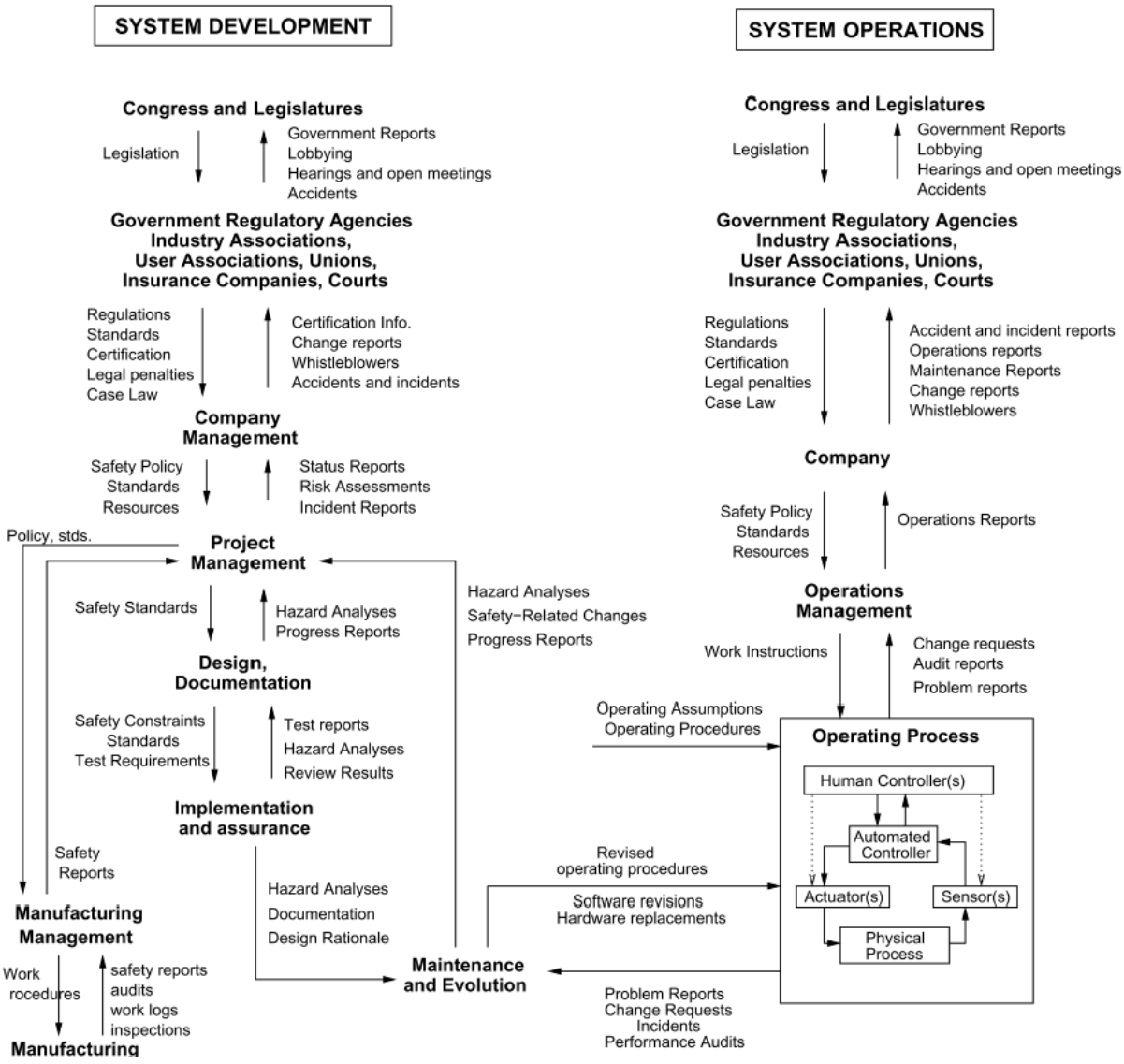


그림 19 Hierarchical safety control structure의 예

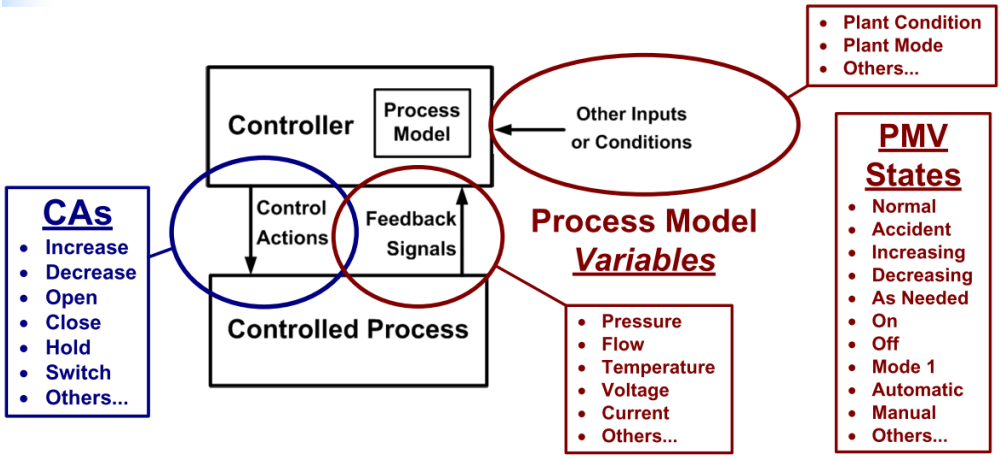


그림 20 Control loop와 control action, process model의 예 [20]

3.5.3 분석 절차

STPA의 분석 단계는 크게 4 단계로 분류할 수 있으며, STAMP 모델을 작성하는 단계와 여기에서 hazard의 원인을 분석해내는 단계로 나누어진다.

1. 시스템 레벨의 accident와 hazard, safety constraints의 식별
2. Control structure의 구축
3. STPA step 1 : unsafe control action의 식별
4. STPA step 2 : unsafe control action의 잠재 원인 식별

위의 4단계 중 앞의 1, 2 단계가 시스템의 요구사항 및 문서를 보고 시스템 레벨의 분석을 수행하고 STAMP 모델을 작성하는 단계이고, 뒤의 두 단계가 STPA에 해당한다. Control structure의 작성이 완료되면 여기서 발생할 수 있는 unsafe control action을 찾게 되며 STPA에서는 이 unsafe control action을 네 가지로 분류하고 있다

1. Not providing control action causes hazard
2. Providing control action causes hazard
3. Wrong timing/order of control action causes hazard
4. Control action stopped too soon/applied too long

위와 같은 4가지에 해당하는 시스템 레벨의 unsafe control action을 찾은 후 STPA step 2단계에서

이에 대한 원인을 분석하고 발생 가능한 시나리오의 확인 후 이를 제거하게 된다.

3.5.4 장•단점

- 장점
  - FTA나 FMEA같은 chain of event 기반의 기존 모델을 사용하는 기법에서 고려하지 못하는 사항들을 분석 가능.
  - Human/socio technical한 부분까지도 분석의 대상에 포함하여 분석 가능.
- 단점
  - STPA는 functional analysis 기법이므로 자세한 구현 수준에 대한 분석은 다루지 않음.

3.5.5 적용 사례

[21]에서는 NASA에서 STPA를 이용하여 일본의 JAXA와 함께 H-Ⅱ Transfer Vehicle에서 발생할 수 있는 hazard와 그 원인에 대해 분석한 사례가 있고 [20][22][23]에서는 원자력 발전소의 컨트롤 시스템을 대상으로 하여 발생할 수 있는 accident에 대해 분석을 수행하였으며, [24]에서는 의료기기의 분석 등 다양한 분야에 사용됨을 확인할 수 있다.

4 Reference

1. Lawrence, J. Dennis. *Software safety hazard analysis*. Division of Reactor Controls and Human Factors, Office of Nuclear Reactor Regulation, US Nuclear Regulatory Commission, 1996.
2. Radatz, Jane, Anne Geraci, and Freny Katki. "IEEE standard glossary of software engineering terminology." *IEEE Std* 610121990.121990 (1990): 3.
3. IEEE Std. 1074 - 1997, “Standard for Developing Software Life Cycle Processes”.
4. IEEE Std. 1228 - 1994, “Standard for Software Safety Plans”
5. IEEE Std. 829 - 1998, “Standard for Software Test Documentation”

6. IEEE Std. 1008 - 1987, R1993, “Standard for Software Unit Testing”.

7. IEEE Std. 1012 - 1998, “Standard for Software Verification and Validation”.

8. ISO, EN. "IEC 61508." *Functional Safety of Electrical/Electronic/Programmable Electronic Systems* Part 1 (1998): 1998-2001.

9. ISO/DIS 26262-8:2009. Draft International Standard Road vehicles — Functional safety - Part 8: Supporting processes. 2009.

10. RTCA/DO-178B. “Software Considerations in Airborne Systems and Equipment Certification.” 1992

11. RTCA/DO-178C. “Software Considerations in Airborne Systems and Equipment Certification.” 2011.

12. IEC 61513, Nuclear Power Plants – Instrumentation and control for systems important to safety – General requirements for systems

13. Vesely, William E., et al. *Fault tree handbook*. No. NUREG-0492. Nuclear Regulatory Commission Washington DC, 1981.

14. "NuFTA: A CASE Tool for Automatic Software Fault Tree Analysis", Sanghyun Yun, Dong-Ah Lee and Junbeom Yoo, Transactions of the Korean Nuclear Society Spring Meeting 2010, pp.855-856, 2010

15. "고장수목을 이용한 Function Block Diagram의 위험성 분석 기법 연구", 이동아, 김의섭, 유준범, 한국정보과학회 제39회 추계발표회 (KIISE 2012), Vol.39, No.2(B), pp.76-78, 2012.

16. "Software Safety Analysis of Function Block Diagrams using Fault Trees", Younju Oh, Junbeom Yoo, Sungdeok Cha, Han Seong Son, Reliability Engineering and System Safety, Vol.88, No.3, pp215-228, 2005.

17. "Software FMEA analysis for safety-related application software", Gee-Yong Park, Dong-Hoon Kim, Dong Young Lee, Annals of Nuclear Energy, Vol 70, pp.96-102, 2014

18. "Software Failure Modes and Effects Analysis For a Small Embedded Control System", John B. Bowles, Chi Wan, Reliability and Maintainability Symposium, 2001. Proceedings. Annual, 2001

19. de Oliveira, André L., et al. "A Model-Based Approach to Support the Automatic Safety Analysis of Multiple Product Line Products."

20. Torok, R., and B. Geddes. "Systems Theoretic Process Analysis (STPA) Applied to a Nuclear

Power Plant Control System." *Presentation at MIT STAMP Workshop*. 2013.

21. Ishimatsu, Takuto, et al. "Hazard analysis of complex spacecraft using systems-theoretic process analysis." *Journal of Spacecraft and Rockets* 51.2 (2014): 509-522.

22. Lee, Dong-Ah, et al. "Application of system–theoretic process analysis to engineered safety features-component control system." *Proceedings of the 37th enlarged halden programme group (EHPG) meeting, Gol, Norway*. 2013.

23. Thomas IV, John P. *Extending and automating a systems-theoretic hazard analysis for requirements generation and analysis*. Diss. Massachusetts Institute of Technology, 2013.

24. Samost, Aubrey. "Evaluating Systems with Multiple Processes Using STPA: A Case Study in a Medical Intensive Care Unit."