An Extended Process of STPA and Implementation of an Automatic Assistant Tool for Reactor Protection System Software

2015.12.07

서영주

Dependable Software Laboratory
KU 건국대학교 KONKUK UNIV.

Contents

Contents

Introduction

개요

- 소프트웨어의 발전으로 인한 시스템 내 소프트웨어의 중요성 증가
- Safety-critical system의 소프트웨어와 관련된 hazard analysis (HA)의 필요성 증가

STPA

- 기존의 HA 기법들과 다른 accident model을 사용하는 기법
- System theory에 기반한 accident model을 사용함
- Safety를 component 수준이 아닌 전체 시스템 레벨에서 생각함

STPA 적용의 한계점

- STPA는 일반적으로 시스템 레벨의 HA를 수행하는데 사용됨
- 구현 수준의 문제로 accident가 발생할 수 있음
- 일반적인 STPA 분석 시 구현 레벨에서 hazard의 원인 분석이 어려움
- STPA 분석 수행 시 요구사항 및 문서들을 이용한 수동 분석을 수행함

목적

- STPA를 구현 레벨까지 수행하기 위한 프로세스와 자동화 도구의 제안
- 시스템 레벨의 STPA 분석 결과와 구현 레벨의 분석간의 연관관계를 보임
- 구현 레벨의 분석 시 hazard의 원인을 효과적으로 분석할 수 있음을 보임

Case study

- 도구를 통한 자동화를 보이기 위한 case study
- 시스템 레벨-구현 레벨 간의 연관관계를 보이기 위한 case study
- 원자로 보호 시스템의 소프트웨어에 대한 formal specification을 이용해 case study를 수행함

Contents

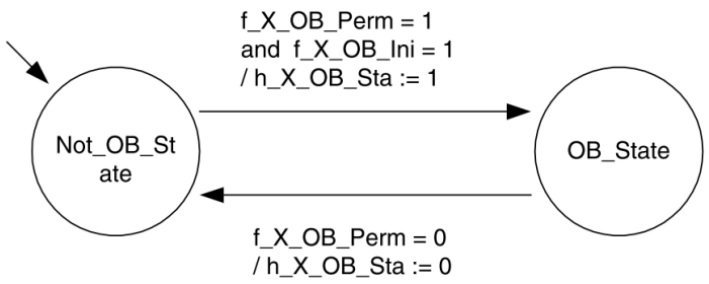
Background

NuSCR

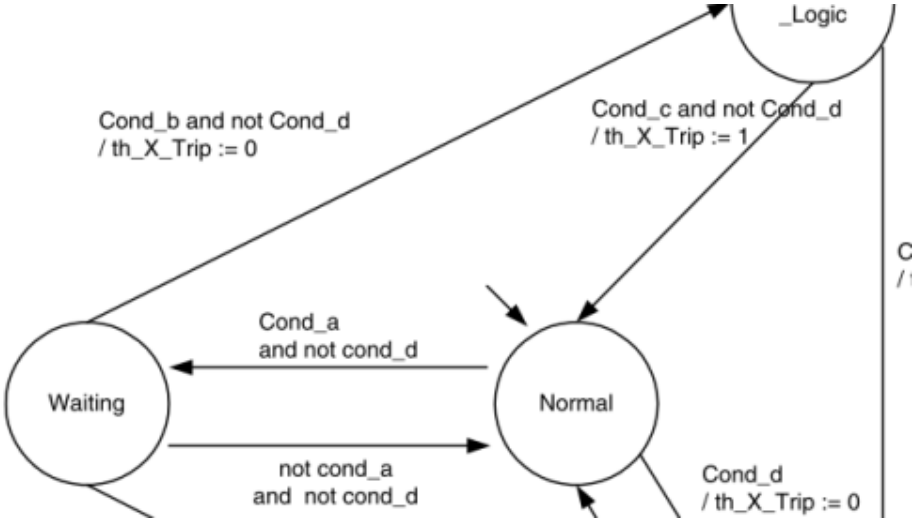
- 원자로 보호 시스템의 동작을 명세하기 위한 formal software requirement specification
- Structured Decision Table (SDT), Finite State Machine (FSM), Timed Transition System (TTS) 의 세 가지 요소를 이용해 시스템의 동작을 나타냄

Conditions	1	2
f_VAR_OVER_PWR_MT_Query = true	F	T
Action	**1**	**2**
f_VAR_OVER_PWR_Val_Out := f_VAR_OVER_PWR_PV	O	
f_VAR_OVER_PWR_Val_Out := f_VAR_OVER_PWR_Manu_Test		O

SDT



FSM

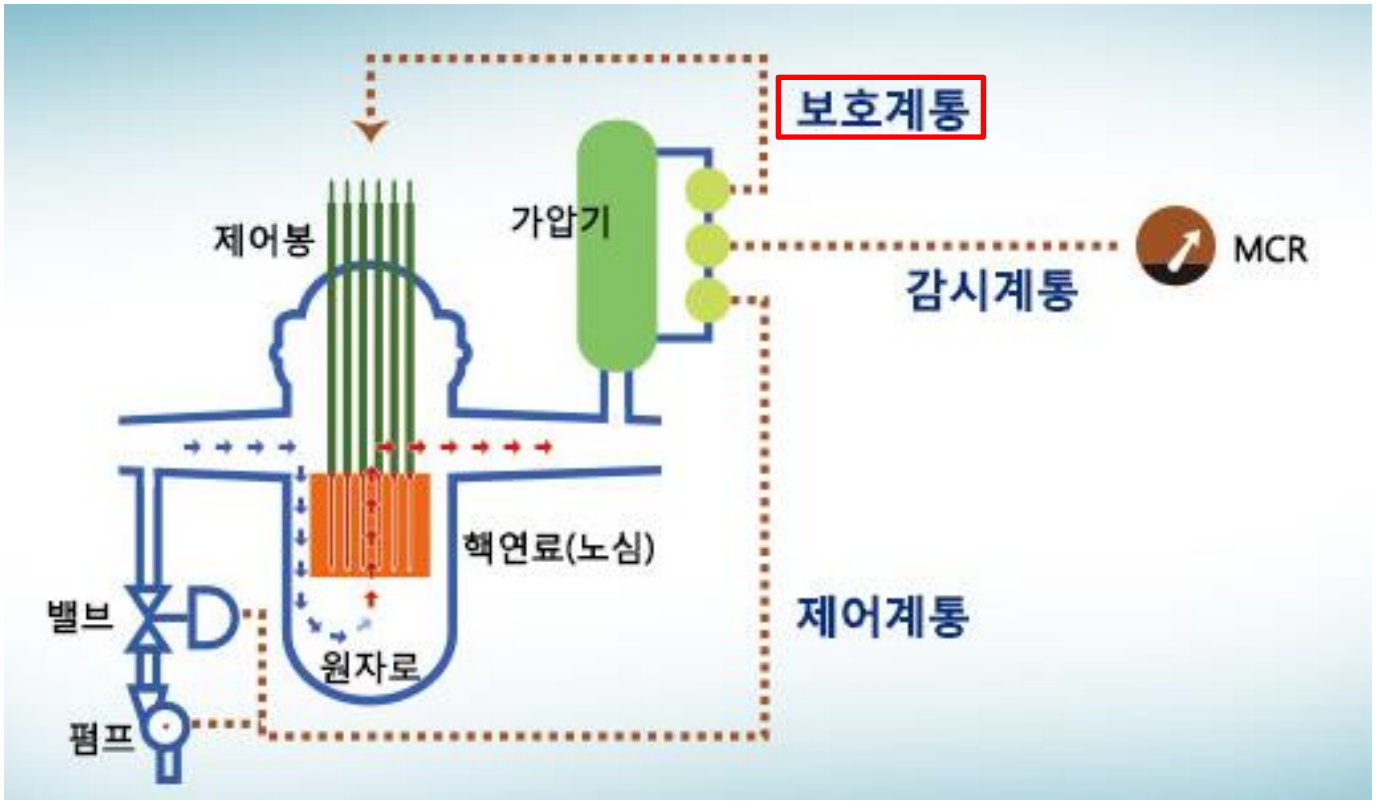


Cond_a : f_X >= k_X_Trip_Setpoint
Cond_b : [k_Trip_Delay, k_Trip_Delay]
(f_X >= k_X_Trip_Setpoint
and h_X_OB_Sta = 0)

TTS

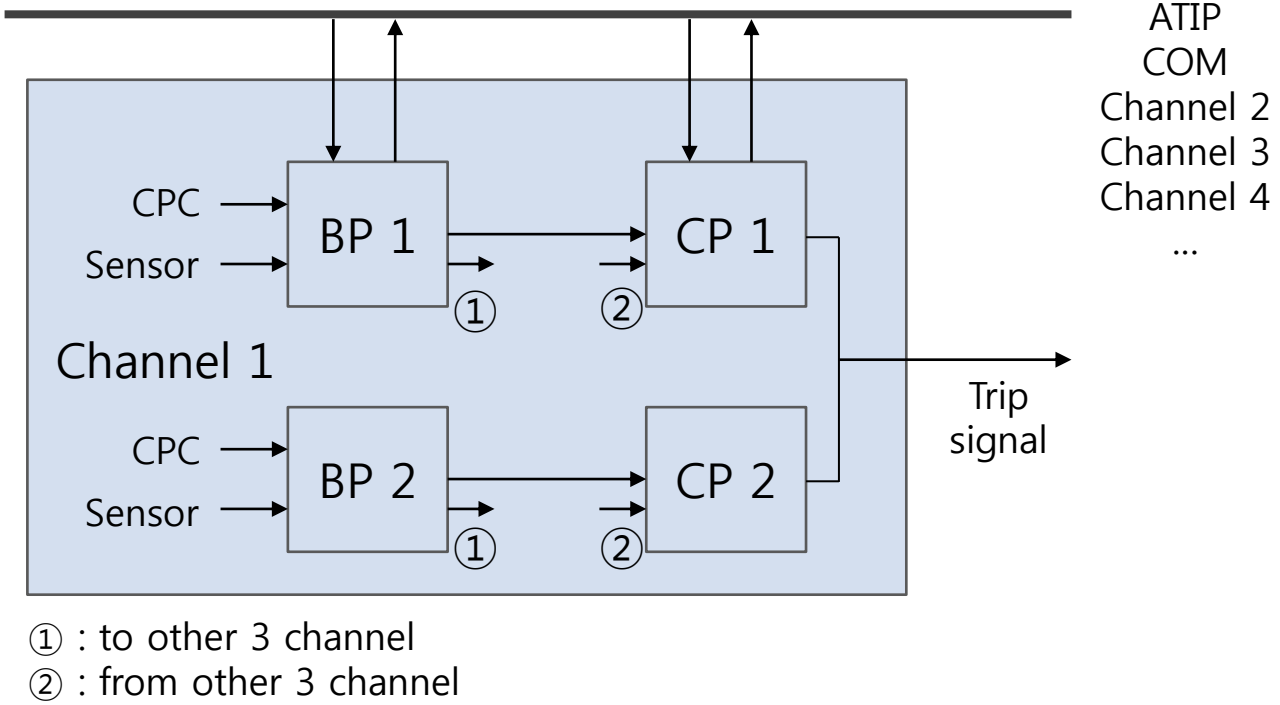
Reactor Protection System (RPS)

- 원자로의 상태를 확인하여 문제가 발생했을 시 원자로를 안전하게 정지시킴
- Programmable logic controller (PLC)라고 하는 하드웨어로 이루어진 시스템



Bistable Processor (BP)

- 센서 값, 사용자 입력 등의 외부 정보를 받아 연산을 수행함
- 연산 결과에 따라 원자로 정지 신호 (Trip signal)를 내보냄
- 내부 동작은 소프트웨어로 구현됨
- NuSCR을 이용해 해당 소프트웨어의 동작을 기술할 수 있음



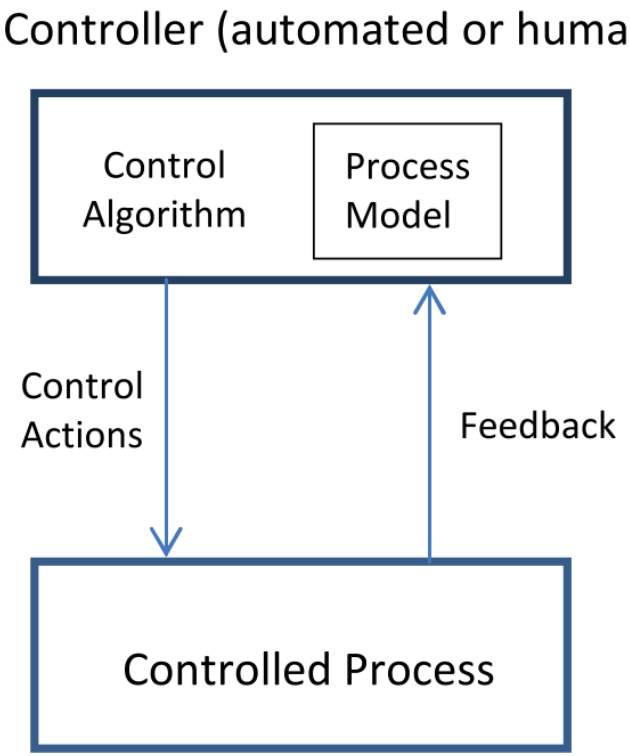
RPS 내부 구조도

Contents

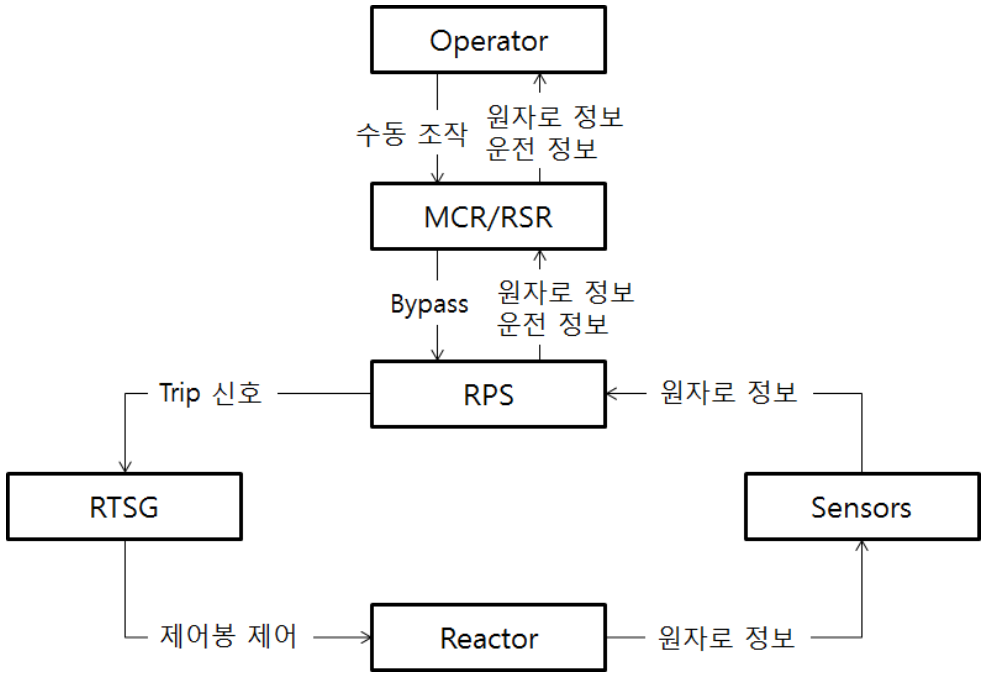
STAMP/STPA

System-Theoretic Accident Model and Processes (STAMP)

- System theory에 기반하여 시스템을 표현하기 위한 모델
- 전체 시스템을 Control structure라는 형태로 나타냄
- Control structure는 여러 Control loop의 조합으로 나타남



Control loop 개념도



RPS의 control structure

Systems-Theoretic Process Analysis (STPA)

- STAMP 모델을 이용해 시스템에서 발생할 수 있는 hazard의 원인을 찾아내기 위한 hazard analysis 기법
- 각 컴포넌트에 문제가 없더라도 컴포넌트간 컨트롤로 인해 accident가 발생할 수 있다고 봄
- 목적 : Accident의 발생의 원인이 될 수 있는 unsafe한 control action (UCA)의 발견 및 원인 분석

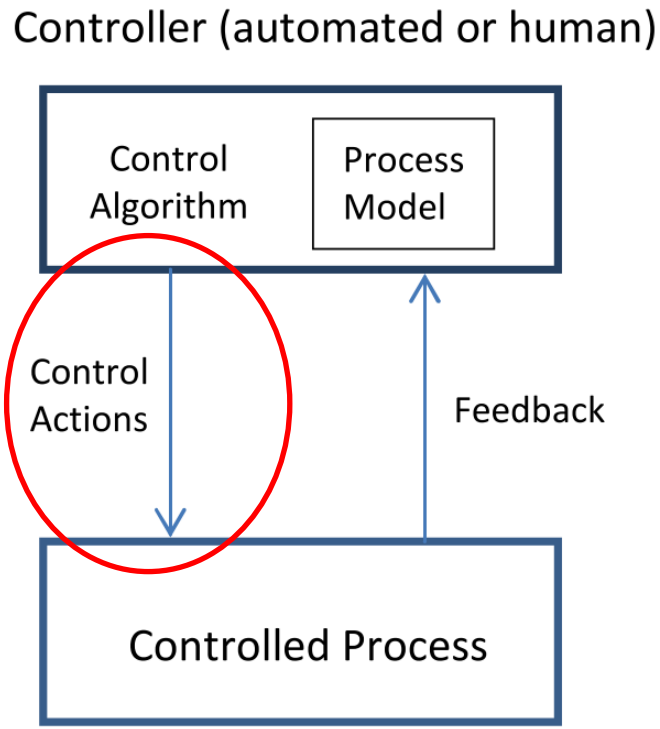
STPA process

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

Unsafe control action 분류

- STPA에서 control action (CA)이 unsafe일 수 있는 경우를 4가지로 분류

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



Control Action	Not providing causes hazard (A)	Providing causes hazard (B)	Wrong timing or order causes hazard (too late) (C)	Stopped too soon or applied too long
Trip	Trip신호가 다음의 상황을 하나라도 만족하는 상황에서 발생하지 않음:	Trip신호가 다음의 상황을 하나라도 만족하지 않는 상황에서 발생함	Trip신호가 다음의 상황을 하나라도 만족하는 상황에서 너무 늦게 발생함:	-

Contents

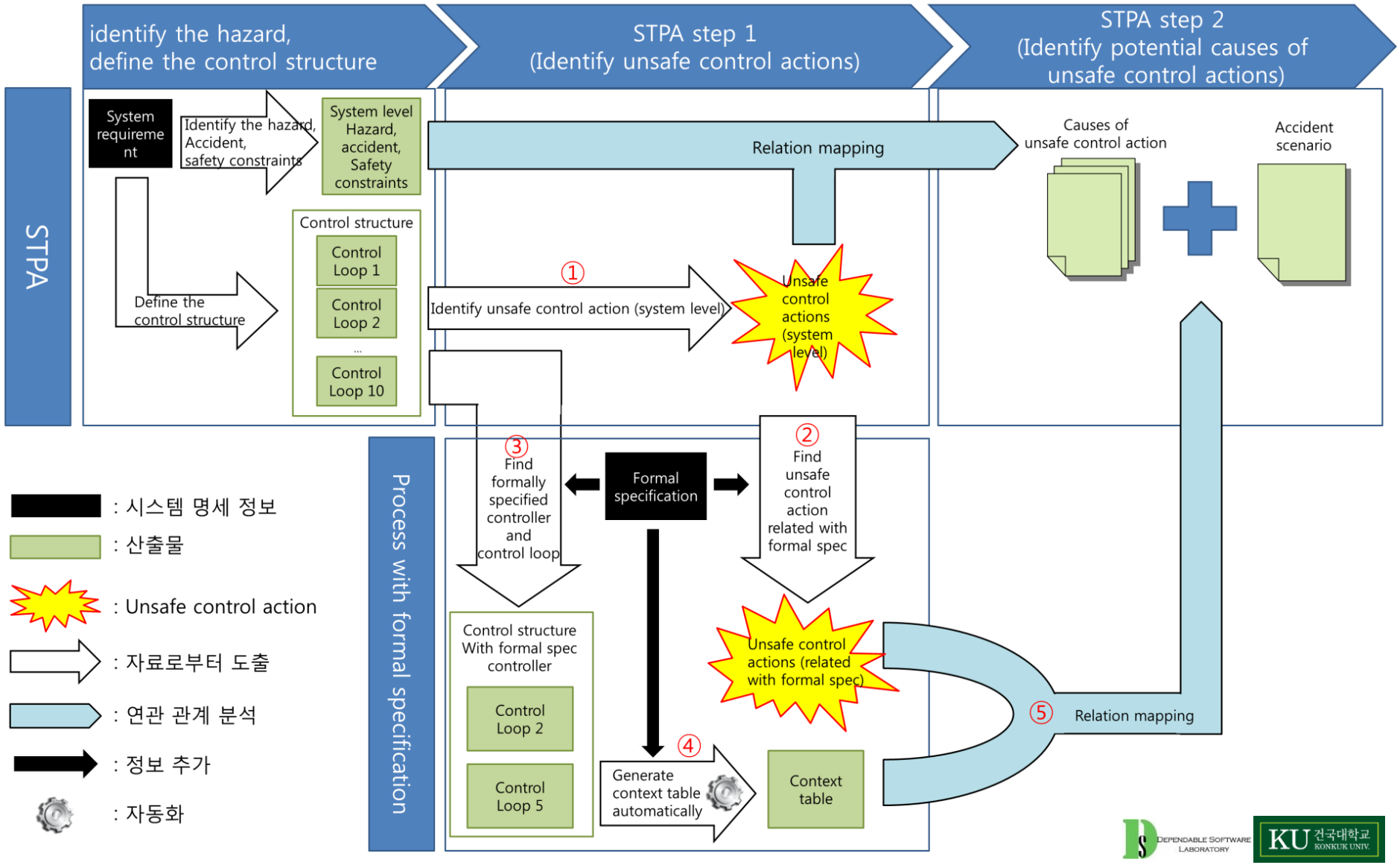
Automatic assistant for STPA on RPS software

Process

- 기존의 STPA에 확장된 과정을 추가

1. 기존의 시스템 레벨의 STPA를 step 1까지 수행
2. Formal specification을 참고하여 시스템 레벨의 UCA 중 소프트웨어의 동작과 관련된 것을 분류
3. Formal specification을 참고하여 Control structure의 control loop 중 소프트웨어가 CA에 영향을 주는 것들을 분류
4. Formal specification과 자동화 도구를 이용해 control loop의 process model과 실행 결과를 테이블로 작성
5. 테이블과 시스템 레벨의 UCA간의 연관관계를 분석하여 STPA step 2의 분석에 사용

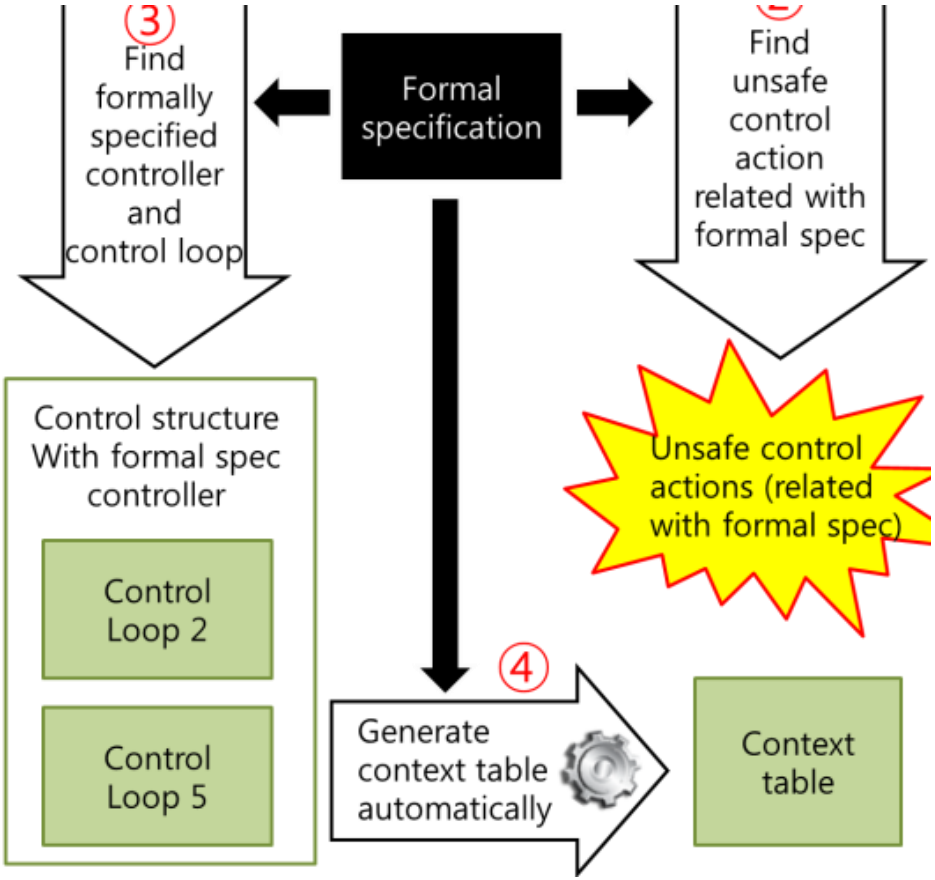
Process


identify the hazard, define the control structure
STPA step 1 (Identify unsafe control actions)
STPA step 2 (Identify potential causes of unsafe control actions)
STPA
System requirement
Identify the hazard, Accident, safety constraints
System level Hazard, accident, Safety constraints
Define the control structure
Control structure
Control Loop 1
Control Loop 2
...
Control Loop 10
Relation mapping
①
Identify unsafe control action (system level)
Unsafe control actions (system level)
Causes of unsafe control action
Accident scenario
Process with formal specification
③
Find formally specified controller and control loop
Formal specification
②
Find unsafe control action related with formal spec
Control structure With formal spec controller
Control Loop 2
Control Loop 5
④
Generate context table automatically
Context table
Unsafe control actions (related with formal spec)
⑤
Relation mapping
: 시스템 명세 정보
: 산출물
: Unsafe control action
: 자료로부터 도출
: 연관 관계 분석
: 정보 추가
: 자동화

Automatic assistant

- 구현 레벨의 분석에서 생길 수 있는 문제점
  - 많은 연산을 필요로 하고 복잡함

- 자동화를 통해 분석 정확도 향상 및 비용 감소를 실현
  - Process model 및 context table의 생성 : ContextTable Maker
  - Context table 정보의 평가 : NuSCR Runner



Automatic assistant

- ContextTable Maker
  - NuSCR formal specification으로부터 변수들의 정보, 초기 값, 상수 값 등을 추출
  - 내부 SDT/FSM/TTS의 transition 및 condition들로부터 변수들의 경계값을 추출하여 Process model 생성
  - Process model의 조합으로 나올 수 있는 경우들을 context table로 저장
- NuSCR Runner
  - Python의 스크립트 언어 특성을 NuSCR formal specification의 수식 계산에 사용
  - ContextTable Maker가 만든 table의 값에 대한 연산 결과를 NuSCR formal specification을 이용해 평가
  - SDT/FSM/TTS의 연결 관계에 따라 계산을 수행하여 결과값을 순차적으로 출력함

```
----------------SDT :  f_VAR_OVER_PWR_Val_Out -----------------
SDT condition :  f_VAR_OVER_PWR_MT_Query == True
condition column :  1
selected SDT action :  f_VAR_OVER_PWR_Val_Out  = f_VAR_OVER_PWR_PV
Result :  f_VAR_OVER_PWR_Val_Out  =  0
----------------FSM :  h_VAR_OVER_PWR_Int_SP -----------------
currentState :  Hi_Limit
transition Id :  1265
FSM condition :  (f_VAR_OVER_PWR_Val_Out -f_VAR_OVER_PWR_Val_Out_t1 <=k_VAR_OVER_PWR_100ms_Rate)
                &(f_VAR_OVER_PWR_Val_Out +k_VAR_OVER_PWR_Trip_Step >=k_VAR_OVER_PWR_SP_Lo_Lim)
                &(f_VAR_OVER_PWR_Val_Out +k_VAR_OVER_PWR_Trip_Step <=k_VAR_OVER_PWR_SP_Hi_Lim)
FSM condition Result :  True
FSM assignments :  h_VAR_OVER_PWR_Int_SP  =f_VAR_OVER_PWR_Val_Out +k_VAR_OVER_PWR_Trip_Step
Result :  h_VAR_OVER_PWR_Int_SP  =  2100
----------------TTS :  th_VAR_OVER_PWR_Trip_Logic -----------------
currentState :  Trip
transition Id :  1363
FSM condition :  f_VAR_OVER_PWR_Val_Out <=h_VAR_OVER_PWR_Int_SP -k_VAR_OVER_PWR_Trip_Hyst
FSM condition Result :  True
FSM assignments :  th_VAR_OVER_PWR_Trip_Logic  = False
Result :  th_VAR_OVER_PWR_Trip_Logic  =  False
```

NuSCR Runner 실행 결과

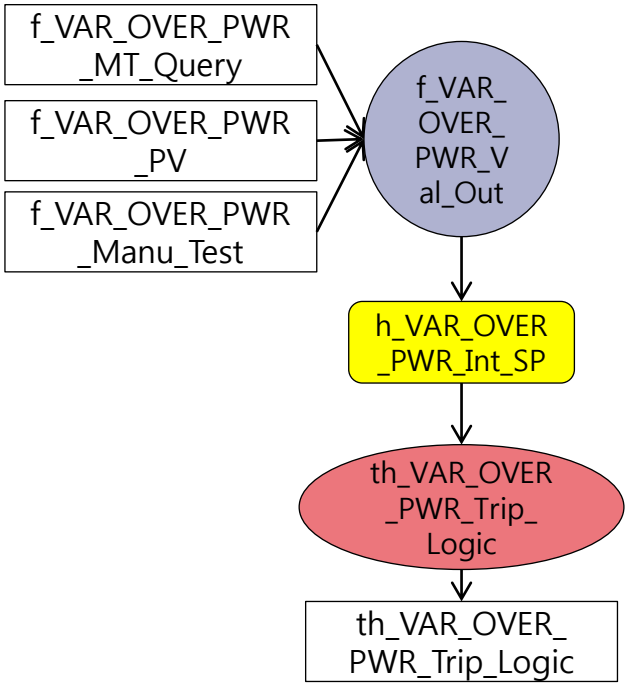
```
th_VAR_OVER_PWR_Trip_Logic=true
f_VAR_OVER_PWR_MT_Query=true
f_VAR_OVER_PWR_Manu_Test=0
f_VAR_OVER_PWR_PV=0
h_VAR_OVER_PWR_Int_SP=0
f_VAR_OVER_PWR_Val_Out_t1=0
h_VAR_OVER_PWR_Int_SP_State=Hi_Limit
th_VAR_OVER_PWR_Trip_Logic_State=Trip
th_VAR_OVER_PWR_Trip_Logic_Start_Time=479
th_VAR_OVER_PWR_Trip_Logic_End_Time=479
```

ContextTable Maker 실행 결과

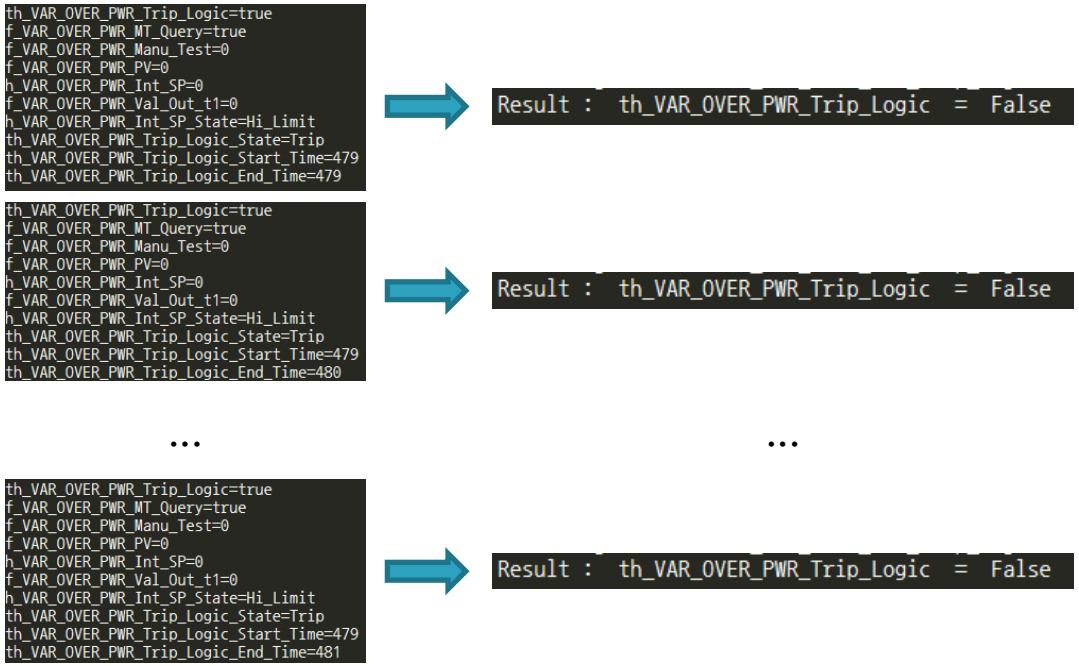
Contents

Automatic assistant

- 대상 : SDT/FSM/TTS를 하나씩 포함하는 NuSCR formal specification
- 목적 : Context table의 자동 생성 및 결과 판정의 자동화 (프로세스의 ④) 확인



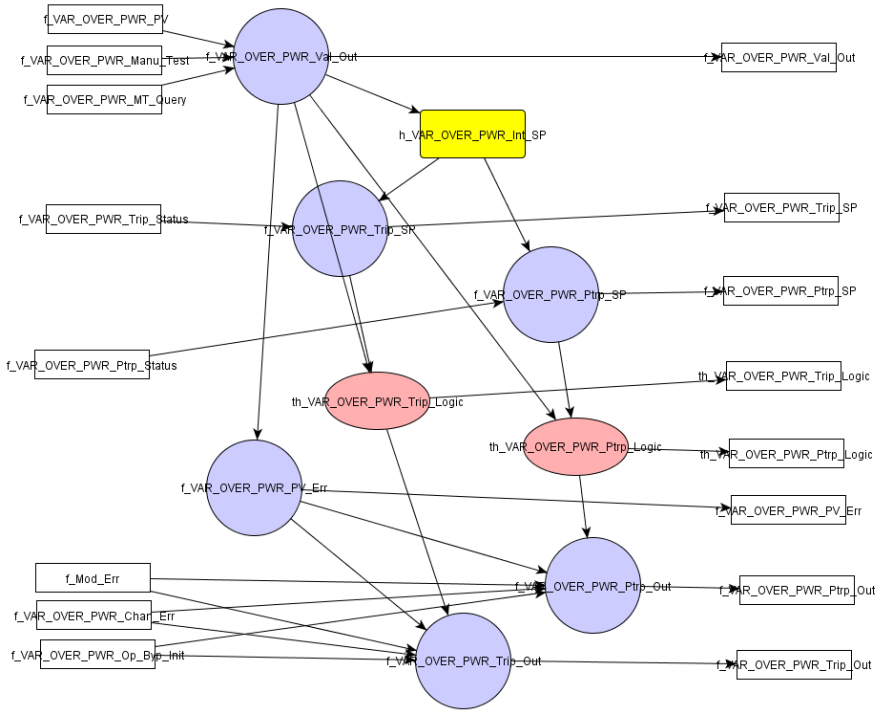
자동화 Case study용 NuSCR 구조



29160개의 context table 조합 생성 및
각각의 판정이 가능

System level mapping

- 대상 : RPS의 BP 내 한 logic (가변 과출력)
- 목적 : system level의 소프트웨어 Unsafe control action (프로세스 ②의 결과)와 RPS의 logic (프로세스 ③의 결과)의 자동화 분석 (프로세스 ④) 후 mapping (프로세스 ⑤) 확인

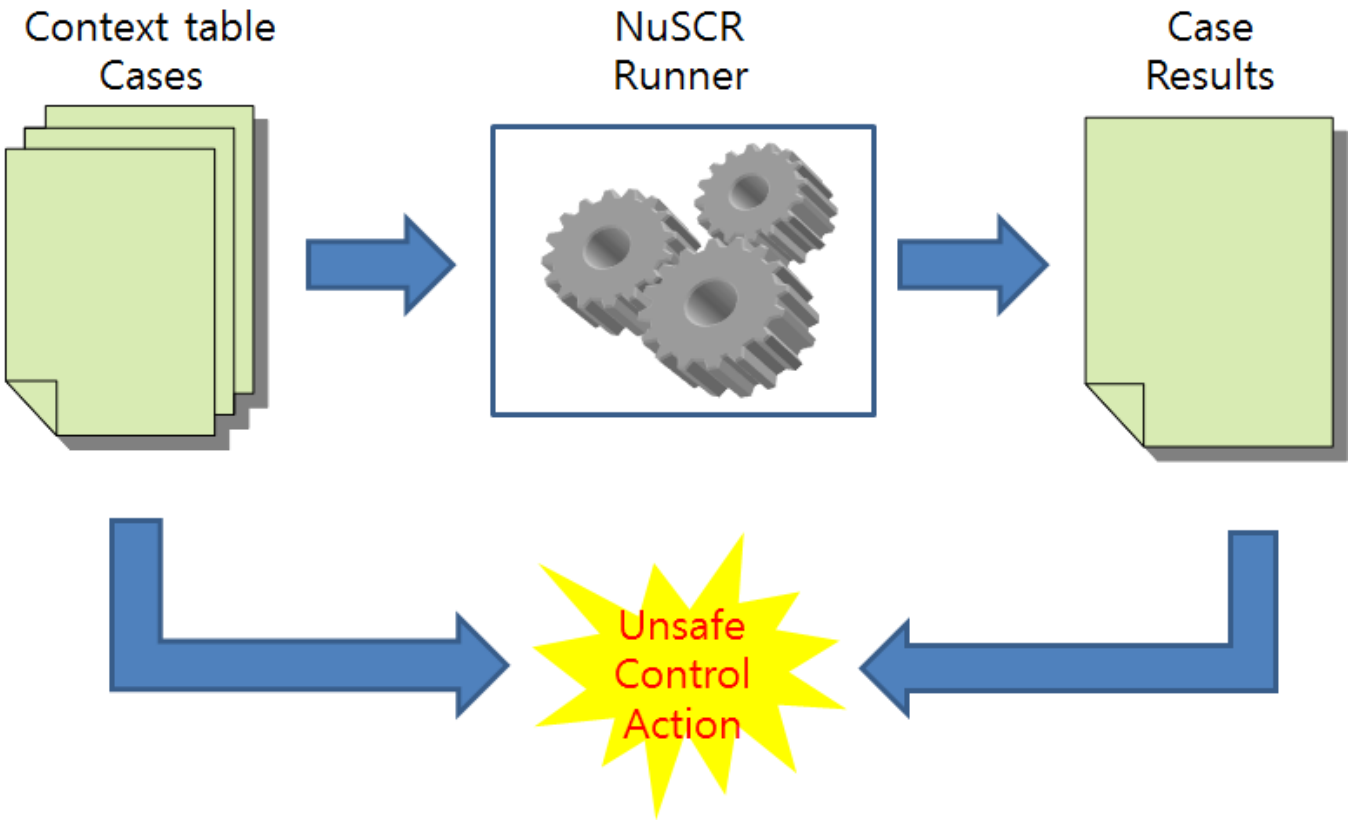


BP의 logic 중 가변 과출력 logic

Variable의 일부와 조합 예

System level mapping

- 자동 분석 결과로부터 UCA이 발생할 수 있는 경우를 찾음
- 구현 레벨의 UCA : 자동화 도구의 연산 결과 (프로세스 ④의 결과)와 다른 결과가 나올 경우



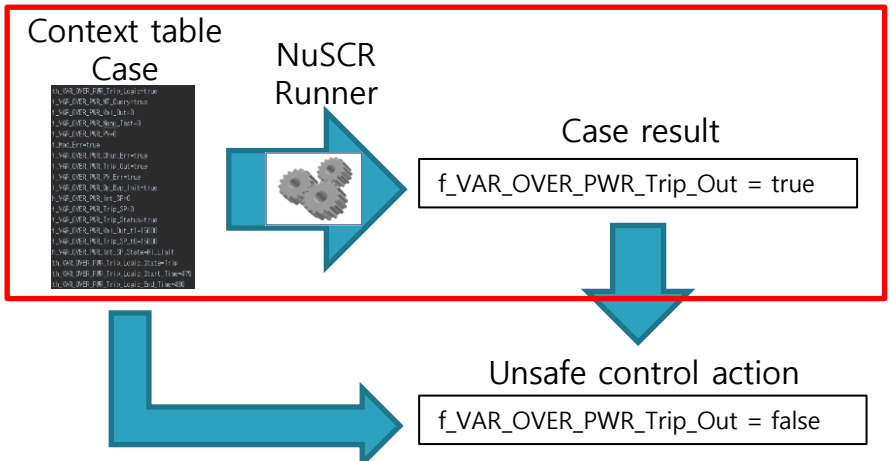
System level mapping case study

- 프로세스 ④에서 찾은 UCA와 system level의 UCA간의 mapping을 수행
- Mapping 결과를 STPA step 2에서 사용할 수 있음

프로세스 ② 결과

Control Action	Not providing causes hazard (A)	Providing causes hazard (B)	Wrong timing or order causes hazard (too late) (C)	Stopped too soon or applied too long
Trip	Trip신호가 다음의 상황을 하나라도 만족하는 상황에서 발생하지 않음:	Trip신호가 다음의 상황을 하나라도 만족하지 않는 상황에서 발생함	Trip신호가 다음의 상황을 하나라도 만족하는 상황에서 너무 늦게 발생함:	-
	1. 중성자속 출력이 (원자로 저출력 기동 시) 급격히 허용기준 이상의 변화율로 증가 2. 중성자속 출력이 (원자로 정상출력 운전 시) 설정된 최대 값을 초과 3. 중성자속 대수(log)출력이 허용치 이상으로 올라갈 경우 4. 계산된 노심최대국부출력밀도가 허용치 이상으로 증가할 경우 5. 계산된 핵비등이탈율이 사용치 이하로 감소하였을 경우 6. 가압기 압력이 설정치 이상으로 상승할 경우 7. 가압기 압력이 설정치 이하로 떨어질 경우 8. 증기발생기 수위가 트립설정치 이하로 떨어질 때 9. 증기발생기 수위가 트립설정치 이상으로 상승할 때 10. 격납용기 내부압력이 트립설정치 이상으로 상승할 경우 11. 증기발생기 압력이 트립설정치 이하로 하강할 때 12. 증기발생기 1차 측 전단 및 후단에서 측정된 압력차(유량)가 급격한 변화율로 감소할 때 13. 증기발생기 1차 측 전단 및 후단에서 측정된 압력차(유량)가 설정치 이하로 감소할 때			

프로세스 ④



⑤ Relation mapping

Contents

Conclusion

Contribution

- STPA를 이용한 HA 시 시스템 레벨의 분석 결과와 구현 레벨의 연관 관계 확인 가능
- 구현 레벨의 분석을 통해 사고의 원인/시나리오의 자세한 분석이 가능함
- 자동화를 통해 분석 오류 및 비용의 감소 기대

Limitation

- Formal specification을 이용한 경계값 추출이 너무 많은 경우의 수를 만들게 됨
- 구현 레벨의 분석 결과와 시스템 레벨의 UCA간의 연관관계를 찾기 위한 해석의 어려움

Future works

- Don’t care case를 생략하는 식으로 생성되는 구현 레벨의 경우의 수를 줄임
- 구현 레벨의 분석 결과의 해석을 돕기 위한 방법을 추가적으로 연구

감사합니다