KCSE 2019
2019년 1월 28일(월) ~ 30일(수),
강원도 평창 한화리조트 (휘닉스파크점)

사이버-피지컬 시스템의 상호운용성 상충관계 분석에 관한 연구

김의섭, 유준범

Konkuk University

Dependable Software Laboratory

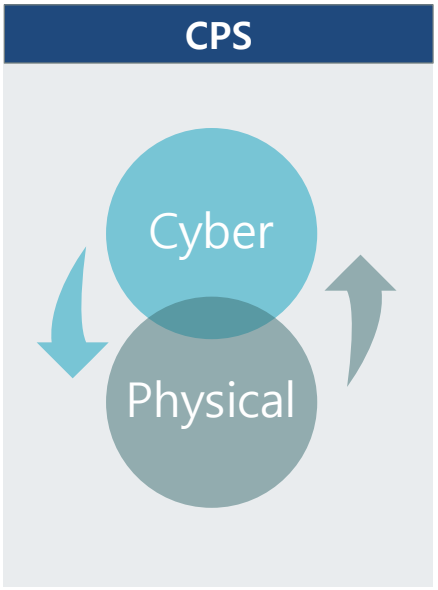
목차

서론

사이버-피지컬 시스템 (CPS – Cyber Physical System)

- 현실 세계의 다양한 물리, 화학 및 기계공학적 시스템(physical systems)이 컴퓨터와 네트워크(cyber systems)를 통해 커뮤니케이션 하여 → 피지컬 시스템을 자율적, 지능적으로 제어하는 시스템 – 네이버 백과사전

- 현실 세계의 정보를 실시간으로 수집(sensing) 하여 지능적, 자율적 연산(cyber process) 후 현실 세계(physical process)에 피드백 하여 운용되는 시스템 및 패러다임

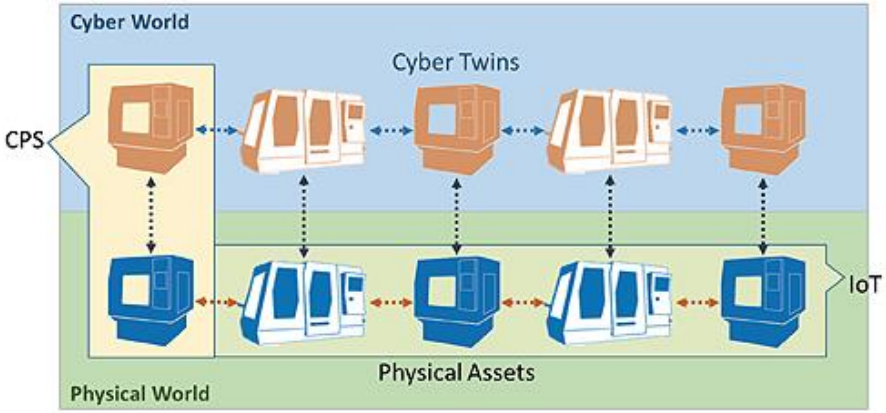


[출처] 융합의 또 다른 이름,
사이버 물리 시스템, 손상혁, 2016

사이버-피지컬 시스템의 예

- 스마트 그리드
  - 효율적인 에너지 관리 및 제공
- 스마트 공장
  - 효율적인 유지보수 지원
  - 안전성 동작 지원
- 도로 교통 통제 시스템
  - 돌발상황 감지 및 교통량 제어

[출처] 융합의 또 다른 이름, 사이버 물리 시스템, 손상혁, 2016



[출처] A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems - Jay Lee, Behrad Bagheri, Hung-An Kao

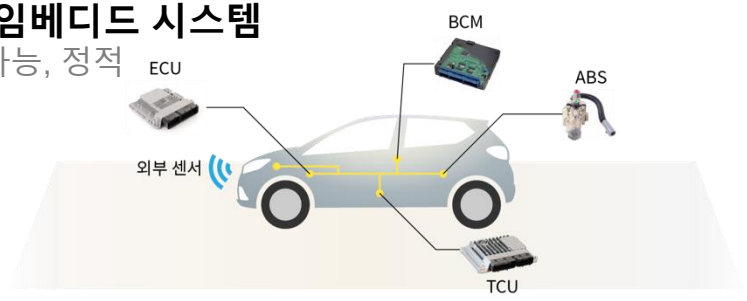
기존 임베디드 시스템과 차이점

기존 임베디드 vs. 사이버-피지컬 시스템

- **기존 임베디드 시스템:**
  - 사용자 및 이벤트에 의해 정해진 동작을 수행하는 **단순 제어 시스템**
  - 시스템들은 실시간 처리, 소형화, 저전력 운용, 저비용을 특징으로 하며 다른 **시스템과의 상호작용 없이 특정 목적을 달성하기 위해 독자적으로 동작**
  - 사용자 요구에 따라 동작하기 때문에 **단방향이고 폐쇄적인 물리 시스템 (+중앙 집중형)**
  - **시스템의 상황 변화를 고려하지 않음**
  - 물리 시스템은 환경, 시간 및 인간과의 상호작용에 따라 시스템의 상태가 변화하게 되는데 임베디드 시스템의 경우 이런 요인들에 따른 결과에 대해 **종합적인 고려 없이 주어진 기능만을 수행**

단일 임베디드 시스템
중앙 집중, 정적

복합 임베디드 시스템
예측 가능, 정적



기존 임베디드 시스템과 차이점

기존 임베디드 vs. 사이버-피지컬 시스템

- **사이버-피지컬 시스템:**
  - **시스템의 상태를 인지하여 필요한 동작을 수행하는 시스템**
  - 물리 세계 정보를 습득, 가공, 계산, 분석하여 그 결과를 엑츄에이터를 통해 물리 세계에 적용, 사이버 세계와 스마트 오브젝트, 인간, 운영 환경을 포함하는 **물리 세계의 긴밀한 상호작용**을 위한 실시간 자율제어 시스템
  - CPS는 **물리 세계에서 발생하는 변화를 감지**할 수 있는 다양한 센서를 통해 환경 인지 기능을 수행

사이버-피지컬 시스템
예측 불가능, 이종 장치, 동적

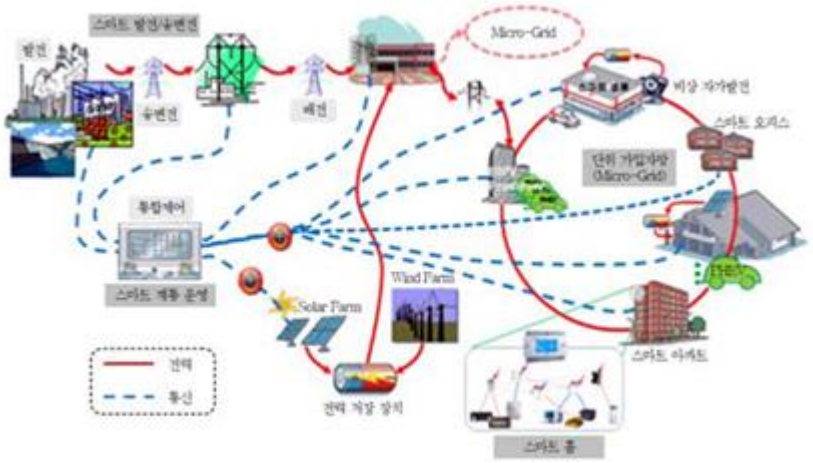
해결 해야 될 문제		
다양성	**복잡성**	**가변성**
다양한 사이버, 현실 공간의 SW 체계 연동 문제	초연결, 분산 환경에서의 CPS SW 복잡성 문제	사이버, 현실 공간 SW 체계의 가용성과 QoS 변화 대응 문제

상호운용성 (Interoperability) 문제

상호운용성 (Interoperability)

- 사이버-피지컬 시스템은 다양하게 변화하는 실세계에서 동작하는 **개방형(open-ended) 시스템**
  - **다양한 이종(heterogeneous)의 시스템이 커뮤니케이션하는 시스템**
- 따라서 동일 또는 이기종의 시스템 및 제품, 서비스가 특별한 제약이 없이 서로 호환되어 사용될 수 있도록 **상호운용성(Interoperability)**이 보장되어야 함
- **상호운용성:**
  - **하나의 시스템이 동일 또는 이기종의 다른 시스템과 아무런 제약이 없이 서로 호환되어 사용할 수 있는 성질을 말한다. - wiki**

- 다양한 이종 시스템간 상호운용
  - Cyber vs. Physical
  - Application vs. Application
  - CPS vs. Other CPS



상호운용성 예

1 뒤섞인 자동 제어 시스템 간섭 오류 자주 발생

2호선: ATS(80·90년대 일본 기술)+ATO(2000년대 독일 기술)+
두 체제 연결 시스템(인터페이스:국산) 등 3가지 시스템 혼재
(선로에는 ATS, ATO 모두 깔려 있음. 열차는 구형 50편. 신형 38편)

1호선: ATS | 3·4호선: ATC(또 다른 기술의 자동제어장치): 미국 기술

2 직류·교류 방식 혼재

서울메트로 직류
코레일 교류

2가지 장비가 혼재돼 고장의 원인, 전류가 달라 운행 중 전철의 모든 기능 일시 중단
(예:1호선 서울역·청량리역)

3 전동차 통행 방향 혼재

1호선: 좌측통행 | 2~4호선: 우측통행

4 한 열차의 차량 연식 달라

열차 6량 → 8량 → 10량으로 확대, 연식 다른 열차끼리 조립하는 경우 흔해

남태령·선바위역의 경우

남태령역	선바위역
서울메트로 관할	코레일 관할
우측통행	좌측통행
직류	교류
ATC	ATS

[출처: 중앙일보] 지하철, 미국·독일·일본 시스템 뒤죽박죽 https://news.joins.com/article/14612053

상호운용성 사고 예

[출처: 중앙일보] 세 차례 오류 신호 … 왜 열차 중단 안 했나 https://news.joins.com/article/5049311

상호운용성 문제 해결 방안

상호운용성 (Interoperability) 문제

- 다양한 시스템 및 제품, 서비스의 결합, 새로운 기능의 추가 등은 사이버-피지컬 시스템의 상호운용성 보장을 저해하는 요소임
- 특히 상충되는 목적을 가진 시스템 및 제품, 서비스가 결합되면, 사이버-피지컬 시스템이 어떤 동작을 할지 예측할 수 없게 되는 문제가 발생

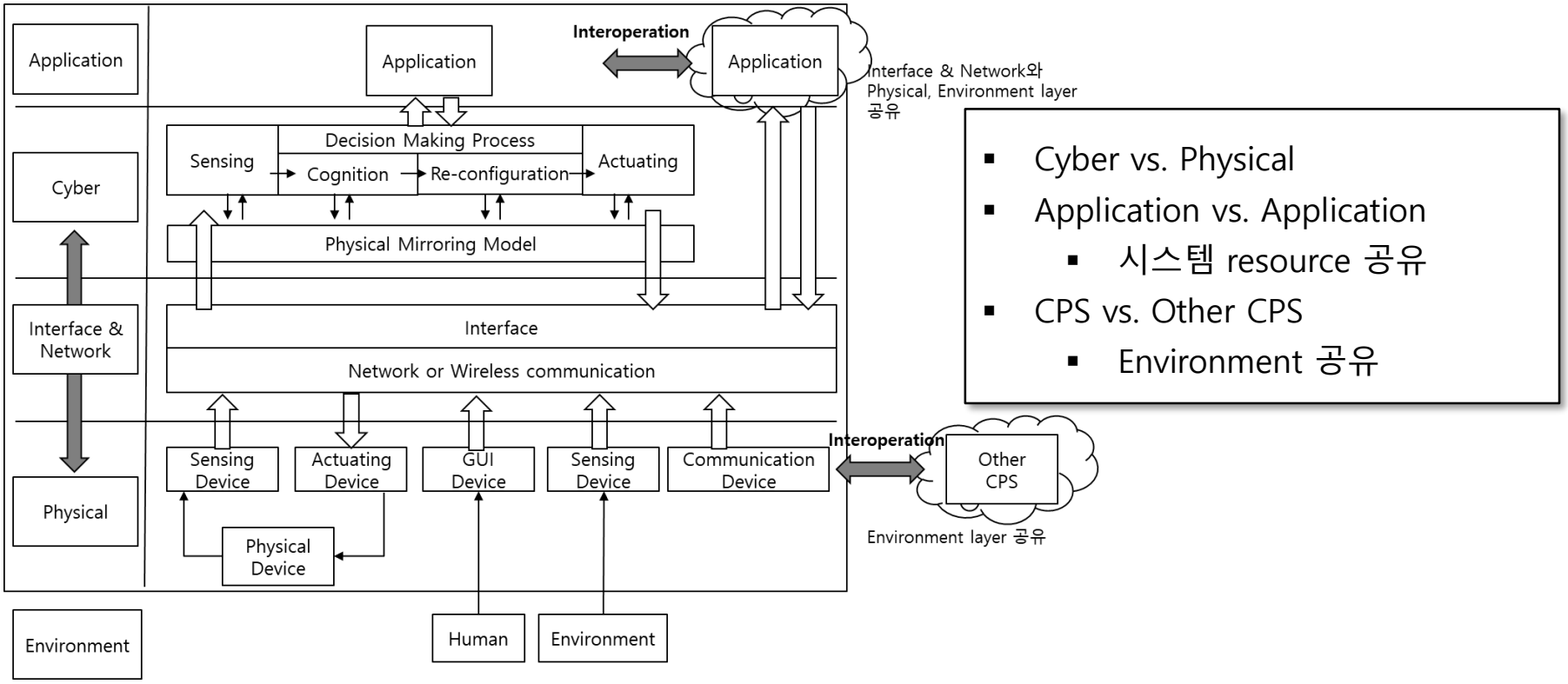
- 이런 문제는 단순하게 시스템의 성능 저하뿐만 아니라 안전을 위협하는 사고로 이어지게 되기 때문에 반드시 **개발 및 설계 단계에서 충분히 분석하고 해결해야 할 문제**

제시하는 해결 방안

- 요구사항 분석 단계에서 기능적 요구사항과 함께 상충관계를 분석 하는 방법 제시
  - 개발 초기에 상충관계를 식별하여 상호운용성을 확보 가능
- 상충관계에 대한 분류 및 분석 방법을 제시
  - 명세 포멧으로 SysML의 확장된 프로파일을 제시

사이버-피지컬 시스템의 상호운용성 상충관계

- **상호운용성**이란 시스템 및 제품, 서비스 가 추가적인 노력 없이도 타 시스템 및 제품, 서비스와 함께 동작할 수 있는 능력
- 사이버-피지컬 시스템은 다양한 **이종의 시스템 및 제품, 서비스가 커뮤니케이션 하는 시스템**으로서 **상호운용성의 보장**이 중요함



CPS 아키텍처와 상호운용성

사이버-피지컬 시스템의 상호운용성 상충관계

- **상호운용성 상충관계**
  - 이종의 시스템이 적절한 상호운용을 하지못해 원하는 기능을 수행하지 못하는 관계

- **상호운용성 상충관계의 두가지 분류**
  - **정적인 상충(구조적 연결성 상충)**
    - 상호 운용되어야 하는 어플리케이션 및 시스템이 적절한 **커뮤니케이션을 하지 못해** 상호 운용되지 못하는 상충관계
  - **동적인 상충(기능적 상충)**
    - 서로 다른 어플리케이션 및 시스템이 각자의 **판단 결과가 불일치**하여 어떤 것을 수행해야 할지 판단하기 어려운 상황인 동적인 상충(기능적 상충)을 의미한다.

상호운용성의 상충

- **정적인 상충(구조적 연결성 상충)**
  - 상호 운용되어야 하는 어플리케이션 및 시스템이 제대로 **커뮤니케이션을** 하지 못해 상호 운용되지 못하는 상충관계
  - **원인:** 데이터 포맷의 불일치, 통신 프로토콜의 불일치, 커뮤니케이션 지연 등

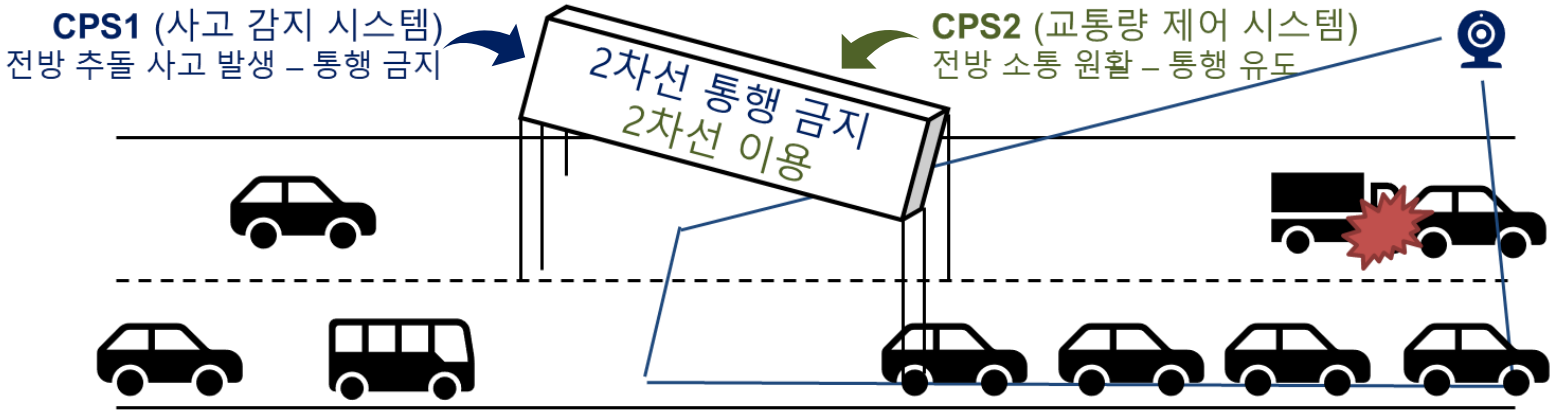
- **정적인 상충**은 **이종**의 서비스 및 어플리케이션, 이종의 네트워크 프로토콜, 제조업체가 다른 디바이스 및 컴포넌트, 다른 사이버-피지컬 시스템과의 상호운용 등 다양한 곳에서 존재
- 뿐만 아니라, 운용 중 다른 시스템 및 제품, 서비스와 **동적인 조합**, **유지보수를 위한 제품의 교체**, 제품 및 어플리케이션의 **업데이트** 등에 의해서 정적 상충 상황이 발생
- 해당 상충을 해결하기 위해서는 **RAMI 4.0 표준이나 AUTOSAR 표준과 같은 산업 표준을 준수 및 인터페이스 매칭** 수행

상호운용성의 상충

- **동적인 상충(기능적 상충)**
  - 서로 다른 어플리케이션 및 시스템이 각자의 연산 결과가 불일치하여 어떤 것을 수행해야 할지 판단하기 어려운 상황인 동적인 상충(기능적 상충)을 의미함
  - **원인:** 커뮤니케이션 지연, 동일한 환경에 대한 불일치 인지, 기능적 상충 등

- 정적인 상충은 참조 모델 및 표준 준수, 인터페이스 메칭 등을 통해 보장 가능한 반면 **동적인 상충은 정적인 상충보다 분석 및 식별하기가 어려움**
- 시스템 차원의 기능 분석 또는 시나리오 기반 단일 기능 분석을 통해 충돌 여부 평가 가능 (정성적)

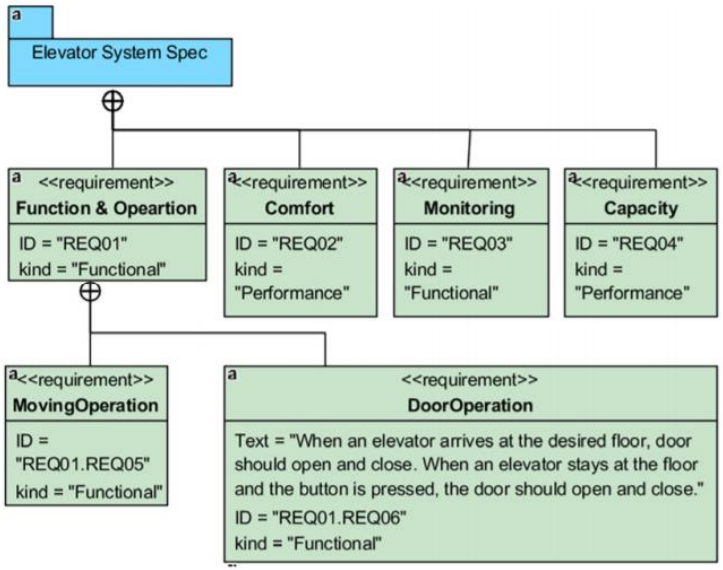
도로 통제 시스템



동적인 상충 예

사이버-피지컬 시스템의 상호운용성 상충관계 분석 방안

- 요구사항 분석 단계에서 기능적 요구사항과 함께 상호운용성의 상충관계를 함께 분석
  - 개요: 이종의 시스템의 요구사항에 상호운용성 요구사항을 추가하여 상충관계를 분석
  - **SysML의 요구사항 다이어그램 확장**



Requirement Diagram	req ReqDiagram
Requirement	«requirement» Requirement name; text="The system shall do" Id="62j32."
TestCase	«testCase» TestCaseName
Requirement containment relationship	«requirement» Parent ⊕ «requirement» Child1, «requirement» Child2
CopyDependency	«requirement» Slave ----«copy»----> «requirement» Master
Derive Dependency	«requirement» Client ----«deriveReqt»----> «requirement» Supplier
Satisfy Dependency	NamedElement ----«satisfy»----> «requirement» Supplier
Verify Dependency	NamedElement ----«verify»----> «requirement» Supplier
Refine Dependency	NamedElement ----«refine»----> «requirement» Client
Trace Dependency	«requirement» Client ----«trace»----> «requirement» Supplier

[출처] 안전 필수 시스템의 개념단계 해저드 분석 프로세스를 위한 SysML 적용 방안, 정보과학회논문지, 2018
[출처] SysML 다이어그램(1) – Requirement Diagram|작성자 한컴MDS

사이버-피지컬 시스템의 상호운용성 상충관계 분석 방안

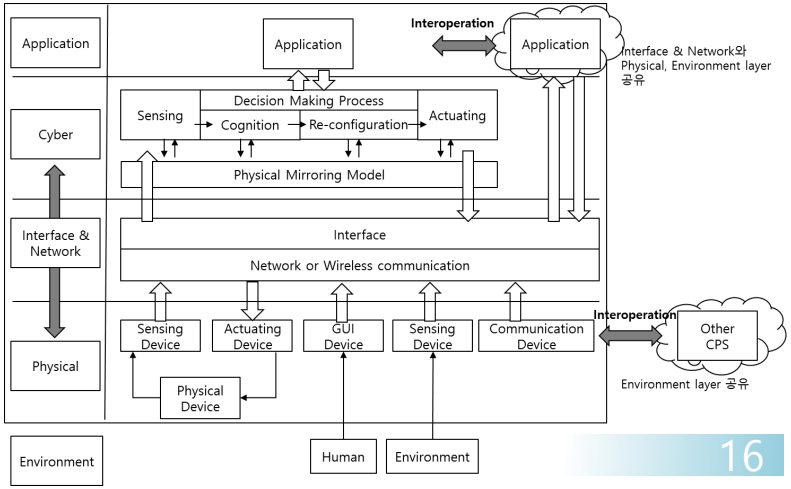
- 요구사항 분석 단계에서 기능적 요구사항과 함께 상호운용성의 상충관계를 함께 분석
  - SysML의 요구사항 다이어그램 확장

Requirement Diagram	req ReqDiagram
Requirement	«requirement» **Requirement name** text="The system shall do" Id="62j32."

<<requirement>>
Requirement Name

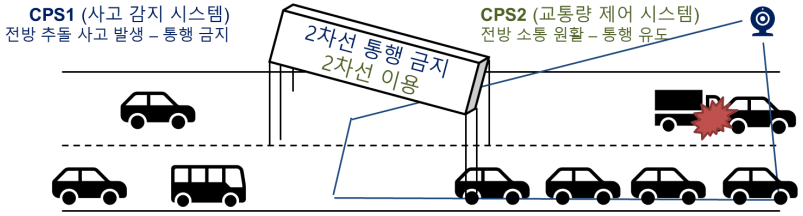
ID:
Text:
Source:
Source Layer:
Target:
Target Layer:
Command:
Decision resource:

ID: 요구사항을 식별할 수 있는 유일한 속성의 식별자
Text: 자연어로 작성된 요구사항의 설명
Source: requirement가 요구되는 요소의 이름
Source Layer: 아키텍처에 대응되는 Source의 Layer
Target: 명령을 내리고자 하는 요소의 이름
Target Layer: 그림 1에 대응하는 Target의 Layer
Command: Target에 요구되는 명령
Decision resource: Source가 Command를 생성하는데 사용한 resource



사이버-피지컬 시스템의 상호운용성 상충관계 분석 방안

- 상충상황 적용 예시
  - **사고 감지 시스템 vs. 교통량 제어 시스템**
  - **2차선 통행 금지 vs. 2차선 이용**



CPS1 요구사항 1
사고 감지 시스템

<<requirement>>
Requirement Name

ID: Req1
Text: 사고 발생시 전광판에 진입 금지 출력
Source: App1
Source Layer: Application
Target: 전광판
Target Layer: Physical
Command: 진입 금지 출력
Decision resource: 자동차 사고

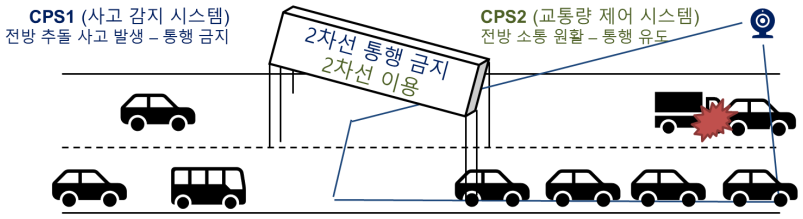
CPS2 요구사항 2
교통량 제어 시스템

<<requirement>>
Requirement Name

ID: Req2
Text: 고속도로 차선의 교통량에 따라 적절한 차선 유도
Source: App2
Source Layer: Application
Target: 전광판
Target Layer: Physical
Command: 차선 별 진입 허용/금지 출력
Decision resource: 도로 교통량

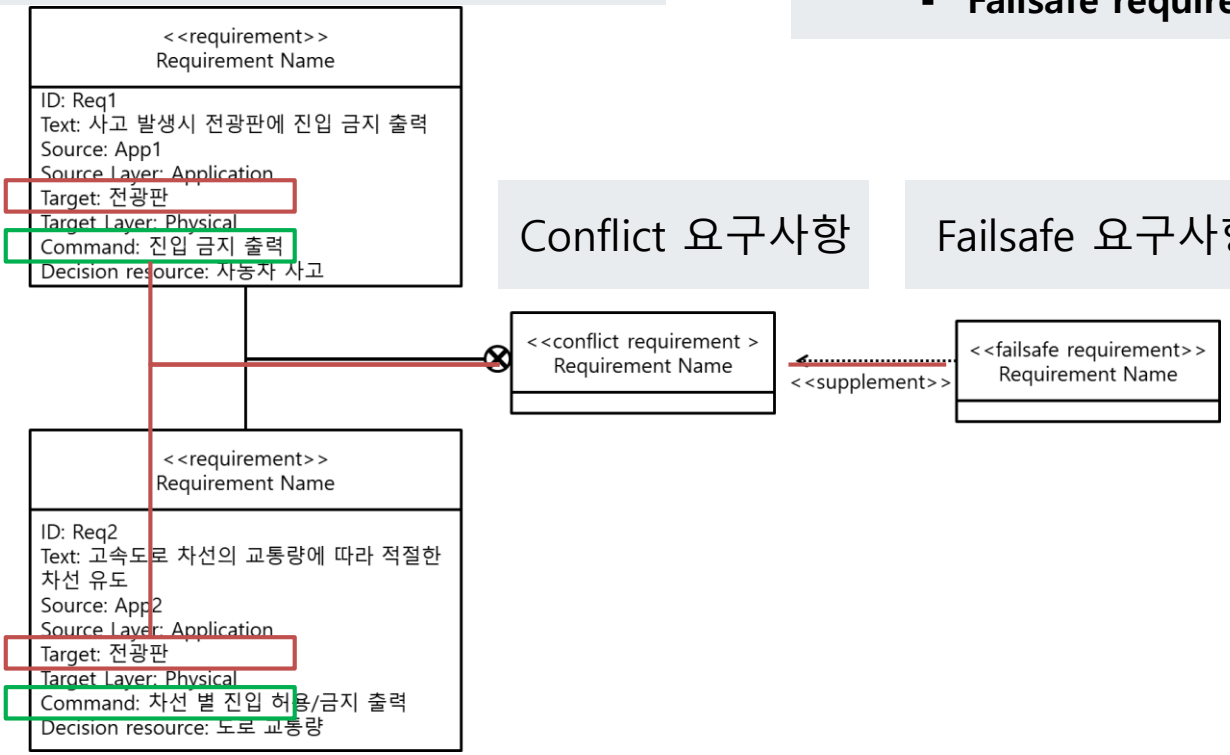
사이버-피지컬 시스템의 상호운용성 상충관계 분석 방안

- 상충상황 적용 예시
  - 사고 감지 시스템 vs. 교통량 제어 시스템
  - 2차선 통행 금지 vs. 2차선 이용
  - **상호운용성 상충 후보 선택 방법**
    - **동일한 Target에 다른 CPS로부터 상이한 Command 발생**



- **상충상황 발생 Candidate**
  - **Conflict requirement와**
  - **Failsafe requirement 작성**

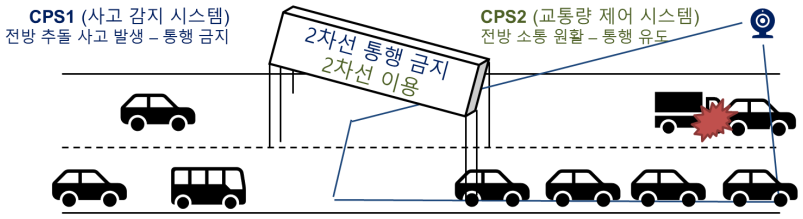
CPS1 요구사항 1
사고 감지 시스템



CPS2 요구사항 2
교통량 제어 시스템

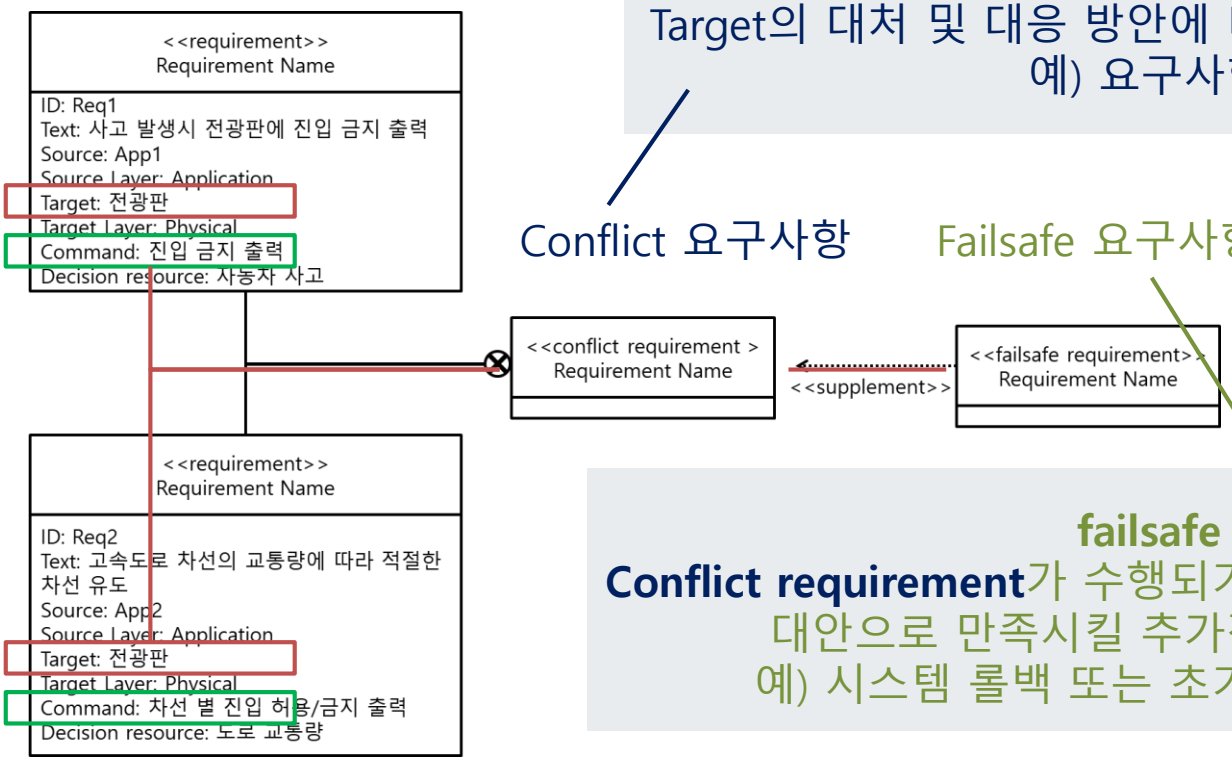
사이버-피지컬 시스템의 상호운용성 상충관계 분석 방안

- 상충상황 적용 예시
  - 사고 감지 시스템 vs. 교통량 제어 시스템
  - 2차선 통행 금지 vs. 2차선 이용
  - 상호운용성 상충 후보 선택 방법
    - 동일한 Target에 다른 CPS로부터 상이한 Command 발생



conflict requirement:
두 개의 어플리케이션 및 시스템이 동일한 Target 시스템에 다른 명령을 내리는 경우 Target의 대처 및 대응 방안에 대한 요구사항
예) 요구사항 우선순위화

CPS1 요구사항 1
사고 감지 시스템

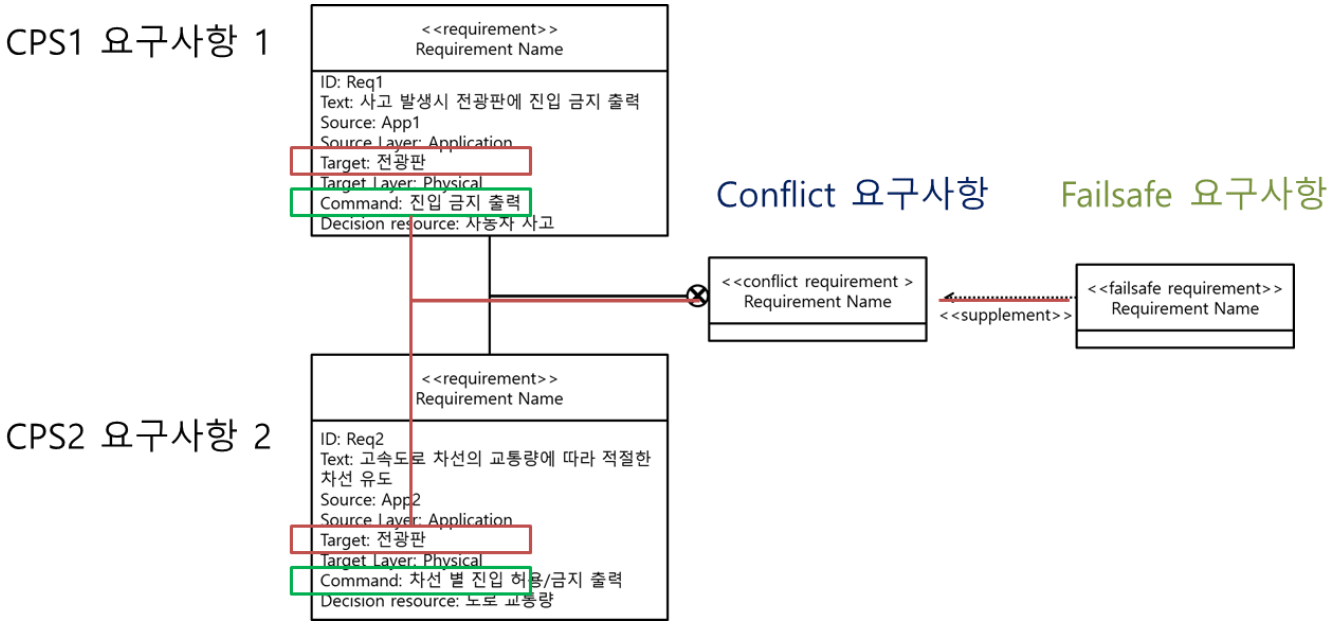


CPS2 요구사항 2
교통량 제어 시스템

failsafe requirement:
Conflict requirement가 수행되기 어려울 경우 대안으로 만족시킬 추가적인 요구사항
예) 시스템 롤백 또는 초기화 등의 기능

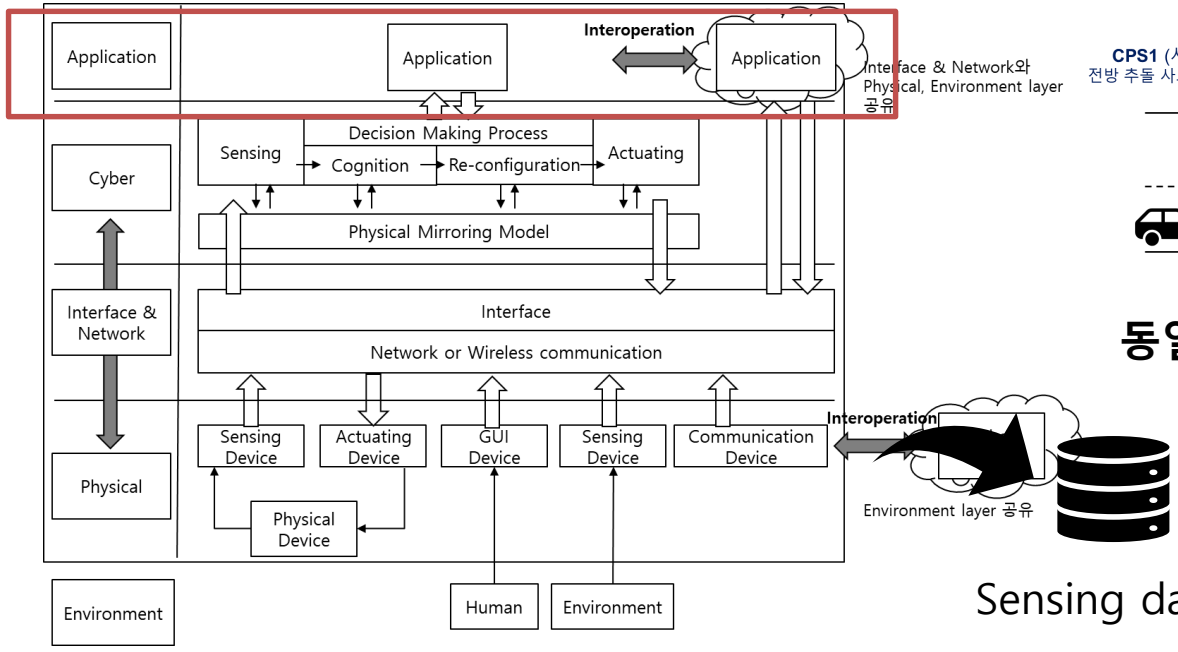
사이버-피지컬 시스템의 상호운용성 상충관계 분석 방안

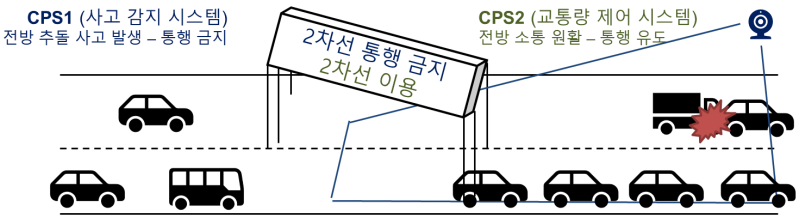
- **상호운용성 상충 후보 선택 방법**
  - ≒ 동일한 Target에 다른 CPS로부터 **상이한 Command 발생**
  - ≒ 동일한 Source가 **상이한 Decision resource을** 바탕으로 command를 생성할 때
  - ≒ 동일한 Source Layer의 Source 들이 **상이한 Decision resource을 바탕으로 command를 생성할 때**



사이버-피지컬 시스템의 상호운용성 상충관계 분석 방안

- **상호운용성 상충 후보 선택 방법**
  - ≒ 동일한 Target에 다른 CPS로부터 **상이한 Command 발생**
  - ≒ 동일한 Source가 **상이한 Decision resource을** 바탕으로 command를 생성할 때
  - ≒ 동일한 Source Layer의 Source 들이 **상이한 Decision resource을 바탕으로 command를 생성할 때**





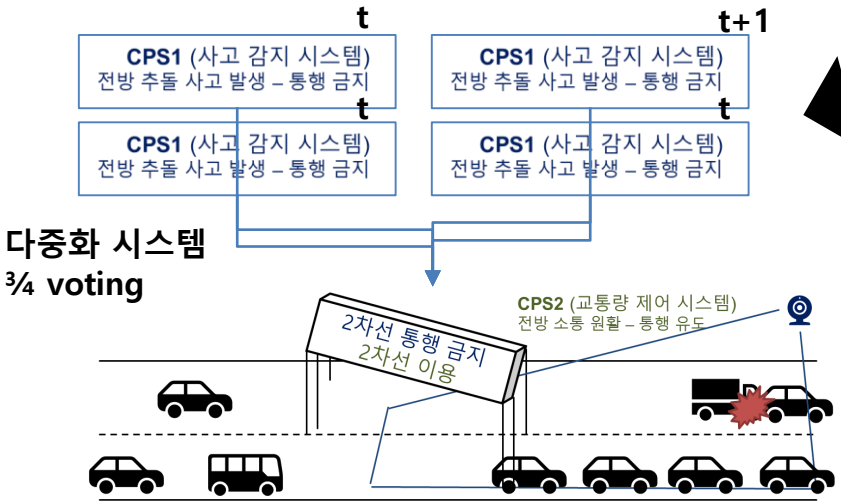
동일하지 않은 상황을 바탕으로 판단

t	...	사고X
t+1	...	사고O
t+2	...	...

Sensing data, DB

사이버-피지컬 시스템의 상호운용성 상충관계 분석 방안

- **상호운용성 상충 후보 선택 방법**
  - ≒ 동일한 Target에 다른 CPS로부터 **상이한 Command 발생**
  - ≒ 동일한 Source가 **상이한 Decision resource을** 바탕으로 command를 생성할 때
  - ≒ 동일한 Source Layer의 Source 들이 **상이한 Decision resource을 바탕으로 command를 생성할 때**



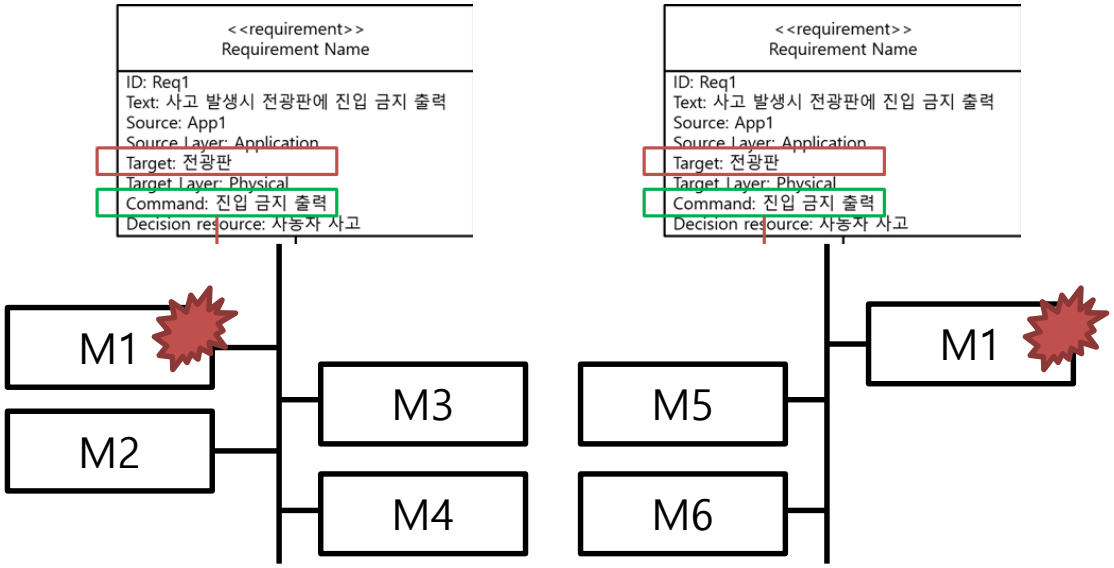
Sensing data, DB

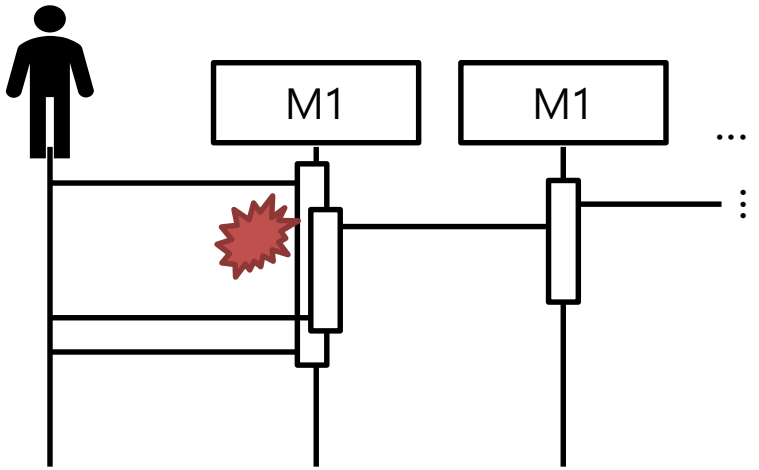
동일하지 않은 상황을 바탕으로 판단

t	...	사고X
t+1	...	사고O
t+2	...	...

사이버-피지컬 시스템의 상호운용성 상충관계 분석 방안

- **+ (optional) 상호운용성 상충 후보 선택 방법**
  - 모델링 레벨에서의 피드백
  - ≒ 동일한 Target(≒ **동일한 Model Element**)에 다른 CPS로부터 상이한 Command 발생





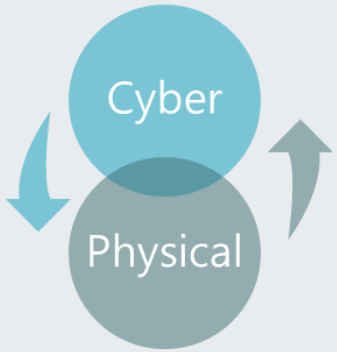
요구사항과 디자인 모델링과의 추적성이 선행 되어야 함

상호운용성 상충 요구사항 활용 방안

- **상호운용성 상충 후보 활용 방안**
  - 사이버-피지컬 시스템의 안전성 확보 기여

<CPS 어플리케이션의 복합적인 안전성 평가 지원을 위한 통합 추적성 분석 기술>
전략: IEC-61508 Safety Lifecycle 적용을 통한 안전성
체계적인 안적성 분석 방법 + 추적성 확보를 통한 안전성 확보

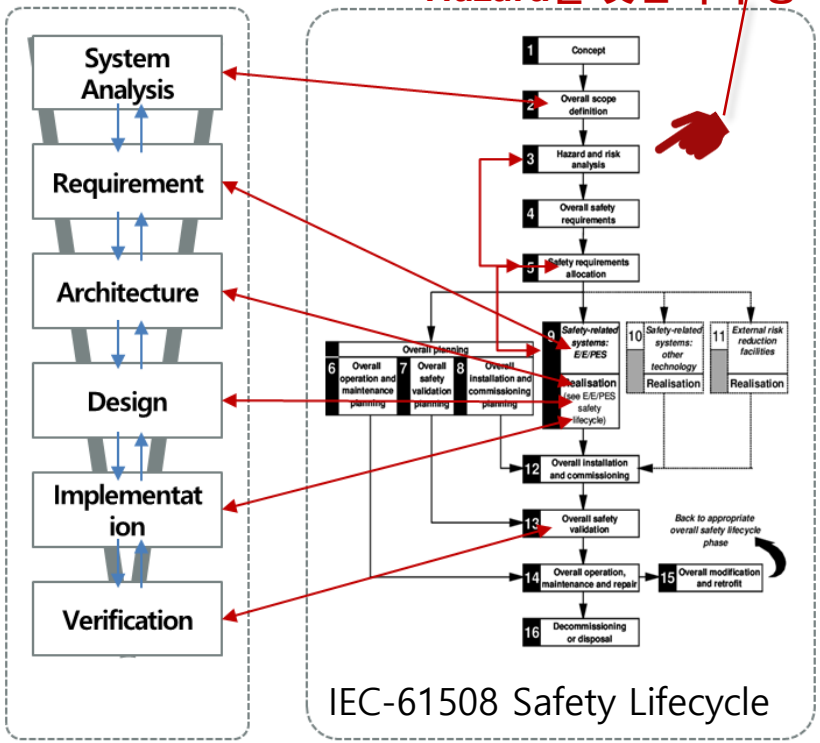
CPS



Safety Critical CPS

: Accident가 심각한 재산적 피해 및 인명 사고로 이어지는 System

얼마나 다양하고 많은 Hazard를 찾는지가 중요



Safety Lifecycle

Accident

발전소의 **폭발**은 심각한 피해를 준다

Hazard

온도가 올라가면 폭발한다

Safety Function

온도가 올라가면 **냉각기능**을 수행한다

Development (+Traceability)

기능 개발 + **추적성**을 확보한다

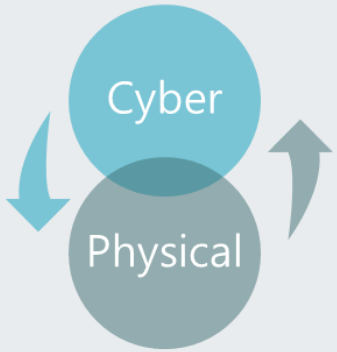
상호운용성 상충 요구사항 활용 방안

문제:
Safety Analysis는 전문가의 지식 및 노하우에 에 의존적
완벽한 분석 불가능 ≒ 끝없는 분석과 노력 필요
CPS에 적합한 Safety analysis 필요

특히 상호운용성의 상충관계 부분은 식별하기 매우 어려움
→ 제시하는 연구가 기여할 것으로 판단

한 통합 추적성 분석 기술>

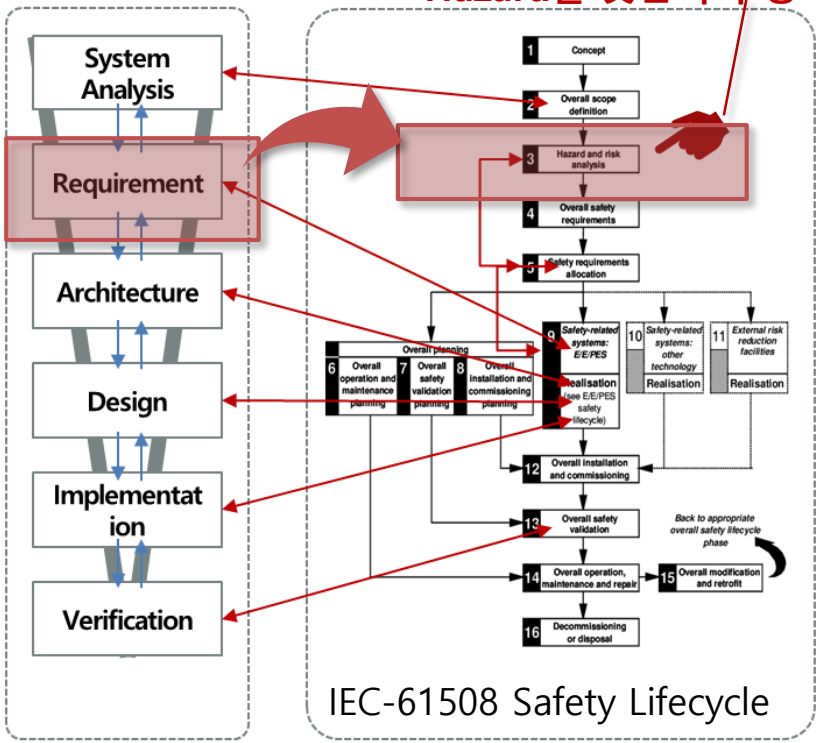
CPS



Safety Critical CPS

: Accident가 심각한 재산적 피해 및 인명 사고로 이어지는 System

얼마나 다양하고 많은 Hazard를 찾는지가 중요



IEC-61508 Safety Lifecycle

Safety Lifecycle

Accident

발전소의 **폭발**은
심각한 피해를 준다

Hazard

온도가 올라가면 폭발한다

Safety Function

온도가 올라가면
냉각기능을 수행한다

Development (+Traceability)

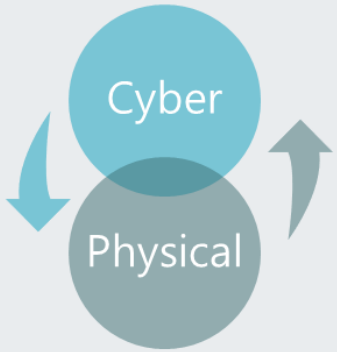
기능 개발
+ **추적성**을 확보한다

상호운용성 상충 요구사항 활용 방안

앞으로>
- **도구로 구현하여 시스테믹한 상충관계 후보의 식별 지원**
  - Ex) 동일한 Target에 다른 CPS로부터 상이한 Command 발생
- **모델링 레벨과의 추적성 확보를 통해 상충관계 후보 식별 지원**

분석 기술>

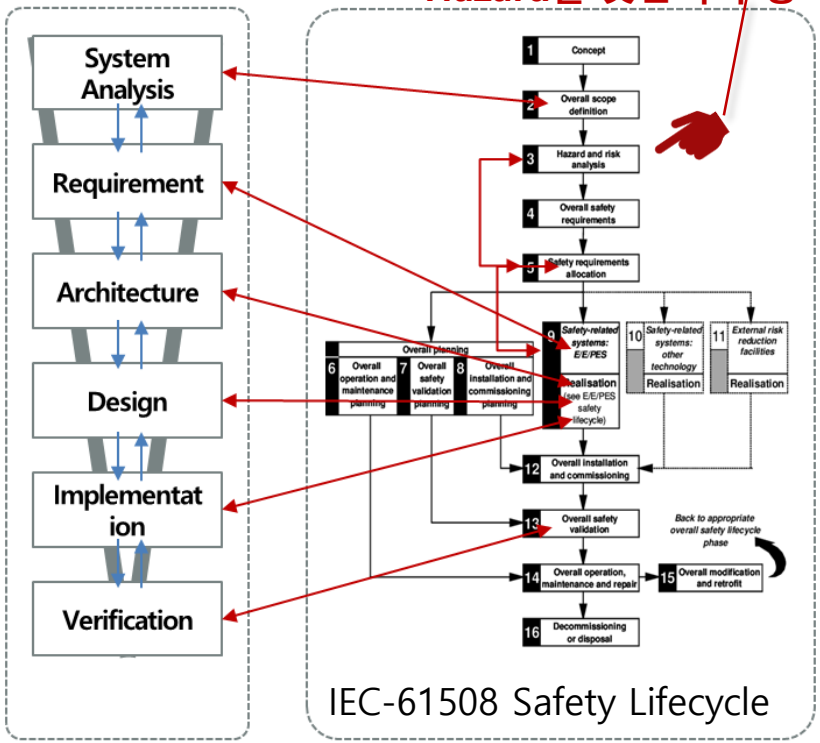
CPS



Safety Critical CPS

: Accident가 심각한 재산적 피해 및 인명 사고로 이어지는 System

얼마나 다양하고 많은 Hazard를 찾는지가 중요



IEC-61508 Safety Lifecycle

Safety Lifecycle

Accident

발전소의 **폭발**은 심각한 피해를 준다

Hazard

온도가 올라가면 폭발한다

Safety Function

온도가 올라가면 **냉각기능**을 수행한다

Development (+Traceability)

기능 개발 + **추적성**을 확보한다

상호운용성 상충 요구사항 활용 방안

문제:
- Requirement Analysis 단계에서는 실제 Physical asset 및 model에 대한 정보가 부재하거나 불확실 할 수 있음

상호보완:
- 일정수준의 개발 진행 후 추적성을 통해 Physical asset 및 model에 대한 정보 보완 후
  - 상충관계에 대한 보다 정교한 분석 수행 가능

문제:
개발의 과도한 backward process는 개발 지연을 유발
가능한 요구사항 분석단계에서 식별 가능하면 유용

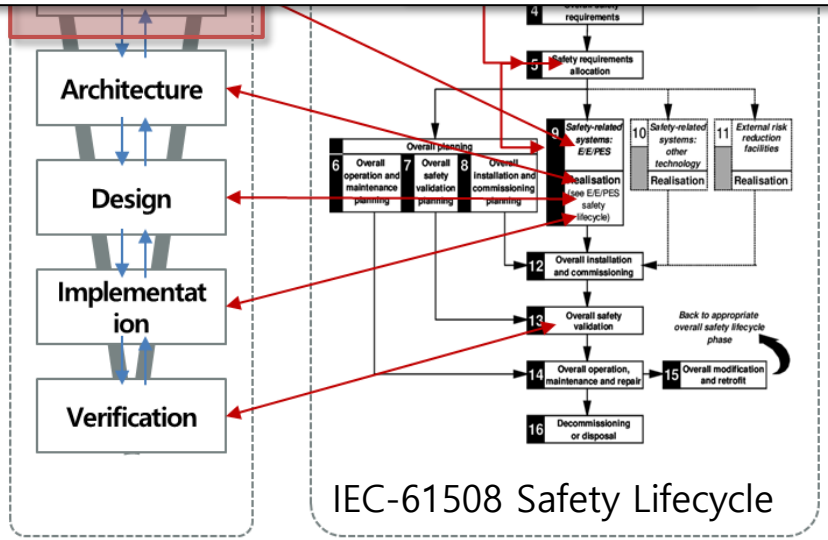
적성 분석 기술>

Hazard를 찾는지가 중요

Physical

Safety Critical CPS

: Accident가 심각한 재산적 피해 및 인명 사고로 이어지는 System



IEC-61508 Safety Lifecycle

Safety Lifecycle

Accident

발전소의 **폭발**은 심각한 피해를 준다

Hazard

온도가 올라가면 폭발한다

Safety Function

온도가 올라가면 **냉각기능**을 수행한다

Development (+Traceability)

기능 개발 + **추적성**을 확보한다

결론 및 향후 연구

- **사이버-피지컬 시스템의 상충관계에 대한 분류 및 정의**
  - 정적인 상충(구조적 연결성 상충)
  - 동적인 상충(기능적 상충)

- **사이버-피지컬 시스템의 상충관계 분석을 위한 방안 제시**
  - SysML의 확장 / CPS layer를 위한 아키텍처 제시
  - 상충관계의 후보 식별 방안

- **사이버-피지컬 시스템의 상충관계 분석 방안의 활용 방안**
  - 사이버-피지컬 시스템의 안전성 향상에 기여할 것으로 기대

- **향후 연구**
  - **상충관계 후보 식별 방안에 대한 적합성 검증 및 확장**
  - **도구로 구현하여 실제 사용 가능하도록 지원**

Q & A

감사합니다.

atang34@Konkuk.ac.kr
http://dslab.Konkuk.ac.krs