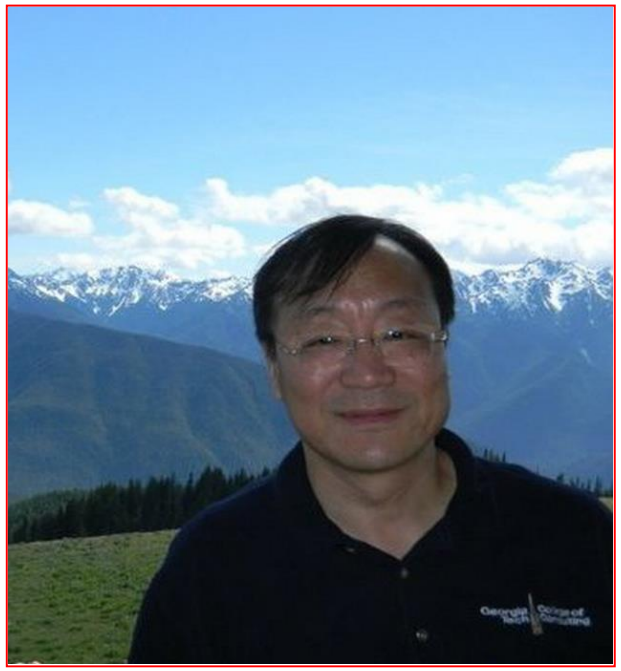
자동차 소프트웨어 신뢰성, 안전성, 그리고 보안

고려대학교

최진영

choi@formal.korea.ac.kr

강사 소개



- 서울대 컴퓨터공학과 학사
- University of Pennsylvania 박사
- 고려대 컴퓨터학과 교수(전)
- 고려대 사이버국방학과 교수(현)
- 고려대 정보보호대학원 교수(현)
- SW개발보안연구센터장(전, 행자부지원)
- 융합SW연구센터장(전), ITRC (IITP지원)
- 정형기법연구실 책임교수
- 관심분야:
  - 정형기법(정형명세, 정형검증)
  - 시큐어 소프트웨어공학,
  - 소프트웨어 및 공급망 보증 (SSCA, Software and Supply Chain Assurance)
  - 안전필수, 기능필수 시스템

순서

1. 들어가기
2. ISO 26262 이전
3. 제어기와 전자장치
4. IEC 61508 탄생 : 전기/전자/프로그램 가능한 기기의 안전성
5. ISO 26262 탄생 : 자동차의 안전성
6. ISO 26262 의 미래 : 자동차 안전성과 보안성
7. 안전성과 보안성의 통합
8. 결론

들어가기 : 자동차는 기계공학? 전자공학? 소프트웨어공학?

21세기 자동차산업 트렌드 읽기

글 : 데스크(webmaster@global-autonews.com)

승인 2008-10-08 05:54:21 목록

21세기 자동차산업 트렌드 읽기

글/채영석(글로벌오토뉴스 국장)

인류 최대의 문명의 이기라 일컬어지고 있는 자동차가 근본적으로 그 모양과 내용이 달라져 가고 있다. 무엇이 어떻게 달라지는지 정리해 본다.

1. 미래 자동차 신기술은 90%가 전기 전자 제품

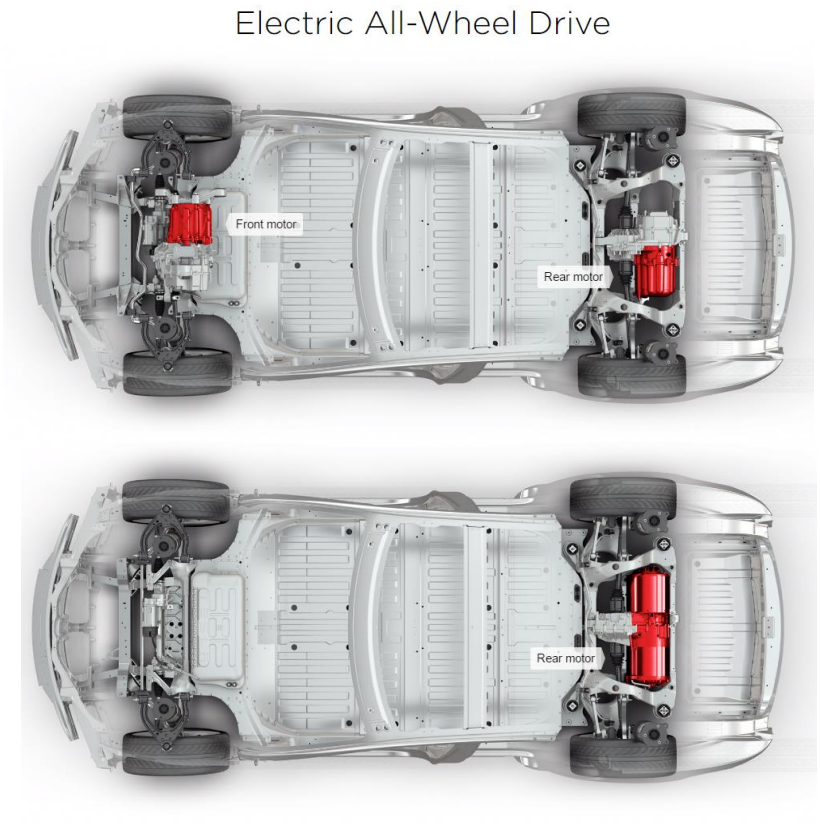
무엇보다 큰 특징은 21세기의 자동차는 빠른 속도로 전자 제품화 되어간다는 것이다. 2010년경이 되면 자동차를 생산하는데 드는 비용의 35~40% 가량이 전기 전자부품이 점하게 될 것으로 보인다. 또한 앞으로 우리가 접하게 될 자동차 신기술은 90% 이상이 전기 전자 분야에서 이루어진다는 분석도 있다.

이는 이미 BMW의 iDrive와 아우디의 MMI, 메르세데스 벤츠의 커맨드 시스템 등을 통해 진행되어 왔던 내용들이다. 중요한 것은 그런 흐름이 위에 예로 든 프리미엄 브랜드 뿐 아니라 양산 브랜드들에게까지도 피할 수 없는 과제라는 것이다.

이런 변화는 우선 자동차회사의 인력 보충에서의 변화를 예고한다. 다시 말해 기계공학 출신보다 전기전자 기술 분야의 인력을 집중적으로 채용할 수밖에 없다는 것이다. '기계공학의 종합 예술작품'이라고까지 했던 자동차 부문에서의 이런 변화가 우리에게 주는 의미는 적지 않다고 할 수 있다.

흔히 자동차는 파워 트레인과 차체, 섀시 등 3대 요소로 구분했고 그것을 바탕으로 기술 개발 분야의 인력구성도 이루어져왔다. 여기에 전장품이 가장 비중이 큰 부문 중 하나로 대두되어 있다는 것이다. 더불어 세그먼트의 세분화가 진행되고 그에 따른 하이테크 개발력이 필요하게 되었다. 이런 추세에 따라 현재에도 각 자동차회사들은 적게는 10%, 많게는 20%의 연구개발 인력을 전기전자부문에 할당하고 있다. 또한 심하게는 엔지니어의 절반 가까이가 전기전자 부문에 종사할 것으로 전망되고 있다. 그렇다면 앞으로 자동차산업의 주류는 더 이상 기계 공학이 아니라 전기전자학가 될 것은 자명하다.

그에 따라 자동차회사들도 자동차 개발의 주력을 전기전자에 둘 수밖에 없다. 더불어 기계와 전기전자가 각각 따로 운영되던 상황에서 이제는 통합이 될 수밖에 없는 상황이 도래했다. 이런 움직임은 이미 유럽의 자동차회사들을 중심으로 진행이 되고 있다.



ISO 26262 이전 : 기계산업

참조: 디자인정글

- 신뢰성 (Reliability)
  - 어떤 부품/소재나 제품시스템 등이 주어진 조건 (사용, 환경 조건)하에서 고장없이 일정기간 (시간, 거리, 사이클)동안 최초의 품질 및 성능을 유지하는 특성.
  - 신뢰성이라는 단어는 1816년 탄생
  - 현대의 신뢰성 개념은 1940년대 미군에 의해 정립됨.
  - 특징: 일정 시간이 지나면, 기계가 마모가 되어 신뢰성은 감소함.
  - 안전성과의 관계: 신뢰성은 고장과 관련이 있으므로, 신뢰성과 안전성은 서로 연관되어 있음, 즉, 신뢰성이 높으면 안전성도 높다.
  - 신뢰성이 높은 기계제품을 만드는 기술은 오랜 경험과 연구로 확보되어 있음.

전자장치의 출현

참조: www.conceptcarz.com

사진: S. Howard

- GM 캐딜락 세빌에 처음으로 모토롤라 6802 기반 트립 컴퓨터 탑재 (1978)
- 디지털 속도계 와 디스플레이.
- 기계식 트립 컴퓨터는 스웨덴 택시메터 생산기업에서 생산 (1950s)
- 2004년 아큐라 TL 트립 컴퓨터
  - 평균 속도 (MPH)
  - 평균 마일리지 (MPG)
  - 주어진 기간 동안 주행 거리

소프트웨어 신뢰성(reliability)

- 소프트웨어가 정의된 작동환경에서 다양한 부하에서도 정확하고 일관된 결과들을 반복적으로 만들어 낼 수 있는가에 대한 확신할 수 있는 정도
  - "a measure of confidence that the software produces accurate and consistent results, which are repeatable under low, normal, and peak loads in the intended operational results"

  ※ D. S. Hermann, Software Safety and Reliability, IEEE Computer Society Press, 2000

- 무기체계 내장형 소프트웨어가 실제 무기체계 작동환경에서 무기체계 운영기간 동안 사용자 요구사항에 제시된 기능을 반복적으로 수행 시 정확하고, 일관성 있게 작동할 수 있는 소프트웨어 능력

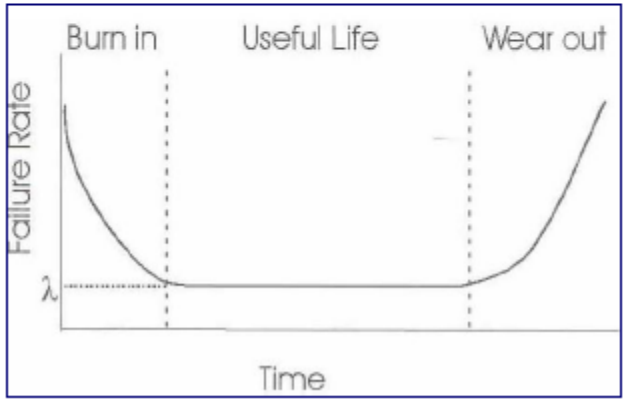
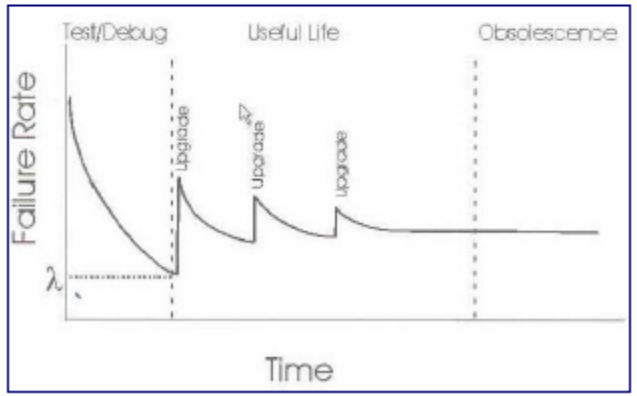
  ※ 무기체계 내장형 소프트웨어 ㅐ 신뢰성 확보방안 연구, 정보통신기술협회, 2012

- 명세된 (요구된) 대로 소프트웨어가 서비스를 수행할 수 있는 능력 (소프트웨어 내에 있는 결함 밀도)

  "The ability of the system to deliver services as specified."

  ※ Sommerville, : Software Engineering 10th edition, 2016

소프트웨어 신뢰성 vs. 하드웨어 신뢰성

항목	하드웨어	소프트웨어
고장원인	설계, 생산, 유지보수, 마모 등.	설계 결함, 코딩 결함
고장발생	ㅇ 대부분 제품의 소모(Wear)나 외부 요인에 발생 ㅇ 고장 발생 전에 잠재적인 증상이 나타남	ㅇ 제품의 소모(Wear)와 관계가 없음 ㅇ 고장/에러에 대한 사전 증상 없이 갑작스럽게 발생
시간관련	ㅇ 시간과 밀접한 관계가 있으며, 운영시간이 늘어날수록 고장률이 떨어지다가 증가	ㅇ 반드시 시간 의존적이지는 않음
환경영향	외부 환경적 조건에 많은 영향을 받음	외부 환경적 영향에 상관없이 동작하며, 내부 환경적 영향에 민감하게 반응함
신뢰성 곡선		

참조: 무기체계 소프트웨어 신뢰성 시험,
방위산업기술지원센터 SW기술팀, 2014

100% 신뢰할 수 있는 소프트웨어?

- 소프트웨어의 (이론적) 100% 신뢰성
  - 명세에서 정의된 모든 입력에 대해 소프트웨어어가 정의된 서비스를 수행하면 됨
- 문제점
  - 입력의 개수가 너무 방대하다
    - 입력의 정수 변수 10개인 경우 (2^32)^10 개의 입력이 가능
  - 실시간 시스템의 경우 시간이 입력이 되기에 100% 실행하기 불가능
  - 동시성이 존재하는 경우 그 의미론적 해석으로 상태 폭발이 발
  - 반응형 시스템(Reactive System, 주로 임베디드 소프트웨어)은 종료가 되지 않으며 또한 입력이 환경에 영향을 받음.
- 결론
  - 100% 신뢰성 소프트웨어는 일반적으로 불가능함
  - 모든 소프트웨어는 오류를 포함함
  - 소프트웨어 개발 시 잠재된 오류를 최소가 되도록 노력 필요.
  - 특히 숨어있는 오류로 안전성이 문제가 됨

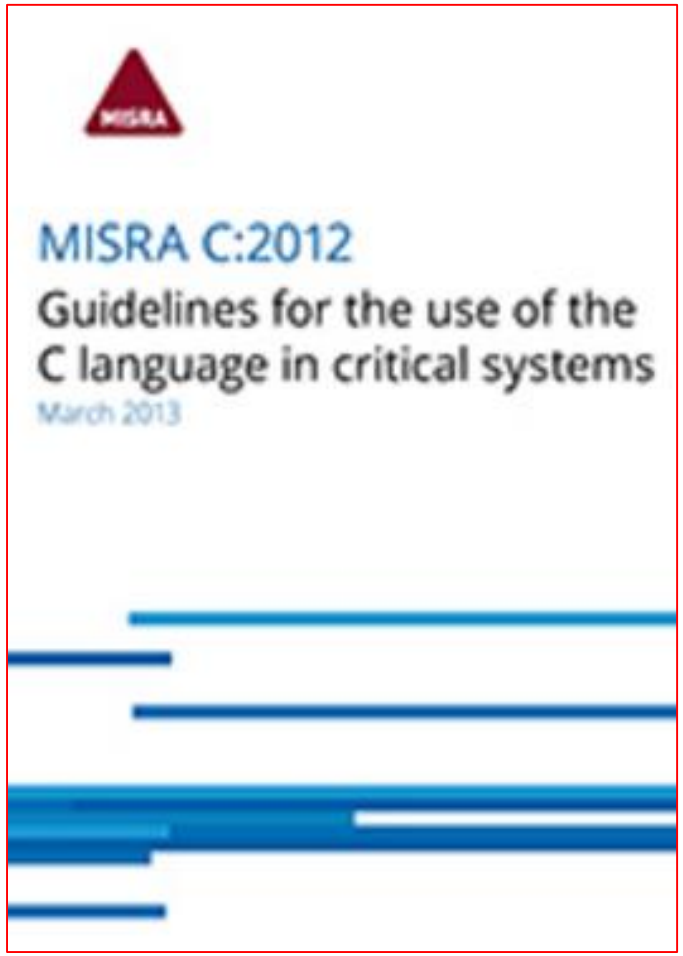
오류의 최소화 (비용대비) : 방어적 프로그램

- 방어적 프로그래밍 (Defensive Programming)
  - 소프트웨어 버그를 줄임 (코딩시)
  - 소스 코드를 이해하기 쉽게 작성함.
  - 생각 못한 입력이나 사용자 반응에도 불구하고 예상 가능하도록 작성함.

- 코딩 규칙의 사용
  - MISRA C/C++ (Motor Industry Software Reliability Association)
  - 사업/기관을 위한 코딩 규칙 정의 및 적용
    - 방위사업청 코딩 규칙

- 룰 체커 (Rule Checker) 활용
  - 정적분석도구의 일종
  - **코딩 규칙의 검증 시험**

MISRA-C



- MISRA (Motor Industry Software Reliability Association)
- 목적은 임베디드 시스템의 관점에서 SW 안전성 (Safety), 보안성(Security), 이식성 (Portability), 신뢰성 (Reliability) 등을 높이기 위함.
- 현재는 자동차 분야 뿐만이 아니라, 우주항공, 통신분야, 의료기기, 철도분야, 등 다양한 분야에 활용이 되고 있음
- 지속적 발전을 하고 있음.
  - MISRA C:1998
  - MISRA C:2004
  - MISRA C:2012
  - MISRA C:2012 Amendment 1 (2016)
  - MISRA C: 2012 Addendum 2 (2018)
  - MISRA C:2012 Addendum 3 (2018)
  - MISRA C:201X
  - MISRA C:202X

오류의 최소화 (비용대비)

- 소프트웨어 개발 프로세스 활용
  - 명세 오류 예방/제거
  - 설계 오류 예방/제거
  - 코드 오류 예방/제거
  - 단계별 테스트 방법 사용
    - 단위 테스트
    - 모듈 테스트
    - 통합 테스트
    - 시스템 테스트

SW 신뢰성을 위한 테스트 종류

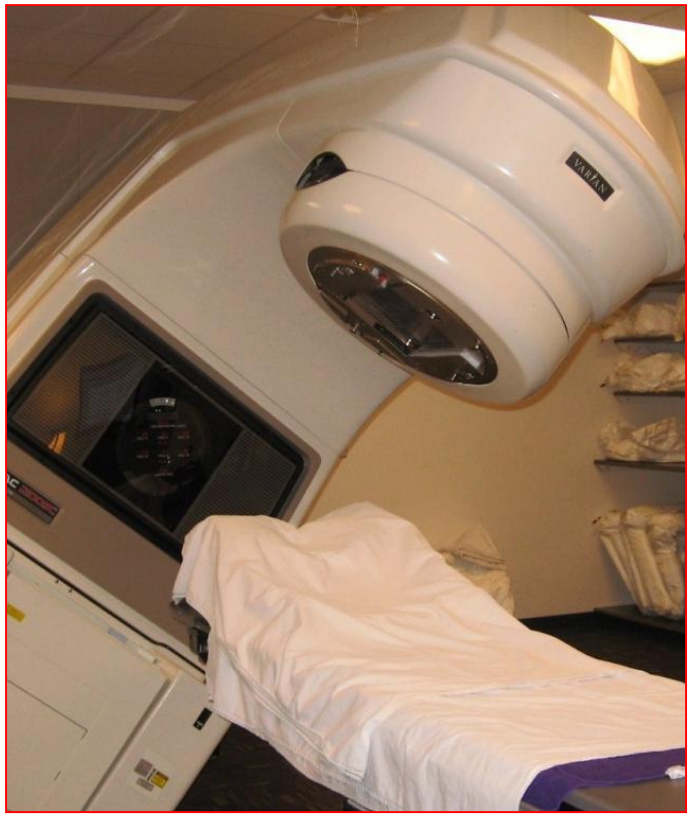
- 테스트
  - **확인 테스트** (Validation Test) : 목적은 결함을 찾는 것이 아님
  - **오류 테스트** (Defect Test) : 목적은 결함을 찾는 것임.

- 코드 실행 여부
  - 정적 테스트 (Static Analysis, Static Test) : 코드 미 실행
    - 코딩 규칙 체크
    - 실행시간 오류 검출 (버퍼오버플로우, divide by zero, 등)
  - 동적 테스트 (Dynamic test) : 코드 실행
    - 코드 실행률 (Code Coverage)
    - 문장 커버리지 : 프로그램 내 코드 실행 여부
    - 분기 커버리지 : 해당 분기문의 분기별 실행 여부
    - MC/DC (Modified Condition/Decision Coverage) : 독립적 변화에 따른 모든 조합에 대한 실행 여부

- 입력 데이터 여부
  - 일반 테스트
  - 심볼릭 테스트 : Concolic test

- 프로그램 내부 정보 여부
  - Black Box test
  - White Box Test

제어기기 사고



참조:http://hackaday.com/2015/10/26/killed-by-a-machine-the-therac-25/

- Thera-25 사고
- AECL 방사선 암 치료기
- 1982년 SW 내장형 제어 모듈 탑재 후 서비스 개시
- 1985년 ~ 1987년: 6번의 사고
- 3명 사망
  - 치사량을 초과하는 X선에 암환자가 노출
- SW 오류로 인명이 살상이 된 첫 사례
  - 제어 프로그램의 오류
  - 소프트웨어의 안전성에 대한 충분한 분석이 없이 하드웨어 안전장치 제거
- SW 공학에 Safety 개념 도입
  - 소프트웨어 내에 재난으로 이어지는 에러/오류가 없음
  - Software Engineering 10th edition, Ion Sommerville, 2016
- Reliability ≠ Safety
  - 단, 100% 신뢰성을 가진 시스템은 안전성이 보장됨.
- IEC 61508 제정
  - Functional safety of electrical/electronic/programmable electronic safety-related systems

SW 안전성(SW Safety)?

- 시스템 안전성 (Safety) 정의
  - Freedom from those condition that can cause death, injury, occupational illness, or damage to or loss of equipment or property to the environment

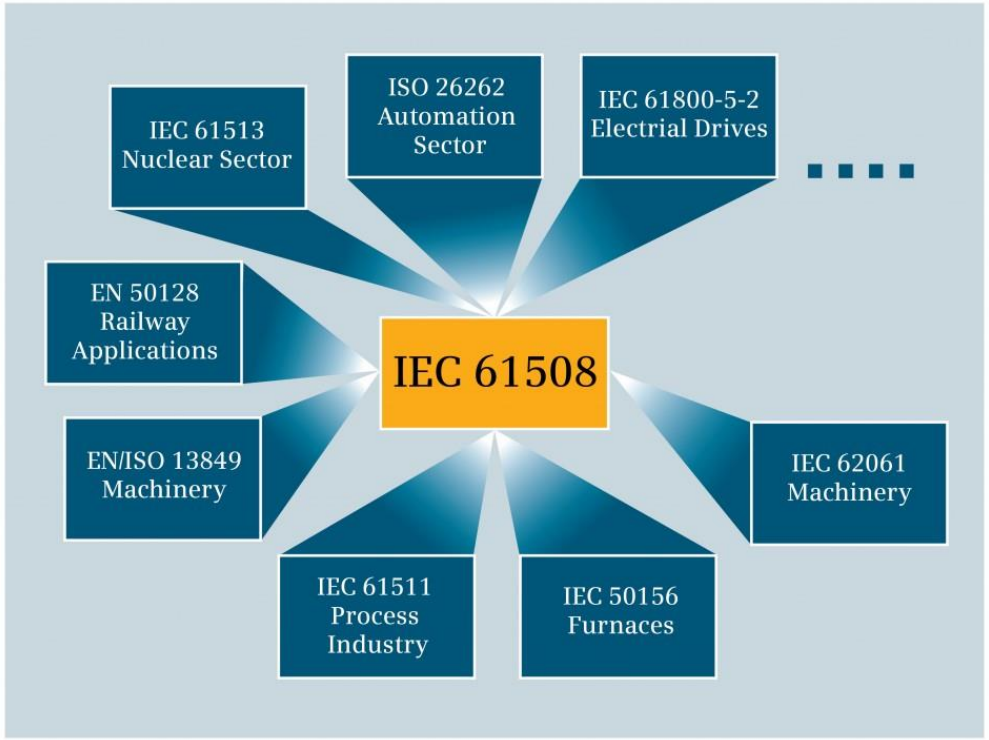
    ※ Air Force System Safety Handbook, 2000, MIL-HDBK-336B
- 기능 안전
  - Part of the overall safety that depends on a system or equipment operating correctly in response to its inputs, including the safe management of likely operator errors, hardware failure and environmental changes.

    ※ Focus Topics: Functional Safety, TUV SUD, 2016
- Safety-critical (안전필수, 고안전, 안전중심) 정의
  - A term applied to a condition, process or item of whose proper recognition, control performance, or tolerance is essential to safe operation or use; e.q., safety, critical function,, safety critical path ..

    ※ Air Force System Safety Handbook, 2000
- 소프트웨어 안전성 :
  - 소프트웨어 시스템이 재앙(인명의 손상, 시스템의 파괴 등) 없이 작동하는 능력
  - 소프트웨어 내에 시스템을 재앙으로 이끄는 오류가 없는 정도

    ※ Software Engineering 10th edition, Sommerville, 2016

IEC 61508

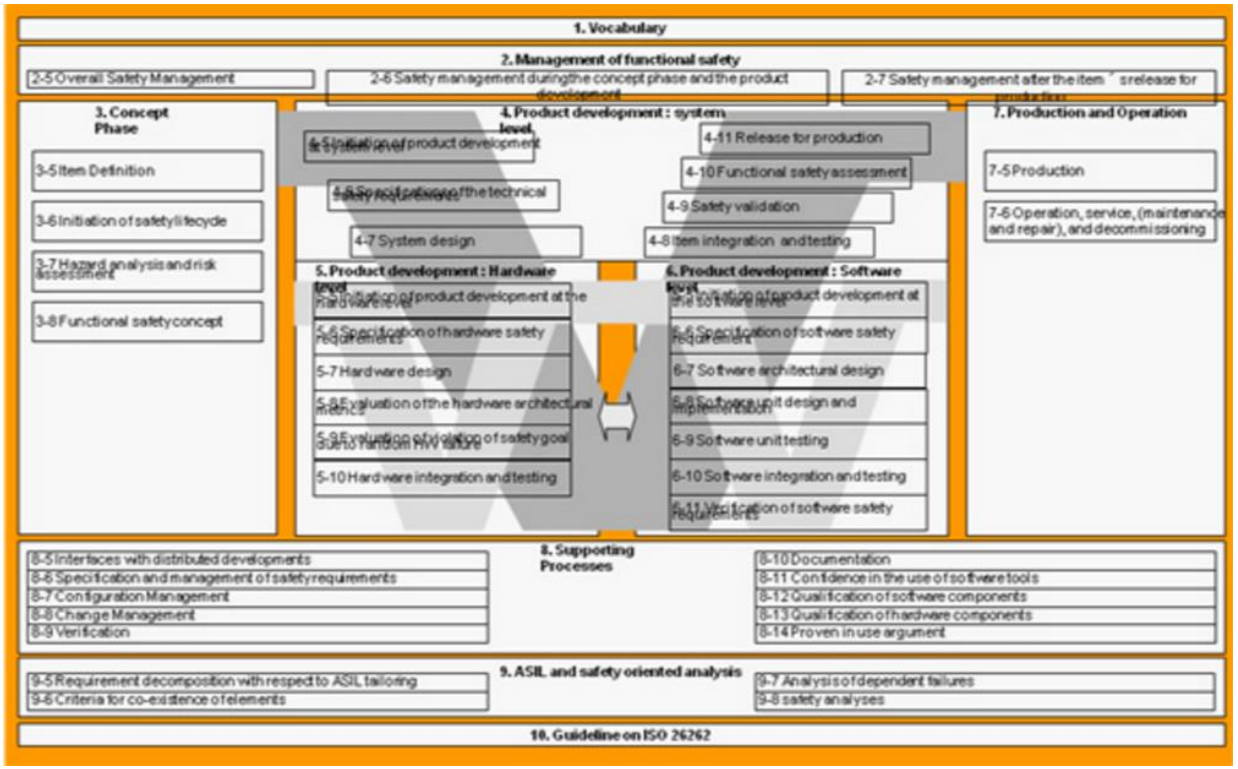


- 전기/전자/프로그램 가능한 전자 안전 관리 시스템의 기능안전 규칙의 국제 표준
- 기능 안전
  - E/E/PE 안전 관련 시스템의 정확한 기능, 다른 기술 안전 관련 시스템과 외부적인 위험 감소 설비에 의존하는 제어 대상 장비와 제어 대상 장비를 제어하는 시스템 관련된 부분적 또는 전반적인 안전
  - 컴퓨터 기반 시스템은 점차 안전기능에도 이용되고 있음.
- 안전 생명 주기 포함
- IEC 61508 에서 보는 위험(risk)
  - 위험은 항상 존재 (Zero risk can be never reached)
  - 안전은 개발 초기부터 고려되어야 함
  - 허용되지 않는 위험은 줄어야 한다.
- Safety integrity level (SIL)

소프트웨어 신뢰성 vs. 안전성

	신뢰성	안전성	비고
범위	전체 SW	일부분	
요구사항 분석	요구사항분석	기능 안전 분석	
명세	비정형/준정형	정형기법	
설계	비정형/준정형	정형기법	
코딩	방어적 프로그램 (코딩 규칙)	방어적 프로그램 (코딩규칙)	
테스팅	중요	오류삽입테스트	
보증	SW 품질 보증 (SW Quality Assurance)	SW 안전 보증 (SW Safety Assurance)	
새 기법		Safety Case	

ISO 26262:2011



- 목적
  - 자동차 분야에서 안전성 생명주기 (관리, 개발, 생산, 운영, 서비스, 폐기) 를 제공하며 이러한 생명 주기에서 발생하는 필요 엑티비티를 자세히 제공
  - 전 개발 주기에서 기능 안전성를 포함 (요구사항, 설계, 구현, 통합, 검증, 확인, 형상 등)(
  - ASIL (Automotive Safety Integrity Levels) 제공
    - 자동차 관련 위함 기반 분석
  - ASIL 을 이용하여 허용할 수 있는 위험 정도를 구현하기 위해 필요한 안전 요구사항을 명세
  - 구현될 충분하고 허용 가능한 안전성 레벨을 확인 및 확중을 위한 요구사항 제공

- 표준화 일정
  - 2009년 7월 : 국제 표준 드래프트 배포 (dis)
  - 2011년 4월 : 최종 국제 표준 프래프트 배포(FDIS)
  - 2011년 11월 : Part 1 ~ 9 국제 표준으로 확정
  - 2012년 8월 : Part 10 국제 표준으로 확정
  - 2018년 3월 : 2nd 에디션 배포 (공식배포 12월?)

Verification and Validation (ISO 26262)

- Verification(검증) : 올바르게 만들고 있는 가?
- Validation (확인/검정) : 올바른 것을 만들었는 가?

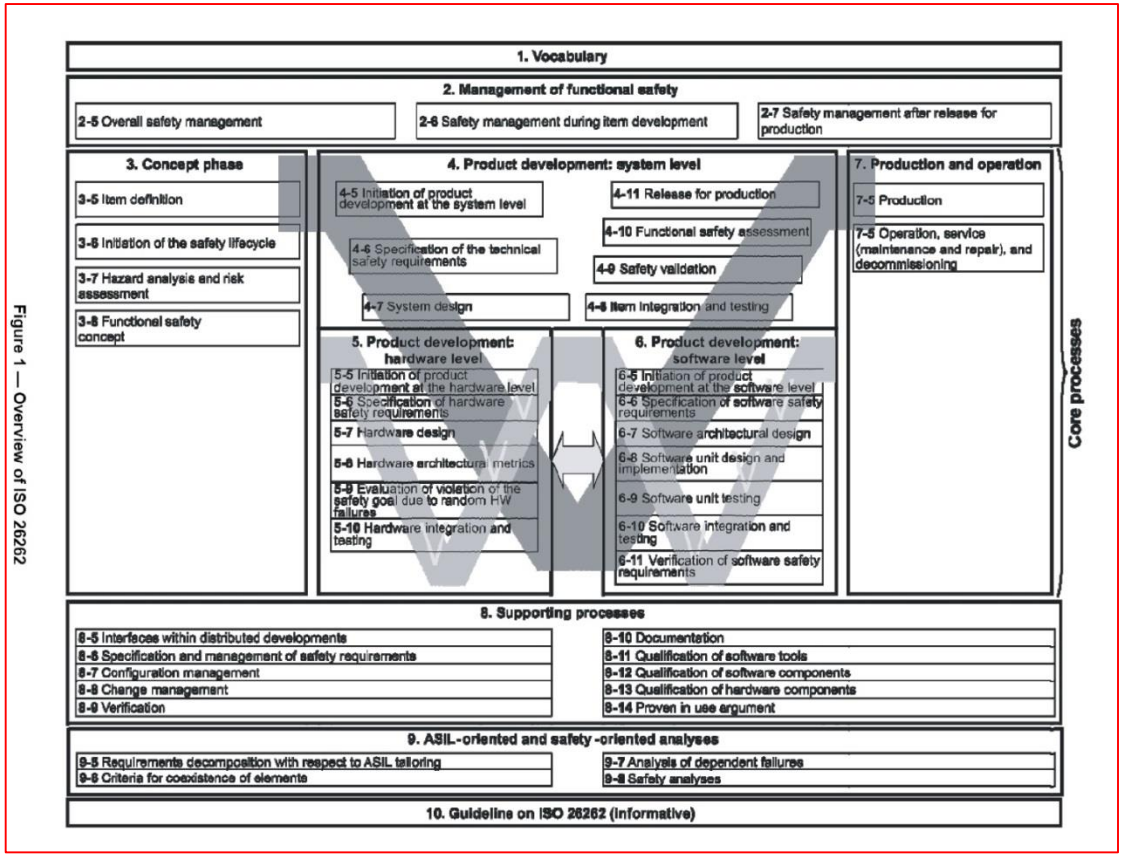


Figure 1 — Overview of ISO 26262

자동차와 소프트웨어

This Car Runs on Code

It takes dozens of microprocessors running 100 million lines of code to get a premium car out of the driveway, and this software is only going to get more complex

Posted 1 Feb 2009 | 5:00 GMT
By **ROBERT N. CHARETTE**

The avionics system in the F-22 Raptor, the current U.S. Air Force frontline jet fighter, consists of about 1.7 million lines of software code. The F-35 Joint Strike Fighter, scheduled to become operational in 2010, will require about 5.7 million lines of code to operate its onboard systems. And Boeing's new 787 Dreamliner, scheduled to be delivered to customers in 2010, requires about 6.5 million lines of software code to operate its avionics and onboard support systems.

These are impressive amounts of software, yet if you bought a premium-class automobile recently, "it probably contains close to 100 million lines of software code," says Manfred Broy, a professor of informatics at Technical University, Munich, and a leading expert on software in cars. All that software executes on 70 to 100 microprocessor-based electronic control units (ECUs) networked throughout the body of your car.

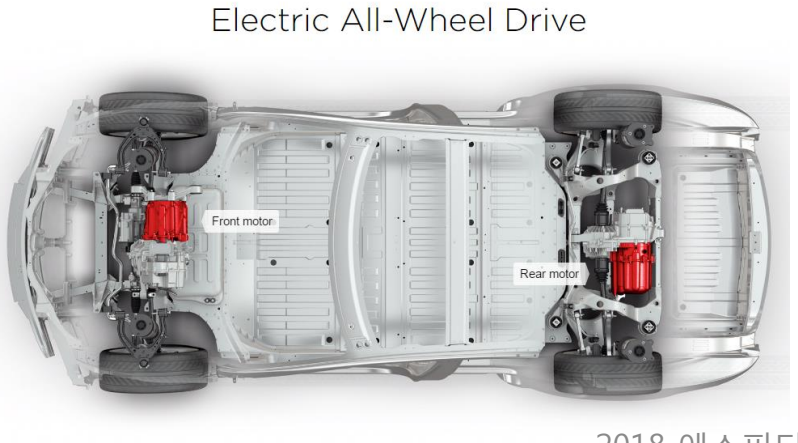
Alfred Katzenbach, the director of information technology management at

- 제너널 모터스 올스모빌 토로나도 (1977)
  - 역사상 첫 임베디드 소프트웨어가 내장된 자동차
  - ECU 를 이용하여 전자식 점화 시간 제어
  - 50,000 라인의 소프트웨어 (1981)
- F-22 Raptor : 170만 라인
- F-35 JSF (Joint Strike Fighter) : 570만 라인
  - JSF C++ Coding standard
- 보잉 787 : 650만 라인
- 벤즈 S-Class : 1억 라인
  - ECU 수 : 100
  - 네트워크 수 : 5
  - 케이블 길이 : 2마일
  - OS 수 : 10+
  - 비용 : 전체 비용의 50%
- 테슬라 로드스터 (2008), 모델 S (2012)
  - 소프트웨어 활용의 새로운 혁신
  - 리눅스 기반 컴퓨터 시스템이 대부분의 기능 제어
  - 소프트웨어 업데이트는 3G 통신망으로
    - 버그 수정
    - 새로운 기능 탑재

자동차와 소프트웨어 : 새로운 시대

2015 Tesla Model S P85D

현대 아이오닉 엔진룸



- 테슬라 로드스터 (2008), 모델 S (2012)
- 소프트웨어 활용의 새로운 혁신
- 리눅스 기반 컴퓨터 시스템이 대부분의 기능 제어
- 소프트웨어 업데이트는 3G 통신망으로
  - 버그 수정
  - 새로운 기능 탑재
- 소프트웨어 기반 전자식 파워트레인
- 듀얼 모터 P85D (2014) : 전방 후방 모터 사이의 토크를 10^{-3} 초 단위로 조절.
  - 트윈 터보 V-12 엔진 이상
  - 691마력, 3.1 초에 60MPH 가속
  - 유투브 동영상 : https://youtu.be/9cA1doO_9h8
- 자동차 제조사는 소프트웨어 기업

자동차의 새로운 위협: 사이버 시큐리티

IT 제품이 된 자동차… 치명적 해킹 위험에 떨고있다

[세계는지금]

"원거리 차량 해킹 언제든 가능"

인터넷에 연결된 커넥티드 카, 차량의 모든 기능 원격 조정

2년 만에 무선 해킹 비약적 발전… 美 정치권도 제도 보완에 나서

해커들의 원격 해킹으로 브레이크가 작동하지 않게 된 스포츠유킬리티 차량 지프 체로키가 도로변 구덩이에 빠져있다. 와이어드 제공

2년 전 발생한 독립언론 버즈피드 소속 마이클 해스팅스 기자(당시 33세)의 죽음은 많은 의

한국일보, 2015년 8월 2일 ,

2. Jeep 체로키 원격 조종 사건

올해 피아트 크라이슬러 (Fiat Chrysler)는 140만대가 넘는 자동차들에 대한 리콜 조치를 취했었죠. 이 이유는 2014년형 지프 체로키 (Jeep Cherokee)에 장착된 UConnect 기능에 치명적인 보안 취약점이 발견되었기 때문이었죠. UConnect 기능은 전화부터 자동차 내부 엔터테인먼트 그리고 와이파이 핫스팟을 제공하는데, 해커들은 이 UConnect 연결을 이용해 자동차의 IP주소를 알아냈고 미국 전역 어디에서나 무선으로 원격조종이 가능하도록 만들었습니다. 또한 해커들이 엔터테인먼트 시스템을 조종하는 칩에 접근해서 자동차의 펌웨어를 다시 만져 엔진이나 브레이크까지 손을 댈 수 있게 했죠. 피아트 크라이슬러 브랜드로는 치명적인 해킹 사례를 남기게 되었습니다.

올해 최악의 해킹 사례 12건, IT 인터넷, 2015, 12

사이버 시큐리티 101

- 개발시 보안성(시큐리티)를 고려하지 않은 시스템은 100% 해킹 가능하다.
- 자동차 개발시 ISO 26262 최고의 ASIL 을 받았다고 해킹에 안전한 것은 아니다.
- 즉, Safety ≠ Security
- 그렇지만, 해결 방법은 매우 비슷하다.

소프트웨어 보안성 (Security)

- 소프트웨어 보안성
  - 실수 또는 고의의 외부 침입으로부터 시스템을 보호할 수 있는 능력
  - The ability of the system to protect itself against accidental or deliberate intrusion.

  ※ Software Engineering 10th edition, Sommerville, 2016.

- 소프트웨어 보안(Software Security)의 다른 정의
  - 소프트웨어가 포함하고 있는 취약점(보안약점)의 밀도

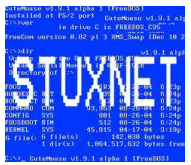
- 소프트웨어 보안의 어려운 점
  - 제로데이 취약점을 줄여야 함.
  - 취약점의 원인이 되는 보안약점을 줄여야 함.
  - 프로그래밍 언어의 의미론이 명확하지 않아, 모르는 보안약점이 존재함.
  - 공격자는 새로운 보안약점을 지속적으로 찾아내고, 이를 활용하여 제로데이 취약점 (사이버 무기)를 개발하고 있음.

- 명세에 정의 된 영역의 입력이 아닌 정의가 안된 범위의 입력을 사용
  - 예) SQL 삽입공격, 버퍼오버플로 공격 등

- 즉, 일반적인 테스트 방법 과 다른 방법이 활용

소프트웨어 보안은 보안 소프트웨어로 ?

(일반적인) 버그 vs. 보안 약점



- 소프트웨어로 인한 자동차 고장/사고의 원인
  - 일반적인 소프트웨어 버그
  - 외부 사이버 공격 (취약점, 보안약점)
- (일반적인) 버그
  - 정상적 작동 중에 설계/코딩 등의 잘못으로 발생하는 버그
  - 명세에 정의된 범위내의 입력으로 발생
  - SW 품질, 신뢰성, 안전성 저하
  - Therac 25 사고, Arian 5 사고, 세계종말 버그 (구 소련 레이더 장비 버그)
- 보안약점 (Weakness)
  - 공격자의 의도로 악용가능한 버그
  - 명세에 정의된 범위가 아닌 입력으로 발생
  - 사이버 공격에 사용
  - 스턱스넷, 해킹, 하트블리드

Software Weakness (보안약점)

보안약점이란, 소프트웨어 내에 있는

1. 소프트웨어 구현, 코딩, 설계, 구조에 내포된
2. Bug, flaw, fault, error, mistake.

- cwe.mitre.org

임베디드 SW 용 (특히 자동차용) 보안약점이 연구 중

Software Vulnerability (취약점)

취약점이란, 소프트웨어 내에 있는

1. Bug, flaw, error, mistake, weakness (보안약점) 으로
2. 공격자가 이러한 weakness 에 접근 하여
3. 시스템의 정보보증을 위태롭게 함.

- wikipedia.com

임베디드 소프트웨어 용 (특히 자동차 용) 보안 취약점 연구중

In computer security, a **vulnerability** is a weakness which allows an attacker to reduce a system's information assurance. Vulnerability is the intersection of three elements: a system susceptibility or flaw, attacker access to the flaw, and attacker capability to exploit the flaw.[1] To exploit a vulnerability, an attacker must have at least one applicable tool or technique that can connect to a system weakness. In this frame, vulnerability is also known as the attack surface.

소프트웨어 안전성 VS 소프트웨어 보안성

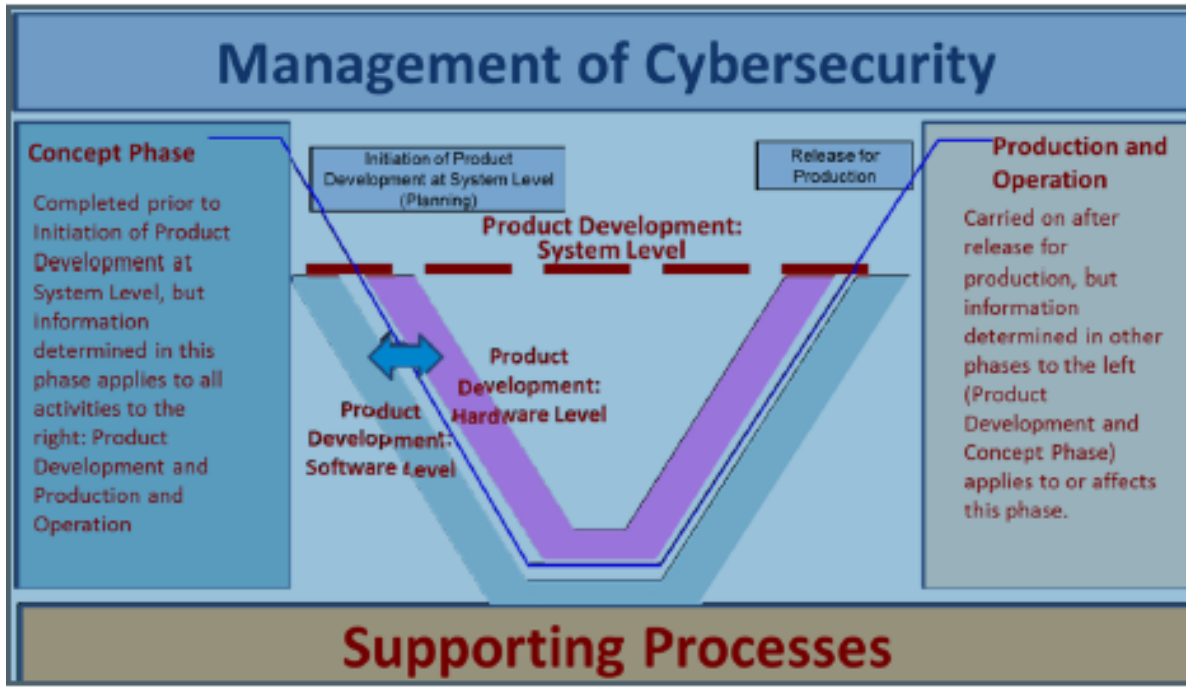
소프트웨어 버그로 인한 안전성 사고는 확률로 결정되나

소프트웨어 취약점이 있는 경우는 항상 (100%) 공격이 가능

MISRA-C 2012: Amendment 1

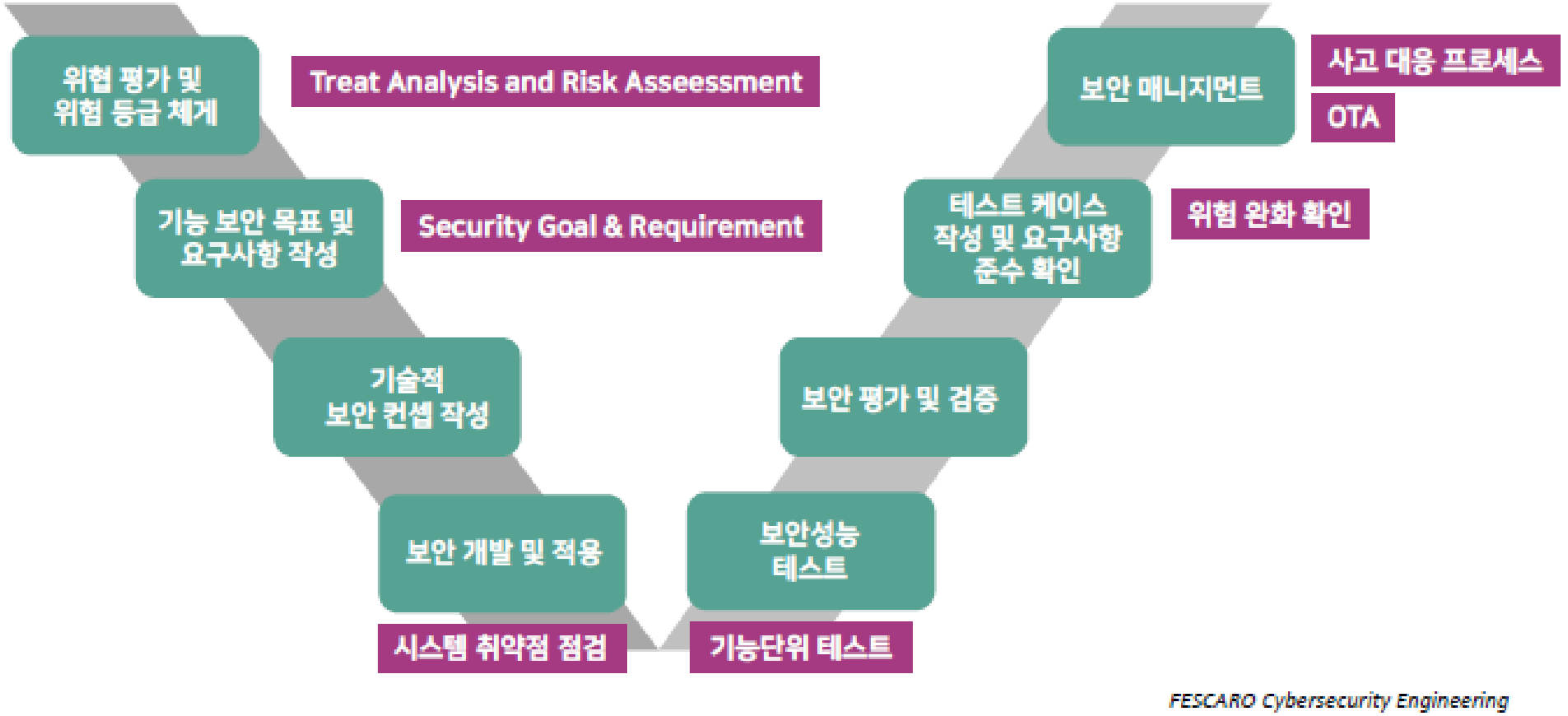
- MISRA C;2012 에 MISRA 가이드라인에서 요구하는 시큐리티 커버리지를 포함하는 시큐어 코딩룰을 포함
- MISRA 프로그래밍 언어 가이드라인의 모든 문서와 호환성을 가지고 있음
- 미래 MISRA C 가이드라인의 방향성과 일치함
- 본 가이드라인을 활용하면 개발자는 보안 취약점을 파생시킬 수 있는 코딩 방식을 피할 수 있게 해줌. 또한 코드를 더욱 읽기 편하고 관리 편하게 해줌
- 14개의 새로운 C 코딩 룰 : ISO C Secure 가이드라인에서 중요시하는 시큐리티 관련 분야를 커버함.
- 시큐리티 취약점과 관련이 있는 신뢰할 수 없는 데이터 (untrustworthy data) 에 관련된 이슈를 지적함.

SAE J3061



- SAE : Society of Automotive Engineers (미국 자동차 공학회)
- Cybersecurity Guidebook for Cyber Physical Vehicle Sysyetm
- ISO/SAE 21434 로 재정 중
- 자동차에서 1) 사이버시큐리티 설계, 구현하는 제계적인 엔지니어링 프로세스를 제공해 줌. 2) 침입을 모니터링하고 대응하는 엔지니어링 프로세스를 제공해 줌 3) 서비스와 운행 중 취약점에 대해 설명하는 엔지니어링 프로세스를 제공해 줌

ISO/SAE 21434 V-Cycle



미래 방향: 안전성과 보안성을 만족하는 효율적 통합 개발

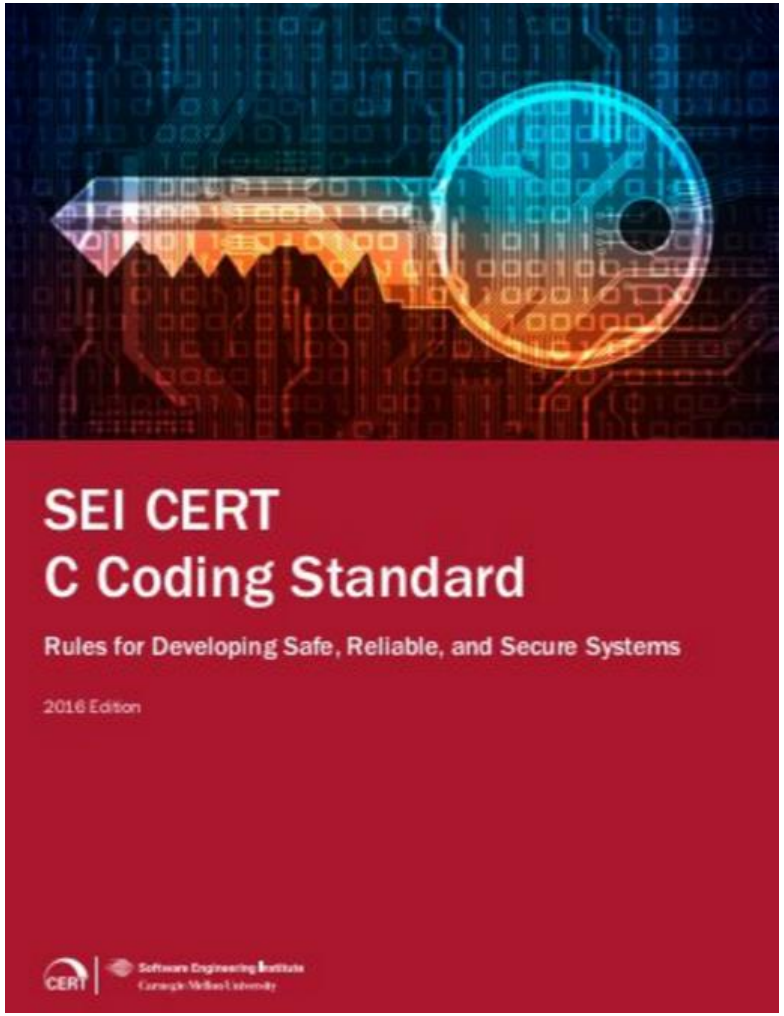
- 파워트레인
  - 에너지 효율
  - 예상하지 못한 속도 변경
- 운전자 보조장치
  - 자율 운전
  - 신호의 혼란
- 연결 (connectivity)
  - 24기간 연결
  - 갑작스런 운전자 방해

소프트웨어 신뢰성, 안전성, 보안성에서 통합 속성으로

	신뢰성	안전성	보안성	비고
범위	전체 SW	일부분	전체 SW	
요구사항 분석	요구사항분석	+ 기능 안전 분석	+ 보안 요구사항 분석	
명세	비정형/준정형	정형기법 (SIL4)	위협모델/위험 분석	
설계	비정형/준정형	정형기법(SIL4)	시큐어 설계	
코딩	방어적 프로그램 (코딩 규칙)/ 정적분석[1)]	방어적 프로그램 (코딩규칙)/ 정적분석[2)]	시큐어 코딩/ 정적분석	
테스팅	중요	오류삽인테스트	보안 테스트	
보증	SW 품질 보증 (SW Quality Assurance)	SW 안전 보증 (SW Safety Assurance)	SW 보안 보증 (SW Security Assurance)	
새기법		Safety Case	Security Case	
통합 보증 기법	SW Assurance			

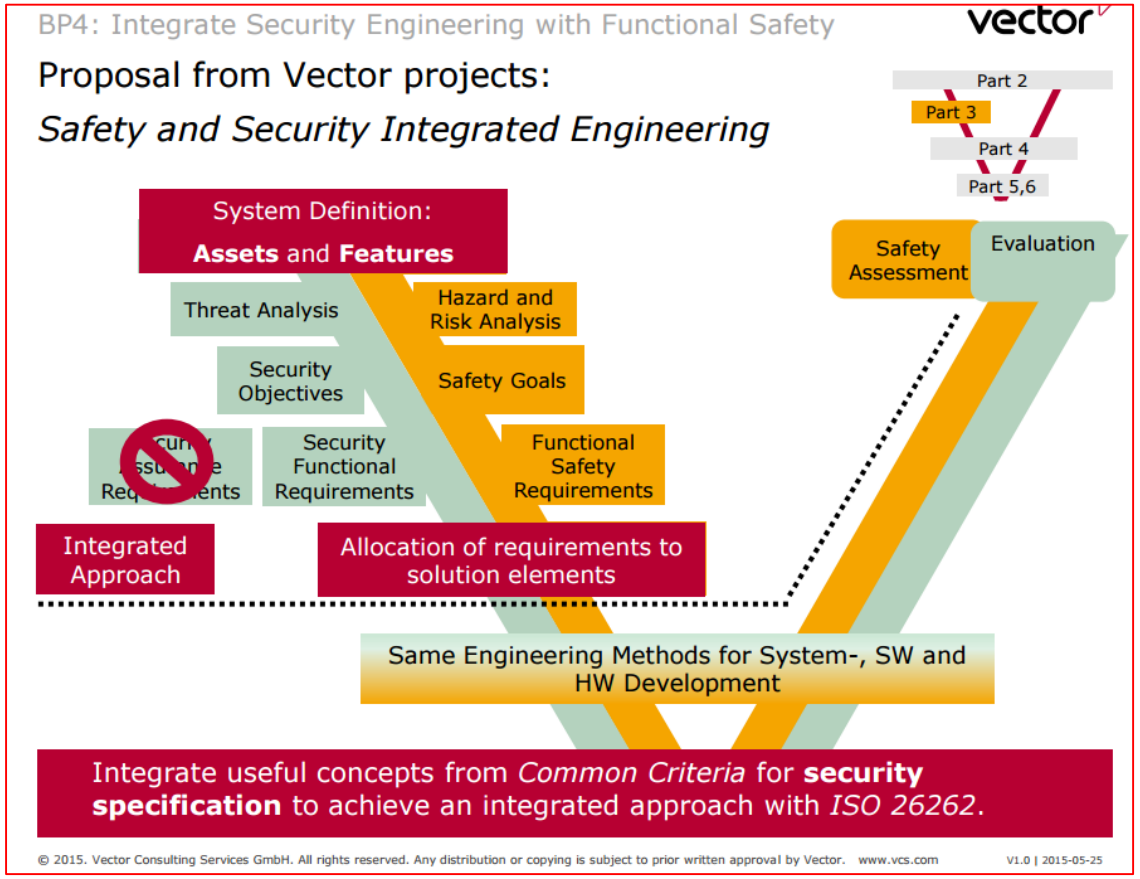
1) 명세에 정의된 입력범위
2) 명세에 정의된 입력범위 이외

MISRA-C 발전



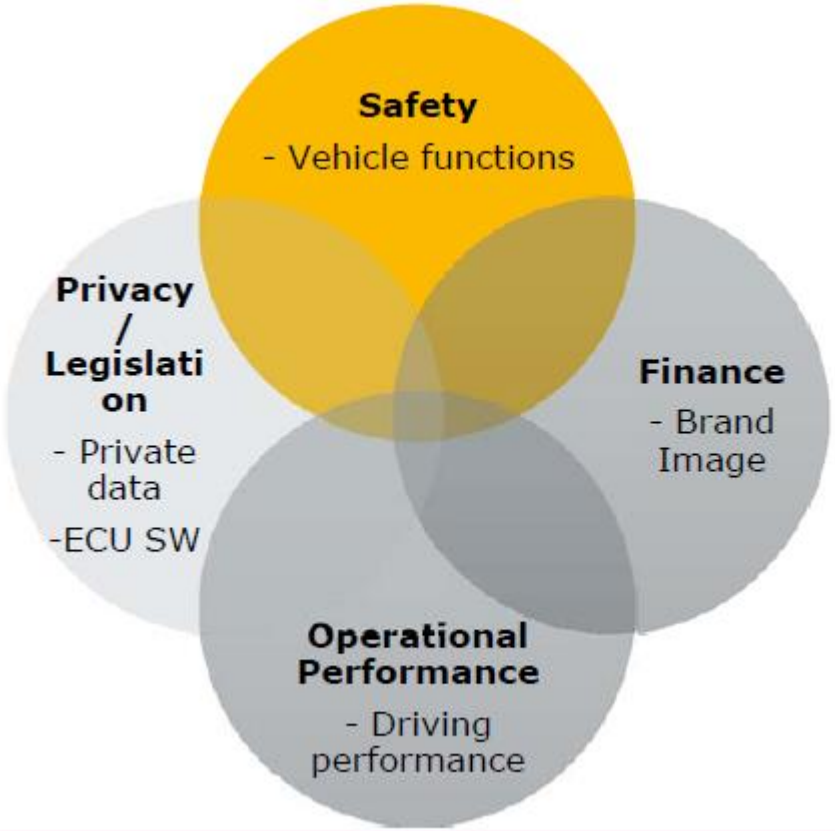
- MISRA-C
  - SW 안전성 중심에서 SW 안전성, 보안성 중심으로 변화하는 중
- SEI CERT C Coding Standard
  - 초기에는 SEI CERT C Secure Coding 으로 시작
  - SW 보안성 중심에서 SW 보안성, 안전성 중심으로 변화하는 중
- 최종에는 모든 코딩 표준/규칙은
  - 신뢰성, 안전성, 보안성을 포함하는 규칙으로 발전하고 있음
- MISRA-C:202X 는 보안성을 포함하는 코딩 표준으로 발전 예상

ISO 26262 발전 예상



- 2018년 ISO 26262에 SW 시큐리티 언급
- 202X년 ISO 26262 에 Safety 와 Security 를 함께 구현하는 Integrated 엔지니어링 기법 예상.
- Features vs. Assets
- Hazard/Risk Analysis vs. Threat/Risk Analysis
- Safety Goal vs. Security Objective (Goal)
- Functional Safety Requirement vs Security Functional Requirement

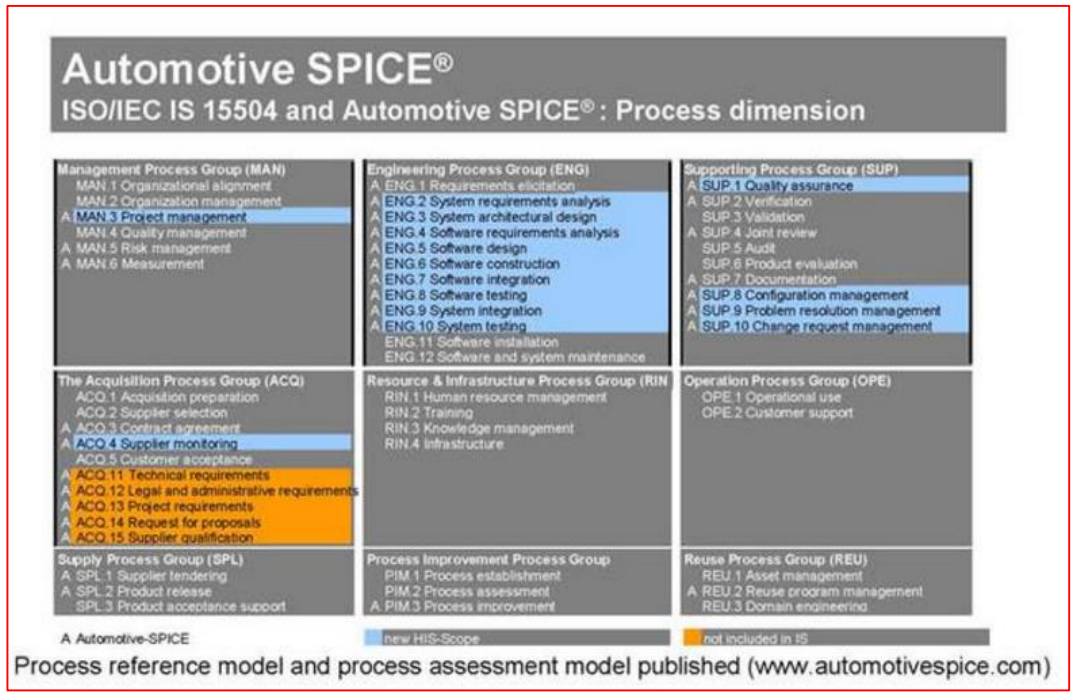
새로운 요구사항 분석



위협 분석

- 안전
  - 오동작으로 인한 부상
- 재정
  - 브랜드 이미지 손상으로 인한 판매 감소
- 운전 성능
  - 문이 잠겨야 함.
- 개인정보/준법
  - 개인정보 유출
  - ECU 소유권

자동차 제조기업의 성숙도 (BSIMM)



Process reference model and process assessment model published (www.automotivespice.com)

- 현재는 Automotive SPICE 활용중
  - 보안 관련 엑티비스 부족
- BSIMM = 일종의 CMMI 로 기업의 소프트웨어보증 성숙도 (신뢰성 + 안전성 + 보안성)
- Building Security in Maturity Model (BSIMM, “bee simm”으로 읽음)
- BSIMM 은 소프트웨어 보안의 현재 상태를 반영함.
- 4개의 도메인
  - 관리
    - 정책 및 메트릭
    - 준수 및 정책
    - 교육
  - 정보
    - 공격 모델
    - 보안 특징 및 설계
    - 표준 및 요구사항
  - 소프트웨어 개발 생명주기 touchpoints
    - 아키텍처 분석
    - 코드 리뷰
    - 보안 테스팅
  - 배포
    - 침입테스팅
    - 소프트웨어 환경
    - 형상관리 및 취약점 관리

새로운 속성 : Resilience (지속성)

[영상] 한국 자동차 센서 교란 공격에 성공...후방감지센서 조작 가능해

기사승인 2016.11.13 14:32:27

가- 가+

- POC2016서 中 첸얀, 자동차 센서 오작동 유발 공격 연구결과 발표

▲첸얀(Chen Yan), POC2016에서 자동차 해킹 발표

중국 제쟝대학교 박사과정 학생인 첸얀(Chen Yan)은 지난 10일 한국에서 열린 POC2016 "Can You Trust Autonomous Vehicles: Contactless Attacks against Sensors of Self-Driving Vehicles"을 주제로 발표를 진행했다.

자동주행 자동차에 사용되고 있는 Ultrasonic(울트라소닉) 센서와 MMW 레이더 등에 대한 스푸핑 및 재밍 공격 시연, 특히 테슬라 S모델에 대한 실제 공격 데모를 선보였다. 특히 국내 자동차에 대해서도 실제 테스트를 진행해 시스템 조작이 가능하다는 것을 공개했다. 그는 국내 자동차뿐만 아니라 대부분의 해외 자동차에도 공격이 가능하다고 밝혔다.

- 센서에 변조된 입력값을 제공
- US 록히드 마틴 RQ-170 사건
  - GPS 신호를 조작하여 이란 공항에 착륙하게 함.
- 자동차도 다양한 센서가 있고, 안전성, 보안성이 만족을 해도 외부 입력 자체가 조작된 경우 해결할 수 없음.
- 다양한 센터를 이용하여 해킹된 센서 입력을 확인하여 공격을 받았음을 인지하고 지속적으로 운항가능하게 함.

결론

"Those who will be able to conquer software will be able to conquer the world.”

-- Tadahiro Sekimoto, president, NEC Corp.