Research Report

연구보고서 2017-008

CPS의 기능, 비기능적 소프트웨어 요구사항 분석

Functional, non-functional Software requirements analysis of Cyber Physical System environment

진회승 / 서영희 / 김원태

2018.04.

SPRi 소프트웨어정책연구소

이 보고서는 2017년도 과학기술정보통신부 정보통신·방송연구개발사업의 연구결과로서 보고서 내용은 연구자의 견해이며, 과학기술정보통신부의 공식입장과 다를 수 있습니다.

목 차

표 목 차

그 림 목 차

요 약 문

1. 제 목

CPS의 기능, 비기능적 소프트웨어 요구사항 분석

2. 연구 목적 및 필요성

제4차 산업혁명은 사회, 경제적, 기술적 변화로 나타났으며, 그 중 CPS는 디지털과 인간을 포함한 자동차, 항공기 등 제품, 크게는 공장, 도시 등 물리세계의 융합을 통해 생산성, 편리성과 안전성 등의 증가가 가능하게 하는 시스템이다. CPS는 시스템 자체가 복잡하고 다양한 응용 도메인 간에 차이점이 많이 있으나, CPS라는 개념 하에 속하는 여러 시스템은 관련 기술과 구축방법에서 공통적으로 활용이 가능한 영역이 있다. 이에 CPS 응용도메인에서 연구를 시행하고 시스템을 구축할 때, 활용이 가능한 CPS 공통의 특성을 정의하고 구체화하는 연구가 필요하다.

이 연구에서는 CPS 공통의 특성을 도출하고, 그 특성별 요구사항을 정리하여 CPS 응용 도메인 연구나 구축을 위해 기초 자료로써 사용하고자 한다. 이를 위하여 이 보고서에서는 첫째, CPS의 특성에 대해 연구하여, 공통적인 CPS 시스템 요구사항을 도출하고자 한다. 다음으로 CPS를 이루는 기술 중 소프트웨어로 구성되어진 CPS SW플랫폼 구조를 제안하고자 한다.

3. 연구의 구성 및 범위

이 연구는 총 7장으로 구성되어 있다.

제 1장은 서론으로 CPS의 소프트웨어 요구사항 분석에 대한 연구의 배경, 필요성, 연구의 목적 및 연구 수행 방법을 제시한다.

제 2장은 이론적 배경으로 CPS의 정의 및 발전단계, 요구공학, 소프트웨어 플랫폼에 대해 설명하고, 선행연구에 대해 분석한다. 첫 번째로 이 연구의 대상 시스템인 CPS의 정의, 특징, 응용 도메인, 수준별 단계에 대해 알아보겠다. 두 번째로는 소프트웨어 요구공학의 정의, 특성 및 필요성에 대해 다룬다. 세 번째로 CPS 플랫폼 연구를 위해 소프트웨어 플랫폼, CPS 플랫폼과 유사한 플랫폼인 IoT 플랫폼에 대해 살펴본다.

제 3장은 CPS 요구사항 분석의 틀을 설명한다. 사용자 요구사항 분석의 틀로

NIST(National Institute of Standards and Technology, 미국 표준기술연구소)의 CPS 프레임워크를 활용하고자 하며, 시스템 요구사항 분석의 틀로 CPS 특징을 이용하고자 한다. CPS 특징은 ‘시스템 확장성(Scalability), 융복합성(Composability), 상호작용성(Interactivity), 고신뢰성(Dependability), 실시간성(Timing), 상호운용성(Interoperability), 지능성(Intelligence)’로 선정하고 시스템 요구사항을 위한 기준으로 삼는다.

제 4장은 미국의 NIST의 CPS 프레임워크를 활용하여 정의된 CPS 프레임워크를 이용하여, CPS 분석의 대상 도메인으로 선정한 자동차, 항공기, 스마트공장의 사용자 요구사항을 분석한다. 자동차, 항공기, 스마트공장에서 CPS 프레임워크를 이용하여, 각각의 요구사항을 추출한다.

제 5장은 CPS의 특성인 확장성, 융복합성, 상호작용성, 고신뢰성, 실시간성, 상호운용성, 지능성을 기준으로 자동차, 항공기, 스마트 공장의 시스템 요구사항을 도출하고, 각 도메인의 공통의 시스템 요구사항을 추출한다.

제 6장은 IoT 플랫폼과 CPS의 공통의 요구사항을 추출하여, IoT 플랫폼 상에서 CPS 요구사항의 구현 가능한 정도를 검증하고, CPS 플랫폼 구축을 위해서 필요한 기능들을 정의하고 개념적 CPS 플랫폼을 제안한다.

제 7장은 결론으로서 연구의 요약, 향후 연구 방향, 연구의 한계 등에 대해 논의한다.

4. 연구 내용 및 결과

이 연구는 CPS 특성을 기반으로 CPS 공통의 요구사항을 도출하는 부분과, 도출된 CPS 공통의 요구사항과 CPS의 응용도메인인 스마트공장의 요구사항을 만족하는 CPS SW 플랫폼에 대한 연구로 구성되어 있다.

첫 번째 연구 내용은 CPS 특성을 기반으로 CPS 공통의 요구사항을 도출하는 것이다. CPS의 사용자 요구사항을 추출하기 위해서 자동차, 항공기, 스마트공장을 분석할 수 있는 공통적인 틀로 미국의 NIST(National Institute of Standards and Technology, 미국 표준기술연구소)의 CPS 프레임워크를 활용한다. NIST의 CPS 프레임워크는 CPS 분석 연구 방법론과 그것을 설명하는 용어를 포함하고 있어 CPS에 대한 보편적인 분석을 가능하며, 이를 활용하되 본 연구의 목적인 소프트웨어 관련 요구사항 추출을 위해 필요한 요건들만 선택하고 수정한 CPS 프레임워크를 정의했다.

이 연구에서 사용자 요구사항을 정의함에 있어서 국제 표준문서, 논문, 보고서 및 국내 최고 산학연 전문가들 인터뷰 등을 통해 요구사항이 올바르고 모호하지 않도록 하였다.(Correctness, Unambiguous) 특히, 자동차와 항공 표준에 포함되지 않은 자율주행차 및 무인기 측면에서 CPS 요구사항을 추가하여 CPS 요구사항의 완전함

(Completeness)을 높였다.

CPS 프레임워크는 기능(functional), 인간(human), 신뢰성(trustworthiness), 시간(timing), 데이터(data), 조립(Composition)의 6개의 측면과 각 측면들과 관련된 관심사로 구성되어 있다. 자동차, 항공기, 스마트공장에 대해 6개의 측면과 관심사 관점에서 사용자 요구사항을 정리하였으며, 자동차에서 66개, 항공기에서 83개, 스마트공장에서 96개의 사용자 요구사항을 추출하였다.

기능적인(Functional) 측면은 관심사 중 Actuation, Controllability, Measurability, Sensing 등은 자율주행자동차, 무인항공기 등의 요구사항을 나타내 지능성이 필요한 CPS에서 더욱 요구되는 사항이라 볼 수 있었다. 인간적(Human) 측면은 자동차의 경우는 개발 시의 역할, 자율주행차의 경우는 운전자와의 관계를 나타냈다. 신뢰성(Trustworthiness) 측면은 CPS에서 중요하게 다루어져야 하는 것으로, 안전, 보안, 복구, 신뢰성 등 CPS를 구현하기 위해서 반듯이 제공되어야 하는 요건이 추출되었다. 시간(Timing) 관련해서는 실시간성, 시간 동기화, 정확성, 시간 간격에 대한 조건과 지연 시처리 방법 등이 요구되었다. 조립(Composition) 측면에서는 모듈화, 모형화, 계층화되어 변경 없이 쉽게 복잡한 시스템이 서로 연동이 요구되었다.

도출한 각각의 기능은 CPS 특성을 기반으로 자동차, 항공, 스마트공장 시스템 요구사항으로 정리했다. CPS 특성은 선행연구를 통하여 시스템 확장성(Scalability), 융복합성(Composability), 상호작용성(Interactivity), 고신뢰성(Dependability), 실시간성(Timing), 상호운용성(Interoperability), 지능성(Intelligence)으로 선정하였다. 자동차, 항공, 스마트공장의 각각의 시스템 요구사항 중 공통의 요구사항을 추출하여, CPS 공통의 시스템 요구사항으로 정의했다. 시스템 요구사항은 자동차에서 108개, 항공기에서 62개, 스마트공장에서 102개, 공통으로 35개를 추출하였다.

시스템 확장성에 속하는 공통 시스템 요구사항은 통신 관련 기능이 대부분이었다. 스마트공장의 경우는 공장 도메인 기능에 속하는 공정 관련 사항과 공장 구성요소 변경에 관한 요구사항이 많아 공통 기능에는 포함하지 않았다. 융복합성, 고신뢰성, 상호운용성에 속하는 자동차 요구사항은 자동차 표준인 AUTOSAR에 관련된 것이 많았으며, 스마트공장의 경우도 도메인에 관련된 요구사항이 많았다. 지능성에 포함된 요구사항은 도메인에 속하는 요구사항이 많이 공통의 요구사항으로 도출하기는 어려운 면이 있다. 상호작용성의 경우는 통신에 관련된 일반적인 요구사항이 많아 공통요구사항으로 도출되었다.

두 번째 연구 내용은 CPS SW 플랫폼에 대한 것이다. 도출된 CPS 특성별 시스템 요구사항은 CPS SW 플랫폼의 기본 기능으로 구성된다. IoT 플랫폼 기반의 CPS SW 플랫폼을 정의하기 위해, 기존에 존재하는 IoT 플랫폼이 어느 정도 CPS의 기능을 충족하는

지 검증했다. 검증 대상은 스마트공장과 CPS 기본 기능으로, 검증을 위한 IoT 플랫폼은 OneM2M 표준을 만족하는 Mobius 플랫폼으로 선정했다. 추출한 CPS의 기능이 IoT 플랫폼에서 어느 정도 만족되는 지 검증하고, IoT 플랫폼이 제공하지 못하는 CPS 기능을 정리하였다.

추출한 CPS 시스템 요구사항 중 OneM2M 표준이 제공하는 요구사항을 분석하여 검증이 가능한 37종의 시스템 요구사항을 도출하였다. CPS 시스템 요구사항 중 CPS 특성별로 확장성 7종, 융복합성 6종, 상호작용성 12종, 고신뢰성 3종, 실시간성 1종 그리고 상호운용성 8종이 도출되어 총 37종의 요구사항이 검증 대상으로 식별되었고 지능적 요구사항은 AI와 같은 지능형 서비스와 관련된 요구사항들이 많아 IoT 플랫폼을 통한 검증에는 어려움이 있어 검증 대상에서 배제되었다.

CPS 시스템 요구사항 충족을 검증하기 위한 Mobius 기반 검증 테스트베드를 설계, 구축, 검사 및 결과분석 과정에서 SW 개발 프로세스를 적용하였다. 먼저, 검증 테스트베드에 대한 다양한 사용자 요구사항 및 이를 지원하는 디바이스・서버・네트워크 측면에서의 시스템 요구사항을 정의하였고, 검증 테스트베드 네트워크 구조를 설계 및 구현하였다. IoT 플랫폼 기능으로서 검증이 가능하다 판단되는 CPS 시스템 요구사항들을 검증하기 위한 검증용 프로그램들을 설계하고, 구현하였다. 마지막으로, 테스트케이스를 활용한 시험계획과 절차를 정의하고 검증 결과를 도출하였다.

검증결과 37종의 검증 대상 CPS 요구사항들은 IoT 플랫폼에서 구현이 가능함을 확인하였으며, IoT 플랫폼이 제공하지 못하는 CPS 요구사항들을 추가하여, CPS SW 플랫폼을 제안하였다. CPS 플랫폼에서 추가적으로 제공해야 하는 기능들은 CPS 서브시스템의 유연한 변경을 위한 상황 파악 및 제어 기능, 모델링과 시뮬레이션을 통한 최적화 기능, CPS 서브시스템의 복잡성을 추적가능하게 하는 시각화 기능, 실시간성이 보장되는 오류 검출 및 이상상황 대응 기능, AI 등을 이용한 지능형 서비스 등이다.

이 연구에서 CPS SW 플랫폼을 서비스, 연동, 네트워크 계층으로 구분하고, 서비스 계층은 시스템 확장성, 융복합성, 상호작용성, 고신뢰성, 실시간성, 상호운용성, 지능성의 CPS 특성을 지원하는 기능과 모델링과 시뮬레이션, 지능형 데이터 처리/분석 기능으로 구성하였다. 연동 계층은 검증 결과에서도 확인이 되었듯이 대부분 IoT 플랫폼의 기능을 사용하여 구성할 수 있다. 네트워크 계층의 요소는 IoT 플랫폼에서 제공하는 기능에 추가하여 실시간성을 보장하는 기능이 추가되어야 한다.

이 연구에서 각각의 요구사항 리스트를 제공함으로써 구체적인 요구사항을 정의하였으며, CPS에서 필요한 요구사항을 CPS 플랫폼 형태로 구성하여 종합적인 CPS 시스템 요구사항을 제공하였다.

5. 정책적 활용 내용

이 연구결과는 자동차, 항공, 스마트공장의 사용자와 시스템 요구사항과 CPS 공통의 시스템 요구사항을 도출하고, CPS SW 플랫폼 구조를 제안함으로써 CPS 연구와 개발을 위한 정책 방향 마련에 기초 자료로써 활용될 수 있다.

특히 우리나라 경제에서 제조업 의존율은 다른 나라에 비해 상당히 높으나, 제조업 경쟁력이 저하되고 있어 유연성, 실시간성, 신뢰성, 지능성을 만족하는 스마트공장으로의 전환이 시급하다. 이 연구 결과가 스마트공장 소프트웨어 플랫폼을 포함한 대규모 CPS 플랫폼 구축을 위한 정책 기초 자료로써 활용 가능한 것으로 기대한다.

6. 기대효과

이 연구결과는 융복합 시스템의 소프트웨어 개발 프로젝트 시 우리나라에서 개발, 테스트 등 다른 단계에 비해서 기술 역량이나 경험이 부족한 요구사항 분석 단계에 활용되어, 성공적인 CPS 개발에 기여할 것이다. 또한 초기 단계인 CPS 연구 및 개발에 초석이 되어, 국내 CPS를 활용한 관련 산업 발전과 국제 경쟁력 확보에 기여하리라 기대한다.

SUMMARY

CPS, an important technology of the fourth industrial revolution, is a system that enables increase of productivity, convenience and safety through the convergence of digital and physical world. The CPS system is complex and there are many differences between various application domains of CPS. However, many systems belonging to the concept of CPS need to study and define CPS common characteristics that can be used in related technologies and development methods. In this study, I will summarize the requirements for each CPS characteristic and use it as basic data for CPS application domain research and development.

This study consists of research that derives common requirements of CPS based on characteristics and suggests CPS software platform that meets requirements. Automotive, aircraft, and smart factories were selected for analysis in order to extract the CPS user requirements. It uses the CPS framework of the National Institute of Standards and Technology (NIST) in the United States as an analytical framework. NIST's CPS framework includes a CPS analysis research methodology and terminology to describe it, allowing for a universal analysis of CPS. The NIST's CPS framework was used to define the revised CPS framework for software requirements extraction.

In defining the user requirements in this study, international standards documents, papers, reports, and interviews with top industry, academy experts in the country ensure that the requirements are correct and unambiguous. Completeness of CPS requirements has been increased by adding CPS requirements for autonomous vehicles and UAVs.

The CPS framework consists of six aspects of functional, human, trustworthiness, timing, data, and composition and concerns related to each aspect. User requirements are summarized in terms of six aspects and concerns about automobiles, aircraft, and smart factories. I have extracted 66 user requirements from the car, 83 user requirements from the aircraft, and 96 user requirements from the smart factory.

Each user requirement derives the automotive, aeronautical, and smart plant system requirements using CPS characteristics. CPS characteristics can be classified into the following categories: system scalability, composability, interactivity, dependability, timing, interoperability, and intelligence. Respectively. Common requirements of each of the system requirements of automobile, aviation, and smart factory were extracted and defined as CPS common system requirements. In this study, 108 automotive system requirements, 62 aircraft system requirements,

102 smart plant system requirements, and 35 common system requirements were extracted.

The second part is about the CPS software platform. The system requirements for the derived CPS characteristics should be provided as the basic functions of the CPS software platform. To this end, it was verified to what extent the existing IoT platform satisfies the functions of the CPS. The verification targets are smart factories and CPS basic requirements. The IoT platform for verification was selected as a Mobius platform that meets the OneM2M standard. I verified the degree of satisfaction of the extracted CPS functions on the IoT platform and summarized the CPS functions that the IoT platform does not provide.

I analyzed the requirements of the OneM2M standard among the extracted CPS system requirements and derived 37 system requirements that can be verified. Among CPS system requirements, there are 7 types of scalability, 6 types of composability, 12 types of interactivity, 3 types of dependability, 1 type of timing and 8 types of interoperability. Intelligent requirements were excluded from verification because of difficulties in verification through the IoT platform which does not provide intelligent services such as AI.

As a result of the verification, it is confirmed that the 37 CPS requirements to be verified can be implemented in the IoT platform. In addition to the CPS requirements not provided by the IoT platform, the CPS software platform is proposed by dividing into the service, interworking and network layers. The additional functions that the CPS platform should provide are context awareness and control for flexible changes to the CPS subsystem, optimization through modeling and simulation, visualization to track the complexity of the CPS subsystem, detection and abnormal situation response function, and intelligent service using AI.

In this study, the CPS system requirements are organized in the form of CPS platform, and comprehensive requirements are defined by providing a list of requirements. It is expected that CPS can be developed more fully and reliably by reducing the time and effort by utilizing the results of this research as a reference material for companies and researchers who wish to develop CPS in systematic research and application domain of new CPS.

CONTENTS

제1장 서 론

제1절 연구 배경 및 필요성

제4차 산업혁명은 사회, 경제적, 기술적 변화로 나타났으며, 특히 첨단 정보통신(ICT, Information and Communication Technologies) 기술이 경제·사회 전반에 융합되어 혁신적인 변화가 나타나는 현상이다. 제4차 산업혁명의 사회적 요인은 원거리 근무가 가능한 업무 환경의 변화, 신흥 중산층 증가, 노령화 등으로 있으며, 기술적 요인은 ABCi(AI, Big Data, Cloud, IoT), CPS(Cyber Physical Systems, 사이버물리시스템) 등의 ICT 신기술의 발전이다.[1] 사회적, 기술적 요인으로 인해 국내외에서 새로운 산업과 일자리가 나타나고 있다.

그 중 CPS는 디지털과 인간을 포함한 자동차, 항공기 등 제품, 크게는 공장, 도시 등 물리세계의 융합을 통해 생산성, 편리성과 안전성 등의 증가가 가능하게 하는 시스템이다. 유럽, 미국 등 기술 선진 국가에서는 장기적으로 계획을 세워 CPS에 대한 연구가 이루어지고 있으며, CPS 응용 도메인인 교통, 제조, 환경, 헬스케어, 스마트시티 등에 대한 연구 투자도 계속되고 있다.

유럽은 Framework Programme 7(FP 7) ARTEMIS(Advanced Research & Technology for EMbedded Intelligent Systems)와 Horizon 2010을 통하여 임베디드 시스템(embedded System)[2]과 사이버물리시스템(Cyber Physical Systems)에 대한 연구 투자를 활발히 진행하고 있다.(2007년~2020년)[3] 미국은 미국의 대통령과학기술자문위원회(President's Council of Advisors on Science and Technology, PCAST)의 보고서(2007년, 2010년)에서 CPS를 국가 경쟁력 강화를 위한 최우선 연구 과제로 선정하였으며, 2009년부터 NSF(National Science Foundation, 미국 국립과학재단)을 통한 대규모 연구 지원을 시작하였다.[4]

---

1) World Economic Forum(2016), The Future of Jobs, Employment, Skills and Workforce Strategy for the Fourth Industrial Revolution pp6~8, 2016.1.
2) 기계나 기타 제어가 필요한 시스템에 대해, 제어를 위한 특정 기능을 수행하는 컴퓨터 시스템으로 장치 내에 존재하는 전자 시스템, 임베디드 시스템은 전체 장치의 일부분으로 구성되며 제어가 필요한 시스템을 위한 두뇌 역할을 하는 특정 목적의 컴퓨터 시스템
3) https://artemis-ia.eu/ , artemis 홈페이지
4) PCAST(2010), Designing a Digital Future: Federally Funded Research and Development in Networking and Information Technology, 2010.12.26

이에 우리나라에서도 CPS의 중요성을 정부·연구기관·기업이 인식하고 관련 연구에 적극적으로 투자하고 관련 제품을 생산하는 것이 필요하다. 그러나 현재는 CPS 개념에 대한 여러 의견이 있으며, 그 특성에 관해서도 구체적으로 정의된 것을 찾아보기 어렵다.

물론 스마트도시, 스마트공장, 스마트 그리드 등 각각의 CPS 응용도메인에 대한 연구는 진행되고 구축되고 있으나, 각각의 CPS 응용도메인이 공통점을 가지고 연구를 진행하고 있는 것은 아니라, 서로 다른 중복적인 연구와 별개의 구축이 진행되고 있다. 새로운 CPS를 구축을 위해서 참고자료가 될 만한 CPS 관련 연구는 미국, 유럽 등에서 장기적으로 이루어지고 있으나, 아직은 개념적인 단계이거나 개별 응용 시스템에 대한 사례 연구를 하고 있는 실정이다. 국내에서는 CPS 자체의 연구보다는 개별 응용도메인 연구에 주력하고 있다.

CPS는 시스템 자체가 복잡하고 다양한 응용 도메인 간에 차이점이 많이 있으나, CPS라는 개념 하에 속하는 여러 시스템은 관련 기술과 구축방법에서 공통적으로 활용이 가능한 영역이 있다. 이에 CPS 응용도메인에서 연구를 시행하고 시스템을 구축할 때 활용이 가능한 CPS 공통의 특성을 정의하고 구체화하는 연구가 필요하다.

제2절 연구 목적

이 연구에서는 CPS 공통의 특성을 도출하고, 그 특성별 요구사항을 정리하여 CPS 응용 도메인 연구나 구축을 위해 기초 자료로써 사용하고자 한다. 이를 위하여 본 보고서에서는 첫째, CPS의 특성에 대해 연구하여, 공통적인 CPS 요구사항을 도출하고자 한다. 다음으로 CPS를 이루는 기술 중 소프트웨어로 구성되어진 CPS SW플랫폼[5] 구조를 제안하고자 한다.

CPS를 구축하기 위해서 가장 먼저 시행해야 할 것은 요구사항 분석이다. 소프트웨어 개발 과정은 분석, 설계, 구현, 테스트, 이행으로 이루어지며, 이 중 요구사항 분석단계는 소프트웨어 개발의 시작단계로 구축해야 하는 시스템을 분석하

5) 응용 프로그램을 작성하고 실행하는 데 사용되는 소프트웨어 환경. 여기에는 응용 프로그램 개발을 위한 컴파일러, 유틸리티 및 인터페이스와 같은 소프트웨어 도구와 응용 프로그램을 실행하기 위한 런타임 엔진이 포함. 자동차의 경우 AUTOSAR, 항공의 경우 ARNIC 653 등이 표준화된 플랫폼

고 밑그림을 그리는 단계이다. 이 밑그림이 구축하고자 하는 시스템의 목적과 기능에 부합하지 않으면, 구축되는 시스템은 사용자가 원하지 않는 다른 시스템이 될 것이다. 이에 요구사항 분석단계는 시스템 구축 시 가장 기본이 되는 단계이며, 서로 다른 도메인에 속하는 고객과 개발자가 요구사항의 의미를 동일하게 이해해야하기 때문에 어려운 단계이기도 하다.

특히 CPS는 여러 시스템이 복잡하게 연결되어 물리시스템과 사이버시스템이 동적으로 움직이고, 제한된 시간에 동시에 적용해야 하는 복잡한 요구조건이 필요하다. 이러한 복잡한 요구사항은 요구사항의 중복과 충돌의 정도가 심하여, 요구사항을 정제하고 추적하기가 쉽지가 않다.

CPS 특성을 연구하여 도출된 공통의 요구사항은 CPS 응용도메인에서 시스템 구축 시 시행해야 하는 요구사항 단계의 반복 작업을 방지하며, 필수적인 요구사항 도출에도 활용된다. CPS 시스템은 시스템 위의 시스템(System of System, SoS)[6]으로 매우 복잡하고 고려해야 하는 특성도 많아 도출된 CPS 기본 요구사항은 활용가능성이 많다.

또한 국내 소프트웨어 개발과정에서 가장 취약한 부분이 요구사항 분석 및 관리이기 때문에 CPS 기본 요구사항의 제공은 CPS 응용 도메인에서 올바르고, 모호하지 않고, 완료되고, 일관성 있고, 안전성 있는 요구사항 도출을 지원한다.

이를 위하여 CPS가 기본적으로 가지는 특성을 기반으로 CPS SW플랫폼을 개발할 때 기본적으로 필요한 기능에 대해 분석하고 정리한다. CPS의 기본적인 소프트웨어의 요구사항을 추출・분석・명세하여 향후 CPS를 개발하고자 할 때 요구사항 분석 가이드를 제공하고, CPS SW 플랫폼에 필요한 기능을 정리하여 개념적 CPS SW 플랫폼을 제안하고자 한다.

6) Systems of Systems (SoS)는 공통 목표를 위해 융복합된 대규모의 이기종 시스템을 말하며 독립적으로 운영되고 관리 될 수 있는 구성 요소로 구성됨

제3절 연구 내용

본 연구는 CPS 특성을 기반으로 CPS 공통의 요구사항을 도출하는 부분과, 도출된 CPS 공통의 요구사항과 CPS의 응용도메인인 스마트공장의 요구사항을 만족하는 CPS SW 플랫폼에 대한 연구로 구성되어 있다. 목적하는 CPS SW 플랫폼은 기존 IoT 플랫폼을 활용하여 어느 정도 구축이 가능한지 검토하여 추가로 필요로 하는 기능을 도출하고 개념적 CPS SW 플랫폼을 제안한다.

이론적 배경에서는 본 연구의 중심 개념인 CPS 정의, 특성, 중요성, 수준별 단계 등에 대해 정리한다. CPS의 응용도메인인 자율주행차, 스마트공장, 스마트 도시의 수준별 단계에 대해 조사하고, CPS의 수준별 단계에 대해서도 정의하였다.

또한 본 연구의 목적인 요구사항 도출을 위한 배경지식으로 요구공학, 요구사항의 분류, 도출 방법 등에 대해서도 정리한다. CPS 플랫폼의 구축에 활용하기 위한 CPS 플랫폼과 유사한 플랫폼인 IoT(Internet of Things, 사물인터넷) 플랫폼에 대해서 정의, 기능, 기존 플랫폼 등에 대해 확인한다. 선행 연구로 CPS 관련 연구와 임베디드 시스템과 IoT에서 요구공학 관련 연구에 대해 조사한다.

CPS 사용자 요구사항과 시스템 요구사항 분석을 위한 요구사항 분석의 틀에 대해 정의한다. 사용자 요구사항을 추출하기 위한 틀로 미국의 NIST(National Institute of Standards and Technology, 미국 표준기술연구소)의 CPS 프레임워크[7]를 사용한다. NIST의 CPS 프레임워크는 CPS 분석 연구 방법론과 그것을 설명하는 용어를 포함하고 있어 CPS에 대한 보편적인 분석을 가능하며, 이를 활용하되 본 연구의 목적인 소프트웨어 관련 요구사항 추출을 위해 필요한 요건들만 선택하고 수정하여 사용한다. 시스템 요구사항 분석을 위한 틀은 CPS 공통 특성을 활용한다. CPS 공통의 특성은 선행연구를 통하여 시스템 확장성(Scalability), 융복합성(Composability), 상호작용성(Interactivity), 고신뢰성(Dependability), 실시간성(Timing), 상호운용성(Interoperability), 지능성(Intelligence)으로 선정하였다.

첫 번째 연구 내용은 CPS 응용 도메인 중 자동차, 항공기, 스마트공장을 분석 대상 도메인으로 선정하여 사용자 요구사항을 도출하고, CPS 특성을 기반으로 각각의 도메인의 시스템 요구사항과 CPS 공통의 요구사항을 도출하는 것이다.

NIST의 CPS 프레임워크를 이용하여 재정의한 CPS 분석틀의 활용하여 연구 대상 도메

7) NIST CPS PWS(2016), Framework for Cyber-Physical Systems Release 1.0, 2016.5.

인의 대표적인 소프트웨어 표준인 AUTOSAR(AUTomotive Open System ARchitecture, 개방형 자동차 표준 소프트웨어 구조)[8], ARINC(Aeronautical Radio, Incorporated) 653[9], RAMI 4.0(Reference Architectural Model Industrie 4.0)[10]의 스마트공장 소프트웨어 관련 표준 등을 분석하여 자동차, 항공, 스마트 공장에서 필요한 기능을 도출한다. 도출한 각각의 기능은 CPS 공통 특성을 기반으로 자동차, 항공, 스마트공장 시스템 요구사항으로 정리한다. 자동차, 항공, 스마트공장의 각각의 요구사항 중 공통의 요구사항을 추출하여, CPS 공통의 시스템 요구사항으로 정의한다. 도출된 시스템 요구사항은 기능 요구사항과 비기능 요구사항으로 분류하여, 요구사항 도출 시 간과하기 쉬운 비기능 요구사항에 대해 강조하였다.

두 번째 연구 내용은 CPS SW 플랫폼에 대한 것이다. 도출된 CPS 특성별 시스템 요구사항은 CPS SW 플랫폼의 기본 기능으로 제공되어야 하며, 이를 위해 기존에 존재하는 IoT 플랫폼이 어느 정도 이 기능을 충족하는 지 검증한다. 검증 대상은 스마트공장과 CPS 기본 기능으로 자동차, 항공의 경우는 단일 시스템으로서 성격이 아직은 강해, 연결 중심의 IoT 플랫폼 활용은 연결 중심 시스템인 스마트공장을 대상으로 선정했다. 물론 자율주행자동차, 커넥티드카의 경우 다른 시스템과 연결되어 데이터 통신이 중요해지고 있고, 항공의 경우도 지상 관제시스템과 연결될 경우 정보 교환이 많아지고 그 범위가 확대되나,[11] 본 연구에서는 제외하였다.

검증을 위한 IoT 플랫폼은 OneM2M 표준 만족하는 플랫폼을 선정한 후 CPS SW 플랫폼 기능과 IoT 플랫폼 비교하여 IoT 플랫폼이 제공하지 못하는 CPS 기능을 분리한다. IoT 플랫폼은 OneM2M 표준 준수하고 오픈소스인 Mobius[12]를 이용하여 검증 플랫폼을 구성하고, 추출한 CPS SW 플랫폼과 IoT 플랫폼의 공통 시스템 요구사항이 제대로 작동하는 지 검증한다.

이를 통하여 IoT 플랫폼 위에 동작하는 CPS SW 플랫폼의 기본 기능에 대해 제안한다.

---

8) 전 세계 여러 완성차 업체, 부품업체들과 개발도구 생산자들(BMW, 보쉬, 콘티넨탈, 포드, GM, 푸조, 도요다, 폭스바겐, 덴쏘, 벡터 등)이 협력하여 공동 개발하고 표준화를 진행한 차량용 소프트웨어 플랫폼

9) Avionics Application Standard Software Interface, ARINC에서 규정한 항공용 애플리케이션 소프트웨어를 개발하기 위한 규격이며, 항공 SW인증 기준인 DO-178B Level A을 만족

10) 독일 Industry 4.0의 관련 기술을 적용한 참조모델로써 3차원적 해석을 통해 Industry 4.0에 관련된 기술을 체계적으로 분류하고 정의한 것

11) Volkan Gunes et all(2014), A Survey on Concepts, Applications, and Challenges in Cyber-Physical Systems, KSII TRANSACTIONS ON INTERNET AND INFORMATION SYSTEMS, 2014.12.

12) 전자부품연구원이 개발한 oneM2M을 기반으로 하는 IoT플랫폼으로써 최근 Mobius 2.0을 오픈소스로 배포

제4절 연구 방법

이론적 배경으로써 요구공학과 요구사항 분석 관련 특징 및 기법에 대한 관한 관련 문헌을 조사한다. CPS 및 소프트웨어 플랫폼에 대해 분석한다.

CPS는 자율주행자동차, 무인항공기 등을 포함한 스마트 교통시스템, 스마트 그리드, 스마트도시, 스마트공장 등 그 범위가 광범위하여 본보고서에서는 연구 대상을 자동차, 항공기, 스마트공장으로 그 범위를 한정하고, 자동차, 항공, 스마트공장, IoT 플랫폼 등의 전문가 인터뷰를 시행하였다.

<표 1-1> 전문가 인터뷰 현황

분야	연구소・대학	기업
SW 기술	RTOS(Real time OS), AI(2)	
자동차	AUTOSAR, 자율주행 (3)	AUTOSAR (1)
항공	ARNIC 653 (1)	ARNIC 653 (1)
스마트공장	개념, 플랫폼(1)	개념, 개발현황(5)
IoT	플랫폼(1)	플랫폼(1)
CPS	플랫폼(2)	

※ 괄호()는 전문가 수

자동차, 항공기, 스마트공장 구축을 위한 소프트웨어 표준 플랫폼과 관련 자료를 분석하고, 인터뷰 내용을 종합하여 3개 도메인에 대한 각각의 CPS 관련 시스템 기능을 추출한다. 사용자 요구사항과 시스템 요구사항을 도출하기 위한 분석의 틀은 3장에서 자세히 설명한다.

도출된 시스템 기능에서 CPS 소프트웨어의 공통적인 기능을 추출하고, 향후 CPS 개발을 위한 개념적인 CPS SW플랫폼을 제안한다. 이를 위해서 여러 IoT 플랫폼 중 표준 플랫폼 하나를 선정하여, 추출된 CPS 기능과 비교하고, IoT 플랫폼 상에서 동작하는 CPS 기능을 검증한 후, IoT 플랫폼 위에서 동작하는 CPS SW플랫폼 구조에 대해 제안한다.

제2장 이론적 배경

이 장에서는 CPS 분석을 통한 소프트웨어 요구사항 도출 및 CPS 소프트웨어 플랫폼 기능 제안에 앞서 배경지식으로서 CPS, 소프트웨어 요구공학, 소프트웨어 플랫폼에 대해 설명하고, 선행연구에 대해 분석한다.

첫 번째로 CPS의 정의, 특징, 응용 도메인, 수준별 단계에 대해 알아보겠다. 두 번째로는 점차 복잡해지는 소프트웨어 개발에 있어 필수적인 소프트웨어 요구공학에 대한 정의, 특성 및 요구공학의 필요성에 대해 다룬다. 세 번째로 소프트웨어 플랫폼, CPS 플랫폼과 유사한 플랫폼인 IoT 플랫폼의 특성에 대해 살펴본다. CPS, 요구공학에 대한 선행 연구를 분석한다.

제1절 CPS의 정의 및 발전 단계

1. CPS 정의 및 특성

CPS는 2007년 8월에 미국 대통령과학정책자문위원회(PCAST, President's Council of Advisors on Science and Technology)에서 발간한 보고서의 권고로 출발하였으며,[13] NSF는 연구 지원을 위한 자금을 지원하고 있다. 학계, 정부 및 산업계의 CPS 전문가들의 협력을 촉진하기 위해 미국에서 가상조직(Cyber-Physical Systems Virtual Organization ,CPS-VO)[14] 이 설립되었다. 유럽도 ARTEMIS(Advanced Research and Technology for Embedded Intelligence Systems), Horizon 2020을 통하여 대규모로 지속적인 CPS 연구를 지원하고 있다.

기존의 임베디드 시스템 연구에서는 주로 사이버시스템, 즉 소프트웨어에서의 연산기능에 초점을 맞추어왔지만, CPS는 물리 세계와 사이버시스템 간의 상호작용이 더 중요하다. 현실 세계의 다양한 센서가 감지한 신호들이 유·무선 통신을 통하여 컴퓨터에 전달되어 분석·처리되고, 다시 통신과 작동장치(Actuator)를 이용하여 물리시스템을 모니터링하고 제어하는 기능이 점차 현실화되면서 CPS가 등장하게 되었다고 볼 수 있다.

CPS는 임베디드 시스템이 확장되고, 진화된 개념이다. 임베디드 시스템은 컴퓨터 하드웨어, 소프트웨어, 전용 기능을 수행하도록 설계된 추가 기계 또는 기타 부품으로 구성되며,

13) PCAST(2007), Leadership Under Challenge: Information Technology R&D in a Competitive World
14) https://cps-vo.org/

어떤 경우에는 임베디드 시스템이 자동차의 ABS(Anti-lock Breaking System)처럼 큰 시스템 또는 제품의 일부라고 Jack Ganssle과 Mike Barr가 정의[15]하였다. Robert Oshana는 임베디드 시스템은 대규모 시스템의 특정 기능을 수행하기 위해 설계된 컴퓨터 시스템으로 종종 하나 이상의 실시간 컴퓨팅 제약 사항을 갖고 있으며 이는 하드웨어와 기계적 부분을 포함하는 큰 장치의 부분으로서 내장된다고 정의[16]하였다.

위키피디아에서는 임베디드 시스템을 더 큰 기계적 또는 전기 시스템 내에서 전용 기능을 갖춘 컴퓨터 시스템으로, 종종 실시간 전산 제약 조건을 가지고 있다고 정의[17]한다. 일반 임베디드 시스템의 특성은 범용 제품과 비교할 때 저전력 소모, 소형 크기, 견고한 작동 범위 및 낮은 단위당 비용 등이 있다. 이것은 제한된 프로세싱 자원의 대가로 이루어지므로 프로그램하고 상호 작용하는 것이 훨씬 더 어려워진다. 임베디드 시스템은 디지털시계 및 MP3 플레이어와 같은 휴대용 장치부터 신호등, 공장 제어기, 자동차, 항공기와 같은 복잡한 시스템까지 매우 다양하다.

대부분의 임베디드 시스템은 저렴한 비용으로 지속적인 조치를 수행하도록 설계되었다. 이러한 시스템의 대부분은 하드웨어 및 소프트웨어 측면에서 성능에 제약이 존재한다. 예를 들어, 일부 기능을 실행하는 동안 시스템이 높은 속도를 필요로 할 때 실시간으로 작동해야 하지만 다른 기능에서는 속도가 더 느릴 수도 있다.[18] 특히 많은 양의 데이터를 처리해야하는 시스템의 경우, 범용 임베디드 시스템의 속도 또는 비용을 특성화하는 것은 어렵다. Robert Oshana는 임베디드 시스템은 최종 사용자의 다양한 요구에 부응하는 범용 컴퓨터와는 극명하게 비교되며 범용 시스템에 적용됐던 소프트웨어 시스템의 개발 방법과 기법, 툴이 임베디드 시스템에는 쉽게 적용되지 못한다고 주장했다.

임베디드 소프트웨어는 이산 연산을 수행하는 컴퓨팅 장치에서 동작하지만, 결과가 적용되거나 표출되는 장치는 컴퓨터가 아닌 물리적 대상, 특히 기계인 경우가 대부분이다. 임베디드 소프트웨어는 미리 정의된 목적을 위해 물리적 입력 및 그 가공된 데이터를 이용하여 적절한 반응을 제공하기 위해 설계된 소프트웨어이며, 실시간성, 결정성, 반응성, 이질성, 생존성, 자원 및 환경의 제약 등의 특성을 가진다. [19]

CPS(Cyber Physical Systems, 사이버물리시스템)는 컴퓨팅 환경에서 물리적 환경 정보(데이

15) Jack Ganssle(2008), Mike Barr, Embedded Systems Dictionary
16) Robert Oshana & Mark Kraeling(2013), Software Engineering for embedded systems
17) 위키피디아, “Embedded system”
18) Di Paolo Emilio, Maurizio(2015), Embedded Systems Design for High-Speed Data Acquisition and Control, Springer
19) 한국정보통신기술협회(2011), 임베디드 SW정의 및 분류 가이드라인, 정보통신단체표준(TTAS)

터) 처리 결과로 물리 세계, 시스템 또는 프로세스를 제어하는 고신뢰 시스템(Dependable Systems)이다.

[그림 2-1] CPS 개념도

자료 : 손상혁(2016), 융합의 또 다른 이름, 사이버물리시스템, 지식의 지평

우리가 살고 있는 물리세계는 인간, 자동차, 항공기, 교통 시스템, 집, 로봇 등이 실제 공간에 존재한다. 이러한 물리 세계를 제어하기 위해 데이터를 수집하고 연산이 이루어지는 소프트웨어가 존재하는 컴퓨팅 공간을 사이버 세계라고 할 수 있다. CPS는 고성능 컴퓨팅(Computing), 통신(Communication), 실시간 제어(Control)에 기반을 둔 오류에 강한 지능형 시스템이며 인간 주변의 물리세계에 장착된 스마트센서[20]를 통해 데이터가 수집되고 연결된 통신망을 통해 연동된 소프트웨어가 데이터 분석을 통해 물리 현상을 분석하고 제어하는 기술이라고 할 수 있다. 물리세계는 예측가능하지 않고, CPS는 조절 가능한 환경에서 동작하지 않기 때문에 하위 시스템 오작동에 관해서 처리가 가능해야 하고, CPS는 신뢰성이 중요하게 된다 [trustworthiness].[21]

2000년대 초반 CPS 초기 연구에서 연구자들은 CPS를 다음과 같이 정의하여 CPS의 가장 큰 특징을 물리 세계와 사이버 세계의 통합이라고 주장하였다. RAJ Rajkumar는 CPS를 “컴퓨팅 및 통신에 의해 관찰, 조정, 제어 그리고 통합되는 작업을 수행하는 공학시스템”[22]이라고 설명한다. Edward Lee는 CPS를 “계산과 물리 프로세스의 통합”[23]이라고 설명한다.

20) 기능이 단순하고 정밀도가 낮으며 사용이 불편한 이전의 센서에 비해 센싱소자와 신호처리가 결합하여 데이터 처리, 자동보정 자가진단, 의사결정 기능을 수행하는 ‘소형, 경량, 고성능, 다기능, 고편의성, 고부가가치의 센서’

21) Edward A. Lee(2008), Cyber Physical Systems: Design Challenges, Object Oriented Real-Time Distributed Computing (ISORC), 2008 11th IEEE International Symposium on

22) Ragunathan(RAJ) Rajkumar, Insup Lee, Lui Sha, and et all.(2010), Cyber-physical systems: The next computing revolution, in Proc. of 47th IEEE/ACM Design Automation Conf., pp. 731-736

Peter Marwedel은 CPS가 “물리적 환경이 통합된 임베디드 시스템”[24]이라고 설명한다. Helen Gill은 “계산에 의해 통합, 관찰 및 제어되는 물리적, 생물학적, 공학적 시스템”[25]이라고 설명한다.

NIST에서는 CPS는 “물리적인 구성 요소와 연산 구성 요소로 구성된 상호작용 네트워크를 포함하는 스마트 시스템” 이라고 정의했다 [Interactivity, Intelligence]. NITRD(Networking and Information Technology Research and Development, 미국 국가과학기술위원회)에서는 CPS는 “인간 사용자를 포함한 물리적인 세계를 감지하고 상호작용하도록 설계된 임베디드 센서, 프로세서 및 액추에이터를 갖춘 스마트 네트워크 시스템으로, 안전에 중요한 응용 프로그램에서 실시간성과 성능 보장을 지원” 한다고 했다 [Interactivity, trustworthiness, Interoperability, Timing].[26]

국내 연구자들은 CPS를 다음과 같이 정의한다. CPS는 다양한 센서를 통하여 물리 세계의 정보를 수집·분석하고 그 결과를 작동장치(Actuator)를 통하여 물리 세계에 즉시 적용하여, 사이버 세계와 물리 세계가 데이터를 이용하여 서로 긴밀하게 연결되어 협력하는 시스템이다 [Composability, Interactivity].[27] CPS는 사이버 세계와 인간, 운영환경을 포함하는 물리세계의 긴밀한 상호 작용을 위한 실시간 자율 제어 시스템이다 [Interactivity, Timing].[28] CPS는 우리가 살아가는 물리 세계와 센서, 작동장치(Actuator), 임베디드 컴퓨팅 시스템 등으로 구성된 사이버 세계와의 융합하여 안전하고 신뢰성 있게 분산제어하는 지능형 시스템 구축 기술이다 [Composability, trustworthiness, Intelligence].[29]

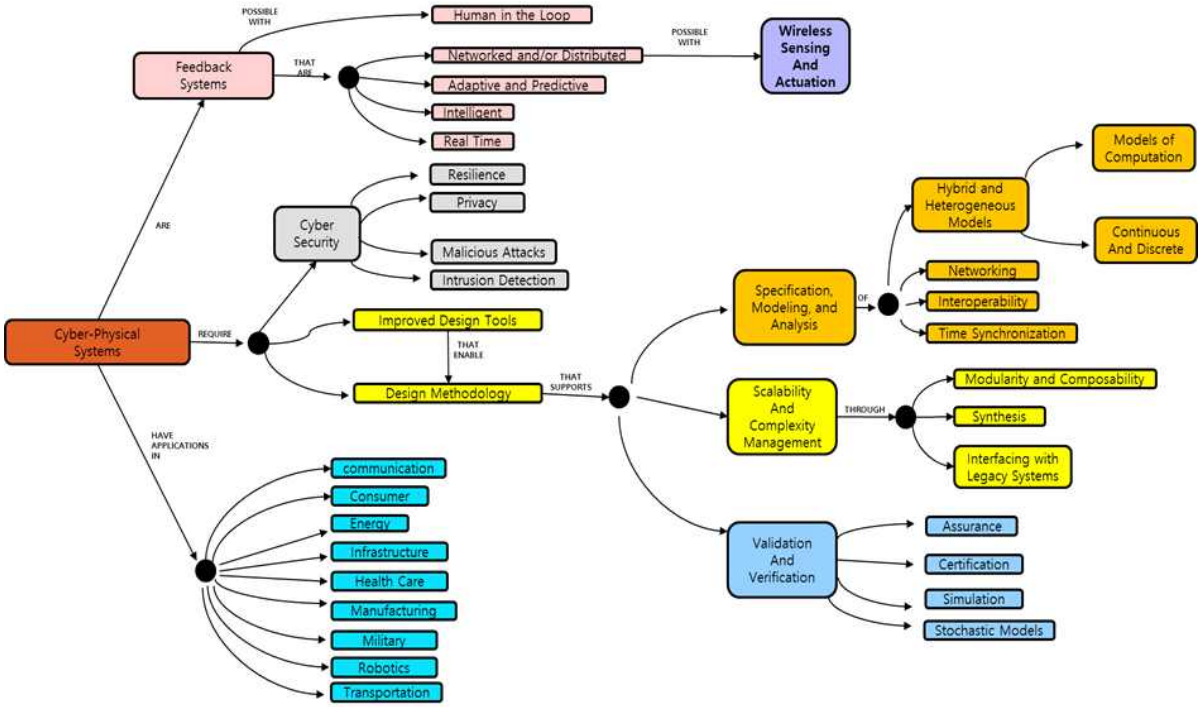
아래 그림은 UC 버클리 EECS(Electrical Engineering and Computer Sciences)에서 CPS의 개념과 관련 기술을 맵 형태로 구성한 내용이다. CPS는 시스템과 사람 간 시스템과 시스템 간 피드백이 가능한 시스템이다. 네트워크에 의해 분산된 제어 시스템이며, 고도의 적응성과 예측가능성을 가지며, 실시간성과 지능성을 제공한다[Interactivity, Interoperability, Composability, Timing, Intelligence]. 요구되는 기술로는 복구성(Resilience), 보안, 악의적인 공격, 침입 탐지와 같은 사이버 보안 기술이 있으며, 이기종 시스템, 네트워크, 시간동기화

---

23) Edward A. Lee(2006), Cyber-Physical Systems - Are Computing Foundations Adequate?, Position Paper in NSF Workshop On Cyber-Physical Systems: Research Motivation, Techniques and Roadmap
24) Peter Marwedel(2010), Embedded System Design, Springer, 2nd Edition
25) Helen Gill(2010), Cyber-Physical Systems: Beyond ES, SNs, and SCADA, Presentation in the Trusted Computing in Embedded Systems (TCES) Workshop, 2010.6.25.
26) NITRD(2012), CPS Vision Statement
27) 원명규, 장덕우, 장정훈 외(2014), 사이버물리시스템 기반의 스마트 시티 기술, 한국통신학회지, 31(8), 2014.7.,pp45-53
28) 손상혁(2016), 융합의 또 다른 이름, 사이버물리시스템, 지식의 지평
29) 한국전자통신연구원(2016), 가상-실공장 연동을 위한 CPS 기반 스마트팩토리 기술, 2016.11.

를 위한 모델링, 복잡성 관리, 검증 등의 설계 방법론이 있다 [trustworthiness, Interoperability, Composability, Timing].

[그림 2-2] CPS 관련 개념 맵



자료 : UC 버클리 EECS(Electrical Engineering and Computer Sciences), http://cyberphysicalsystems.org/

NSF(National Science Foudation, 미국 국립과학재단)는 CPS를 전산 알고리즘과 물리적 구성 요소의 완벽한 통합을 기반으로 구축되고 설계된 엔지니어링 시스템이라 정의했다 [Composability, Interactivity]. 또한 CPS는 오늘날의 단순한 임베디드 시스템보다 훨씬 뛰어난 기능, 적응성, 확장성, 복구성, 안전성, 보안성 및 유용성을 가능하게 한다고 했다 [Scalability, trustworthiness, Interoperability]. [30] NIST는 CPS에서 필요한 특징은 확장성, 상호운용성, 신뢰성, 융복합성, 실시간성 등이라고 했다 [Scalability, Interactivity, trustworthiness, Interoperability, Composability, Timing].[31]

위에 자료들에 나온 CPS의 특성을 종합해보면 **'시스템 확장성(Scalability), 융복합성(Composability), 상호작용성(Interactivity), 고신뢰성(Dependability), 실시간성(Timing), 상호운용성(Interoperability), 지능성(Intelligence)'** 이다. 고신뢰성(Dependability)은 외부 환경의 영향이나 시스템 내부의 오류에도 불구하고 시스템 성능과 산출물의 감소가 없이 목적하는

30) https://www.nsf.gov/funding/pgm_summ.jsp?pims_id=503286
31) NIST CPS PWS(2016), Framework for Cyber-Physical Systems Release 1.0, pp2-4, 2016.5.

기능을 수행할 수 있는 특징이다.

이러한 특징을 가지고 있는 CPS 시스템 구축에는 네트워크로 연결된 데이터 추출, 데이터 조작, 사이버시스템과 물리시스템의 유연한 연결, 물리시스템 제어 및 모니터링, 안전성 확보 등의 기능들이 필요하다.[32] CPS 기술요소는 센서 등의 디바이스, 통신 기술을 통한 가상세계의 만들어진 데이터의 연결·데이터 관리, 시뮬레이션과 모델링을 통한 분석 및 실제 시스템의 제어 기술이며, 대규모 CPS 지원을 위해서는 다양한 디바이스 지원, 무결성, 실시간성, 고성능 대역폭 등의 지원을 위한 '소프트웨어 플랫폼'이 필요하다.[33] 특히, CPS 기술 중 미래 예측, 협동 및 합의 의도 파악, 머신 러닝 등 지능적 기술은 순수한 소프트웨어 기술로 연구가 필요하다.[34]

즉, CPS는 지능형 플랫폼, 클라우드 서비스, 시뮬레이션 플랫폼, 상호운용성 플랫폼 등을 포함하는 서비스를 제공하며, 이를 위해서는 임베디드 시스템, 시뮬레이션, 클라우드 서비스, 센서 네트워크 등 기존의 기술을 활용해야 한다.[35] CPS 구축을 위해서는 피드백이 일어나는 이벤트를 가진 시간 관련 시스템 제어, 물리·화학·안전·실시간·에너지 제한·견고성 등의 물리적 법칙을 소프트웨어에 적용, 정보와 거대한 네트워크를 관리할 수 있는 CPS 아키텍처요소들의 연구가 필요하다.[36]

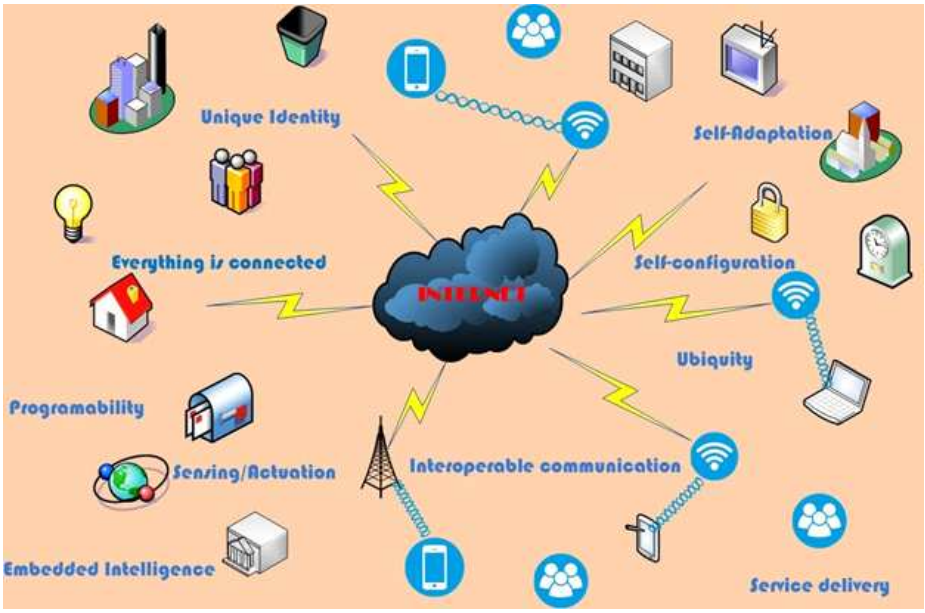
CPS는 다수의 센서 및 작동장치(Actuator)들이 사물인터넷(Internet of Things, 이하 IoT)으로 연결되어 구성되는 시스템 위의 시스템 (SoS, System of system) 으로, 개별 전자·기계 장치의 조작 정도로 적용범위가 한정되어 있는 임베디드 시스템들과 비교하여 규모와 복잡도가 매우 클 뿐 아니라 물리 세계와 밀접하게 상호작용하는 시스템으로서 차별화된다.[37] CPS는 IoT 기술과 많은 부분이 연계되어 있지만 IEEE(2015)[38]에 의하면 아래와 같은 차이점이 있다. CPS는 물리 시스템에서 수집된 정보의 공유를 넘어 효율성을 높이고 특정 목표를 달성하는 시스템 제어 기술로 볼 수 있고, IoT는 통신 관점에서 수많은 객체를 연결하는

32) Jay Lee 외(2014), A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems, 2014.12.
33) 박종만(2015), 중소기업 스마트공장 기술 동향과 이슈, The Journal of Korean Institute of Communications and Information Sciences, Vol.40 No.12, 2015.12.
34) acatech(2015), Integrated research agenda Cyber physical systems, CPS 기술은 센서 융합, 패턴 인식의 물리적 감지, 인공지능 등으로 미래 예측 가능한 자동화 기술, 인공지능을 통한 협동 및 합의, 의도 파악과 인간 모델링을 통한 인간과 기계의 상호작용, 머신 러닝 기술, 2015.3.
35) Jesús Angel García Sánchez, Jorge J. Rodríguez Vázquez(2014), Open CPS Platforms, CPS20: CPS 20 years from now visions and challenges workshop, 2014.4.
36) Rajkumar et al(2016), Cyber-Physical Systems: The Next Computing Revolution, Design Automation Conference (DAC), 47th ACM/IEEE, 2010.6.
37) 원명규, 장덕우, 장정훈 외(2014), 사이버물리시스템 기반의 스마트 시티 기술, 한국통신학회지, 31(8), 2014.7.,pp45-53
38) IEEE(2015), Towards a definition of the Internet of Things, 2015.05.

기술이라고 볼 수 있다.

CPS와 유사 개념인 IoT(Internet of Things, 사물인터넷)은 각종 사물에 센서와 통신기능을 내장하여 모든 사물들이 인터넷을 통해 연결되고 서로 정보를 주고받을 수 있게 해주는 기술 및 서비스를 의미한다.[39] IoT시스템은 고유한 글로벌 식별자를 사용하여 단일 "사물"을 식별하고 언제 어디서나 액세스 할 수 있는 시스템이다.[40] IoT는 이미 존재하거나 향후 등장할 상호 운용 가능한 정보 기술과 통신 기술을 활용하여 다양한 실재 및 가상 사물 간의 상호 연결을 통해서, 진보된 서비스를 제공 수 있게 하는 글로벌 인프라스트럭쳐이다.(ITU-T의 Y.2060) IoT는 사람, 사물, 공간, 데이터 등 모든 것이 인터넷으로 서로 연결되어, 정보가 생성・수집・공유・활용되는 초연결 인터넷이다.[41]

[그림 2-3] IoT 시스템의 특징과 범위



자료 : IEEE(2015), Towards a definition of the Internet of Things (IoT)

IoT는 인간과 사물, 서비스 세 가지 분산된 환경 요소에 대해 인간의 명시적 개입 없이 상호 협력적으로 센싱, 네트워킹, 정보 처리 등 지능적 관계를 형성하는 사물 공간 연결망이라고도 정의하며, 센싱 기술, 유무선 통신 및 네트워크 인프라 기술, IoT 서비스 인터페이스 기술이 필요하다.[42] 또한 표준화된 통신 프로토콜, 이기종 IoT 연결을 위한 미들웨어를 이용하

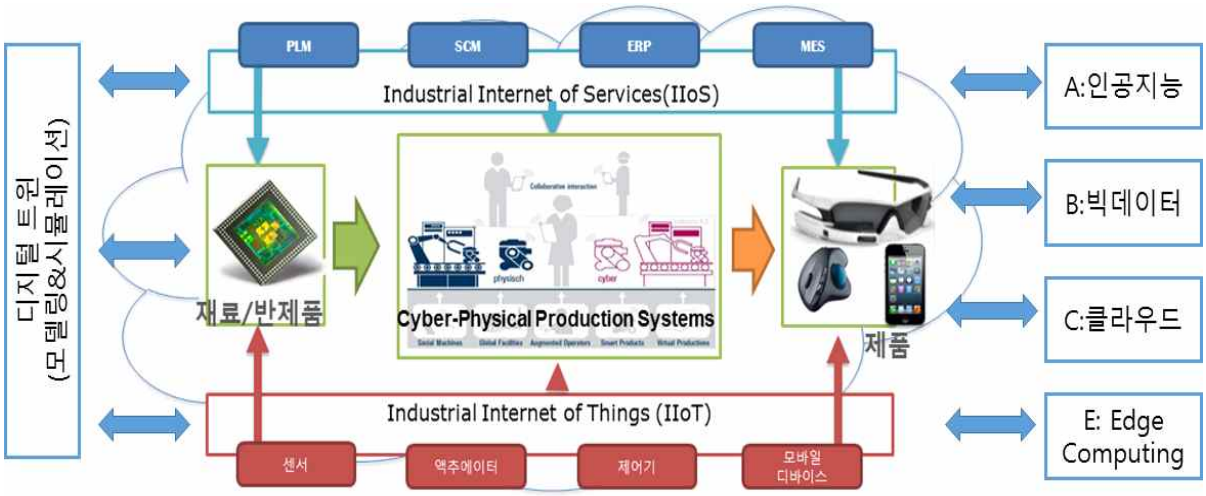
39) 위키피디아
40) IEEE(2015), Towards a definition of the Internet of Things
41) 미래창조과학부(2017), 제품서비스 개발을 위한 IoT와 oneM2M의 이해
42) 민경석(2013), 사물인터넷(Internet of Things), 한국인터넷진흥원

여 사물을 연결하고 빅데이터를 모으는 기술이 필요하다. 제조분야에서는 IoT 기술 중 저전력 네트워킹, 임베이드 운영체계, 센서 데이터 최적화 기술 등이 핵심 기술[43]이라고 한다.

IoT 시스템은 복잡한 "서비스"를 제공하고 복잡한 프로세스의 실행을 지원하기 위해 많은 양의 정보가 상호 연결되면 복잡성의 수준이 커진다. IoT에서는 사물들 간의 통신이 가장 기본적인 기능이나 현재 개발된 많은 IoT 서비스들의 경우, 동일한 제조사의 디바이스 간 서비스나 이해당사자 중심으로 IoT 생태계를 구축하는 현상이 뚜렷하다. 그러므로 서로 다른 제조사로부터의 사물들, 다른 여러 서비스 영역간 진정한 개방형 IoT 서비스가 이루어지기 위해서는 상호운용성 확보를 위한 국제 표준의 준수가 필요하다.

독일의 인더스트리 4.0 자료나 이를 수정하여 만든 스마트공장추진단의 스마트공장 개념도에는 다음과 같이 CPS와 IoT 관계를 보여주고 있다. 다양한 센서, 작동장치(Actuator), 제어기, 각종 모바일 디바이스로부터 데이터를 수집하는 IIoT(Industrial of IoT) 계층은 상위 시스템과 서비스계층으로 제어 데이터를 전달하고, 지능형 제어 계층인 CPPS(Cyber Physical Production Systems)는 IIoT 계층과 IIoS(Industrial Internet of Services) 계층에서 수집된 데이터를 기반으로 모델링과 시뮬레이션을 통해 최적의 공정을 실시간 설계하고 공정을 재구성하여, 고객의 다양한 생산요구 및 시스템 오류에 대응한다.[44]

[그림 2-4] 스마트공장 개념도



자료 : 스마트공장 사업소개(2015.12.), 스마트공장추진단, 2017년 산업기술 R&BD 전략에서 그림 수정

43) 박종만(2015), 중소기업 스마트공장 기술 동향과 이슈, The Journal of Korean Institute of Communications and Information Sciences, Vol.40 No.12, IoT는 센서, 프로세서, 통신 능력을 가진 지능자산, 데이터통신 구조, 데이터 분석 및 응용, 의사결정으로 구성, 2015.12.

44) 진회승(2017), 스마트공장 성공을 위한 소프트웨어 역할과 과제, SPRI 이슈리포트

그러나 NIST에서는 CPS와 IoT가 종종 같은 의미로 사용되기 때문에 중첩되는 부분이 있어서 NIST의 CPS 프레임워크에 설명된 접근법을 IoT에도 동일하게 적용할 수 있다고 했다. 이는 본 연구자가 기존의 IoT 플랫폼과 CPS 플랫폼의 기능을 비교하고자 하는 근거가 된다. 현재 개발되고 사용하는 수많은 IoT 플랫폼이 존재하며, 이 중 하나를 선택하여, IoT 플랫폼이 CPS의 기능을 어느 정도 지원가능한지 확인해보고자 한다.

2. CPS 연구 필요성

CPS는 물리 세계에서 정보를 수집하고 조작하여 산업 및 생활방식을 편리하고 효율적으로 바꾸어 국가 산업 기반의 변화를 가져올 수 있는 제4차 산업혁명에서 강조되는 기술의 하나이며, 미국, 유럽 등 기술 선진국에서는 국가적으로 필요성을 인식하여 장기적으로 투자하는 시스템이고, 산업적으로는 자동차, 항공, 에너지, 제조 등 전통산업에 소프트웨어 기술이 결합되어 새로운 서비스를 창출하는 융·복합 산업에 적용 가능하여 7%이상의 연평균 성장률을 예상하는 유망한 분야이다.

CPS 기술은 3C (컴퓨팅/Computing, 통신/Communication, 제어/control) 기술들의 고도화 영향으로 기존 산업용 물리시스템 (예, 전통 제조, 자동차, 항공)이 새로운 지능형 물리시스템 (예, 스마트공장, 자율자동차, 무인항공기)으로 진화하면서 산업계에 필수적으로 요구되는 기술이다.

미국, 유럽 등은 CPS를 차세대 핵심기술로 인식하고 관련 기술 개발을 위해 총력을 기울이고 있다. 미국은 오바마 대통령이 2013년 2월 혁신 프로젝트에 CPS를 포함, 이를 기반으로 제조, 운송 등 산업분야에서 스마트시스템 구축을 통해 경제 개발 및 일자리 창출 기대하고 있다. 독일은 인더스트리 4.0의 주요 핵심 기술인 CPS, 특히 모델링 및 시뮬레이션 기술을 통해 지능형 스마트 공장 구현으로 생산성 증가 등 신 부가가치 창출을 기대하고 있다. EU는 2007년부터 CPS 및 임베디드 시스템에 Frame Programme 7에서 70억 달러를 투자했고, 후속 프로젝트인 Frame Programme 8(HORIZON 2020)을 통해 연구를 수행하고 있다.

CPS 시장은 세계 각국의 활성화 대책 수립 · 시행과 주요기업의 참여로 관련 시장의 급격한 성장 예상되며, 전 세계 CPS 시장은 연평균 7.2% 성장하여 2020년에는 시장규모가 약 2.3조 달러에 달할 것으로 전망[45]되고 있다. 반면 전통적인 임베디드 시장은 연평균 0.7% 성장할 것으로 예측되어 CPS 시장이 지속적인 성장을 통해 전통적 임베디드 시스템 시장

45) IDC(2016), Worldwide Embedded and Intelligent Systems Forecast, 2016-2020, 2016.05.

규모를 빠르게 넘어설 것으로 보인다.

<표 2-1> 전통적인 임베디드 시스템과 CPS 시장 비교

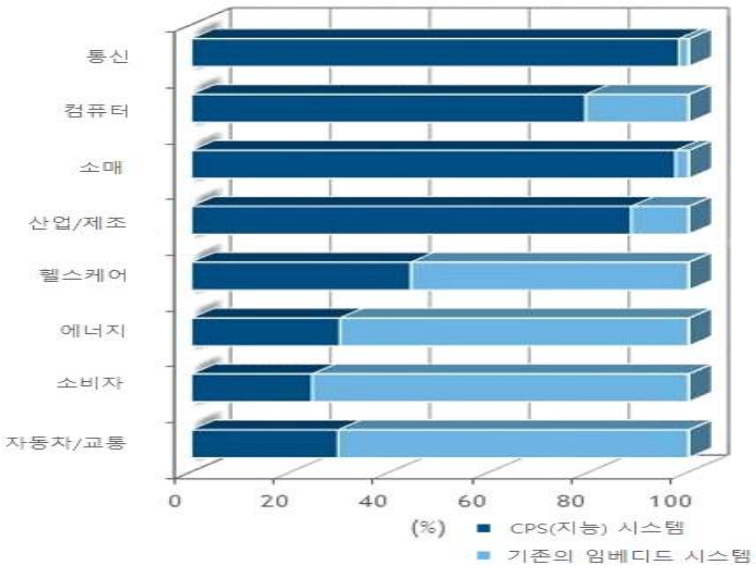
(단위: 십억불)

구분	2015	2016	2017	2018	2019	2020	CAGR (%)
전통적 임베디드 시스템	1,748.6	1,719.4	1,731.6	1,796.9	1,815.4	1,809.6	0.7
CPS (지능형 시스템)	1,615.3	1,727.0	1,890.7	2,032.5	2,167.6	2,290.8	7.2
합계	3,363.9	3,446.4	3,622.3	3,829.4	3,983.0	4,100.4	4.0

자료: IDC(2016), Worldwide Embedded and Intelligent Systems Forecast, 2016-2020

산업별 구분측면에서는 통신과 소매 그리고 산업/제조 분야에서 기존의 임베디드 시스템보다 지능 시스템인 CPS가 차지하는 시장 규모가 더 크다는 것을 확인 할 수 있으며 헬스케어, 에너지, 자동차/교통 분야의 경우, 아직까지 기존 임베디드 시스템의 시장 규모가 60% 이상을 차지하고 있음을 알 수 있다.

[그림 2-5] 산업별 임베디드 시스템/CPS 시장 규모 비교



자료: IDC(2016), Worldwide Embedded and Intelligent Systems Forecast, 2016-2020

앞으로 2020년까지 웨어러블, 자동차 분야, 드론, 에너지, 스마트 홈, 스마트 빌딩 영역에서 높은 연간 성장률을 보일 것으로 예측되며 시장 규모가 커질 것으로 전망되고 있다.

<표 2-2> 분야별 CPS 시장 전망

(단위: 백만 불)

구분	주요 시장 부문	2015	2020	CAGR (%)
웨어러블	스마트 웨어러블	2,181	43,837	82.2
ADAS	컨트롤러, V2X	956	8,100	53.3
자동차 엔터테인먼트	인포테인먼트 시스템	7,562	31,328	32.9
드론	드론	5,760	22,213	31
상용 자동차 텔레메틱스	드라이버 관리	414	1,560	30.4
게이트웨이	IoT 게이트웨이	1,019	2,805	22.4
에너지	인버터 제어	522	1,415	22.1
스마트홈	게이트웨이, 보안, 에너지 관리	3,532	8,242	18.5
스마트빌딩	빌딩 관리, 에너지 관리, 조명 컨트롤	25,181	57,146	17.8
기타	디지털 TV, 스마트폰, 태블릿, 서버 등	1,568,185	2,114,199	5.7
합계		1,615,312	2,290,839	7.2

자료 : IDC, Worldwide Embedded and Intelligent Systems Forecast, 2016-2020 에서 편집

3. CPS 응용 도메인

CPS 기술은 컴퓨팅・통신・제어 기술의 발전을 기반으로 다양한 융합 IT 분야로 확장 가능한 기술이며, 스마트 무인기, 스마트 국방, 스마트 그리드, 스마트 공장, 스마트 헬스케어 등 고신뢰성이 요구되는 도메인에 사용된다.

[그림 2-6] CPS 기술 적용 도메인

자료: 김원태(2015), 하이브리드 모델기반 고신뢰 CPS(Cyber-Physical Systems) SW 개발 환경, 2015.12

CPS 적용 도메인별 규모와 기능은 다음과 같다.

<표 2-3> CPS 적용 도메인 및 기능

도메인	규모/기능
스마트 제조	Medium Scale; 생산이나 물류에서 생산성 향상
응급 구조	Medium/Large Scale; 공공안전, 재난, 기반시설의 위험 관리
항공	Large Scale;항공 시스템의 운영 관리
기반시설	Large Scale;물, 전기, 가스 ,오일 등 생성, 분배 관리
건강	Medium Scale;환자의 건강상태를 확인하고 의료 처치
스마트 운송	Medium/Large Scale;실시간으로 교통상태 관리 및 안전 향상

자료: Volkan Gunes et all(2014), A Survey on Concepts, Applications, and Challenges in Cyber-Physical Systems, KSII TRANSACTIONS ON INTERNET AND INFORMATION SYSTEMS, 2014.12.

CPS는 전통산업, 에너지, 교통, 안전, 국방 분야에 다음과 같이 적용하여 서비스화 할 수 있다.

<표 2-4> CPS 적용 예

적용 분야	서비스	내 용
전통산업	스마트 공장 차세대 생산 기반 기술	・CPS의 모델&시뮬레이션, 컴퓨팅 플랫폼을 이용한 기존 생산 프로세스의 획기적 변화와 생산성 향상을 이룰 수 있는 제품 생산프로세스 기반 기술
에너지 생산 및 송전	스마트 그리드, 스마트빌딩	・전력망 수급 균형 유지 및 에너지 효율이 최적화된 스마트 그리드 구성 단말 시스템 및 체계 개발 ・Zero 에너지 빌딩 구축
의료 및 헬스케어	스마트 헬스케어	・네트워킹 기능 탑재 고신뢰 의료기기, 의료로봇 ・노인, 장애인 등을 위한 낙상사고 보호 등 헬스케어 서비스
운송 시스템	스마트 교통	・스마트 교통을 구성하는 요소들 사이의 신뢰성 있는 데이터 서비스 제공 ・교통상황의 데이터를 이용하여 교통사고 사망자・부상자 최소화, 교통 정체 최소화 시스템
안전	소방・방재 서비스	・각 응급 상황별 모델 생성 및 제어 모델을 개발 ・고신뢰 통신미들웨어를 이용한 실제 소방・방재 장비와의 연동과 이를 이용한 고신뢰 고지능형 소방・방재 서비스
국방	전투체계 시스템	・모델&시뮬레이션을 이용한 체계적이고 고성능의 전투 체계 시스템 ・네트워크로 연결된 자율형 로봇 기반의 국방 시스템

자료: 김원태 외(2010), CPS 기술동향, 정보통신산업진흥원 주간기술동향 통권 1455호, 2010.7.21.
원명규 외(2013), 사이버물리시스템의 현재와 미래: 응용 어플리케이션 관점에서의 접근, 한국통신학회지(정보와 통신) 30(10), 2013.9. pp62-69에서 편집

4. CPS 발전 단계

현재 CPS는 미성숙한 기술로 미국과 유럽 등 기술 선진국에서도 연구 및 적용이 진행 중에 있다. 이에 따라 CPS 성숙도에 따른 CPS의 단계를 나누어 CPS 연구 및 구축을 수행하는 것이 CPS의 실질적인 연구에 한 가지 방법이다. 다음은 자동차, 스마트공장, 스마트도시의 수준별 단계에 대해 정리하였다.

미국 도로교통안전국은 자율주행차에 대한 수준별 단계를 정의하고 있다. 자동차의 자율주행 수준에 따라 단계를 정의하고 운전자의 책임 소재를 규정하고 있다.

<표 2-5> 자율주행차 단계 구분

단 계	구 분	범위/기능
Level 0	비자동 (No Automation)	운전자가 항상 브레이크, 속도조절, 조향 등 차량 제어와 도로 모니터링 등 안전 조작에 책임
Level 1	자동 기능 (Function-specific Automation)	자동차가 각 기능이 독립적으로 작동하는 차량 제어에 대한 제한된 권한을 가지고 주행 예) 크루즈컨트롤, 자동브레이킹
Level 2	자동 조합 기능 (Combined Function Automation)	주행 환경에서 두 개 이상의 제어 기능이 조화롭게 작동. 운전자는 모니터링 및 안전에 책임을 지고 자동차 제어권을 소유하고 차량 제어 가능 예) 스마트크루즈컨트롤과 차선중앙유지
Level 3	제한된 자율주행 (Limited Self-Driving Automation)	특정 교통 환경에서 자동차가 모든 안전 기능을 제어하며 자동차가 모니터링 권한을 갖되 운전자가 제어가 필요한 경우는 경보신호 제공
Level 4	완전 자율주행 (Full Self-Driving Automation)	자동차가 모든 안전 기능을 제어하고 도로 상태를 모니터링하며, 운전자는 목적지 혹은 운행을 입력

자료: 미교통국 도로교통안전국 (NHTSA, National Highway Traffic Safety Administration)

또한 스마트공장 추진단은 스마트공장에 대한 수준별 단계를 다음과 같이 정의하고 있다. 데이터 수집방식과 제어 방식에 따른 자동화 단계와 시스템 자동화에 따른 공장 운영 방식에 따라 단계를 나누고 있다.

<표 2-6> 스마트공장 단계 구분

단 계	자동화 단계	공장운영 단계
고도화	제어자동화 및 디지털식별이 결합된 IoT 이용 자동화	CPS, IoT, 빅데이터를 이용한 자가진단과 제어능력을 갖춘 지능형 생산
중간수준2	설비 제어 자동화	실시간 의사결정 및 설비 직접 제어
중간수준1	설비로부터 실시간 데이터 수집	설비로부터 집계된 실적 중심의 공장운영 분석
기초수준	바코드, RFID를 이용하여 기초적 물류정보 수집 수준	공정물류 중심의 실적관리 수준
ICT 미적용	엑셀 활용 정도	시스템을 갖추고 있지 못한 상태

자료: 스마트공장 추진단 홈페이지

스마티도시의 경우는 다음의 단계를 거쳐 발전하였다.

<표 2-7> 스마트도시 발전 단계

단 계	시 기	내 용
1 단계	1996년 ~ 2002년	온라인상의 부분적인 도시 정보화
2 단계	2003년 ~ 2011년	가상과 현실공간을 융합하는 도시 정보화
3 단계	2012년 이후	플랫폼, 데이터 분석 등 기술을 이용한 고도화 및 융합화

자료: 한국정보화진흥원(2016), 스마트시티 발전전망과 한국의 경쟁력, 2016.11.07.

또한 구조적 관점에서 스마트도시 발전 단계를 다음과 같이 구분할 수도 있다.

<표 2-8> 스마트도시 단계 구분

단계	구 분	내 용
1	기반구축	스마트시티의 본격적 구축을 위한 기반이 조성되는 단계로 도시혁신과 스마트시티 인프라(도시 인프라, ICT 인프라, 공간정보 인프라)를 성숙시키기 위해 ICT와 건설 산업이 주도
2	수직적 구축	개별 분야와 서비스별로 수직적 연계·통합을 이루어 도시 운영 효율성을 높이는 단계로 IoT 계층 구축
3	수평적 구축	연관되는 기능과 업무들이 데이터와 플랫폼을 공유하여 더욱 고도화된 분석과 서비스를 제공하는 단계로 데이터 공유가 중요
4	도시 플랫폼	도시가 하나의 플랫폼처럼 기능하는 단계
5	미래도시	스마트시티를 넘어 본격적인 지능사회로 진화하는 단계

자료: 한국정보화진흥원(2016), 스마트시티 발전전망과 한국의 경쟁력, 2016.11.07.

자율주행자동차, 스마타공장, 스마트도시 등의 단계를 분석하여 CPS 단계를 다음과 같이 구분하였다. 첫 번째 단계는 개별 시스템 구축하고, 다음단계는 구축된 정보를 기반으로 의사결정 지원한다. 다음 단계는 결정된 사항을 기반으로 물리시스템을 제한적으로 제어하며, 그 다음 단계는 시스템을 전체적으로 제어한다.

<표 2-9> CPS 단계 구분

단계	내 용	사 례
0	물리시스템과 사이버시스템 간 연동 없음 물리시스템과 사이버시스템 독립적으로 구축	사람의 입력에 따른 길찾기 정보 제공
1	수집된 정보를 기준으로 의사결정 가능	네비게이션 (GPS에 의한 실시간 위치 정보 제공)
2	결정된 사항을 기반으로 일부 물리시스템 제어	단일기능 자율주행 자동차 일부 공장자동화
3	단일 시스템 완전 제어	자율주행차, 공장자동차
4	전체 시스템을 연동하여 정보를 수집하고 의사결정하여 제어함 (집중화 되어나 분산된 데이터 저장소 존재하여 실시간 데이터 교환 가능)	ITS 시스템에 의해 제어되는 자율주행 자동차 각종 부품사와 완제품사가 연결된 완전 자동 공장

제2절 요구공학(Requirement Engineering)

CPS는 복잡하고 대규모의 시스템이기 때문에 요구사항 도출이 어렵기도 하지만, 여러 시스템이 융복합되어 있기 때문에 요구사항이 중복되고 충돌이 일어나 요구사항 명세서 작성이 더욱 어렵다. 충돌이 일어나는 요구사항을 해결하고, 중복된 요구사항을 제거하는 작업은 쉬운 작업이 아니며, 이 때문에 CPS 구현에서 있어서 요구공학은 다른 시스템에서보다 더 중요하다.

이 절에서는 요구사항을 정의하고 관리하는 요구공학 개요와 소프트웨어 공학 지식 체계인 SWEBOK(Software Engineering Body of Knowledge)[46]에 소개되어 있는 요구사항의 8가지 세부 분야를 통해 전반적인 요구사항 관련 사항을 살펴보고 소프트웨어 요구공학의 필요성에 대해 설명한다. 요구사항의 분류와 도출 기법에 대해 설명한다.

1. 요구공학 개요

1) 요구공학의 이해

요구공학(Requirement Engineering)이란 소프트웨어 개발 과정에서 요구사항을 정의, 문서화 및 유지 관리하는 프로세스를 의미한다. 요구공학 관련 ISO 표준인 "ISO/IEC/IEEE 29148:2011(E)" [47]에 의하면 요구공학은 시스템, 소프트웨어 또는 서비스가 충족되어야 하는 요구사항을 설정하고 유지·관리하기 위해 취득자(Acquirer, 공급 업체로부터 제품 또는 서비스를 입수하거나 조달하는 이해관계자)와 공급자의 영역 사이를 상호 중재하는 통합적(interdisciplinary) 기능으로 정의하고 있으며 요구사항을 발견, 도출, 개발, 분석, 검증 방법 결정, 검증, 의사소통, 문서화 및 관리하는 것을 포함한다.

SW공학백서 2017[48]에서는 "SW요구 공학은 요구사항을 구조적, 이론적, 논리적으로 접근하고 최종산출물에 보다 가깝게 구현할 수 있도록 지원하는 것"이라고 정의하고 있다. 즉, 요구공학은 요구사항의 추출, 분석, 명세, 검증/확인 및 관리 등을 개발과정 동안 수행하며 소프트웨어 요구사항 명세서를 최종산출물로 도출한다. SW개발과정에서 요구공학 활동을 통해 관련된 이해관계자들이 요구사항을 명확히 이해하고 검증하여 사전에 문제를 걸러내면 불필요한 비용과 시간을 절감할 수 있다.

---

46) P. Bourque et all.(2014), Guide to the Software Engineering Body of Knowledge, Version 3.0, IEEE Computer Society, 2014.

47) ISO/IEC/IEEE 29148:2011(E)은 IEEE 표준으로 라이프사이클 동안 시스템 및 소프트웨어 제품 및 서비스에 대한 요구공학과 관련된 프로세스 및 제품에 대한 조항을 포함

48) 김태열 외(2017), 소프트웨어 공학백서, 정보통신산업진흥원

여기서 요구사항이란 요구(need)와 관련 제약 조건을 해석하거나 표현하는 것[49]으로 소프트웨어 요구사항은 소프트웨어를 통해 어떤 기능을 원하는지, 기능을 수행하는데 있어 제약사항이 무엇인지를 표현하는 것이다. SW공학백서 2017에서 "요구사항이란 이용자가 어떤 문제를 해결하거나 목표를 달성하기 위해 필요로 하는 조건이나 능력을 의미하며, 계약을 수행하거나 표준에 맞추거나 만족스러운 산출물을 생산하기 위해 시스템의 전체 혹은 일부가 갖추어야 하는 조건이나 능력, 요구의 총체를 의미한다." 고 서술하고 있다.

요구공학(Requirements Engineering)이란 용어는 1977년 M.W. Alford[50]에 의해 처음 사용된 것으로 추정되나 이후 일반적으로 사용되지는 않았다가 1990년대에 요구공학 분야에 있어서 몇 가지 중요하고 급진적인 변화가 있었다. 1990년대 초반에 2년에 한번 씩 교차로 개최되는 IEEE 후원 국제 컨퍼런스와 심포지움의 출현과 함께 국제 학습지 발간[51]에 의해 요구공학은 자체적인 연구 분야로 부각되었다. 1990년대 후반까지, 이 분야는 여러 나라에서 소규모 회의와 워크샵을 추가로 지원할 만큼 충분히 성장했다. 그 중 하나는 1993년부터 지금까지 진행되고 있는 국제 요구공학 컨퍼런스(International Requirements Engineering Conference)[52]가 있다.

소프트웨어 요구공학에 대해서 소프트웨어 공학의 주요 기술 집합 중 가장 체계적으로 정립·활용되고 있는 모델인 SWEBOK(Software Engineering Body of Knowledge, 소프트웨어 공학 지식 체계) Guide V3.0[53]을 토대로 살펴보고자 한다. SWEBOK은 소프트웨어 요구사항, 설계, 구현, 시험, 품질, 전문가 등 총 15가지 지식영역으로 구성되어 있다.

SWEBOK에서 15가지 지식영역 중 첫 번째로 소개되는 소프트웨어 요구사항 부분은 SW 관련 이해관계자들의 요구를 파악하는 지식영역으로 요구사항 기본, 요구사항 프로세스, 도출, 분석, 명세, 검증, 실제적 고려사항, 소프트웨어 요구사항 도구의 8가지 세부 분야로 나뉘어 기술되어 있다. 8가지 분야에 대한 상세 설명은 아래와 같다.

---

49) ISO/IEC/IEEE 29148(2011), Systems and software engineering-Life cycle processes-Requirements engineering, 2011.

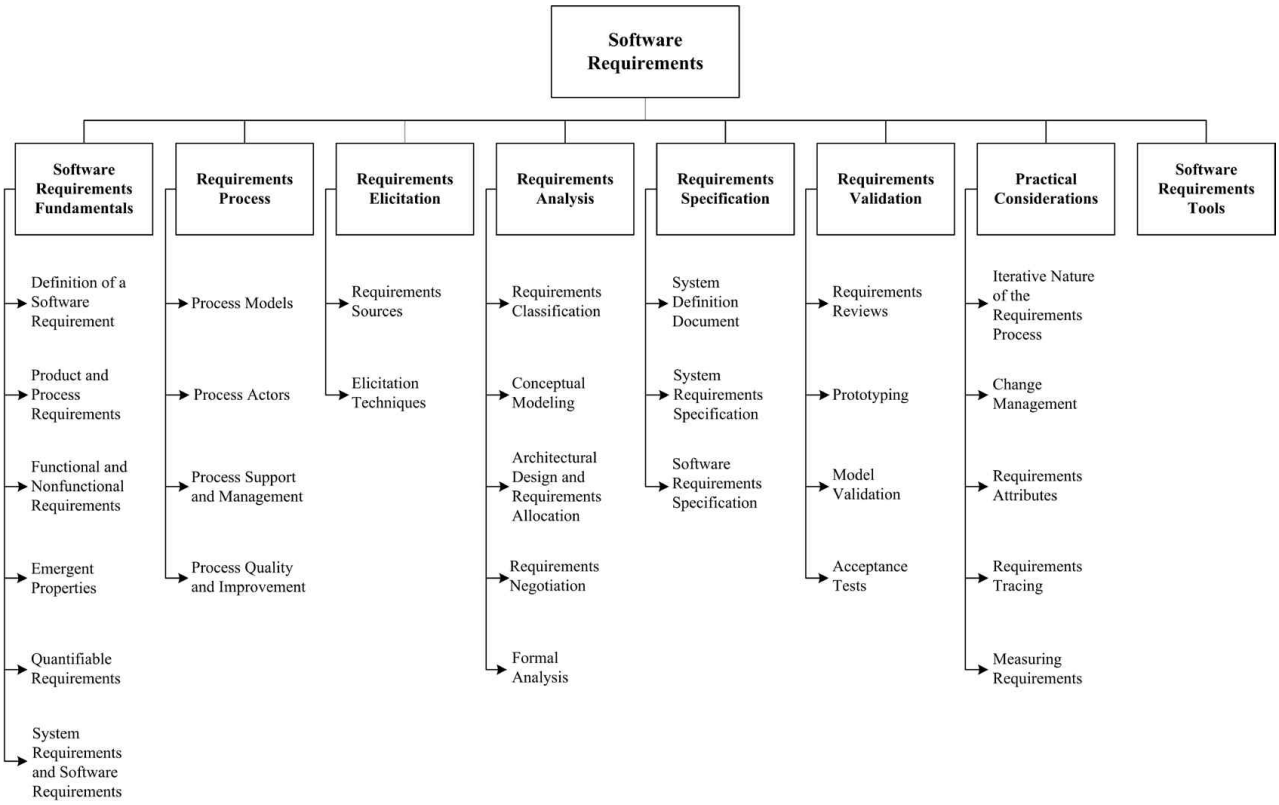
50) M. W. Alford(1977), A Requirements Engineering Methodology for Real-Time Processing Requirements, IEEE Transactions on Software Engineering, 1977.

51) Loucopoulos, P. & Potts, C. (Ed.)(1996), Requirements Engineering Journal. Springer Verlag, 1996.

52) International Requirements Engineering Conference는 1993년부터 시작된 요구공학에 관한 국제 심포지움으로 2017년 27회 컨퍼런스가 개최됨

53) P. Bourque, Richard Fairley(2014), Guide to the Software Engineering Body of Knowledge, Version 3.0, IEEE Computer Society, 2014.

[그림 2-7] 소프트웨어 요구사항 분야 관련 주제



자료 : Software Engineering Body of Knowledge, Version 3.0, IEEE Computer Society, 2014.

(1) 소프트웨어 요구사항 기본 (Software Requirements Fundamentals)

소프트웨어 요구사항 기본에 관련된 내용으로 요구사항의 정의와 제품 및 프로세스 요구사항, 기능, 비기능 요구사항의 차이 등에 대해 설명하고 있다. 요구사항이란 특정 문제를 해결하기 위해 소프트웨어 개발 또는 적용을 통해 외부로 드러나야 하는 속성으로 주어진 자원 내에서 반드시 검증 가능하여야(verifiable) 한다.

제품 요구사항은 개발될 소프트웨어 자체에 관한 것이며, 프로세스 요구사항은 소프트웨어 개발의 제약 조건(constraint)과 관계가 있다. 또한 기능 요구사항은 소프트웨어가 수행할 기능을 기술한 부분이며, 비기능 요구사항은 해당 솔루션을 제한하기 위해 행동하는 것으로 제약 사항 또는 품질 요구사항으로 불리기도 한다. 소프트웨어 요구사항은 정량적으로 최대한 명확하게 표현하여 모호함을 최대한 배제해야 한다.

(2) 요구사항 프로세스 (Requirements Process)

요구사항 프로세스의 경우에는 프로세스 모델과 담당자의 역할, 프로세스 지원 및 관리

방법, 품질 관리에 대해 설명한다. 요구사항 프로세스는 소프트웨어 개발 생명 주기(Software Development Life Cycle, 이후 SDLC)의 초기의 활동이 아니라, 프로젝트 초기에 시작되어 생명 주기 전반에 걸쳐 수정 및 관리 되어야 하는 활동이라고 볼 수 있다. 소프트웨어 요구사항을 형상 항목[54]으로 식별하고 SDLC 전반에서 타 산출물과 동일한 형상관리 활동을 통해 관리하는 활동으로 조직과 프로젝트 전반에 적용되어야 한다.

요구사항 개발 시는 사용자, 시장 분석가, 규제 관련자, 소프트웨어 개발자 등의 다양한 이해관계자가 포함된다. 먼저 이해관계자를 식별하고, 이해관계 및 상충관계를 분석하여 요구사항을 추출해야 한다. 프로세스의 품질 평가 및 개선활동은 비용절감과 소프트웨어 제품의 적시성, 고객 만족도를 높이기 위한 목적으로 필요한 활동이다.

(3) 요구사항 도출 (Requirements Elicitation)

요구사항 도출은 요구사항의 출처와 소프트웨어 개발자가 소프트웨어 요구사항을 수집하는 방법에 관련된 내용이다. 소프트웨어가 해결해야하는 문제에 대한 이해를 하는 첫 번째 단계로, 이 과정을 요구사항 획득이라 부르기도 한다. 올바른 요구사항 도출 프로세스의 기본 원칙 중 하나는 다양한 이해관계자 간의 효율적인 의사소통이라고 볼 수 있다. 또한 요구사항 추출의 핵심 요소는 프로젝트 범위를 알려주고 소프트웨어의 목적 및 우선순위에 따라 고객의 요구를 충족시키는지 확인하는 것이다.

요구사항 추출 과정에서는 소프트웨어에 영향을 주는 모든 잠재적 원인을 파악하는 것이 중요하다. 다양한 요구사항을 인지하고 관리하며 개발하려는 소프트웨어의 목표, 도메인 지식, 이해관계자, 운영 및 조직 환경을 파악하는 것이 그 핵심이라고 볼 수 있다. 도출 관련 기술은 인터뷰, 시나리오(예, use case), 프로토타입, 브레인스토밍과 같은 미팅을 진행하거나 비즈니스가 이루어지는 조직 환경의 관찰 등이 있다. 보다 자세한 내용은 아래 “요구사항의 도출 기법” 에서 설명한다.

(4) 요구사항 분석 (Requirements Analysis)

요구사항 분석을 통해 요구사항 간 상충 관계를 파악하고 해결하며 소프트웨어의 범위를 파악하고, 소프트웨어가 운영환경과의 상호작용을 어떻게 할 것인지 확인하는 과정이 필요

54) 형상 항목(Configuration Item)이란 소프트웨어의 변경사항을 체계적으로 관리하는 것을 말하는 형상관리(Configuration Management)의 대상으로 개별적인 요구사항 문서나 소프트웨어, 모델, 계획 등을 포함 (wikipedia)

하다. 먼저 요구사항 분석을 위해서는 요구사항의 분류가 필요하다. 요구사항은 기능/비기능, 하나 이상의 높은 수준의 요구 사항으로부터 파생여부/이해관계자 또는 다른 출처에 의해 소프트웨어에 직접 도출되는지 여부, 제품/프로세스 요구사항, 요구사항 우선순위, 범위, 요구사항 변경 가능성 등으로 구분할 수 있다.

실제 문제에 대한 모델 개발은 요구사항 분석의 핵심이라고 할 수 있다. 이는 문제에 대한 해결책의 제시뿐만 아니라 문제 이해에 중점을 두며 실제 관계 및 의존성을 반영한다. 모델의 종류는 유스케이스 다이어그램, 데이터 제어 흐름, 목표 기반 모델, 상태 모델, 사용자 상호작용, 객체 모델, 데이터 모델 등이 있다. 모델링 표기법 중 많은 대부분은 UML(Unified Modeling Language)에 속한다. 모델링 표기법의 선택은 개발하고자 하는 소프트웨어의 특성에 따라 달라질 수 있으며 소프트웨어 개발자가 경험해 본 표기법을 선택하는 것이 생산적일 수 있다. 만일 고객이 선호하거나 반대로 친숙하지 않은 표기법이 있는 경우, 이에 의해서 결정되기도 한다.

아키텍처 설계(Architectural Design)는 요구사항 프로세스와 소프트웨어 설계(Design) 단계가 겹치는 지점이며 두 작업을 완전히 분리하는 것은 매우 어렵다. 요구사항 분석 프로세스를 통해서 요구사항을 구체적인 컴포넌트에 지정하는 과정이 필요하다. 예를 들어 어떤 요구사항 집합이 A라는 컴포넌트에 지정되면, 개별 요구사항을 더 자세히 분석하여 해당 요구사항을 충족시키기 위해 다른 컴포넌트와 상호 작용해야하는 방식에 대한 추가 요구사항을 발견 할 수 있다. 대규모 프로젝트에서 이러한 지정작업은 각 하위 시스템에 대한 새로운 분석을 시작할 수 있게 한다.

요구사항 협상은 요구사항 분석과정에서 중요한 부분으로 “갈등(conflict) 해결”이라고도 할 수 있다. 서로 호환되지 않는 기능을 요구하는 이해관계자 간의 갈등, 기능과 비기능 요구사항의 충돌, 자원과 요구사항 사이의 조정이 필요한 경우가 발생하므로 이러한 선택에 대해 개발자 스스로 결정하기 보다는 이해관계자가 함께 타협을 이루어야 한다.

(5) 요구사항 명세 (Requirements Specification)

대부분 “명세(specification)”란 용어는 제품의 설계 목표에 대한 수치 또는 제한(limit)을 할당하는 것을 의미하나. 소프트웨어 공학 측면에서의 소프트웨어 요구사항 명세는 체계적으로 검토, 평가 및 승인 될 수 있는 문서를 제작하는 것을 뜻한다. 복잡한 시스템의 경우, 총 3가지 문서를 생성하는데 그 문서는 시스템 정의, 시스템 요구사항 및 소프트웨어 요구사항 문서이다.

먼저 시스템 정의서는 도메인 관점에서 요구사항을 정의하는 문서로, 사용자나 고객이 독자층에 포함되기 때문에 해당 도메인 언어로 기술되어야 한다. 그 내용에는 시스템의 전체 목표에 대한 배경 정보, 시스템 요구사항, 대상 환경, 제약 조건, 가정, 비기능 요구사항, 시스템 정보를 표현하기 위한 개념 모델과 사용자 시나리오, 데이터, 워크플로우 등이 포함된다. 시스템 요구사항 명세서는 여객기와 같이 규모가 상당히 큰 소프트웨어와 하드웨어 컴포넌트가 섞여 있는 시스템을 만드는 경우 작성하며, 이 경우 소프트웨어 요구사항 명세서는 시스템 요구사항으로부터 파생된다.

소프트웨어 요구사항 명세서(Software Requirements Specification, 이후 SRS)는 소프트웨어가 해야 할 일과 제약사항에 대해 고객과 계약자 또는 공급자 간 합의의 기반이 되며 소프트웨어 요구사항 정의 문서와 통합하여 작성하기도 한다. 소프트웨어 설계 전에 엄격한 요구사항 평가를 수행하여 재설계에 대한 위험을 제거할 수 있다. 조직은 효과적인 검증(Verification) 및 확인(Validation) 계획을 수립하기 위해 소프트웨어 요구사항 명세서를 사용할 수 있다. 소프트웨어 요구사항은 종종 자연어로 기술되기도 하나, SRS는 정형(formal) 또는 준정형(semiformal) 형식으로 표현되기도 한다. 적절한 표기법을 선택하면 소프트웨어 아키텍처의 요구사항을 자연어보다 정확하고 간결하게 설명할 수 있다.

(6) 요구사항 확인 (Requirements Validation)

요구사항 문서는 검증 및 확인 절차를 거칠 수 있다. 소프트웨어 개발자가 요구사항을 이해하고 있는지 확인하고, 요구사항이 기업의 표준을 준수하였는지, 이해가 가능하도록 작성되었는지 등을 확인하고 일관성과 완전성을 검증하는 과정은 매우 중요하다. 고객과 개발측 대표를 포함한 다양한 이해관계자가 문서를 검토해야 한다. 요구사항 검토(review)는 검토자가 그룹을 이루어 오류, 잘못 설정된 가정(assumption), 불명확성, 표준 준수 여부 등을 확인하는 것이다.

프로토타이핑은 타당성의 검증이나 성능 평가를 위해 미리 시험 삼아 만들어 보는 모형 제작방법으로 소프트웨어 요구사항에 대한 개발자의 해석을 검증하고 새로운 요구사항을 도출하기 위한 수단으로 활용 가능하다. 프로토타이핑은 요구사항 확인 외의 다양한 용도로 사용이 가능하며, 소프트웨어 개발자의 가정을 해석하고 피드백을 받기에 좋다. 하지만 프로토타입의 품질 문제로 인해 핵심적인 기본 기능에 주의를 집중하지 못할 위험을 포함하기도 한다.

요구사항의 핵심 속성은 최종 제품이 요구사항을 만족하는지 여부를 확인 가능하도록 하

는 것이다. 요구사항은 기본적으로 소프트웨어가 개발이 완료 되었을 때 요구조건을 만족하였는지 여부를 검증 가능해야 하며, 각 요구사항을 어떻게 검증할지를 계획하는 절차인 승인(acceptance) 테스트 케이스 설계는 매우 중요하다. 비기능 요구사항에 대한 승인 테스트 설계는 다소 어렵기 때문에 먼저 정량적으로 표현할 수 있는 지점까지 세분화하여 분석을 시작해야 한다.

(7) 실제적 고려사항 (Practical Considerations)

요구사항 프로세스는 소프트웨어 개발 생명주기 전체에 걸쳐 진행되며, 개발하고자 하는 소프트웨어의 요구사항의 변경관리와 유지보수는 소프트웨어 공학 프로세스의 성공을 위한 핵심 요소이다. 모든 조직이 요구사항을 문서화하고 관리하는 문화가 있는 것은 아니지만, 제품과 조직이 성장할수록 제품의 변경사항에 대한 영향을 평가하기 위해 문서화가 필요해 진다. 요구사항 문서화 및 변경관리가 요구사항 프로세스의 성공에 중요한 역할을 한다고 할 수 있다.

소프트웨어 산업은 점차 개발주기가 단축되는 압박이 나타나고 있으며 특히 경쟁이 치열하고 시장 주도적인 분야일수록 심해지는 경향을 보인다. 또한 대부분의 프로젝트는 환경의 제한을 받게 되고, 기존 소프트웨어의 유지보수성 작업이 많다. 따라서 요구사항 프로세스를 이해 관계자로부터 추출하여, 개발팀에 전달하는 선형적이고 결정론적인 프로세스로 요구사항 프로세스를 구현하는 것은 비현실적일 수 있다.

대신 요구사항 프로세스는 일반적으로 설계 및 조달 결정을 내릴 수 있는 충분한 수준의 품질 및 세부사항을 위해 반복된다. 대부분은 요구사항에 대한 이해는 설계와 개발이 진행되면서 이루어진다. 따라서 요구사항의 수정은 개발주기 중에 지속되며 요구사항이 변한다는 것은 요구공학의 이해에 대한 핵심적인 사항이다. 이는 분석 과정에서의 오류 때문이기도 하지만 주로 환경의 변화로 인해 발생한다. 그러므로 요구사항의 변화는 지속적으로 관리되어야 하며 이는 검토와 승인 과정을 통해 수행되어야 한다. 이를 위해 요구사항의 추적, 영향 분석 등이 요구된다. 여기서 요구사항의 추적은 요구사항이 변경될 때 영향 분석을 수행하기 위한 기본이라고 할 수 있다. 따라서 요구사항 프로세스는 소프트웨어 개발의 초기 작업이 아니라 전체 생명 주기에 걸쳐 있다고 할 수 있다.

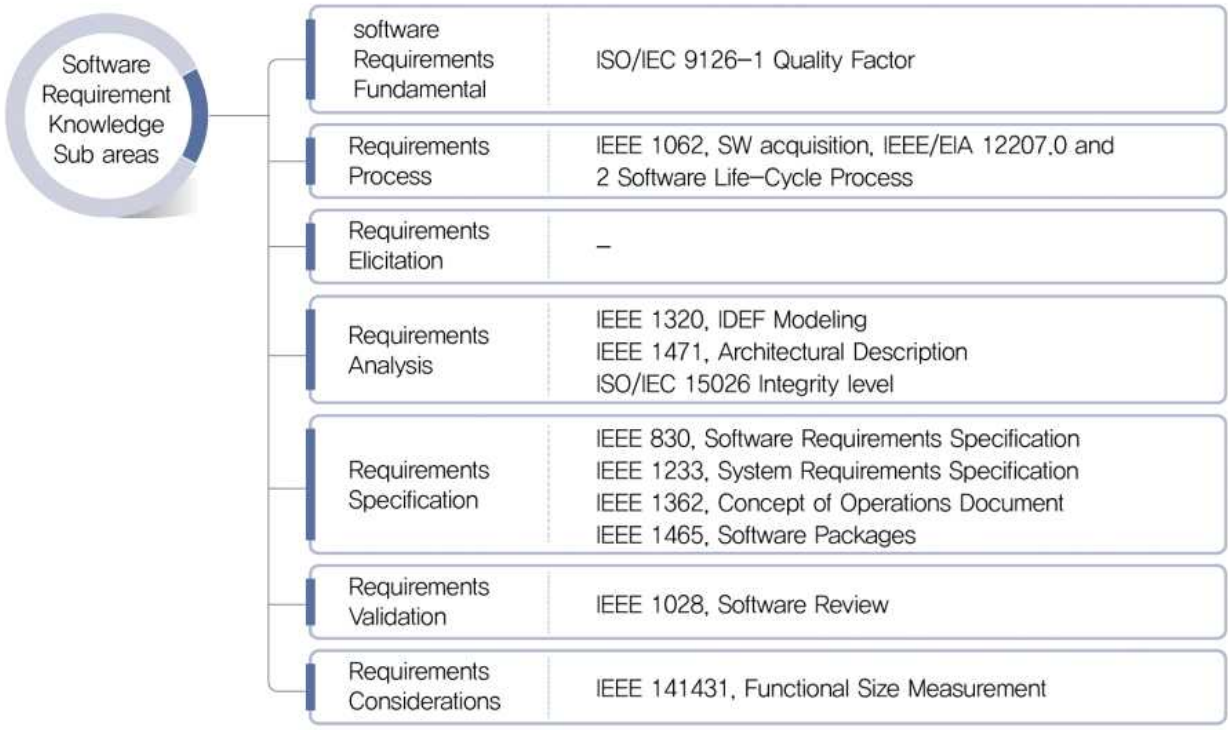
(8) 소프트웨어 요구사항 도구 (Software Requirements Tools)

소프트웨어 요구사항 관련 도구는 크게 모델링을 위한 도구와 요구사항을 관리하기 위한 도구라는 두 가지 범주로 나뉜다. 요구사항 관리 도구는 일반적으로 문서화/관리 문서, 추적

및 변경 사항을 폭넓게 지원한다. 실제로 추적과 변경 관리는 도구에 의해 지원되는 경우에만 실행 가능하므로 도구의 역할이 상당하다고 볼 수 있다.

다음은 위에서 설명한 SW 요구사항 관련 지식 체계와 관련된 국제 표준을 정리한 내용이다. 간단히 설명하자면, 요구사항 분석을 위한 국제 표준의 경우, 모델링과 관련한 "IEEE 1320.1 for functional modeling" 이 있고 아키텍처 관련은 "IEEE 1471-2000, Recommended Practice for Architectural Description of Software Intensive Systems" 이 있다. 요구사항 명세 관련 표준의 경우, IEEE1233-98이 시스템 요구사항을 다루고 있으며 소프트웨어 요구사항의 경우는 IEEE 1465(ISO/IEC 12119와 유사), IEEE 830 for the production and content of the SRS를 참고할 수 있다. 소프트웨어 요구사항 명세 분야인 IEEE 830에 대해서는 다음 절에서 보다 자세히 설명하겠다.

[그림 2-8] SW요구사항 관련 지식체계 및 국제표준



자료 : SW공학백서 2017, 정보통신산업진흥원

2) 요구공학의 필요성

요구공학은 소프트웨어가 만족해야 하는 기능, 조건 및 제약 조건을 찾아내고 분석을 통해 문서화하며 점검하는 프로세스로써 소프트웨어 공학의 토대라고 할 수 있다. 만일 고객

과 개발자가 요구사항의 의미를 서로 다르게 이해하는 경우, 소프트웨어 개발은 실패할 가능성이 높아질 수 있다. 프레더릭 브룩스(1987)[55]은 소프트웨어 개발에서 가장 어려운 부분은 무엇을 만들지 정확하게 결정하는 것으로 구체적인 요구사항을 수립하는 것은 매우 어렵고, 요구사항이 제대로 식별되지 않으면 프로그램이 제대로 동작하지 않을 수 있을 뿐만 아니라 나중에 바로잡기는 더 어렵다고 기술하였다. 이를 통해 개발 주기 동안 요구사항을 도출하고 분석, 검증 및 관리하는 요구공학의 중요성을 알 수 있다.

또한 기업 규모별 교육 필요 현황과 교육 제공 현황의 차이를 분석한 결과, 아래 그림과 같이 ① 프로세스 설계, ② 형상관리, ③ 요구공학 교육과정 순으로 교육 필요도가 높게 나타났다. 이를 통해 점차 복잡해지고 다양한 기술의 융합이 이루어지고 있는 소프트웨어 개발에 있어서 요구공학과 관련된 연구나 교육에 대한 기업의 필요성이 높다는 것을 알 수 있다.

<표 2-10> 기업 규모별 교육 수요 현황 및 교육 필요도

구분 (단위 : %)	의사 소통	비즈니스 도메인	프로세스	개발 언어	테스트	Tool	품질	개발 방법론	설계, DBMS	측정 및 분석	프로세스 설계	동료 검토	프로젝트 관리	형상 관리	요구 공학
대기업	91.9	88.9	90.9	76.8	86.9	89.9	90.9	87.9	70.7	86.9	62.6	80.8	56.6	44.4	39.4
중소기업	72.9	65.2	63.0	66.7	57.3	56.1	53.3	53	56.7	50.1	52.4	46.7	49.3	49.9	48.1
교육 필요도*	**0.5**	**4.0**	**20**	**2.7**	**16**	**3.1**	**4.4**	**14.6**	**12.4**	**15.6**	**22.9**	**18.5**	**20.2**	**21.7**	**20.5**

자료 : SW공학백서 2017, 정보통신산업진흥원, * 교육필요도 = 교육제공현황 - 교육수요현황

2. 요구사항의 구분

1) 사용자 요구사항과 시스템 요구사항

소프트웨어의 요구사항의 독자는 다양한 유형으로 구성되며 각자의 역할과 상황에 따라 요구사항에 대한 이해도는 달라질 수 있다. 사용자 요구사항은 보통 사용자 측면에서 이해할 수 있도록 시스템이 사용자에게 제공해야 할 서비스와 준수해야 할 제약사항에 대해 설명한 내용이다. 시스템 요구사항은 시스템의 기능과 서비스, 동작 시 제약사항에 대해 개발자의 입장에서 보다 더 상세히 서술한 요구사항이다. 보다 자세한 사용자 요구사항과 시스템 요구사항에 대한 설명은 아래와 같다.

55) Frederick P. Brooks(1986), No Silver Bullet: Essence and Accidents of Software Engineering

(1) 사용자 요구사항

사용자 요구사항은 소프트웨어가 가져야 할 속성으로 사용자에게 친숙한 표현과 쉬운 용어를 사용하여 충분히 이해할 수 있도록 작성되어야 한다. 사용자의 이해를 돕기 위해서는 사용자 요구가 반영된 기능과 제약 조건 등을 자연어, 다이어그램을 활용하여 표현하는 것이 도움이 된다. 다만 자연어를 사용하는 경우, 애매해지거나 심지어 다르게 해석하는 경우가 발생하므로 핵심적인 기능을 중심으로 단순하게 기술하는 것도 하나의 방법이다. 일반적으로 사용자 시나리오나 유스케이스, 사용자 스토리 등으로 표현될 수 있다. 사용자 요구사항을 작성하는 경우, 비슷한 프로젝트 경험을 가진 전문가를 선정하거나 회사의 표준 양식을 만들어 사용하는 것이 도움이 되며 요구사항을 수집할 때 해당 출처를 기재하여 중복이나 혼란을 막는 것도 효율적일 수 있다.

(2) 시스템 요구사항

시스템 요구사항은 개발자가 소프트웨어를 설계하는데 사용되는 요구사항으로 전문적이고 기술적인 표현을 사용한다. 완전하고 일관성 있는 작성이 중요하며 구조적인 방법론이나 언어, 유스케이스 다이어그램을 사용하여 표현하기도 한다. 위와 같이 두 가지로 요구사항을 구분하는 경우에는 사용자 요구사항으로부터 시스템 요구사항을 도출하면서 내용의 누락이나 변경이 일어나지는 않았는지 일관성 확인이 필요하다.

표 2-11은 정신 건강관리를 위한 환자정보 시스템의 요구사항에 대한 예시로써 사용자 요구사항의 정의와 시스템 요구사항의 차이점을 보여 준다. 시스템 요구사항 명세는 일반적인 형태로 작성된 사용자 요구사항에 비해 시스템이 제공해야 할 서비스에 대해 보다 구체적으로 표현하고 있음을 알 수 있다. 또한 하나의 사용자 요구사항은 여러 개의 시스템 요구사항으로 확장이 가능하다는 것을 확인할 수 있다.

<표 2-11> 사용자 및 시스템 요구사항 예시

사용자 요구사항 정의	시스템 요구사항 명세
1. Mentcare 시스템은 해당 월 동안 각각의 클리닉에서 처방한 약품 비용을 보여주는 월 관리 보고서를 매달 생성해야 한다.	1.1 처방한 약품에 대한 요약, 비용과 그 약을 처방한 클리닉을 매월 마지막 업무일에 생성해야 한다. 1.2 시스템은 해당 월의 마지막 업무일 17시 30분 이후에 그 보고서를 생성해야 한다. 1.3 각각의 클리닉에 대해 보고서가 작성되고, 개별 약품명과 전체 처방 회수, 처방한 복용량과 처방한 약품의 전체 가격의 리스트를 보여 줄 수 있어야 한다. 1.4 다른 용량 단위로 처방이 가능한 경우 (가령 10mg, 20mg 등)에는 각각의 용량 단위에 대한 별도의 보고서를 생성할 수 있어야 한다. 1.5. 관리 접근 제어 목록에 포함되어 있는 해당 권한이 있는 사용자에게만 약품 비용 보고서에 대한 접근을 허가해야 한다.

자료 : Ian Sommerville(2016), 소프트웨어 공학 제10판

2) 기능 요구사항과 비기능 요구사항

소프트웨어 시스템 요구사항은 크게 기능 요구사항과 비기능 요구사항으로 분류될 수 있다. 기능 요구사항은 시스템이 제공해야 하는 서비스와 특정 입력이나 동작에 대해 시스템의 동작 방식에 대한 것이다. 비기능 요구사항은 시스템이 제공하는 서비스나 기능에 대한 제약사항으로 시간적 제약이나 개발 프로세스 및 표준에 의한 제약 등이 포함된다. 그러나 실제로 기능과 비기능 요구사항의 차이는 위에서 설명한 정의처럼 명확하게 구분되기는 어렵다. 허가된 사용자에 대한 접근을 제한하는 보안과 관련된 사용자 요구사항은 비기능 요구사항으로 보일 수 있지만 사용자 인증에 대한 기능 요구사항이다. 이에 아래에서 좀 더 자세히 기능과 비기능 요구사항에 대해 살펴보고자 한다.

(1) 기능 요구사항

시스템의 기능 요구사항은 시스템이 무엇을 해야 하는지를 설명한다. 이 요구사항은 개발하려는 소프트웨어의 유형이나 예상 사용자, 요구사항 작성 시 조직의 접근 방식에 따라 달라진다. 사용자 요구사항으로써의 기능 요구사항은 주로 시스템 사용자가 이해할 수 있는 추상적인 방법으로 설명된다. 보다 구체적인 시스템의 기능 요구사항의 경우에는 시스템 기

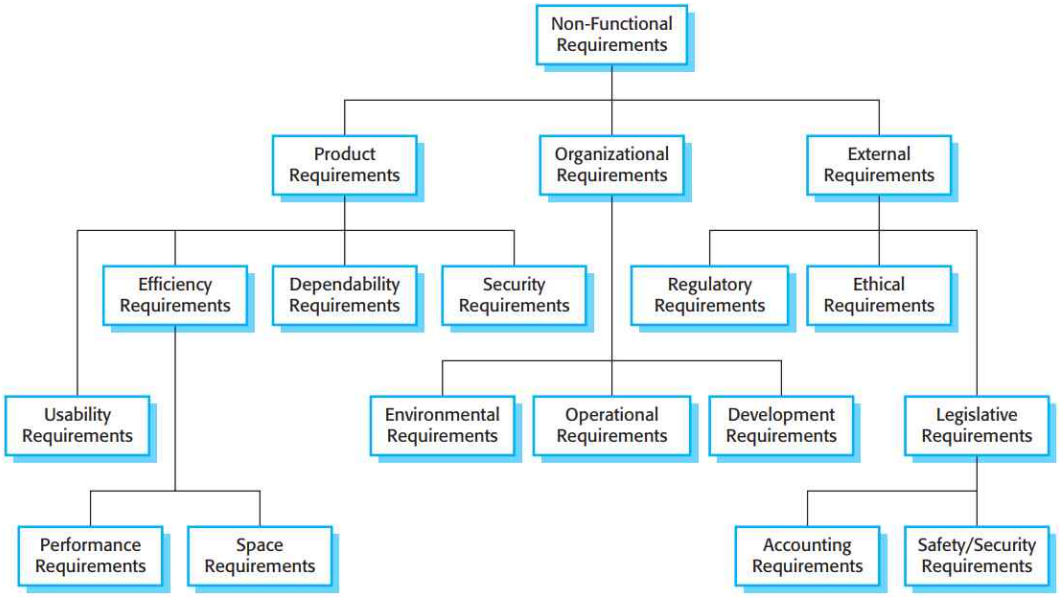
능, 입력 및 출력, 예외 사항 등을 상세히 설명한다. 시스템의 기능 요구사항은 시스템이 무엇을 해야 하는지를 설명하는 일반적인 요구사항부터 로컬 작업 방식이나 조직이 사용하던 기존 시스템을 반영하는 매우 구체적인 요구사항까지 다양하다. 또한 원칙적으로, 시스템의 기능적 요구사항 명세는 완전하고 일관적이어야 한다. 완전성은 사용자가 요구하는 모든 서비스가 정의되어야 한다는 것이고 일관성은 요구사항이 모순되는 정의를 가지면 안 된다는 것을 의미한다. 크고 복잡한 시스템의 경우, 요구사항의 일관성과 완전성을 얻기가 사실상 매우 어렵다. 요구사항 명세서를 작성할 때 실수와 누락이 발생하기도 하고, 대규모 시스템에는 많은 이해관계자들이 존재하며 이들이 서로 다른 요구사항을 가질 수 있으므로 일관성을 갖기가 어려워진다. 이때는 다수의 이해관계자 사이에서 상호간의 이해와 소통을 통한 정기적인 회의로 요구사항을 도출하고 도출된 요구사항의 확인(validation) 과정을 통해 각 요구사항을 어떻게 검증할지를 계획하여 미리 문제를 걸러내어 비용과 시간을 절감하는 것이 필요하다.

(2) 비기능 요구사항

비기능 요구사항은 시스템이 제공하는 특정 서비스와 직접적인 관련이 없는 요구사항으로 신뢰성, 응답 시간 및 점유율과 같은 시스템 특성이나 입출력 장치의 기능이나 다른 시스템과의 인터페이스에 사용되는 데이터 형식과 같은 시스템 구현에 대한 제약사항을 정의 할 수 있다. 성능, 보안 또는 가용성과 같은 비기능 요구사항은 주로 시스템 전체의 특성을 정하거나 제한하며 때에 따라 개별적인 기능 요구사항보다 중요할 수도 있다. 안전이 매우 중요한 항공기 시스템에서 신뢰성 요구사항을 충족시키지 못하면 안전 인증을 받을 수 없듯이 특정 비기능 요구사항을 충족시키지 못하면 전체 시스템을 사용하지 못하는 경우가 발생할 수 있다.

아래 그림은 비기능 요구사항의 유형을 보여주고 있으며 크게 제품 요구사항과 조직 요구사항, 외부 요구사항으로 나뉜다. 첫 번째인 제품 요구사항은 제품의 동작을 지정하거나 제한한다. 시스템의 실행 속도나 요구되는 메모리양과 같은 성능 요구사항과 허용 가능한 실패율, 보안요구사항이나 유용성 등 신뢰성 관련된 요구사항이 여기에 속한다. 두 번째는 광범위한 시스템 요구사항으로 볼 수 있는 조직 요구사항으로 개발 언어, 프로세스 표준, 운영 환경을 지정하는 요구사항이 포함된다. 세 번째는 시스템 외부요인과 개발 프로세스에서 파생된 요구사항으로 시스템이 지켜야 하는 법률 관련 규제 요구사항이 포함될 수 있으며 대중에게 수용될 수 있는 윤리적 요구사항도 포함된다.

[그림 2-9] 비기능 요구사항의 유형



자료 : Ian Sommerville(2011), Software Engineering 9th edition

3. 올바른 요구사항을 도출하기 위한 방법과 특징

올바른 요구사항을 도출하기 위해 활용 가능한 요구사항 도출 관련 기법을 살펴보고 요구사항 도출의 결과물인 소프트웨어 요구사항 명세서(Software Requirements Specification, 이후 SRS)의 특징에 대해 IEEE Std 830 1998 표준을 기반으로 설명하고자 한다.

1) 요구사항 도출 기법

칼 위거스(2017)[56]는 소프트웨어 프로젝트에서 요구사항을 도출하기 위해 다양한 기법을 소개한다. 도출 기법은 크게 이해관계자들과 상호작용을 촉진하는 활동과 자신에게 필요한 정보를 찾는 독립적 활동으로 나뉠 수 있다. 이해관계자들과 상호작용을 통한 촉진 활동의 경우, 사용자 요구사항을 도출하기 위한 좋은 방법이라면 독립적 도출 기법은 사용자가 미처 인지하지 못한 기능 등을 제시하여 요구사항의 도출을 보강할 수 있다. 대부분의 프로젝트에서는 두 활동을 조합하여 활용한다.

56) 칼 위거스, 조이비티(2017), Software Requirements 3, 모든 프로젝트 이해관계자가 알아야 할 요구공학의 정석과 실천법, 2017.

(1) 촉진 활동을 통한 요구사항 도출 기법

먼저 촉진 활동에는 인터뷰, 워크샵, 포커스 그룹, 관찰, 설문지 등의 기법이 존재한다. 첫 번째, 인터뷰 기법은 요구사항 도출의 가장 전통적인 방법으로, 사용자가 원하는 소프트웨어를 개발하기 위한 가장 좋은 방법은 사용할 사람에게 직접 묻는 것이다.

인터뷰 방식은 대규모 워크샵 방식보다 사용자가 더 편안하게 느낄 수 있으며 시간이 많지 않은 사용자에게 좋은 방법이 될 수 있다. 인터뷰는 목적에 맞는 자료를 미리 준비하는 것이 필요하고 적극적인 경청자세와 요구사항을 도출하기 위해 아이디어와 대안을 먼저 제시하는 것이 중요하다. 두 번째는 워크샵 방식으로, 워크샵에는 사용자 및 개발자, 검증자 등의 다양한 이해관계자가 참여한다. 워크샵 기법은 일정의 압박으로 인해 빠른 요구사항 도출이 필요한 경우나 이해관계자간 의견 충돌에 대한 해결이 필요한 경우에 유용하다. 워크샵을 개최할 때는 자료 초안을 미리 준비하고 명확한 계획과 시간 할당 및 인원 구성 등의 운영 규칙을 수립하여 시간 낭비를 피하고 집중도를 높일 수 있도록 해야 한다. 세 번째 기법은 포커스 그룹을 활용하는 것이다. 포커스 그룹이란 기능과 비기능 요구사항의 도출을 위해 모집된 사용자의 대표그룹이다. 이 기법은 상용화될 제품 개발이나 최종 사용자를 대면할 준비가 되지 않았을 경우 유용하다고 볼 수 있다. 네 번째는 관찰방식으로 사용자가 작업을 수행하는 방법을 관찰하는 것이다. 요구사항에 대해 사용자에게 직접 설명을 듣더라도 작업의 체화로 인해 세부적인 사항을 모두 설명하지 못하는 경우가 발생하게 되는데 이런 경우에 관찰 방식을 활용하면 효과적이다. 관찰은 사용자의 업무를 방해하지 않는 선인 2시간 정도로 시간을 제한하는 것이 좋다. 관찰의 방식은 2가지가 있으며, 사용자가 바쁜 경우에 수행하는 ‘조용한 관찰’과 사용자의 업무를 중단시키고 질문을 하는 ‘상호관찰 방식’이 있다. 이 때 추후 분석을 위해 문서화가 필요하며 사용자가 허용한다면 녹화를 할 수도 있다. 마지막은 설문지를 통한 기법으로 저렴하고 많은 사용자에게 정보를 도출할 수 있으며 지리적인 한계를 극복할 수 있다는 장점이 있다. 설문지를 작성하는 경우, 통계적으로 활용하고자 한다면 개방형 질문보다는 두 개 이상의 보기를 가진 폐쇄형 질문을 사용하여야 결과에 대한 공통점을 끌어낼 수 있다. 설문지 설계에 대한 전문가의 자문을 받는 것도 좋은 결과를 도출하는 방법이 될 수 있다.

(2) 독립적 요구사항 도출 기법

독립적인 요구사항 도출 기법으로는 시스템 인터페이스 및 사용자 인터페이스 분석, 그리고 문서 분석 기법이 있다. 먼저 시스템 인터페이스 분석은 개발하고자 하는 시스템과 연

결된 시스템에 대한 분석을 진행하는 것이다. 개발하려는 시스템과 연결된 시스템 중 요구사항으로 도출이 가능한 기능을 파악하는 것이 필요하며 이미 다른 시스템에서 개발되어 있어 구현할 필요가 없는 기능을 발견할 수도 있다. 다음으로 사용자 인터페이스 분석은 사용자 요구사항과 기능적 요구사항을 찾기 위해 기존 시스템을 분석하는 것으로 기존 시스템을 직접 사용하거나 불가능한 경우, 화면 캡처를 활용하거나 유사한 시스템을 사용해 볼 수 있다. 이를 통해 미처 생각하지 못한 잠재적인 기능을 찾아 낼 수 있으며 전체적인 서비스의 목록을 파악하는데 도움이 된다. 마지막으로 문서 분석 기법은 기존의 소프트웨어나 유사한 프로그램의 요구사항 명세서나 관련 문서, 사용 설명서 등에 대한 검토과정이다. 관련 문서를 미리 숙지하고 불필요한 기능이나 유지관리가 필요한 기능 등을 확인하여 이를 토대로 촉진활동의 시간을 단축할 수도 있다.

2) 올바른 요구사항 명세서의 특징

(1) 요구사항 명세서의 본질

소프트웨어 요구사항 명세서(SRS)는 특정 환경에서 특정 기능을 수행하는 소프트웨어나 프로그램에 대한 사양이다. SRS는 공급자나 고객, 또는 둘 모두에 의해 작성 될 수 있다. 요구사항 명세서의 작성자가 다루어야 하는 기본적인 문제는 소프트웨어가 무엇을 하기로 되어 있는지에 대한 기능성과 사람들과 상호작용 및 시스템의 하드웨어/소프트웨어와 어떻게 상호작용을 하는지에 대한 외부 인터페이스, 다양한 소프트웨어 기능의 속도, 가용성, 응답 시간, 복구 시간 등에 관련된 성능이 있다. 또한 휴대성, 정확성, 유지 관리성, 보안 등의 속성에 대한 것과 필요한 모든 표준의 유효성, 구현 언어, 데이터베이스 무결성, 리소스 제한, 운영 환경 등에 필요한 표준 규격이 있는지에 대한 설계 제약 조건이 있다. SRS 작성자는 설계 또는 프로젝트의 요구 사항을 SRS에 배치하지 않아야 한다.

(2) 올바른 요구사항 명세서의 8가지 특징

좋은 요구사항 명세서가 되려면 다음의 8가지 특징을 가져야 한다. 올바름과 모호하지 않음, 완료성과 일관성, 그리고 중요성 및 안정성에 대한 평가와 검증, 수정, 추적이 가능해야 한다는 것이다. 보다 자세한 내용은 아래 표 2-12에서 설명하였다.

<표 2-12> 올바른 요구사항 명세서의 8가지 특징

특성	상세 설명
올바름 (Correctness)	SRS는 소프트웨어에 명시된 모든 요구사항이 충족될 경우에만 정확함, 정확성을 보장하는 도구나 절차는 존재하지 않음, SRS는 시스템 요구사항 사양, 다른 프로젝트 문서 및 기타 적용 가능한 표준과 같은 적합한 사양과 비교하여 동의하는지 확인해야 함
모호하지 않음 (Unambiguous)	SRS는 문서에 명시된 모든 요구사항이 하나의 해석만을 갖는 경우에만 명확하다고 볼 수 있음, 모호성을 없애기 위해 하나의 고유 용어를 사용하여 최종 제품의 각 특성을 설명해야 한다. 특정 맥락에서 사용된 용어가 여러 의미를 가질 수 있는 경우, 그 용어의 보다 구체적인 내용을 어휘 사전에서 설명해야함, SRS는 문서를 만드는 사람들과 사용하는 사람들 모두에게 모호하지 않아야 함
완전함 (Completeness)	SRS가 전성을 갖추기 위해서는 기능, 성능, 설계 제약, 속성 또는 외부 인터페이스와 관련된 모든 중요한 요구사항, 특히 시스템 사양에 의해 부과된 외부 요구사항을 확인하고 처리해야 함, 가능한 모든 입력 데이터에 대한 소프트웨어의 응답에 대한 정의를 제공하고 유효한 입력 값과 유효하지 않은 입력 값에 대해 모두 응답 가능해야 함
일관성 있음 (Consistent)	일관성은 내부 일관성을 의미, SRS가 시스템 요구사항 명세서와 같은 일부 상위 문서와 일치하지 않으면 올바른 것이 아님
중요성 및 안정성에 대한 평가 (Ranked for importance/stability)	SRS는 각 요구사항에 해당 요구사항의 중요도 또는 안정성을 나타내는 식별자가 있는 경우, 중요도 및 안정성에 대해 등급이 필요함, 일반적으로 소프트웨어 제품과 관련된 모든 요구사항은 똑같이 중요하지는 않음. 일부 요구사항은 특히 생명이 중요한 응용 프로그램의 경우에서는 필수적일 수 있지만 다른 응용 프로그램은 선택적일 수 있으므로 SRS의 각 요구사항을 소프트웨어에 따라 명확하게 구별해야 함
검증 가능 (Verifiable)	SRS 문서 내에 명시된 모든 요구사항이 검증 가능한 경우에만 검증 작업을 수행할 수 있음, 즉 사람 또는 기계가 소프트웨어 제품이 요구사항을 충족하는지 확인할 수 있는 한정되고 비용 효율적인 절차가 있는 경우에만 요구사항을 검증 할 수 있음, 검증 가능한 예제는 다음과 같음(ex, 프로그램의 출력은 이벤트의 20초 이내 X 60%의 시간 내에 산출되어야 한다, 이벤트의 30초 이내 X 100% 시간 내에 생산되어야 한다.)
수정 가능 (Modifiable)	수정을 위해서 SRS은 요구사항에 대한 모든 변경사항을 쉽고 일관되게 만들 수 있는 구조 및 스타일을 갖추는 것이 필요함, 목차, 색인 및 명시적 상호 참조를 사용하여 일관성 있고 사용하기 쉬운 구조를 갖추면 추후 수정이 용이해짐, 이 때 요구사항은 중복되지 않아야 하며 다른 요구사항과 혼용하기보다는 각 요구사항을 별도로 표현하는 것이 바람직함, 중복은 종종 SRS의 가독성을 높여주지만 중복된 문서가 업데이트될 때 문제가 발생할 수 있음, 중복이 필요한 경우에는 SRS는 수정 가능하도록 명시적 상호 참조를 포함해야 함

3) 소결

CPS와 같이 복잡한 시스템을 개발하는 경우, 요구사항은 좀 더 복잡해지며 고객과 개발자 사이에 요구사항의 의미를 서로 다르게 이해하게 되면 개발 후반으로 갈수록 이를 바로잡기 힘들다. 이로 인해 프로젝트가 실패할 가능성이 증대될 수 있으므로 개발 초기에 개발하고자 하는 소프트웨어에 대해 명확하게 분석하여 요구사항을 도출하는 것은 매우 중요하다. 요구사항은 살펴본 것과 같이 이해관계자와 인터뷰나 프로토타입 제작, 브레인스토밍, 조직 환경의 관찰 과정 등의 요구사항 도출 기법을 통해 보다 명확해질 수 있으며 다양한 이해관계자와의 요구사항 협상을 통해 요구사항을 조정하는 절차가 필요하다. 이러한 요구사항 분석 과정은 사용자의 요구를 추출하고 목표를 설정하여 어떠한 방식으로 해결할지를 결정하는 단계로 무엇(what)에 초점을 둔다고 볼 수 있다. 요구사항 도출 과정에서 요구사항은 기능 요구사항과 비기능 요구사항으로 구분이 가능하다. 시스템이 수행해야 할 행위들을 구체화 한 기능 요구사항과 제품의 품질 기준 등을 만족시키기 위해 소프트웨어가 가져야 하는 성능(응답시간, 처리량 등), 신뢰도, 보안성, 운용상의 제약, 안정성, 유지보수성 등과 같은 행위적 특성으로 비기능 요구사항을 구분하여 도출한다. 이렇게 도출한 요구사항의 결과로써 요구사항 명세서가 작성되면 개발 주기 내 요구사항 변경에 대한 추적과 함께 소프트웨어 요구사항이 일관성 있고, 연계성 있게 작성되었는지 검증이 필요하다. 기존의 요구사항 관련 연구나 표준을 살펴본 결과, 요구사항의 중요성을 강조(이재기 외(2008)[57], 김상수 외(2006)[58])하거나 프로세스적 접근방법 혹은 요구공학적 관점에서의 세부적인 내용에 대한 방법론(최정은 외(2002)[59], 변정원 외(2012)[60], 박수진 외(2011)[61])에 대해 논의하고 있으나 한 단계 더 나아가 제4차 산업혁명에 대응하여 점차 복잡해지는 융복합적인 시스템에 적용될 구체적이고 필수적인 기능과 비기능 요구사항을 직접 도출하거나 도출된 요구사항을 직접 검증하는 과정을 거치는 연구의 수행이 필요한 상황이다.

57) 이재기, 남궁한, 정연서(2008), 요구사항의 도출과 우선 순위화, 대한전자공학회 학술대회, pp.115-120.
58) 김상수, 인호, 최순황, 박수용(2006), 국방 소프트웨어의 가치창출을 위한 요구공학 방법론, 정보과학회지, pp.5-13.
59) 최정은, 최순규, 이선아(2002), 소프트웨어 요구사항 관리 사례 연구, 한국정보과학회 학술발표논문집, pp.445-447.
60) 변정원, 류성열, 김진수(2012), 사례 기반의 요구사항 정형화 및 선정 평가 기법, 한국컴퓨터정보학회논문지, pp.149-161.
61) 박수진, 박수용(2011), 임베디드 시스템을 위한 그레이-박스 기반의 소프트웨어 요구사항 명세 기법, 정보과학회논문지, pp.485-490.

제3절 소프트웨어 플랫폼(Software Platform)

이 절에서는 소프트웨어 플랫폼의 의미와 필요성에 대해 알아보고 CPS 플랫폼의 유사 플랫폼인 IoT 플랫폼에 대해 조사하겠다. CPS는 복잡하고, 변경 가능성이 많아 플랫폼이 절대적으로 필요하다. CPS를 구축하기 위한 공통요구사항을 기반으로 CPS 플랫폼 구조 제안이 이 보고서의 두 번째 목적으로 플랫폼의 개념과 CPS 플랫폼으로 활용가능하리라 가정한 IoT 플랫폼에 대해 알아보겠다. CPS와 관련되어 있거나 CPS의 구성요소 또는 유사 개념으로는 IoT, 클라우드 시스템, SoS(System of Syetem), 빅데이터 등이 있다. IoT는 물리적인 환경에서 사물을 관찰하고, 통신하여, 효율적 관리를 위해 필요한 정보를 수집하기 때문에 IoT는 CPS와 유사 개념으로 간주된다.[62] IoT의 정보 수집 기능은 CPS에서 활용이 가능하기 때문에 IoT 플랫폼이 CPS 플랫폼의 기반 플랫폼으로 활용가능한 지 확인한다.

1. 소프트웨어 플랫폼의 의의와 필요성

Carliss Baldwin은 플랫폼은 다른 구성 요소 간의 연결을 제한하여 시스템에서 다양성과 진화 가능성을 지원하는 안정적인 구성 요소 집합이라고 정의한다. 시스템은 다양성이 낮은 구성 요소 집합과 다양성이 높은 구성 요소의 보완 집합으로 구분되는데 다양성이 낮은 구성 요소가 플랫폼을 구성한다고 했다. 플랫폼은 복잡한 제품을 만들기 위한 재사용되고 공유되는 요소이다. [63] 플랫폼은 시스템 중에서 공통의 요소로 구성되고, 플랫폼을 기반으로 개발된 특정 요소들이 시스템이나 제품의 새로 다른 특성을 나타낸다고 볼 수 있다.

소프트웨어 플랫폼(Platform)이란 응용 프로그램(Application)을 작성하고 실행하는 데 사용되는 소프트웨어 환경이다. 여기에는 응용 프로그램 개발을 위한 컴파일러(Compiler)[64], 유틸리티(Utility) 및 인터페이스(Interface)와 같은 소프트웨어 도구와 응용 프로그램을 실행하기 위한 런타임(Runtime) 엔진이 포함되어 있다. 컴퓨터의 경우는 마이크로소프트의 윈도우나 리눅스를 플랫폼이라 지칭할 수 있다. 자동차의 경우 AUTOSAR가 표준 플랫폼이며, 항공의 경우 ARNIC 653 표준을 기반으로 개발된 플랫폼으로 VxWorks 653[65]이 있다.

---

62) Kevin Ashton(2009), That ’ Internet of Things’ thing, RFID Journal

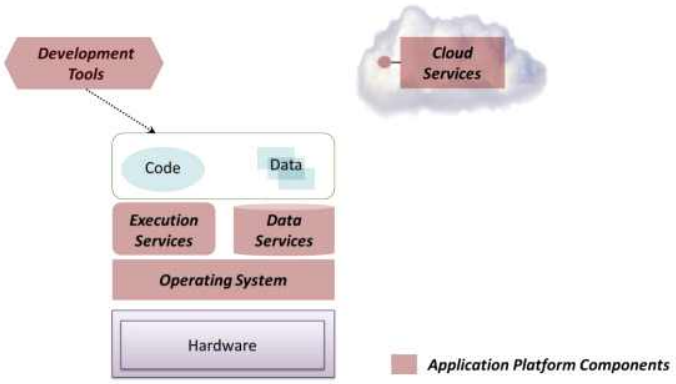
63) Carliss Baldwin, C. Jason Woodard(2008), The Architecture of Platforms: A Unified View, Harvard Business School white paper

64) 컴파일러(compiler, 해석기, 번역기)는 특정 프로그래밍 언어로 쓰여 있는 문서를 다른 프로그래밍 언어로 옮기는 프로그램으로, 소스 코드를 컴파일하는 주요 목적은 대부분 사람에게 이해하기 쉬운 형태의 고수준 언어로부터 컴퓨터나 제품에서 실행 가능한 기계어 프로그램을 만들기 위해서 임

65) 미국 WindRiver사가 ARINC 653 요구사항에 따라 시간과 공간에서 강력한 파티셔닝(partitioning, 분

David Chappell은 소프트웨어 플랫폼은 운영체제(OS, Operating System), 응용 프로그램 실행을 위한 서비스(Execution Services), 데이터 서비스, 클라우드 서비스, 개발 툴로 구성되어 있다고 했다.[66]

[그림 2-10] 어플리케이션 플랫폼



자료 : David Chappell(2011), What is an application platform?, 2011.12.

이는 하드웨어와 융합되지 않은 소프트웨어 플랫폼을 고려한 것으로 생각된다. 하드웨어와 융합된 플랫폼의 경우는 하드웨어와의 인터페이스 기능이 플랫폼의 중요한 구성요소가 될 것이다.

소프트웨어 플랫폼은 응용프로그램을 개발하기 위해 공통적으로 필요한 소프트웨어 기능들을 포함하고 있다. 소프트웨어 개발자가 프로그램의 모든 기능을 모두 개발할 수는 없으며, 그것은 비효율적인 작업이다. 플랫폼은 시스템이 많은 상호 작용 부품으로 구성되어 복잡하고, 요구 사항이 여러 종류로 이루어져 있으며, 향후 기술 개발이 불확실하고 시스템이 예기치 않은 환경 변화에 적응해야하는 경우에 장점을 가지고 있다.[67] 소프트웨어 플랫폼은 공통의 기능을 제공하여 소프트웨어 개발자가 복잡한 시스템을 개발할 수 있게 한다. 또한 플랫폼은 개발자에게 공통의 표준과 인터페이스를 제공하여, 제품의 호환성과 상호운용성을 보장한다.

CPS는 여러 응용도메인이 있으며, 아직 기술과 서비스도 개발 가능성이 많고, 물리 시스템과 사이버 시스템이 융합되어 두 요소의 특징을 모두 가지고 있어 예측이 어렵기 때문에 CPS 플랫폼은 CPS를 개발하는 데 있어서 필수적인 요소이다.

할)을 제공하는 소프트웨어 플랫폼

66) David Chappell(2011), What is an application platform?, 2011.12.

67) Carliss Baldwin, C. Jason Woodard(2008), The Architecture of Platforms: A Unified View, Harvard Business School white paper

2. CPS 플랫폼과 유사 플랫폼 : IoT (Internet of Things) 플랫폼

CPS는 임베디드 시스템이 진화되고 확장된 개념이며, 임베디드 시스템, IoT, 클라우드 시스템 등을 포함하고 있다. NIST는 CPS와 IoT를 유사한 개념으로 언급하기도 했다. IoT 플랫폼을 CPS 플랫폼 구성요소로 활용가능한지 확인이 필요하다. 소형화, 저전력화, 저가격화, 표준화에 의해 비교적 활성화되고 있는 IoT를 위한 IoT 플랫폼의 개념, 표준, 특징, 사례 등에 대해 검토한다.

스마트 홈, 스마트 시티 구축 등에 광범위하게 사용되고 있는 대표적인 IoT 플랫폼 표준인 oneM2M과 OCF(Open Connectivity Foundation)와 스마트공장과 같은 산업 분야에서 사용중에 있는 대표적인 IIoT(Industrial-IoT) 기술인 OPC UA(Open Platform Communications Unified Architecture)[68]의 표준, 특징 그리고 실제로 응용 개발에 활용되어 사용되고 있는 사례 분석을 한다. oneM2M, OCF, OPC UA 표준을 정성적 비교를 하여 어떤 표준이 CPS 플랫폼을 구성에 가장 효율적인지 분석한다.

1) oneM2M Mobius 플랫폼

(1) 개요

oneM2M은 에너지, 교통, 국방, 공공서비스 등 산업별로 종속적이고 폐쇄적으로 운영되는, 파편화된 서비스 플랫폼 개발 구조를 벗어나 응용서비스 인프라 (플랫폼) 환경을 통합하고 공유하기 위한 사물인터넷 공동서비스 플랫폼 개발을 위해 발족된 사실상 표준화 단체 및 표준규격이다.[69] 이 단체는 전세계 지역별 표준 개발기구인 TTA(한국), ETSI(유럽), ATIS/TIA(북미), CCSA(중국), ARIB/TTC(일본), TSDSI(인도) 총 8개의 SDO(Standard Development Organization)가 공동으로 설립하였다.[70]

8개의 SDO 기관 외에도 통신사업자, 장비제조사, 서비스회사 등 약 200여 멤버사가 활발히 활동하고 있으며 2015년 1월에 Release 1, 2016년 8월에 Release 2를 공개하였다.

---

68) OPC Foundation의 데이터 수집 및 제어를 위한 산업 장비 및 시스템과의 통신에 중점을 두고 있으며, 다양한 운영체제나 프로그래밍 언어에 종속적이지 않고, 많은 오픈소스가 개발되어 쉽게 활용할 수 있는 IIoT 플랫폼

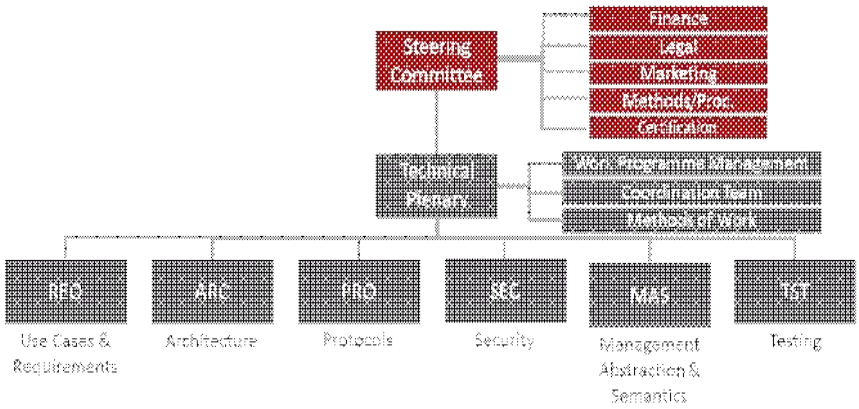
69) 미래창조과학부(2017), 제품서비스 개발을 위한 IoT와 oneM2M의 이해

70) oneM2M 홈페이지, ARIB: Association of Radio Industries and Businesses, ATIS: Alliance for Telecommunications Industry Solutions, CCSA: China Communications Standards Association, ETSI: European Telecommunications Standards Institute, TIA: Telecommunications Industry Association, TSDSI: Telecommunications Standards Development Society, TTA: Telecommunications Technology Association (TTA), TTC: Telecommunication Technology Committee

Release 3의 경우 여러 산업의 기술들과의 연동, 시험 및 인증 등 마켓 확산에 관련하여 주 초점을 맞추어 진행 중이다.

oneM2M의 기술 워킹그룹(6개)은 요구사항을 다루는 Requirement(WG1), 시스템 구조를 다루는 Architecture(WG2), 프로토콜과 관련한 Protocols(WG3), 보안관련 Security(WG4), 장치 관리 및 추상화, 의미과 관련된 Management, Abstraction and Semantics(WG5), 검사 규격을 위한 Test(WG6)로 구성된다.

[그림 2-11] oneM2M 워킹그룹



자료 : oneM2M

(2) 표준

가) 표준 개발 현황

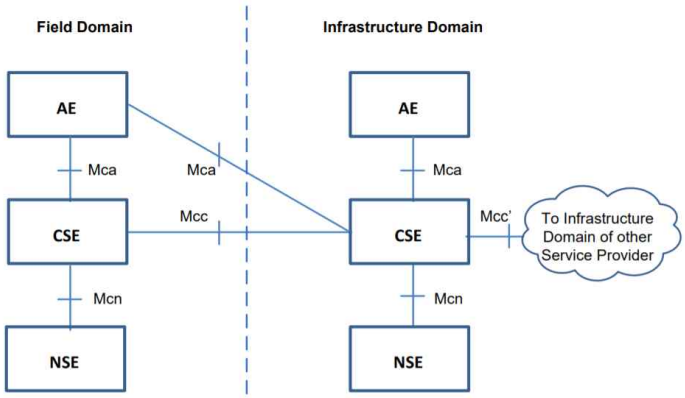
oneM2M Release 1의 아키텍처의 모든 엔티티는 Application 계층, Common Services 계층, Network services 계층 이 세 가지 계층으로 나누어진다.

응용 프로그램 엔티티인 AE(Application Entity)는 응용(Application) 계층에서 M2M 응용 프로그램의 서비스 로직을 구현하는 요소이다. AE의 예로는 함대 추적 어플리케이션, 원격 혈당 모니터링 어플리케이션, 전력 계량 어플리케이션 등이 있다.

공통 서비스 계층의 CSE(Common Services Entity)는 M2M(Machine 2 Machine) 환경의 다양한 AE들이 공통으로 사용할 수 있는 공통 서비스 기능(CSF) 세트들을 모듈화한 것을 나타낸다. CSE가 제공하는 하위 기능들은 공통 서비스 기능으로 논리적이고 정보에 입각하여 개념화될 수 있다.

Network Services Entity(NSE)는 네트워크 서비스들을 CSE에 제공한다. 이러한 서비스의 예로 장치 관리, 위치 서비스 및 장치 트리거링이 있다.

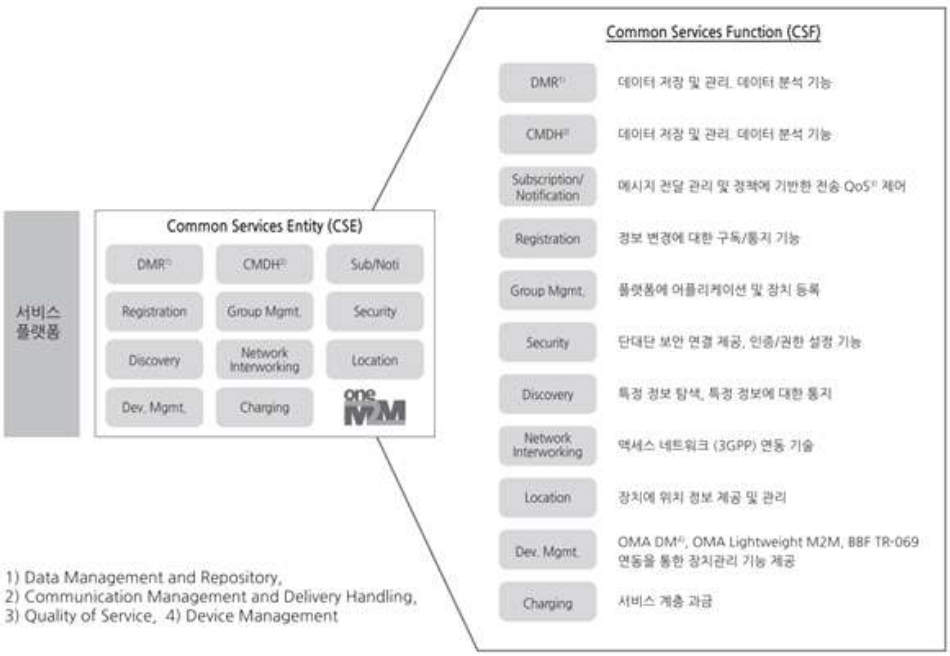
[그림 2-12] oneM2M 기능 구조



자료 : oneM2M

oneM2M 플랫폼이 제공하는 기능을 12개의 공통 서비스 기능(CSF, Common Services Function)으로 정의한다. 공통 서비스 기능은 사물인터넷 서비스 애플리케이션에서 자주 사용되는 기능, 즉 서비스 요구사항을 정의한 것으로 데이터 저장/공유, 구독/통지(Subscription/Notification), 장치 관리, 정보 탐색, 네트워크 연동 기술, 위치 정보, 과금 등의 기능을 포함하며, 보안 기능은 기본적인 인증, 권한 설정 등의 기능을 제공한다.

[그림 2-13] oneM2M Common Services Function



자료 : 김기형(2014), oneM2M 사물인터넷 서비스 플랫폼 표준화 현황

oneM2M 코어 프로토콜 메시지(Primitive)는 Release 1에서 CoAP(Constrained Application Protocol), HTTP 및 MQTT(Message Queuing Telemetry Transport) 프로토콜[71] 메시지를 통해 전송되며 향후 추가 프로토콜을 지원할 수 있도록 특정 메시지 프로토콜에 종속성을 가지지 않도록 개발되었다.

oneM2M Release 2에서는 다양한 인더스트리 사물인터넷 플랫폼 및 네트워크 연동이 목적으로 사물인터넷의 경우 AllJoyn[72], OCF 및 Lightweight M2M 기술과의 연동 규격을 제공한다.

높은 디바이스 및 애플리케이션의 호환성을 보장하기 위해 우선적으로 가전 디바이스에 대한 데이터 모델을 정의하고 있다. 반면 Release 1에서는 가전 제어 및 센싱 정보를 교환하기 위해 사전에 애플리케이션 간 정의한 데이터 모델을 container 및 contentInstance 자원 타입을 이용하였다. container는 IoT 시스템에서의 자원을 전달할 수 있는 데이터 공간이고 그러한 자원 데이터들은 contentInstance 타입에 맞춰 전달된다. Release 2에서는 oneM2M 플랫폼을 이용하는 모든 애플리케이션이 표준에 정의된 가전 디바이스 데이터 모델을 사용함으로써 가전 제조사 및 애플리케이션 개발자 간 별도의 데이터 모델을 정의(container 및 contentInstance 정의)하는 번거로움을 없애고 제품과 애플리케이션 사이의 호환성이 보장된다. 통신 프로토콜은 동시 송수신(Full-duplex)을 지원하는 WebSocket이 추가되었다.

Release 3의 경우, 제조, 자동차, 스마트 그리드, 스마트 도시, 스마트 홈 등의 여러 산업 도메인과 연동으로서 마켓 확산에 주 초점을 두고 개발 중에 있다. 홈 분야 구현(Home Domain Enablement)은 스마트 홈의 가전 디바이스들의 oneM2M 표준을 기반으로 한 유스케이스들을 통해 전체 시스템의 동작들에 대해 정의하였다. 추가적으로 oneM2M 표준이 적용된 플랫폼 및 디바이스 등의 시험과 인증 절차를 강화함으로서 신뢰성 있는 제품 확산에 기여하고자 한다.

나) 표준기반 Mobius와 &Cube 플랫폼

OCEAN(Open allianCE for iot stANdard)은 oneM2M을 준수하는 IoT 공통 플랫폼 개발 및 제공을 목적으로 2015년 1월 KETI(전자부품연구원)과 정부를 주축으로 발족한 연합체이다. OCEAN[73]의 목적은 IoT 표준에 기반 한 오픈소스를 공유하고 회원 간의 공동 작업을 장려

71) MQTT와 CoAP는 인터넷에 기반의 풍부한 리소스를 가진 디바이스로부터 IoT 기반의 제한된 리소스를 가진 디바이스로 통신을 지원하는 경량 IoT 통신 프로토콜

72) IoT 연합 단체인 올신얼라이언스(AllSeen Alliance)에서 표준화한 오픈 소스 기반의 IoT 플랫폼. 올조인은 로컬 영역에서 올조인 기기 간 피투피(P2P: Peer-to- Peer) 통신을 지원하는 IoT 플랫폼

73) 2017년 7월자로 기관, 기업 및 대학 등 727개의 멤버들이 참여하고 있다.

하는 것으로 보다 빠른 IoT 서비스 상업화와 IoT 생태계를 넓히기 위해 노력하고 있다.

OCEAN은 2015년 1월 oneM2M Release 1 기반의 오픈 소스 IoT 플랫폼인 Mobius로서 서버용 플랫폼 ‘Mobius’와 디바이스 환경의 플랫폼 ‘&Cube’를 공개하였고, 이들은 MQTT 프로토콜을 지원한다. 2017년 7월 Release 2의 CoAP, Websocket 등 기능을 추가하여 Mobius 2.0으로서 개정 발표하였다.

Mobius는 oneM2M 표준을 기반으로 하는 오픈 소스 IoT 서버 플랫폼으로서 서로 다른 서비스 도메인의 IoT 어플리케이션에 공통 서비스 기능들을 제공하며 필드 도메인의 디바이스에서 보내온 정보를 저장하거나 사용자로부터 제어신호를 받아 다시 해당 디바이스로 보내는 미들웨어로서 동작한다.

oneM2M 아키텍처에서 Mobius는 인프라 도메인의 클라우드 서버인 IN-CSE[74]를 구현하였다. 또한 Mobius 2.0 버전이 발표되면서 인더스트리 분야 서비스를 위한 높은 신뢰도 및 실시간성, 스트리밍 연계, 의미 기반 데이터 연동, OCF, LWM2M(Lightweight M2M)[75] 등과 같은 타 표준 기술과 연동 등 다양한 기능 지원이 추가되었다.

&Cube 플랫폼은 필드 도메인의 디바이스를 위한 것으로 Rosemary, Lavender 및 Thyme으로 다음의 총 3가지 타입으로 구성된다.

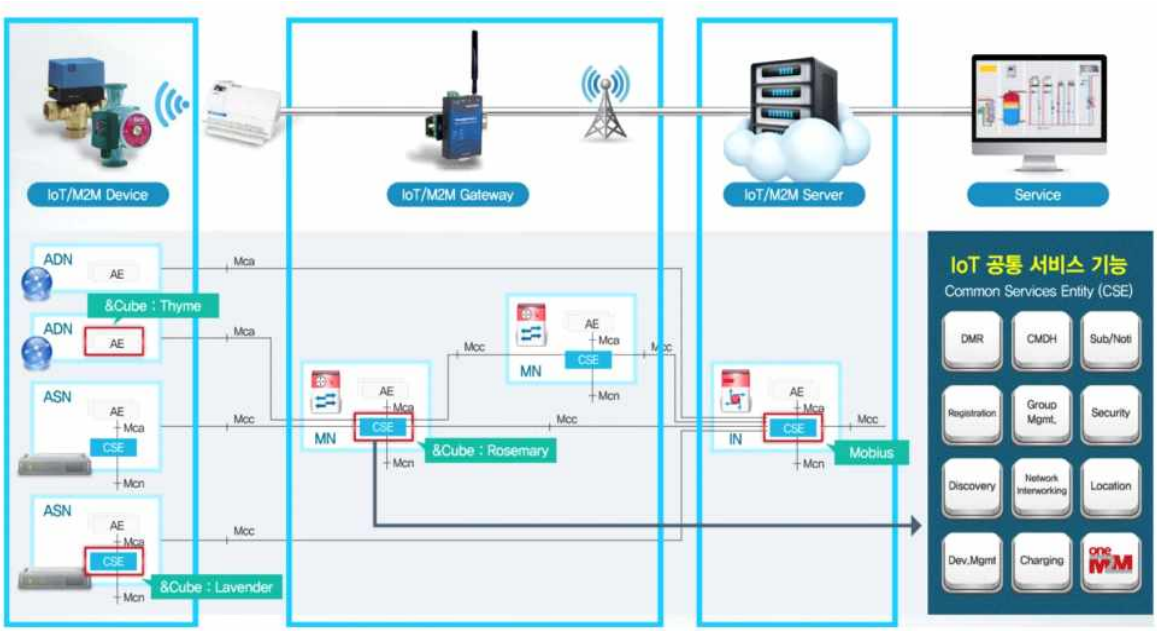
&Cube Rosemary 플랫폼은 MN-CSE[76]를 지원하는 게이트웨이용 디바이스로서 &Cube 플랫폼이 탑재된 하부 디바이스와의 연동 지원

&Cube Lavender 플랫폼은 ASN-CSE[77]를 지원하는 M2M 디바이스로서 &Cube Rosemary 플랫폼과 연계 또는 단독으로 Mobius 플랫폼과 연동 지원

&Cube Thyme 플랫폼은 ADN-AE[78]를 지원하는 경량 디바이스로서 디바이스에 탑재된 센서 및 작동장치(Actuator)와 연계 인터페이스를 통해 수집된 데이터를 Mobius와 연동되며 2017년 7월 발표된 Mobius 2.0 버전에서 Thyme은 자바, Nodejs, 안드로이드, 아두이노를 지원하고 있음

74) Infrastructure Node CSE : 클라우드 서비스 플랫폼과 같은 인프라 노드에 구성된 CSE
75) OMA(Open Mobile Alliance)에서 정의한 소형 장치를 포함하는 다양한 기기 지원을 위한 사물인터넷 기기 관리 표준
76) Middle Node CSE : 홈 게이트웨이 등에 구성된 CSE
77) Application Service Node CSE : 응용 서비스 노드에 구성된 CSE
78) Application Dedicated Node AE: 경량 디바이스 노드에 구성된 AE

[그림 2-14] OCEAN Mobius 플랫폼



자료 : KETI

(3) 특징

oneM2M은 기업 및 기관의 연합 뿐 아니라 유럽, 북미, 한국, 일본, 중국 등 총 8개의 표준화기구들이 함께 표준을 진행하고 있는 만큼 다른 IoT 표준 플랫폼에 비하여 공신력이 높다.

oneM2M의 기술 워킹그룹(6개)은 Use Case 및 요구사항, 시스템 구조, 프로토콜), 보안, 장치관리 및 추상화, 시멘틱(의미), 검증으로 구성된다. 표준 개발을 전체의 그룹에서 진행하는 것이 아닌 각각의 6개의 워킹그룹으로서 해당 분야에 대한 표준화를 진행하며 기술 보고서(TR: Technical Report) 및 기술 규격(TS: Technical Specification)을 개발한다.

oneM2M은 현재 Release 1과 Release 2를 통해 전체적인 아키텍처 구성 및 oneM2M 시스템 구성을 위한 최소한의 표준이 완성되었고 Release 3부터 자동차, 공장자동화 등의 다양한 산업 도메인들에 적용하기 위해 OPC UA, OMG의 DDS(Data Sistribution Service) 등 산업용 기술 표준들과 협력을 진행 중이다.

<표 2-13> oneM2M 표준 발전 단계

분류	Release 1	Release 2	Release 3
기준	최소한 표준	연동	산업도메인 적용
시기	12년 9월 ~ 15년 1월	~ 16년 7월	~18년 2월
특징	12개의 공통 서비스 기능(CSF : Common Service Functions) 정의	AllJoyn, OCF 및 Lightweight M2M 기술과의 연동	- 제조, 자동차, 스마트 그리드, 스마트 홈 등에 적용 - 검증 절차 추가
네트워크	CoAP, HTTP 및 MQTT 프로토콜	WebSocket 프로토콜	

(4) 응용개발 사례

가) 국가

유럽에서는 European Union Almanac라는 사물인터넷 기반의 스마트 시티 프로젝트를 기획, 도시 이탈리아 토리노에서 Telecom Italia와 함께 oneM2M기반의 조명, 자전거 공유 및 로봇 등 광범위한 스마트 시티 응용 어플리케이션을 기획 및 시험 중이다.[79] 한국은 과학기술정보통신부에서 주관, 부산시와 SKT가 컨소시엄을 구성하여 부산 해운대 센텀시티를 중심으로 '사물인터넷 기반 스마트시티 조성 사업'이라는 명칭 하에 oneM2M Release 1를 적용하였다.[80]

나) 기업

SKT는 ThingPlug(씽플러그)라는 oneM2M 표준을 만족하는 통합형 IoT 플랫폼을 2015년 오픈하였다.[81] ThingPlug는 Mobius 프로젝트 성과가 반영된 결과물로서 국제 표준인 oneM2M과 자체 규격(GMMP) 기반의 Open API를 제공하고 SKT의 서비스 인프라의 기능들도 활용할 수 있도록 공개하였다. 차량, 소비재 및 기업용 커넥티비티 선도 기업인 하만은 모바일 기술 개발 회사인 InterDigital과 협력하여 oneM2M 호환되는 IoT 솔루션 개발하였다.[82]

79) http://www.almanac-project.eu/news.php
80) http://k-smartcity.kr/index.php
81) http://news.joins.com/article/17992244
82) https://worldindustrialreporter.com/harman-partners-with-interdigital-on-onem2m-compliant-end-to-end-iot-solutions/

다) 기관

TTA(한국정보통신기술협회)는 oneM2M 글로벌 인증기관으로 지정받았다. 이에 따라 국내뿐 아니라 전세계 모든 제품들을 대상으로 공식 시험·인증 서비스를 한다.[83] 영국의 Buckinghamshire County Council은 운송 서비스 회사인 Arup, WorldSensing and Traak, InterDigital과 연합, oneTRANSPORT solution[84]이라는 county-wide 스마트 운송 모델 및 시스템을 구축하였다.[85] 교외 지역을 위한 스마트 시티 솔루션을 만들기 위해 자동차 번호판 인식 및 교통 카메라 등의 교통 데이터 수집 및 시스템 구축에 oneM2M 표준을 적용하였다.

2) OCF IoTivity

(1) 개요

OCF(Open Connectivity Foundation)는 사물인터넷 서비스 구현 시 경량형 CoAP 프로토콜로 사물인터넷 장치들을 연결하여 장치에 존재하는 자원들을 상호제어 할 수 있게 하는 표준 플랫폼 기술이다.[86]

2014년 7월 OIC(Open Interconnect Consortium)가 삼성, 인텔 등 중심으로 시작하여 2015년 12월 스마트 홈의 대표적 국제표준단체인 UPnP포럼을 통합 흡수하면서 회원사가 100개 이상으로 성장하였고 2016년 2월, 마이크로소프트, 퀄컴 등이 합류하여 확대된 기업 표준화 단체이다. OCF 주축 기업들로 MS, 시스코, 일렉트로룩스, GE, 인텔, 퀄컴, 삼성전자, 아리스, 케이블랩스이며 이들은 반도체와 소프트웨어(SW), 플랫폼, 제조사 등 대표적인 기업으로 사물인터넷 산업을 이끌고 있다.[87]

(2) 표준

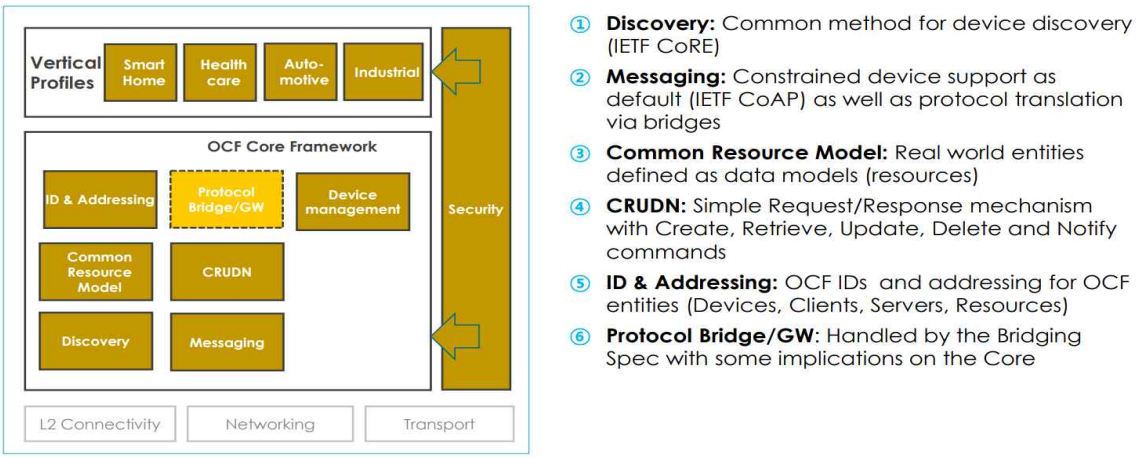
OCF는 다양한 사물인터넷 유무선 연결기술들 상에서 유연하게 탑재되어 동작 가능하도록 프레임워크가 고안되었고, 그 위에 스마트 홈, 자동차, 물류, 헬스케어 등 다양한 사물인터넷 서비스(Profiles)가 가능하도록 개발되었다.

2017년 6월을 기준으로 OCF는 OCF SPECIFICATION 1.0을 배포 중으로, 기존 OIC 1.1 표

83) https://byline.network/2017/02/1-564/
84) https://onetransport.io/
85) http://ir.interdigital.com/file/Index?KeyFile=390952300
86) https://openconnectivity.org/
87) 박수홍(2016), OCF 사물인터넷 오픈소스 플랫폼 표준 동향, TTA Journal Vol.166 pp41-45

준을 기반으로 개발되어 인프라인 중심 프레임워크, 보안, 스마트홈, 헬스케어 등 응용도메인과 리소스로 구성된다.

[그림 2-15] OCF 중심 프레임워크(Core Framework)



자료 : OCF SPECIFICATION INTRODUCTION AND OVERVIEW(2017), OCF, July 2017

리소스(Resource Model)은 사물인터넷 서비스들의 연동 시 리소스, 데이터 모델 및 AllJoyn의 맵핑 등에 관해 제시하고 있으며, 사물인터넷의 특성상 경량 기기들이 많아질 것을 예상, CoAP(Constrained Application Protocol) 표준을 주 프로토콜로 채택하였다.

(3) 특징

OCF는 오픈소스와 표준 규격을 동시에 개발하여 제공하는 방식으로 독창적인 운영 방법을 바탕으로 회원사 각각이 자체적으로 소프트웨어를 개발해야 하는 부담을 줄이고, 오픈소스를 통한 상호연동성 보장으로 빠른 제품 개발 및 확장을 가능하게 한다.

OCF 표준 규격 개발에 있어 자발적인 외부 개발자 참여를 독려하기 위해 오픈소스 프로젝트 운영을 전담으로 하는 비영리 리눅스재단을 통해 IoTivity 프로젝트 운영 중으로 해당 소스코드는 OCF 단체 가입에 관계없이 개발 참여가 가능하다

오픈소스 IoTivity 프로젝트는 아파치(Apache) 2.0 오픈소스 라이선스를 채택하여 오픈소스에 기여하는 모든 기술의 특허를 무상으로 사용토록 하여 표준의 구현에 레퍼런스로 활용된다.

(4) 응용개발 사례

가) 국가

한국에서는 2017년 3월, 삼성전자, LG전자, ETRI가 주관으로 40여개의 업체 및 기관이 협력하여 최대 사실표준기구인 OCF의 공식 한국지부인 OCF 코리아를 설립하였다.[88] OCF 코리아는 OCF의 세계최초 지역포럼으로서, IoT 생태계 확산을 위한 전초기지로서 활동하고 있다.

나) 기업

삼성전자는 2020년까지 가전 전 제품에 스마트 기능을 탑재하고 OCF 주축 멤버로서 다양한 업체와 연동 서비스 제공할 예정이며 업계 최초로 스마트TV, 패밀리허브 냉장고, 에어컨에 OCF 인증을 받았고 추가적인 인증 확대 추진 및 2018년부터 출시되는 스마트가전 전 제품에 OCF 규격 탑재할 예정이다. CISCO, GE, LG, Sony 등의 기업들은 출시 제품들의 OCF 인증을 받으며 타 기업 제품들 간 연결성 확대하고 있다.[89]

다) 기관

커넥티드 차량을 위한 오픈소스 소프트웨어와 표준 개발 커뮤니티인 GENIVI Alliance는 OCF와 협력을 체결함으로서 스마트 홈과 차량 간 안전한 정보 교환 및 통일된 모델을 표준 개발 예정이다.[90]

ETRI(한국전자통신연구원)는 OCF 표준 및 IoTivity 기반의 헬스 케어 디바이스를 개발하였고 OCF의 쉽고 빠른 연결성을 통해 기존의 스마트 홈 분야 위주로 적용하던 OCF 기술을 다양한 산업 분야로 도입하고 있다.

한국전력은 전력업계에서는 세계 최초로 OCF에 가입함으로서 에너지 사물인터넷 분야로 시장을 확대하고자 한다.[91]

---

88) http://www.edaily.co.kr/news/news_detail.asp?newsId=03781846615867912&mediaCodeNo=257&OutLnkChk=Y

89) http://www.zdnet.co.kr/news/news_view.asp?artice_id=20171226180336&type=det&re=

90) http://www.yonhapnews.co.kr/bulletin/2017/02/16/0200000000AKR20170216065100009.HTML?input=1195m

91) http://news.inews24.com/php/news_view.php?g_serial=1034874&g_menu=022100&rrf=nv

3) OPC UA 플랫폼

(1) 개요

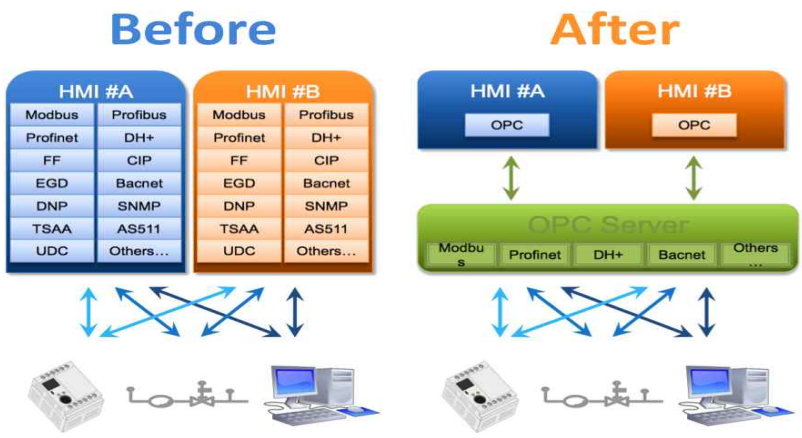
제조 시스템에 서비스 지향 아키텍처(Service Oriented Architecture, SOA)가 도입됨에 따라 보안, 데이터 손실 방지, 중복 데이터 처리 또는 복잡한 데이터 구조 사용과 같은 업계의 새로운 요구사항을 해결하기 위해 OPC 재단은 OPC UA(Unified Architecture) 표준을 개발하였다. OPC UA는 데이터 수집 및 제어를 위한 산업 장비 및 시스템과의 통신에 중점을 두고 있다.

OPC UA는 OPC UA의 전신인 OPC 표준(OPC Classic)의 기능을 통합하고, 데이터의 의미를 좀 더 자세히 전달하는 것과 같은 추가적인 서비스를 제공함으로써 기존의 제약 사항을 해결하였다 OPC 표준(OPC Classic)은 실시간 데이터(OPC DA/Data Access), 경보 및 이벤트 모니터링(OPC AE/Alarms and Events), 데이터 및 기타 응용 프로그램(OPC HDA/Historial DA)에 대한 액세스를 위한 클라이언트와 서버, 서버 및 서버 간 인터페이스를 정의한다.

초기에는 Modbus[92], Profibus[93] 등과 같은 PLC(Programmable logic controller) 특정 프로토콜을 표준화된 인터페이스로 추상화하여 HMI(Human-Machine Interface), SCADA(Supervisory Control And Data Acquisition) 등 상위 시스템과 원활한 연동을 목적으로 개발하였다.

1996년 윈도우의 OLE(Object Linking and Embedding) 연결규약을 프로세스 제어에 적용함으로서 OPC 서버 공통 모듈을 통하여 각각의 응용프로그램이 통신이 가능하게 되었다.

[그림 2-16] OPC 구조



자료 : Thomas J. Burke(2013), OPC Connectivity, OPC Foundation, 2013.10.23

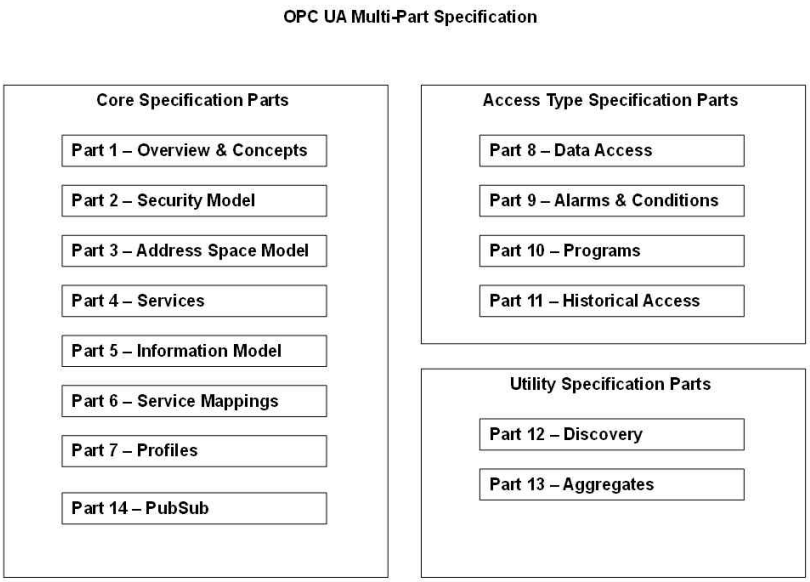
92) 자동화 디바이스 간 통신을 위해 1979년에 Modicon (현재 Schneider Electric)에 의해 개발된 통신 프로토콜

93) BMBF (German department of education and research)이 자동화 디바이스 간 통신을 위해 1989년에 개발한 통신 프로토콜

(2) 표준

OPC UA 표준은 2008년 개별적인 OPC Classic 사양의 모든 기능을 하나의 플랫폼으로 통합하는 서비스 지향 아키텍처로 기존 OPC 표준에 추가로 그림 10과 같이 14개 부분의 표준이 진행 중이다.

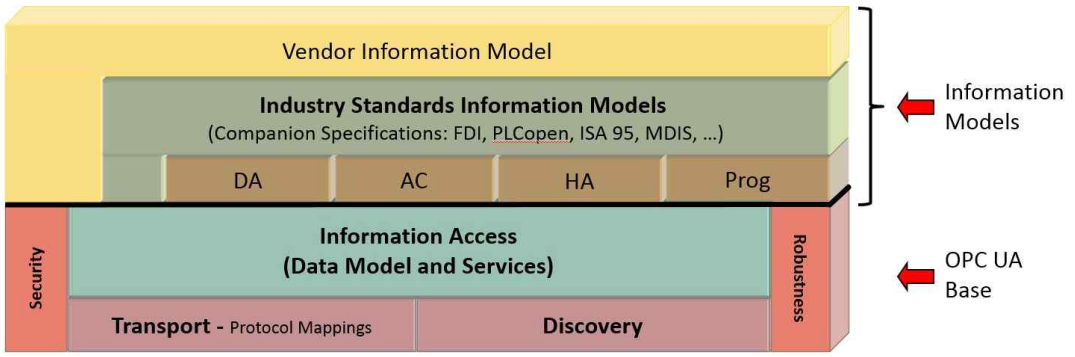
[그림 2-17] OPC UA Specification Organization



자료 : OPC Foundation

Security Model은 다양한 네트워크 및 환경에 놓인 OPC UA의 특성상 사이버 공격에 치명적이므로 이를 대비하기 위한 상황에 맞는 유연한 보안 매카니즘을 제공한다. Address Space Model에서는 서버가 클라이언트를 인식하기 위한 표준화된 방법인 주소 체계를 주로 다룬다. Services Model은 OPC UA의 서버와 클라이언트간 기능들에 대해 정의함으로서 서비스 정의와 구현을 쉽게 할 수 있다. Information Model은 OPC 재단에서 제공하는 Model인 Data Access(DA), Alarm & Event(AE), Historical Access(HA)들을 이용하여 객체 모델링하는 방법을 정의한다. PubSub은 2017년 발표가 된 가장 최신의 표준으로 기존 클라이언트/서버 모형으로는 네트워크에 노드가 많은 경우 한계가 발생하여 PubSub model을 통해 이를 극복하고자 하는 것이다.

[그림 2-18] OPC Unified Architecture



자료 : OPC Foundation

(3) 특징

OPC UA는 윈도우, Apple OSX, 안드로이드, 리눅스 등 다양한 플랫폼에서 구현 가능함으로서 확장성을 보장한다. 따라서 여러 다른 제조업체 제품 간의 데이터 교환을 위해 쉽게 사용할 수 있는 기능을 제공하고 있고, 유연하고, 모듈화 된 자동화 솔루션을 더 용이하게 구현할 수 있다.

더불어 서로 다른 OPC UA 응용 프로그램간의 통신을 위해 여러 가지 프로토콜을 제공하는데, 고성능과 고속도를 제공할 수 있는 바이너리가 포함된 UA TCP, 보안이 강화된 사이트에 인터넷 통신이 요구될 때는 HTTP 및 SOAP을 제공한다.

OPC UA의 데이터 모델링은 생산 데이터, 알람, 이벤트 및 이력 데이터를 OPC UA서버에 통합할 수 있게 하는 정보 모델로 표현된다. Classic OPC에서는 데이터의 값만 나타냈다면 OPC UA에서는 데이터의 값뿐만 아니라 데이터의 의미까지 나타내는 정보 모델을 나타냄으로써 보다 효율적인 데이터 전달을 제공하고 있다.

OPC UA는 실시간성, QoS 등 벤더들의 요구사항을 만족시키기 위해 다양한 기술들과의 결합을 시도 중이다. OPC UA의 실시간 통신 구현을 위해 OPC 재단은 Time-Sensitive Networking(TSN) 기술을 적용하고자 한다. TSN 기술을 토대로 데이터의 정확한 전송 시간을 보장하여 결정론적인 네트워크를 구현할 수 있다. TSN은 IEEE 802.1 TSN Task Group 주도하에 다양한 4차 산업혁명 분야의 통신 요구사항(시간 동기화) 만족을 위한 표준화가 진행 중이다.

OPC UA에 Pub/Sub model 표준을 추가하면서 산업IoT(Industrial Internet of Things, IIoT)의 수많은 요소들의 연결성을 보장하고자 한다. OPC 재단은 Object Management Group(OMG)의 또 다른 IIoT의 실시간 통신 프로토콜인 Data Distribution Service(DDS)와의 결합함으로

상호 보완 및 협업을 진행 중이다. DDS란 데이터 중심 통신 미들웨어로 OMG에서 표준화를 진행하고 있으며, 국방 무기체계 등 고신뢰성과 실시간성이 동시에 요구되는 시스템에 적합하고, 최근 미국의 IIoT 분야에 적극적으로 활용 중이다.

(4) 응용개발 사례

미국은 OPC 재단의 본사가 위치한 장점을 이용하여 표준화에 앞장서고 있으며 IBM, MS 등 서버 및 소프트웨어에 강점을 가진 기업들이 OPC UA 표준 기술 연동을 위해 노력 중이다. Microsoft Azure, IBM Tivoli 등이 OPC UA와 연동을 추진하고 있다.

EU에서는 SAP, 지멘스와 같은 기업들이 OPC UA 표준을 획득하기 위한 움직임이 활발하다. OPC Day Europe 2016에서 SAP는 과거에 다양한 시스템을 연결하기 위해 다양한 프로토콜을 사용하던 방식 대신 통합 시스템 OPC UA 기술 적용하였다.[94]

한국에서는 OPC UA를 통해 실제 제조 시스템에서 사용되는 프로토콜을 통합한 네트워크 데모 시스템 구축할 계획이다. '오토메이션월드+스마트공장엑스포' 전시회에서 다양한 이기종 산업용 이더넷 프로토콜과 무선 기술들을 OPC UA표준을 기반으로 하나의 통합 시스템을 구축한 모델을 시연하였다.[95]

4) IoT 플랫폼 비교

IoT 플랫폼 표준인 oneM2M, OCF, OPC UA를 정리하고, 각 표준은 IoT의 요구사항을 얼마나 만족시키고 있는지 확인한 결과를 다음과 같이 정리하였다.

<표 2-14> IoT 플랫폼 별 요구사항 지원 비교

Requirements		oneM2M	OCF	OPC UA
전체 시스템	다양한 통신 프로토콜 연동	HTTP MQTT CoAP WebSocket	CoAP	AMQP UADP DDS MQTT XMPP HTTPS WebSocket

94) https://www.prosysopc.com/blog/opc-day-europe-2016/
95) http://www.motie.go.kr/motiee/presse/press2/bbs/bbsView.do?bbs_seq_n=159206&bbs_cd_n=81, 2017.3.
96) 공유된 개념에 대한 정형화되고 명시적인 명세(formal and explicit specification). 온톨로지는 단어와

	다른 표준과 연동	OCF 3GPP OPC UA DDS LWM2M OSGi Modbud	oneM2M	oneM2M DDS TSN
	시스템 서비스 및 장치 식별	O	O	O
	다중 장치 및 게이트웨이 간 상호작용	O	O	O
	다양한 산업 도메인 지원	O	X	O
	대규모 확장성 지원	O	X	X
자원 관리	장치 관리 및 구성	O	O	O
	네트워크 구성 검색(topology, protocol)	O	O	O
	그룹화 된 방식의 장치 관리	O	O	O
데이터 시멘틱 처리	**온톨로지(ontology)[96]**	O	X	X
	자원 의미 정보 관리	O	O	O
	온톨로지에 따른 데이터 분석	X	X	X
보안	데이터 기밀성 보장	O	O	O
	데이터 무결성 보장	O	O	O
	시스템 인증 절차 지원	O	O	O
	특정 그룹 보안성	O	O	O
운영	응용 프로그램 모니터링	O	O	O
	응용 프로그램 소프트웨어 관리	O	O	O
	응용 프로그램 실행 상태 구성	O	O	O
통신 관리	통신 QoS 설정 지원	O	O	O
	우선 순위 통신 요청 분류	O	O	O
	데이터 전달 QoS 보장	X	X	X

OCF는 IoT 환경에서의 데이터 보안이나 운영 통신 관리에서 장점을 보이고 있다. 하지만 OCF의 응용도메인은 스마트 홈과 같은 가전기기들이 사용될 수 있는 환경이어서 다양한 산업 도메인을 지원하는데 한계를 보이고 있고 또한 대규모 환경을 구축하는데도 어려움을 보이고 있다.

관계들로 구성된 일종의 사전으로서 생각할 수 있으며, 그 속에는 특정 도메인에 관련된 단어들이 계층적으로 표현되어 있고, 추가적으로 이를 확장할 수 있는 추론 규칙이 포함

OPC UA는 IIoT(Industrial-IoT, 산업IoT) 환경 구축에 사용되는 기술로서 산업 도메인에 적용을 목적으로 두고 있어 다양한 프로토콜 지원과 다른 표준과의 연동 등에 강점을 보이고 보안, 운영, 통신관리 영역에서 IoT 환경이 요구하는 대부분의 기능들을 지원하고 있다. 하지만 OPC UA는 데이터 수집을 주요한 기능으로 제공하고 있어 온톨로지 기능이나 그에 따른 데이터 분석 기능 제공에는 어려움이 있다. 또한 OPC UA는 산업 환경에서의 IIoT 연동 기술로서 실시간성을 지원하는 대규모 시스템을 효과적으로 지원하기 위해서는 어려움이 있다.

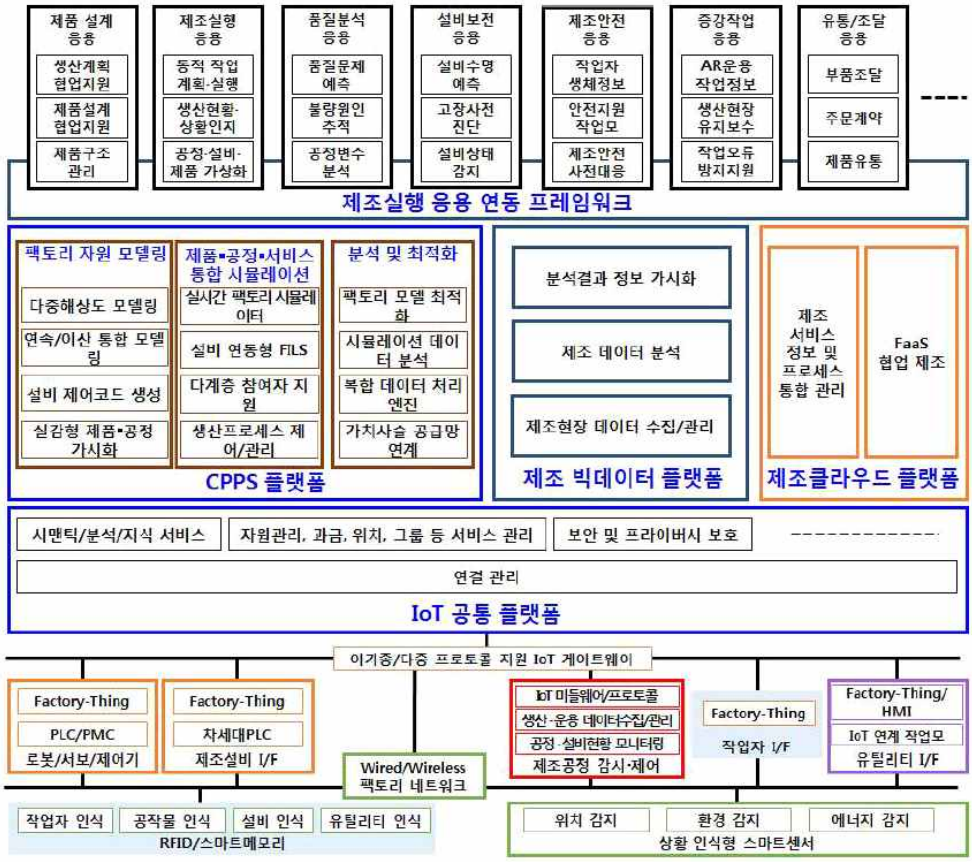
oneM2M 표준은 다양한 프로토콜을 지원하고 여러 표준과 연동을 시도하고 있다. 또한 대규모 환경의 구축을 위한 대부분의 요구사항들을 만족시키고 있어 대규모 CPS 구축을 위한 기반 IoT 플랫폼 구조로 oneM2M을 선택하였고, 요구사항 지원에 대한 검증을 위한 구현된 플랫폼으로 oneM2M 표준 중 가장 널리 사용되는 Mobius 플랫폼을 선택하였다.

CPS와 IoT를 비교하면, IoT는 인터넷에 모든 사물이 네트워크로 연결되어 새로운 서비스를 제공하는 것이며, CPS의 경우는 물리적 시스템을 사이버 시스템이 통합하여 제어하여 가치 창출하는 것이다. IoT와 CPS는 사물이 통신으로 연결되는 시스템이라는 개념과 응용 영역은 비슷하다. 물론 IoT나 CPS에 정의에 대한 다른 시각도 있으나, 본 보고서에서는 CPS를 IoT의 확장 개념이라 보고 연구를 진행하고자 한다. IoT는 사물간의 연결과 데이터 축적을 위한 시스템이라 규정하고, CPS는 IoT에 의해 축적된 데이터를 기반으로 물리세계를 제어하는 시스템이라 규정하고자 한다. 물론 CPS는 데이터의 실시간성이 IoT보다 중요하다.

3. IoT 플랫폼과 CPS 플랫폼과의 관계

본 연구에서는 개념적 CPS 플랫폼을 구성하려고 하며, 이는 IoT 플랫폼 위에서 동작하는 CPS 플랫폼을 정의하고자 한다. 한국정보통신협회의 ICT 제조융합 스마트공장 참조 모델에서는 다음과 같이 CPS 플랫폼과 IoT 플랫폼의 관계를 정의하였다. CPS 플랫폼은 IoT 플랫폼에서 수집한 정보를 이용하여 모델링, 시뮬레이션과 분석을 수행한다.

[그림 2-19] 스마트공장 시스템 모형



자료 : 정보통신단체 표준(2015), ICT 제조융합 스마트공장 참조 모델, 2015.12.

이 모형은 스마트공장에 대한 구조도이나 스마트 교통, 스마트 그리드 등 다른 대규모 CPS 응용도메인에서도 IoT 플랫폼에서 수집하고 정제한 데이터를 이용하여 CPS 플랫폼에서 데이터를 분석하여 물리 세계를 제어하는 모형 구현이 가능하다.

제4절 선행 연구

1. CPS 관련 연구

1) CPS 특성 연구

CPS에 대한 연구는 추상화, 모형 기반 개발, 검증, 인증 등의 개발 관련 사항, 융합시스템, 센서, 네트워크 등에 대한 CPS 개별 요소와 견고성, 신뢰성, 안전성, 보안 등의 특징에 대한 연구가 필요하다.[97] Volkan Gunes는 CPS에 특징으로 고신뢰성(Dependability), 지속가능성(Sustainbility), 보안(Security), 품질 측면의 신뢰성(Reliability), 상호운영성(Interoperability), 예측가능성(Predictability) 등 특징에 대해 도메인별로 연구된 내용을 분류하고 있다.[98] 다음은 이 연구와 관련된 항공, 스마트 제조, 지능형 교통 도메인과 고신뢰성, 보안, 신뢰성, 상호운영성 특징에 대해 정리하였다.

<표 2-15> CPS 응용 도메인에 중요한 특징에 관한 연구

도메인	고신뢰성 (Dependability)	보안 (Security)	신뢰성 (Reliability)	상호운영성 (Interoperability)
스마트 교통	동향, 기술 분석 (NIST) 안전한 자동차 (Koushanfar)	안전한 자동차 (Koushanfar) CPS 보안(Das)	지능형교통 시스템 신뢰 통신(Patil)	동향, 기술 분석(NIST)
항공	CPS 신뢰성(Anwar) 항공 CPS (Sampigethaya)	CPS 안전, 보안, 지속성 (Banerjee) CPS 미래 (Adam) 무인기 (Weber)	무인기(Gertler) 무인기의 안전과 신뢰성(Bhamidipati)	항공기 확장성(Bonnefoy)
스마트 제조	동향, 기술 분석 (NIST) CPS 신뢰성(Anwar)	동향, 기술 분석 (NIST)	CPS 자동화 시스템(Zühlke)	CPS 자동화 시스템(Zühlke)

자료 : Volkan Gunes et all(2014), A Survey on Concepts, Applications, and Challenges in Cyber-Physical Systems, KSII TRANSACTIONS ON INTERNET AND INFORMATION SYSTEMS, 2014.12.에서 편집

97) J. H. Shi, J. F. Wan, H. H Yan and H. Suo(2011), A survey of cyber-physical systems, in Proc. of the Int. Conf. on Wireless Communications and Signal Processing, November 9-11, 2011.

98) Volkan Gunes et all(2014), A Survey on Concepts, Applications, and Challenges in Cyber-Physical Systems, KSII TRANSACTIONS ON INTERNET AND INFORMATION SYSTEMS, 2014.12.

CPS는 물리 세계와 사이버 세계를 융합하는 시스템이므로 복잡하고 두 도메인의 특징을 모두 가지고 있어 추상화, 모델링, 플랫폼 등의 연구가 응용 도메인에 유용하게 활용될 수 있다.

2) CPS 요구사항 관련 연구

CPS 소프트웨어 공학에 관한 연구를 위한 노력이 있다. CPS는 불확실하고 다양한 시스템이기 때문에 소프트웨어 공학의 역할이 중요하다. NESSI (Networked European Software and Services Initiative)는 CPS 소프트웨어 공학 측면에서 문제점을 인식하고, CPS 소프트웨어 응용 프로그램, 서비스의 기술 발전을 위해 구성되었다.[99]

NESSI에서 고려하는 CPS의 소프트웨어 공학 측면의 연구과제는 다음과 같다. 첫 번째 연구과제는 대규모 환경에서 품질과 성능을 관리하는 것이다. 이를 위해서는 비기능 요구사항 관리를 위한 방법론, 기술, 도구를 개발하고 다양성과 불확실성을 관리하고 위한 테스팅 방법을 개발하는 것이 필요하다. 두 번째 과제는 소프트웨어 개발 주기를 지원하는 원칙, 방법, 도구를 지원하는 것이다. 세 번째 과제는 클라우드, 빅데이터와 CPS를 융합하는 것이다. 네 번째는 인간과 CPS 상호작용에서 일어날 수 있는 문제를 해결하는 것이다. 다섯 번째는 CPS의 역동적 구성과 적용을 위한 미들웨어와 플랫폼에 관한 것이다. 마지막으로는 CPS 구축을 위한 새롭고, 강력한 프로그래밍 모형에 대한 것이다. 이와 이 보고서과 관련이 있는 사항은 첫 번째 요구사항과 관련된 부분과, 다섯 번째 플랫폼에 관한 부분이다.[100]

CPS 요구사항 관련 연구는 다음과 같다. CPS를 구축하기 위해서는 기계공학, 전기공학, 컴퓨터과학 등의 기술이 필요하고, 관련자들이 목적하는 CPS에 대한 공감대를 형성하기 위해서는 요구공학에서 해결해야 하는 새로운 과제가 생기게 된다. 이를 위해 자연어 처리를 이용하여 요구사항을 특정 도메인에 맞게 처리하는 것에 대한 연구가 시행되었다.[101]

독립적으로 CPS를 개발하기 위해 수집되고 정제되고, 분류되어야 하는 요구 사항에 대한 방향을 제시하는 CPS 요구 사항 모델에 대한 연구를 하고 제안된 모델을 운송 시스템에 적용한 연구가 있다.[102] 이 연구에서 CPS 요구사항 중 가장 중요한 것 중에 하나는 시간에 관련된 것이라고 했다. 제안된 모델은 업무 분석, 시스템 요구사항 수집, 요구사항 충돌 제

99) Paolo Bellavista, et all.(2014), SOFTWARE ENGINEERING Key Enabler for Innovation, NESSI White Paper, 2014.07.

100) Cristóbal Costa-Soria(2014), Software Engineering for Cyber-Physical Systems, EECA Cluster networking event, 2014.11.12.

101) Stefan Wiesner et all.(2016), Requirements Engineering for Cyber-Physical Systems Challenges in the Context of “Industrie 4.0”, HAL archives, 2016.10.26.

102) Md. Masudur Rahman and Naushin Nower(2017), Requirements Model for Cyber-Physical System, Cornell University Library(https://arxiv.org/)

거, 기능과 비기능 요구사항 정의, 요구사항 분류의 순서로 요구사항 분석을 수행한다. 비기능 요구사항은 신뢰성, 성능, 시간요소 등이다.

국내에서 시행된 연구 중 실시간 협업 CPS의 비기능 요구사항에 대한 연구가 있다. 이 연구에서는 재난 구호 지능 로봇, 자율주행 자동차 등 CPS에 탑재하는 임베디드 소프트웨어를 개발할 때 식별되어야 하는 비기능적 요구사항에 대해 정의하고 있다. 비기능적 요구사항을 ISO/IEC 25010 기준으로 기능적합성, 신뢰성, 운영성, 호환성, 이식성, 안전성 등으로 식별했다. 비기능적 요구사항으로 기능적합성, 신뢰성, 복귀성, 자율 상태 전이 등을 정의하였다.[103]

3) CPS 플랫폼 연구

CPS는 여러 도메인에 적용되기 때문에 공통적인 소프트웨어 플랫폼을 개발하기 보다는 개별적인 도메인에 적합한 소프트웨어 플랫폼을 개발하여 사용하는 것이 실용적이고 현실적인 방법이다. 그러나 CPS라는 개념으로 공통적인 특징을 찾아내고, 정의하여 추상화한 모델을 제공하는 것은 CPS 소프트웨어 연구와 실용 개발에 필요한 작업이다. 이를 위해 NIST(National Institute of Standards and Technology)도 CPS 프레임워크[104]를 정의하고 CPS 연구를 위한 공통 언어와 연구방법론을 제시하고 있다.

CPS 소프트웨어 플랫폼을 개발하려는 적극적인 시도도 있다. 유럽 ARTEMIS 프로젝트의 '개방형 CPS 플랫폼(OCPSP, Open CPS Platforms)' [105]은 특정 환경, 구성 요소 및 도구를 제공할 수 있는 교차 도메인 솔루션이다.[106]

OCPSP는 유럽 프레임워크 프로그램7 (FP7)의 CyPhERS (Cyber-Physical European Roadmap and Strategy, 유럽 CPS 로드맵과 전략) 프로젝트에서 CPS 구조도와 플랫폼을 필요성에 대해 인식하여 제작한 플랫폼이고, 지속적으로 발전되고 있다. OCPSP는 보안, 인간과 기계의 인터페이스, 논리 요소 등 CPS 공통요소, 데이터 관리, 하드웨어와 소프트웨어 미들웨어와 통신을 클라우드를 이용히여 각각의 스마트 서비스에 제공한다.

---

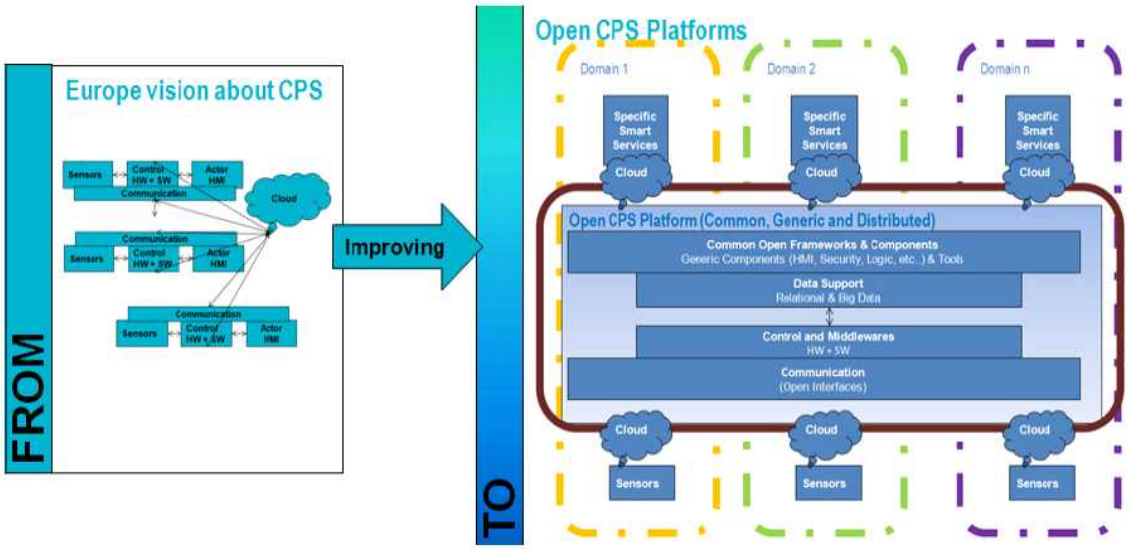
103) 남승우(2017), 실시간 협업의 지능형 CPS 개발을 위한 소프트웨어의 비기능 요구사항 식별, 2017년 한국컴퓨터종합학술대회 논문집 pp543-545, 2017.6.

104) NIST CPS PWG(Cyber Physical Systems Public Working Group, 2016), Framework for Cyber-Physical Systems Release 1.0 , 2016.5.

105) https://www.eurocps.org/eurocps-platforms/

106) Jesús Angel García Sánchez, Jorge J. Rodríguez Vázquez(2014), Open CPS Platforms, CPS20: CPS 20 years from now visions and challenges workshop, 2014.4.

[그림 2-20] CPS 플랫폼



자료 : Jesús Angel García Sánchez, Jorge J. Rodríguez Vázquez(2014), Open CPS Platforms, CPS20: CPS 20 years from now visions and challenges workshop, 2014.4.

CPS를 위한 새로운 소프트웨어와 하드웨어 아키텍처를 연구하는 것을 목표로 하는 AXIOM[107] 프로젝트 (Agile, eXtensible, 고속 I/O 모듈)가 있다. 이러한 시스템의 목표는 실시간으로 반응하고, 할당된 작업에 충분한 연산 능력을 제공하고, 그러한 작업을 위한 가능한 최소한의 에너지를 소비하는 것이다. 추가로 확장성, 프로그래밍 용이성, 비용 효율성 등도 고려한다. 물리세계와의 연결은 아두이노를 이용하며, 응용 어플리케이션 영역은 스마트홈, 스마트빌딩에서 적용을 시도하고 있다.[108] AXIOM 프로젝트는 공통적인 CPS 플랫폼의 개발보다는 실용적인 스마트홈이나 스마트빌딩에 적합한 플랫폼을 개발하는 것이 목표이다.

대부분의 CPS 플랫폼 연구는 전체 시스템에 대한 연구하기보다는 CPS 부분적인 요소나 모니터링, 시뮬레이션, 테스트 등 CPS 관련 도구에 대한 연구가 주로 시행되었다. ITEA[109]의 Flex4Apps[110]는 클라우드, 통신 및 IoT 인프라의 통합 추세의 결과 복잡해진 시스템의 유연한 모니터링 및 최적화 방법을 지원하기 위해 고안한 CPS 플랫폼이다. 대규모의 분산

107) http://www.axiom-project.eu/, 2015년에 시작한 Foundation for Research and Technology - Hellas (FORTH), 이탈리아 Siena 대학등이 수행하는 프로젝트, EU H2020 program, 스페인 정부 등이 지원,
108) Carlos Álvarez, Eduard Ayguadé, Jaume Bosch et al(2016), The AXIOM software layers, Microprocessors and Microsystems 47 (2016), pp262-277
109) ITEA는 Software-intensive Systems & Services (SiSS) 분야에서 R&D 프로젝트를 지원하는 EUREKA 클러스터 프로그램. EURELA는 1987년 유럽의 첨단기술 역량을 높이기 위해 설립한 유럽첨단기술 공동연구기구로 현재는 40개국 이상이 참가
110) ITEA 3 의 CPS 플랫폼, Platform for Application and Infrastructure Flexibility in Cyber-Physical Systems

된 사이버 물리적 시스템을 모니터링하고 최적화하기 위해 가능한 한 관리되는 시스템을 방해하지 않으면서 높은 데이터 볼륨과 시스템 모니터링의 복잡성을 관리하는 솔루션을 제공한다.[111] Flex4Apps는 CPS 플랫폼이라기보다는 CPS 모니터링 및 관리 시스템이라고 정의하는 것이 더 적합하겠다. CPS 시뮬레이션 플랫폼은 네트워크 시뮬레이터, 모델링 도구를 통합하고 시간 동기화 문제에 대한 해결책을 제시하였다.[112]

국내에서는 한국전자통신연구원이 다양한 산업 도메인의 대규모 자율제어형의 CPS가 공통으로 필요한 기술 요구사항을 만족하는 CPS 개발 및 검증 프레임워크(EcoSuite), CPS 통신 미들웨어(EDDS), CPS 자율제어 미들웨어(EcoACE)로 구성되어진 고신뢰 CPS 플랫폼을 개발하였다.[113] 여기서 대규모 자율제어형 CPS란 단위 CPS와 여러 무인시스템들이 연동하여 하나의 CPS를 구성하는 것을 말한다. 이 시스템은 앞으로 시스템의 확장성 보장, 클라우딩 환경 적용, 분산 시뮬레이션 적용 등이 필요하다.

CPS 응용도메인에 대한 플랫폼 연구로 Web 2.0 기반의 자유로운 구성이 가능한 단말 플랫폼, 서버 플랫폼, 저작도구로 구성된 로봇 플랫폼 구현에 대한 연구가 있다.[114]

CPS의 구성요소인 클라우드 플랫폼에 대한 연구도 있다. 구글이나 아마존 등은 이미 자신들의 플랫폼을 구축하여 서비스를 확장하고 있으며, GE나 지멘스 등은 스마트공장을 위한 클라우드 플랫폼을 구축하고 스마트공장 응용프래그램을 위한 서비스를 제공하고 있다. 오픈소스 커뮤니티들이 주도하는 오픈소스 클라우드 컴퓨팅 플랫폼 분석 연구에서는 서비스 모델, 호환성, 지원가상화 기술, 개발언어 등의 기준으로 플랫폼을 비교하고 있다.[115]

이러한 시도들은 물리세계의 수많은 데이터를 이용하여 사이버시스템이 합리적인 판단을 하고 물리세계를 제어하는 지능성, 신뢰성, 보안성, 실시간성을 가진 공통적인 CPS 플랫폼을 구축하기에는 아직 부족한 점이 많다.

앞에서 언급했듯이 CPS 공통 플랫폼의 개발보다는 특정 영역의 CPS 플랫폼 개발이 빠르고 실용적일 수는 있으나, CPS 응용 도메인에서 CPS 구축을 위해 CPS를 개념화하고, 기본 기능과 CPS 구현을 위한 개발, 검증도구를 제공하는 플랫폼을 구축하는 것도 CPS 발전을

111) https://itea3.org/project/flex4apps.html
112) Ahmad T. Al-Hammouri(2012), A comprehensive co-simulation platform for cyber-physical systems, Computer Communications 36 (2012) 8–19.
113) 김원태, 전인걸, 이수형, 박정민(2013), 대규모 자율제어 CPS 소프트웨어 플랫폼 기술, 정보과학회지 31(12), 2013.12., pp16-28
114) 이강희(2012), Web 2.0기반 유비쿼터스 소프트웨어 로봇 플랫폼의 구현, Journal of The Korea Society of Computer and Information July 2012
115) 조충기, 융희용(2015), 오픈소스 클라우드 컴퓨팅 플랫폼 분석 및 비교, 한국컴퓨터정보학회 동계학술대회 논문집 제23권 제1호,2015.1.

위해 필요하다 하겠다.

2. 임베디드 시스템과 IoT에서 요구공학 관련 선행연구

임베디드 시스템 개발에서는 비기능 요구사항에 대한 중요성을 강조하고 있다. G Karsai는 기존 소프트웨어 개발에서는 타이밍, 신뢰성 및 전력 소비와 같은 비기능 요구 사항이 부수적인 요소라면, 임베디드 소프트웨어 개발에서 이러한 비기능 요구사항은 기능 요구사항과 마찬가지로 중요한 역할을 한다고 했다.[116] Teade Punter는 비기능 요구사항은 시장 주도형 임베디드 소프트웨어 개발에서 중요한 성공 요인으로 인식되고 있다고 했다.[117] 뛰어난 기능만을 갖춘 제품으로는 충분하지 않으며 성공적인 제품을 만들기 위해서는 기능 및 비기능 특성을 모두 만족시켜야 한다. 하지만 전통적인 임베디드 시스템의 경우, 특정 기능을 수행하고 제어하는 소프트웨어가 하드웨어에 내장된 형태로 소프트웨어의 복잡도가 자율주행자동차나 스마트 공장과 같은 융・복합시스템에 비해 낮다고 볼 수 있으며 시스템 요구사항이 중요한 부분을 차지하고 있어 사용자와 시스템 간 상호작용에 관련된 요구사항 관련 연구는 다소 부족한 경향이 있다.

IoT에 대한 최근의 연구는 다음과 같다. John Stankovic은 IoT를 새로운 가치를 창출하기 위한 대규모 인프라로 사용하여 위한 빅데이터, 견고성, 개방성, 보안 같은 과제에 대해 정리하고, 다른 분야와의 기술 협력을 제안했다.[118] Ala Al-Fuqaha은 Constrained Application Protocol (CoAP), Data Distribution Service (DDS) 등과 같은 응용 프로그램 프로토콜, Multicast DNS (mDNS) 등 서비스 제공 프로토콜, BLE(Bluetooth Low Energy) 같은 기반 프로토콜 등 가장 관련이 있는 프로토콜을 정리했다. 또한 신뢰성, 이동성, 성능, 확장성, 보안 등 응용프로그램에서 고려되어야 하는 과제에 대한 방대한 요약을 제공한다.[119]

국내 IoT 시스템의 요구사항 관련 연구를 살펴본 결과, IoT 시스템에서의 보안성 관련 연구의 비중이 높으며, 그 이외에는 서로 다른 기기들 간의 상호운용성 확보를 위한 연구 등 특정 속성에 관련된 연구가 집중적으로 수행되고 있다.

---

116) G Karsai(2003), Model-Integrated Development of Embedded Software, Proceedings of the IEEE, 2003.01.

117) Teade Punter(2002), Evaluating Evolutionary Systems, Springer, 2002.12.

118) John A. Stankovic(2014), Research directions for the internet of things, IEEE Internet of Things Journal, vol. 1, no. 1, 2014.2.

119) A. Al-Fuqaha, M. Guizani, M. Mohammadi et all.(2015), Internet of things: A survey on enabling technologies, protocols and applications, IEEE Communications Communications Surveys & Tutorials, 2015.6.

<표 2-16> IoT 시스템의 보안성과 상호운용성 관련 연구

주제	연구 제목	저자 (연구년도)
보안성	사물인터넷 융합 서비스 보안 요구사항	강남희 (2015)[120]
	사물인터넷 보안 요구사항과 oneM2M 표준 보안 기술 분석	허신욱, 김호원 (2017)[121]
	IoT 서비스를 위한 보안	김동희, 윤석웅 (2013)[122]
	IoT 보안 요구사항에 대한 고찰	김다빈, 김경모 외 (2015)[123]
	IoT 기반 컨테이너 보안 장치 및 시스템 성능 평가	문영식, 최성필 외 (2013)[124]
	IoT 디바이스 보안 점검 기준	정용식 (2017)[125]
	A Study on Security Vulnerability Management in Electric Power Industry IoT	이상기 (2016)[126]
상호 운용성	시맨틱 기술을 활용한 글로벌 사물인터넷 상호연동 기술 개발 및 적용	황재영, 안종관 외 (2017)[127]
	IoT 환경에서 서비스 계층 상호운용성 연구	윤주상, 이태진 (2016)[128]
	IoT 상호운용성 및 서비스 개발	고영준, 김용진 (2015)[129]
	IoT 환경에서 상호운용성(Interusability) 개선을 위한 사물 개성(Personality of Things) 모델링 방법에 대한 연구	안미경, 박남춘 (2017)[130]

임베디드 시스템이나 IoT의 경우는 CPS에 비해 소프트웨어가 단순하기 때문에 요구공학에 대한 연구 필요성이 적었던 것으로 생각되며, 하드웨어와 소프트웨어의 융복합 시스템에서 요구사항 도출에 대한 직접적인 선행연구를 찾아보기 어려웠다.

120) 강남희(2015), 사물인터넷 융합 서비스 보안 요구사항, 한국통신학회지(정보와통신), pp.45-50.
121) 허신욱, 김호원(2017), 사물인터넷 보안 요구사항과 oneM2M 표준 보안 기술 분석, 정보과학회지, pp.16-22.
122) 김동희, 윤석웅, 이용필(2013), IoT 서비스를 위한 보안, 한국통신학회지(정보와통신), pp.53-59.
123) 김다빈, 김경모, 조진성(2015), IoT 보안 요구사항에 대한 고찰, 한국정보과학회 학술발표논문집, pp.1072-1074.
124) 문영식, 최성필, 이은규 외(2013), IoT 기반 컨테이너 보안 장치 및 시스템 성능 평가, 한국정보통신학회논문지, pp.2183-2190.
125) 정용식, 차재상(2017), IoT 디바이스 보안 점검 기준, 한국통신학회지(정보와통신), pp.27-33.
126) 이상기, 이세윤, 김정철(2016), A Study on Security Vulnerability Management in Electric Power Industry IoT, 한국디지털콘텐츠학회 논문지, pp.499-507.
127) 황재영, 안종관 외(2017), 시맨틱 기술을 활용한 글로벌 사물인터넷 상호연동 기술 개발 및 적용, 한국통신학회논문지, pp.2208-2216.
128) 윤주상, 이태진(2016), IoT 환경에서 서비스 계층 상호운용성 연구, 한국컴퓨터정보학회 학술발표논문집, pp.77-78.
129) 고영준, 김용진(2015), IoT 상호운용성 및 서비스 개발, 한국통신학회지(정보와통신), pp.31-35.
130) 안미경, 박남춘(2017), IoT 환경에서 상호운용성(Interusability) 개선을 위한 사물개성(Personality of Things) 모델링 방법에 대한 연구, 한국디자인학회 학술발표대회 논문집, pp.74-75.

제3장 CPS 요구사항 분석의 틀

요구사항은 사용자 요구사항과 시스템 요구사항으로 나눌 수 있다. 사용자 요구사항 분석의 틀로 NIST(National Institute of Standards and Technology, 미국 표준기술연구소)의 CPS 프레임워크를 활용하고자 하며, 시스템 요구사항 분석의 틀로 CPS 특징을 이용하고자 한다.

제1절 분석 대상 및 분석 방법

CPS 요구사항을 도출하기 위하여 자동차, 항공, 스마트공장을 선정했다. 각각의 도메인을 분석하기 위해서 연구 대상 도메인의 대표적인 소프트웨어 표준인 AUTOSAR (AUTomotive Open System ARchitecture, 개방형 자동차 표준 소프트웨어 구조), ARINC(Aeronautical Radio, Incorporated) 653, RAMI 4.0(Reference Architectural Model Industrie 4.0)의 스마트공장 소프트웨어 관련 표준 등을 분석하여 자동차, 항공, 스마트 공장에서 필요한 기능을 도출한다.

분석 방법으로는 사용자 요구사항 도출을 위하여 CPS 프레임워크의 측면(Aspects)과 관심사(Concerns)를 이용하여 자동차, 항공, 스마트공장의 요구사항을 분석한다. 시스템 요구사항 도출을 위해서는 CPS 특징인 **‘시스템 확장성(Scalability), 융복합성(Composability), 상호작용성(Interactivity), 고신뢰성(Dependability), 실시간성(Timing), 상호운용성(Interoperability), 지능성(Intelligence)’**를 기준으로 분석을 시행한다.

제2절 분석틀

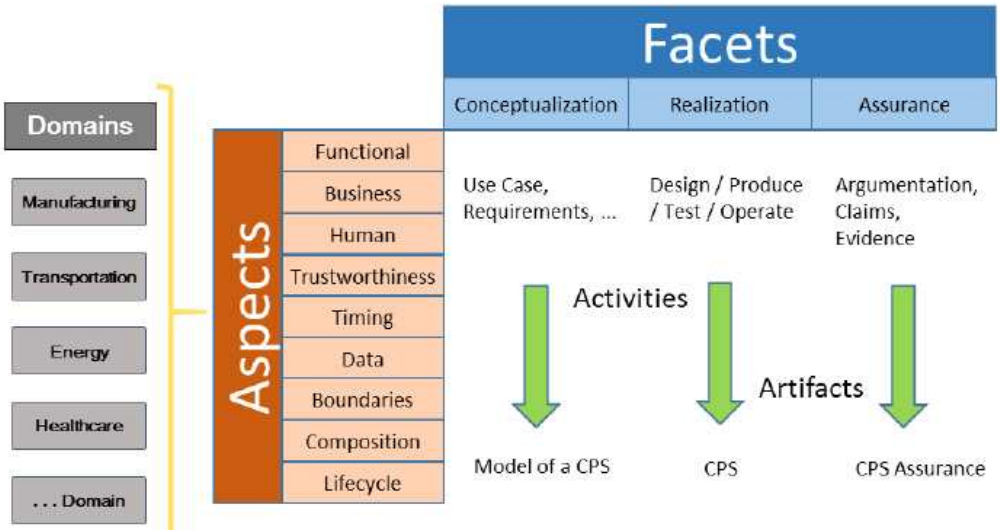
1. CPS 프레임워크를 이용한 사용자 요구사항 분석

CPS는 2장 제1절 CPS의 정의 및 범위에서 확인 했듯이 물리 세계와 사이버 세계의 특징을 동시에 가지고 있고 여러 시스템이 융합된 시스템 위의 시스템(System of system, SoS)으로서 복잡한 기능적, 비기능적 요구사항을 가지고 있어, 요구사항의 추출, 분석, 명세가 쉽지 않다. 이 연구에서는 NIST의 CPS 프레임워크를 이용하여 CPS 프레임워크를 정의하고 CPS에 대해 분석하고자 한다.

NIST CPS 프레임워크는 사용자, 개발자, 아키텍처 등 시스템 개발 관련 이해당사자들 간의 의사소통을 위해 상위 개념, 개념들과의 관계, 관련 용어들을 제공한다. NIST가 CPS 프레임워크를 제작한 목표는 다양한 도메인에서 상호운용이 가능한 CPS 아키텍처를 설명하기 위한 공통 언어를 제공하여, CPS가 여러 도메인 간 상호운용이 가능한 시스템 위의 시스템(System of system, SoS)을 형성 할 수 있도록 하는 것이다. CPS 프레임워크는 응용 도메인에서 CPS 아키텍처를 이해하고 개발할 수 있는 개념적 도구 또는 메타 모형이다.

NIST CPS 프레임워크는 CPS 응용 도메인(Domain), 관심사들(Concerns), 관심사의 집합인 측면(Aspects), 상황들(Facets), 요구 사항, 설계, 테스트 및 검증을 포함하는 속성(Properties)이 포함되어 있다. CPS 구축을 위해서는 요구 사항 도출을 통한 모델링, 설계, 제작, 검사 등을 통한 CPS 구축, CPS의 검증의 작업이 필요하다.

[그림 3-1] NIST CPS 프레임워크 – 도메인, 상황들(Facets), 측면들(Aspects)



자료 : NIST CPS PWS(2016), Framework for Cyber-Physical Systems Release 1.0, 2016.5.

NIST의 CPS 프레임워크는 물리 세계와 사이버 세계로 구성되어진 전체 시스템을 대상으로 분석하는 틀이며, 이 연구에서는 CPS 소프트웨어 대해 집중하기 때문에 NIST의 CPS 프레임워크 중 소프트웨어와 직접적 관련이 없는 부분은 제외한다.

이 연구에 맞게 수정된 CPS 프레임워크를 이용하여 CPS 응용도메인인 자동차, 항공, 스마트공장 도메인의 요구사항을 도출한다. CPS 응용 도메인은 이론적 배경에서도 확인하였듯이 자동차, 항공, 스마트도시, 스마트 헬스케어, 스마트공장 등 다양하여, 이 연구에서는 분

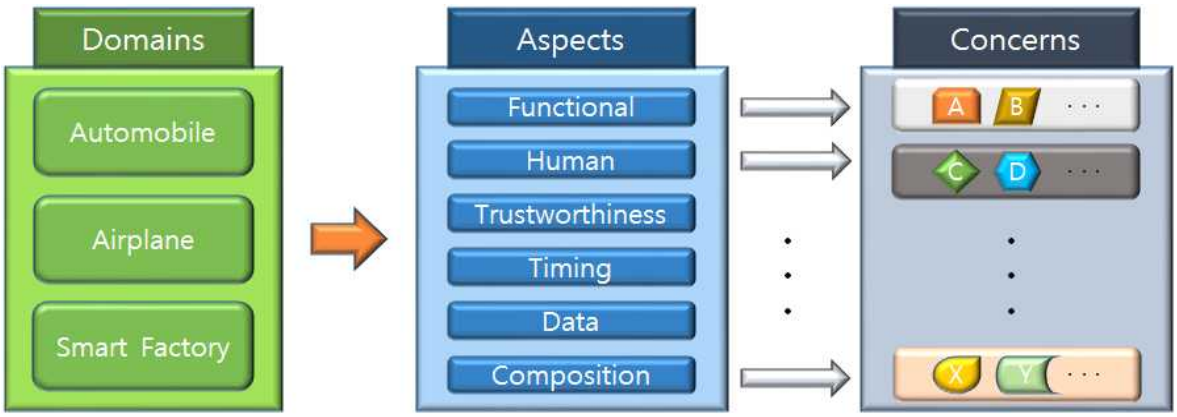
석 대상을 자동차, 항공, 스마트공장으로 제한하였다.

분석의 기본 도구인 측면(Aspect)들은 개념적으로 같거나 혹은 연관된 사용자 요구사항으로 구성되는 것으로 기능(functional), 사업 (Business), 인간(human), 신뢰성(trustworthiness), 시기(timing), 데이터(data), 경계(Boundaries), 조립(Composition), Lifecycle(제품주기)로 분류된다. 사업 (Business) 측면은 기업, 시장, 규제 등에 관한 것이며, 경계(Boundaries) 측면은 도메인 간 연동을 위해 고려할 점이다. Lifecycle(제품주기) 측면은 CPS 전체 시스템 개발에 관한 것이다. 사업(Business), 경계(Boundaries), Lifecycle(제품주기) 측면은 소프트웨어 개발과 직접적인 관계가 없거나, 물리세계와 사이버 세계의 전체 시스템 측면에서 고려해야 하므로 이 보고서에서는 고려 대상에서 제외한다.

NIST CPS 프레임워크 문서에서는 개발 프로세스 관점의 상황들(Facets) 개념도 다루고 있지만, 상황들(Facets)은 CPS개발의 단계 혹은 과정을 나타내는 개념이기 때문에 CPS 요구사항 분석이 주제인 이 보고서에서는 다루지 않기로 한다.

본 연구에 활용되는 측면들은 기능(functional), 인간(human), 신뢰성(trustworthiness), 시기(timing), 데이터(data), 조립(Composition)의 6개이다.

[그림 3-2] CPS 프레임워크 - 도메인들, 측면들, 관심사들(Domain, Aspects, Concerns)



자료 : NIST CPS PW(2016), Framework for Cyber-Physical Systems Release 1.0에서
본보고서에 활용되는 요소를 기반으로 편집

다음은 이 보고서에서 다루는 6개의 각 측면들과 각 측면에 관련된 관심사들(Concerns)에 대해 정리한다.

<표 3-1> CPS 프레임워크의 6가지 측면들에 대한 목록과 설명

측면(Aspect)	설명
Functional	센싱, 작동(Actuation), 제어, 통신, 물리적인 것 등의 기능에 관한 13개 관심사들의 집합. CPS의 기능과 기능 구현을 위해서 관리가능성, 측정가능성, 물리적 특성, 상태 등 고려할 점을 포함. 특히 불확실성(Uncertainty)은 물리적 세계를 CPS의 특징이 드러나는 관심사임
Human	인간의 CPS와 상호작용과 CPS의 부분으로서 상호 작용에 관한 관심사들의 집합
Trustworthiness	보안, 개인정보보호, 안전, 신뢰성, 복구가능성 등 CPS의 신뢰성에 관한 관심사들의 집합
Timing	시간과 주파수 신호들의 생성과 전송, 지연관리 등을 포함하는 CPS에서 시간과 빈도(주파수)에 관련된 관심사들의 집합
Data	퓨전, 메타데이터, 타입, 식별을 포함하는 데이터 상호작용에 관한 관심사들의 집합
Composition	구성 요소의 속성에서 구성 요소 어셈블리의 선택된 속성을 계산하는 기능에 관한 관심사들의 집합

관심사에 대해 면밀히 검토함으로써 각 측면의 특징에 대해 좀 더 자세히 설명하겠다.

다음은 각 측면이 포함하는 각각의 관심사에 대해 정리한다. 기능(functional) 측면은 제어, 통신, 관리, 성능, 센싱 등 13개의 관심사를 포함한다. CPS의 기능과 기능 구현을 위해서 관리가능성, 측정가능성, 물리적 특성, 상태 등 고려할 점을 포함한다. 특히 불확실성 관리가 포함되어 있어, CPS 구현에 있어서 향후 해결해야 하는 문제들을 남겨둔다. 이는 또한 CPS가 아직 불완전하고 해결해야 하는 과제가 있는 시스템임을 증명하는 점이기도 하다.

<표 3-2> Functional 측면(Aspect)의 관심사들(Concerns)

측면(Aspect)	관심사 (Concern)	설명
Functional	Actuation	어떠한 조작에 의해 물리적인 상태의 변화가 일어나는 것
	Communication	CPS 내부 혹은 CPS와 다른 것들의 정보 교환
	Controllability	물리적인 것의 특성을 제어하는 것으로 장소, 시간의 불확실성, 신뢰성, 보안성, 복잡석 등에 의해 고려한 점이 많음

	Functionality	CPS가 제공하는 기능들에 대한 것으로 물리적 상태 변화(Actuation), 통신(Communication), 제어(Controllability)는 다른 시스템에 비해 CPS의 특징적인 기능이기 때문에 따로 분류함
	Manageability	CPS 기능의 관리에 관한 것. 예를 들면 복잡한 CPS에서나 SoS에서의 시간 관리는 새로운 이슈가 되고 있음
	Measurability	CPS의 특성을 측정하는 것에 관한 것
	Monitorability	CPS의 상태나 그것의 동작들에 대한 인식을 얻거나 관리할 수 있는 것들에 대한 것으로 용이성과 신뢰성과 관련
	Performance	요구되는 목표를 만족시키는 능력에 관한 것
	Physical	CPS의 순수한 물리적 특성에 관한 것. 잠금장치, 안전을 포함
	Physical context	물리적인 위치(그리고 불확실성)에 관련한 원하는 동작 혹은 특정한 관찰을 이해하는 것에 관한 것
	Sensing	CPS의 기능을 수행하는데 있어 요구되는 상황인지에 관한 것
	States	CPS의 상태들과 관련된 것들. 예를 들어 CPS의 기능적인 상태는 같은 입력들에 대해 CPS의 다른 반응을 허용하는데 자주 사용됨. 즉 상태에 따라 입력은 같으나 반응이 다를 수 있음
	Uncertainty	불확실성의 영향을 관리하는 것은 CPS에서 근본적인 과제임. CPS에서 불확실성의 원인은 하드웨어 에러에 의한 통계학적(무작위), 지식부족, 소프트웨어 구현의 제한으로 인한 시스템적 불확실성 등임

여기서 Performance 에 대한 관심사는 응용소프트웨어에서 검토되는 ‘성능’이 아니라, CPS의 목적을 얼마나 달성했느냐에 대한 것으로 조금 다르게 해석되어 있어 분석 시 주의해야 한다. Uncertainty에 대한 관심사는 CPS의 특성상 아주 중요하게 다루어져야 한다. 왜냐하면 물리세계는 예측가능하지 않고, CPS는 조절 가능한 환경에서 동작하지 않기 때문에 하위 시스템 오작동에 관해서 처리가 가능해야하기 때문이다.

인간(human) 측면은 CPS에서 인간과 물리시스템에 대해 다룬다.

<표 3-3> Human 측면(Aspect)의 관심사들(Concerns)

측면(Aspect)	관심사 (Concern)	설명
Human	Human factors	CPS가 인간에 의해 어떻게 사용되는지와 관련된 CPS의 특성에 관한 것들
	Usability	CPS가 효과적이고 효율적으로 기능적인 목표와 사용자 만족을 이루는데 사용할 수 있는지에 대한 것

인간도 CPS의 한 부분이며, 인간의 역할이 중요하기 때문에 CPS에서 인간과 시스템의 관계는 간과할 수 없다. 여기서는 인간이 CPS를 활용하여 하는 작업과, 복잡한 시스템인 CPS를 학습하고 반응하여 사용하는 문제에 대해 다루며, 이는 CPS의 안전과 활용성과도 관련이 깊은 부분이다.

신뢰성(trustworthiness) 측면은 보안, 신뢰성, 복구가능성(회복가능성), 안전성, 개인정보 등 5개의 관심사에 관한 것이다.

<표 3-4> Trustworthiness 측면(Aspect)의 관심사들(Concerns)

측면(Aspect)	관심사 (Concern)	설명
Trustworthiness	Privacy	사람이나 기계로부터 CPS에 저장, 생성, 또는 전송되는 데이터에 대한 액세스 권한을 획득하는 것을 막거나 또는, 개인이나 단체가 다른 사람들로부터 자신들의 정보를 보호하는 것과 관련된 것
	Reliability	CPS가 예상 조건에서 안정적이고 예측 가능한 성능을 제공할 수 있는 능력과 관련된 것
	Resilience	CPS가 불안정하고 예기치 않은 상황, 정상적으로 예측 가능하지만 성능이 떨어질 수 있는 부분을 견뎌낼 수 있는지에 대한 것
	Safety	CPS 이해 관계자의 생명・건강・재산, 데이터와 실제 환경에 큰 재앙이 없음을 보장하기 위한 CPS의 능력과 관련된 것
	Security	CPS의 모든 프로세스, 메커니즘, 물리・사이버와 서비스가 의도하지 않은 무단 엑세스, 변경, 손상, 파괴 또는 사용됨으로부터 내부나 외부를 보호하는 능력과 관련된 것들

보안, 신뢰성, 복구가능성, 개인정보 등은 기존의 전통적인 시스템에서도 검토되는 관심사였으나, 안전은 전통 산업과 소프트웨어가 융복합화되면서 더욱 중요시 되는 관심사이다.

시기(timing) 측면은 실시간성 관련한 특성을 다루는 부분이다.

<표 3-5> Timing 측면(Aspect)의 관심사들(Concerns)

측면 (Aspect)	관심사(Concern)	설명
Timing	Logical time	상황이 발생하거나(인과적인 순서 관계) 이벤트가 일어나는 순서에 대한 것들
	Synchronization	모든 연관된 노드들이 요구되는 정확도로 같은 시간 내의 시간 신호 동기화에 관한 것
	Time awareness	설계에 의해 허용되는 시간의 정확성에 관련된 것들. 묘사, 분석 그리고 CPS 설계에 사용되는 모델에서와 실제 구성 요소의 동작에서 명시적으로 사용되는 시간의 유무
	Time-interval and latency	한 쌍의 이벤트들 간의 시간 간격들을 위한 요구사항을 포함하는 시간 요구사항을 명시하는 것

시간과 관련된 관심사는 프로세스 상의 순서도 중요하지만, 여러 시스템이 연동되기 때문에 여러 시스템 간의 시간 동기화와 동일한 시간에 작동하지 않지만 일정 시간 내에 여러 시스템이 작동하는 것에 대한 고려가 중요하다. 여러 시스템이 실시간으로 작동하는 것이 가장 이상적인 것이나, 물리적환경이나 시스템적 제한이 있기 때문에 여러 시스템이 원하는 기능을 달성하기 위해서는 목적 시간 내에 달성하도록 구축하는 것이 필요하다.

데이터(data) 측면은 정보의 의미, 인식, 관리, 관련성, 양, 속도의 6개 관심사에 관한 것이다.

<표 3-6> Data 측면(Aspect)의 관심사들(Concerns)

측면(Aspect)	관심사 (Concern)	설명
Data	Data semantic	시스템 내에서 유지, 생성, 전달되는 동의되고 공유된 데이터의 의미와 관련된 것
	Identity	CPS와 상호작용하거나 CPS를 활용할 때, 독립체(사람, 기계 및 데이터)를 정확하게 인식할 수 있는 능력과 관련된 것
	Operations on	시스템 데이터를 생성, 읽기, 변경, 삭제하는 능력과

	data	CPS 데이터 무결성 구현에 관한 것
	Relationship between data	데이터 집합의 구성 방법, 필요성, 데이터 관련성에 관한 것과 그러한 데이터 집합에서 파생될 수 있는 가치 또는 위험과 관련된 것
	Data velocity	데이터 동작의 실행 속도와 관련된 것
	Data volume	CPS 동작과 연관되는 데이터의 양에 관련된 것

CPS에서 사용되는 데이터는 여러 센서를 통하여 자동으로 생성되는 데이터로 빅데이터가 될 수 있으며, 빅데이터의 경우는 데이터의 양과 속도에 대한 고려가 필요하다. 의미나, 관리, 관련성 등은 기존 데이터 관리 시스템에서 다루면 관심사이다.

조립(Composition) 측면은 적응성, 복잡성, 결합성, 확인이 가능한 용이성에 대해 고려한다.

<표 3-7> Composition 측면(Aspect)의 관심사들(Concerns)

측면(Aspect)	관심사 (Concern)	설명
Composition	Adaptability	새로운 조건, 요구사항 또는 목표를 충족시키기 위해 CPS가 개선 또는 재구성되어 외부 조건이 변할 때, 의도된 목적을 달성하기 위한 CPS의 능력과 관련된 것들
	Complexity	기존 컴포넌트와 다양한 인터페이스 등의 구성 요소들 간의 상호작용의 이종성과 다양성 때문에 필요한 CPS 동작에 대한 이해와 관련된 것들
	Constructivity	CPS 모듈러 구성요소들(하드웨어, 소프트웨어 및 데이터)을 결합하여 사용자 요구 사항을 충족시키는 기능과 관련된 것들
	Discoverability	독립체(사람, 기계)가 구성 요소의 기능을 활용하기 위해 CPS 구성 요소를 관찰하고 이해할 수 있는 용이성과 신뢰성에 관련된 것들

CPS는 물리시스템과 사이버시스템이 융복합된 복잡한 시스템으로 CPS 구성요소들의 조합을 위해서는 각기 요소들의 기능을 이해하고, 상호작용이 가능하게 요소들을 조합할 필요가 있다. 또한 여러 요소들로 구성되기 때문에 각각의 시스템을 주기적으로 개선되고 변경되어 CPS는 이러한 상황을 모두 허용해야 한다.

속성의 구성요소인 요구사항은 여러 다른 이해 관계자(stakeholder)들에 의해서 표현되어

지는 그들의 독특하고 공동의 관점들에서 표현되어지는 것들이며, CPS 프레임워크를 기반으로 하는 연구방법론을 이끄는 핵심적인 기본 항목이다. 이 사용자 요구사항들은 개발, 검증 등의 CPS개발 과정에서 계속적으로 활용되어진다. 요구사항은 올바르고, 모호하지 않고, 일관성이 있게 도출되어 되어야 하지만, CPS가 여러 시스템이 융복합된 시스템이기 때문에 용이하지 않은 작업이다. 이 보고서의 목적의 첫 번째가 목적하는 CPS의 요구사항 도출을 위한 기본적인 요구사항을 도출하는 것이며, CPS 프레임워크는 이 작업을 위해 활용된다.

지금까지 정리한 CPS 프레임워크의 6가지 측면과 각 측면들의 관심사를 이용하여 자동차, 항공, 스마트공장의 사용자 요구사항을 추출하고 분류한다.

2. CPS 특성에 따른 시스템 요구사항 분석

제2장 1절에서 CPS의 특성은 **‘시스템 확장성(Scalability), 융복합성(Composability), 상호작용성(Interactivity), 고신뢰성(Dependability), 실시간성(Timing), 상호운용성(Interoperability), 지능성(Intelligence)’** 라고 정리하였다. 도출된 CPS 특성은 CPS가 목적한 바를 이루기 위한 기본적인 기능으로 자동차, 항공, 스마트도시, 스마트 헬스케어를 구축하기 위해서는 반듯이 제공되어야 하는 것이다. CPS 시스템 요구사항을 확장성, 융복합성 등 7가지 특성에 따라 도출, 분류하고, CPS 플레임워크에 의해 도출된 사용자 요구사항과 상호 연결시켜 요구사항을 추적한다.

CPS 프레임워크를 이용하여 추출한 자동차, 항공, 스마트 공장의 사용자 요구사항은 확장성, 융복합성 등 7가지 CPS 특성에 따라 재분류하고, 자동차, 항공, 스마트 공장의 시스템 요구사항과 연결시킨다.

이와 같이 도출된 시스템 요구사항은 CPS 플랫폼이 제공해야할 기본 기능으로 고려하고, CPS 플랫폼 구조 설계에 이용된다.

제4장 CPS 사용자 요구사항 분석 및 도출

이 장에서는 미국의 NIST의 CPS 프레임워크[131]를 활용하여 정의된 CPS 프레임워크를 이용하여, CPS 분석의 대상 도메인으로 선정한 자동차, 항공기, 스마트공장을 분석하려고 한다. NIST CPS 프레임워크는 CPS 분석 연구 방법론과 그것을 설명하는 용어를 포함하고 있으며, 이는 CPS에 대한 보편적인 분석을 가능하게 한다. 구체적인 도메인에서 CPS 프레임워크를 이용하여, 각각의 기능을 추출하여 CPS의 공통 기능을 추출하고자 한다.

본보고서에서 다루는 CPS 프레임워크를 구성하는 주요 요소들은 도메인들(Domains), 측면들(Facets), 관심사(Facets), 요구사항(Requirements) 이다.

먼저 도메인(Domains)은 CPS의 여러 응용 영역들을 의미하며 분석대상은 자동차, 항공기, 스마트공장으로 한정한다.

이 보고서에서 사용하는 측면들은 기능(functional), 인간(human), 신뢰성(trustworthiness), 시기(timing), 데이터(data), 조립(Composition)의 6개이며, 각 측면들에 연관된 관심사에 대해서는 제3장에서 정리하였다.

CPS 프레임워크의 측면과 관심사를 이용하여 자동차, 항공, 스마트공장의 요구사항을 분석한다. 각각의 도메인을 분석하기 위해서 연구 대상 도메인의 대표적인 소프트웨어 표준인 AUTOSAR(AUTomotive Open System ARchitecture, 개방형 자동차 표준 소프트웨어 구조), ARINC(Aeronautical Radio, Incorporated) 653, RAMI 4.0(Reference Architectural Model Industrie 4.0)의 스마트공장 소프트웨어 관련 표준 등을 분석하여 자동차, 항공, 스마트 공장에서 필요한 기능을 도출한다.

제1절 자동차 사용자 요구사항

기존의 자동차 시스템이 엔진, 바디, 샤시와 같은 물리기계적인 요소로 이루어진 시스템이었다면 현대와 미래의 자동차는 기존 물리기계적인 요소에 소프트웨어와 통신 등의 ICT 요소들이 점차 추가된 CPS로 발전되고 있다. 자율주행차, 커넥티드카 등의 다양한 명칭의 미래자동차는 이제 자동차가 운전자의 수동적인 조작에 의해서만이 아닌 라이다, 레이더, 초

131) NIST CPS PWG(Cyber Physical Systems Public Working Group, 2016), Framework for Cyber-Physical Systems Release 1.0 , 2016.5.

음파, 카메라 센서 등의 물리 시스템들에 의해 인지된 정보가 AI와 같은 지능적인 소프트웨어에 의해 처리 판단되어 운전자에게 많은 도움을 줄 뿐 아니라 궁극적으로는 자율주행할 수 있는 시스템으로 개발되고 있다.

이번 절에서는 NIST CPS 프레임워크 분석 방법에 기반을 두어 자동차의 측면들과 사용자 요구사항에 따른 세부 항목들을 설명한다. 자동차의 6가지 측면들과 사용자 요구사항들에 대한 세부 항목 도출을 위해서 자동차 소프트웨어 구조 표준인 AUTOSAR(AUTomotive Open System ARchitecture) 표준 문서의 3장 요구사항 추적(Requirements Tracing)과 4장의 세부 요구사항(Requirements Specification), 그리고 자율주행 자동차 관련 소프트웨어 기술 동향 논문[132]을 참조하였다.

1. 자동차 요구사항 분석 대상

자동차의 요구사항 분석을 위해 AUTOSAR(AUTomotive Open System ARchitecture) 표준과 자율주행차 관련 소프트웨어를 분석 대상으로 선정하였다.

AUTOSAR는 전 세계 여러 완성차 업체 및 부품업체들이 협력하여 공동 개발하고 표준화를 진행한 차량용 소프트웨어 플랫폼이다.[133] AUTOSAR는 차량용 소프트웨어의 재사용성, 확장성 및 호환성을 개선하고, 자동차 생산 비용 절감 및 새로운 자율주행자동차 기능 개발의 발판을 마련하는 것을 주요 목표로 한다. AUTOSAR는 중요한 시스템 함수(또는 기능) 표준화, 응용 소프트웨어 인터페이스 영역 간 경계 표준화, ECU(Electronic Control Unit,전자제어장치) 네트워크 상에서 유연한 통합, 증가된 제품과 절차 복잡성 통제, 비용 효율적인 규모 가변성, 전 수명 주기에 걸친 유지보수를 제공한다.

이를 위해 소프트웨어의 모듈화, 소프트웨어 계층 분리를 위한 API(application programming interface), 실행환경(Runtime Environment)을 통한 ECU(Electronic Control Unit)와 통신 기능의 기술이 포함되어 있다.[134] AUTOSAR는 크게 ECU에 관련되는 기본 소프트웨어 (Basic Software, BSW), 실행환경(Runtime Environment, RTE), ECU로부터 독립적인 응용 소프트웨어 (Application Software, ASW)로 논리적으로 나뉘어 있다. 기본 설계는 RTE 개념을 도입하여 응용 소프트웨어와 하드웨어관련 소프트웨어인 BSW를 분리함으로써,

---

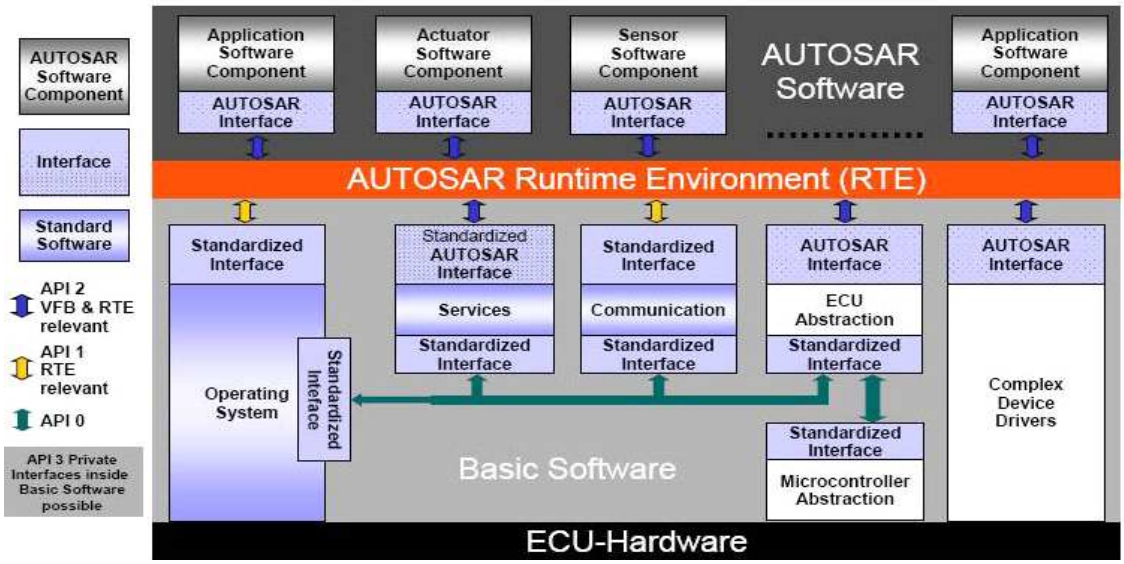
132) 장승주(2016), 자율주행 자동차 관련 SW기술 동향, 한국통신학회지(정보와통신), 33(4),27-33

133) 2005년 AUTOSAR 표준을 처음으로 1.0버전 릴리즈를 배포한 이후 계속된 버전 업데이트를 진행하여 2016년에는 4.3버전 릴리즈를 배포

134) 위키피디아, https://ko.wikipedia.org/wiki/AUTOSAR

하드웨어에 독립적인 응용 서비스를 개발할 수 있도록 하는 것이다.

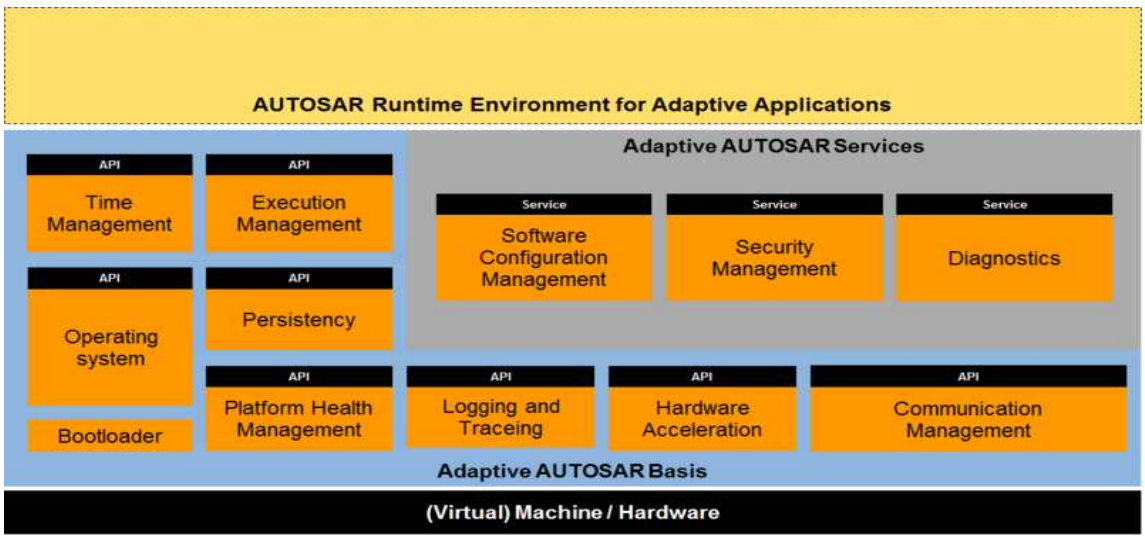
[그림 4-1] AUTOSAR 기반 소프트웨어와 ECU 동작 예시



자료 : AUTOSAR

AUTOSAR Adaptive Platform은 고성능 자율주행과 같은 기능들을 구성하는 고성능 컴퓨팅 ECU들을 위한 2017년에 새로 발표한 AUTOSAR의 솔루션이다.[135]

[그림 4-2] AUTOSAR Adaptive Platform 구조



자료 :AUTOSAR

135) 2017년 3월 첫 AUTOSAR Adaptive Platform 표준 릴리즈

다음은 AUTOSAR 요구사항 리스트로 그 중 CPS의 6가지 측면과 관련되지 않은 기능은 제외했다.

<표 4-1> AUTOSAR 요구사항 리스트

번호	요구사항	설명	해석
1	RS_Main_00010	AUTOSAR shall support the development of safety related systems.	안전 시스템 지원
2	RS_Main_00011	AUTOSAR shall support the development of reliable systems	신뢰 시스템 지원
3	RS_Main_00030	AUTOSAR shall support development processes for safety related systems	안전 시스템 개발 프로세스 지원
4	RS_Main_00060	AUTOSAR shall provide a standardized software interface for communication between Applications	표준화 제공
5	RS_Main_00080	AUTOSAR shall provide means to describe a component model for Application Software	모듈화
6	RS_Main_00100	AUTOSAR shall provide standardized Basic Software	기본 소프트웨어 표준화
7	RS_Main_00120	AUTOSAR shall provide means to assure interoperability of AUTOSAR implementations on application level (RTE) and bus level.	상호운영성 보장
8	RS_Main_00130	AUTOSAR shall provide an abstraction from hardware	하드웨어와 분리
9	RS_Main_00140	AUTOSAR shall provide network independent communication mechanisms for applications	소프트웨어별 통신 독립
10	RS_Main_00150	AUTOSAR shall support the deployment and reallocation of AUTOSAR Application Software	소프트웨어 이행 지원
11	RS_Main_00160	AUTOSAR shall provide means to describe interfaces of the entire system.	전체 시스템과 상호작용
12	RS_Main_00170	AUTOSAR shall provide secure access to ECU	보안성
13	RS_Main_00180	AUTOSAR shall provide mechanisms to protect intellectual property in a shared development process	지적 자산 보호
14	RS_Main_00190	AUTOSAR shall support interoperability with non-AUTOSAR software on the same ECU	상호운용성
15	RS_Main_00200	AUTOSAR specifications shall allow resource efficient implementations	효율성
18	RS_Main_00230	AUTOSAR shall support network topologies including gateways	네트워크 상호운영성 추가
19	RS_Main_00250	AUTOSAR methodology shall provide a predefinition of typical roles and activities in work-share model	개발 시 역할 정의
20	RS_Main_00251	AUTOSAR methodology shall support roles and rights in a work-share model	개발 시 역할 지원
21	RS_Main_00260	AUTOSAR shall provide diagnostics means during runtime, for production and services purposes	진단 수단 제공

22	RS_Main_00270	AUTOSAR shall provide mitigation strategies towards new releases	업그레이드 지원
23	RS_Main_00280	AUTOSAR shall provide standardized communication interfaces for the onboard data exchange between ECUs with different software platforms	통신 표준화
24	RS_Main_00290	AUTOSAR shall support the verification of its specifications	검증 신뢰성 추가
25	RS_Main_00300	AUTOSAR shall provide data exchange formats to support work-share in large inter and intra company development groups	개발 시 정보 교환 양식 제공
26	RS_Main_00310	AUTOSAR shall support hierarchical Application Software design methods	모듈화
30	RS_Main_00350	AUTOSAR specifications shall be analyzable and support according methods to demonstrate the achievement of safety related properties.	안전 검증
31	RS_Main_00360	AUTOSAR shall support management of vehicle diversity	차량 다양성 관리 지원
32	RS_Main_00400	AUTOSAR shall provide a layered software architecture	계층화
33	RS_Main_00410	AUTOSAR shall provide specifications for routines commonly used by Application Software to support sharing and optimization	최적화를 위한 공통 모듈
34	RS_Main_00420	AUTOSAR shall use established software standards and consolidate de-facto standards for basic software functionality	기능성 제공 표준
35	RS_Main_00430	AUTOSAR shall support established automotive communication standards	자동차 통신표준 지원
36	RS_Main_00435	AUTOSAR shall support automotive microcontrollers	마이크로콘트롤러 지원
37	RS_Main_00440	AUTOSAR shall standardize access to non-volatile memory	비휘발성메모리 접근 표준화
38	RS_Main_00450	AUTOSAR shall standardize access to general purpose I/O	범용 I/O 접근 표준화
39	RS_Main_00460	AUTOSAR shall standardize methods to organize mode management on Application, ECU and System level	관리 표준화
40	RS_Main_00480	AUTOSAR shall support the test of implementations	테스트 지원
43	RS_Main_00510	AUTOSAR shall support secure onboard communication	보안성
44	RS_Main_00514	AUTOSAR shall support the development of secure systems	보안성

자료 : AUTOSAR(2016), Requirements on AUTOSAR Features, AUTOSAR CP Release 4.3.0, 2016.11.30.

다음에서 AUTOSAR 기본 요구사항을 기능적 측면, 인간적 측면, 신뢰성 측면, 실시간 측면, 데이터 측면, 그리고 조립 측면으로 분류 정리한다.

2. CPS 프레임워크의 측면 관점에서 자동차 요구사항 명세

자동차의 Functional 측면(Aspect)은 AUTOSAR 표준과 자율주행 관련 기술 동향 논문들에서 설명하는 자동차가 갖추어야 할 기능을 서술하고 있다. 다음으로 Human 측면(Aspect)은 자동차는 사람 사이의 상호 작용과 자동차 간 상호작용을 위해 갖추어야 할 기능을 서술하고 있다. Trustworthiness 측면(Aspect)은 자동차에서 신뢰성이 있는 동작을 위해 갖추어야 할 기능에 대해 서술하고 있다. Timing 측면(Aspect)은 자동차의 기능을 수행하는데 요구되는 시간과 관련이 있는 기능에 대해 서술하였다. Data 측면(Aspect)은 자동차에서 송·수신되는 데이터들이 갖추어야 할 기능에 대해서 서술하고 있다. 마지막으로 composition 측면(Aspect)은 CPS에 있는 구성요소들 간의 원활한 동작을 위해 필요한 기능에 대해 서술하였다.

자동차의 사용자 요구사항은 기능적 측면 25종, 인간적 측면 7종, 신뢰성 측면 12종, 실시간 측면 8종, 데이터 측면 7종, 그리고 조립 측면의 7종이 도출되어 총 66종이 도출되었다.

항목 ID는 Auto.XXXX.YYY.###로 표현된다. XXXX는 해당 측면(Aspect), YYY는 관심사(Concern), ###은 일렬 번호를 나타낸다. XXXX로 표현되는 측면은 Func은 기능적(Functional), Hum은 인간적(Human), Trust는 신뢰성(Trustworthiness), Timing은 실시간(Timing), Data는 데이터(Data), Comp는 조립(Composition)을 나타낸다.

1) 기능적(Functional) 측면

자동차의 기능적 측면은 자동차를 구성하는 요소들 즉, 예를 들면 센서들과 작동장치(Actutor) 그리고 각 자동차 파트(엔진/바디/샤시)의 ECU(Electronic Control Unit)들이 어떻게 동작(Actuation), 통신(Communication), 제어(Controllability), 기능 제공(Functionality), 측정(Measurability), 관리(Manageability), 감독/진단 (Monitorability), 성능(Performance), 물리적 특성(Physical), 센싱(Sensing)하는 지에 관련된 것들에 대한 항목들을 그룹화 한 것이다.

자동차의 기능적 측면의 세부 항목 도출을 위해서 자동차 소프트웨어 구조는 AUTOSAR Classic 표준의 RS Features 문서[136]의 3장 요구사항 추적(Requirements Tracing) 파트, 4장 요구사항 상세화 파트와 자율주행 자동차의 사용자 요구사항은 자율주행자동차 관련 SW기술 동향 논문[137]을 각각 참조하였다. 특히, 요구사항이 AUTOSAR 표준에 관련된 관심사

136) AUTOSAR(2016), Requirements on AUTOSAR Features, AUTOSAR CP Release 4.3.0, 2016.11.30.
137) 장승주(2016), 자율주행 자동차 관련 SW기술 동향, 한국통신학회지(정보와통신), 33(4),27-33

(Concern)인 경우, AUTOSAR Classic 표준의 RS Features 문서의 요구사항 항목들의 번호를 항목 ID아래에 괄호 안에 추가하였다. 기능적 측면의 요구사항은 총 25종이 도출되었다.

<표 4-2> 자동차의 Functional 측면의 관심사들(Concerns)과 요구사항들

측면(Aspect)	관심사(Concern)	항목 ID (Autosar 표준 ID)	설명
Functional	Actuation	Auto.Func.Act.001	자율주행 자동차의 제어 SW는 구동장치인 가속기, 감속기 및 조향장치를 작동할 수 있어야 함
		Auto.Func.Act.002	자율주행 자동차는 센서를 통해 주변 정보를 실시간으로 확인하여 위험 상황이라고 판단되는 경우 경보장치를 동작할 수 있어야 함
		Auto.Func.Act.003	자율주행 자동차의 V2X 시스템은 외부 시스템과 통신하여 차량의 운행을 지원해야 함
	Communication	Auto.Func.Comm.001 (RS_Main_00060)	자동차 소프트웨어 구조는 응용들 간의 통신을 위한 표준 소프트웨어 인터페이스를 제공해야 함
		Auto.Func.Comm.002 (RS_Main_00140)	자동차 소프트웨어 구조는 응용들을 위한 네트워크 독립적인 통신 메커니즘들을 제공해야 함
		Auto.Func.Comm.003 (RS_Main_00280)	자동차 소프트웨어 구조는 다른 소프트웨어 플랫폼을 가진 ECU들 간의 온보드 데이터 교환을 위한 표준 통신 인터페이스를 제공해야 함
		Auto.Func.Comm.004 (RS_Main_00430)	자동차 소프트웨어 구조는 인정받은 자동차 통신 표준들을 지원해야만 함
		Auto.Func.Comm.005	자율주행 자동차는 V2X통신을 통해 인프라 및 주변차량과 주행 정보 상호 교환
	Controllability	Auto.Func.Ctrl.001	자율주행 자동차는 조향, 가감속, 기어등의 액츄에이터를 지능적으로 제어할 수 있어야 함
		Auto.Func.Ctrl.002	자율주행 자동차는 V2X 통신을 통해서 차량의 이동을 제어해야 함
	Functionality	Auto.Func.Func.001 (RS_Main_00100)	자동차 소프트웨어 구조는 표준 기본 소프트웨어를 제공해야 함
		Auto.Func.Func.002 (RS_Main_00120)	자동차 소프트웨어 구조는 응용레벨 과 버스레벨에서 자동차 소프트웨어 구현들 간의 상호동작성을 확증하는 수단들을 제공해야 함
		Auto.Func.Func.003 (RS_Main_00410)	자동차 소프트웨어 구조는 공유와 최적화를 지원하기 위해 응용 소프트웨어에 의해 공통적으로

			사용되는 루틴들(routines)에 대한 표준을 제공해야 함
		Auto.Func.Func.004 (RS_Main_00420)	자동차 소프트웨어 구조는 기본 소프트웨어의 기능성을 위해 인정받은 소프트웨어 표준을 사용해야 함
		Auto.Func.Func.005 (RS_Main_00435)	자동차 소프트웨어 구조는 차량 마이크로 컨트롤러들을 지원해야 함
		Auto.Func.Func.006 (RS_Main_00440)	자동차 소프트웨어 구조는 비휘발성 메모리의 접근을 표준화해야함
		Auto.Func.Func.007 (RS_Main_00450)	자동차 소프트웨어 구조는 범용 I/O(GPIO)의 접근을 표준화해야함
		Auto.Func.Func.008 (RS_Main_00480)	자동차 소프트웨어 구조는 구현들의 테스트를 지원해야 만 함
	Measurability	Auto.Func.Meas.001	자율주행 자동차는 앞뒤차량간의 간격을 측정(종방향 속도 측정)할 수 있어야 함
	Manageability	Auto.Func.Mng.001 (RS_Main_00360)	자동차 소프트웨어 구조는 차량 다양성의 관리를 지원해야 함
		Auto.Func.Mng.002 (RS_Main_00460)	자동차 소프트웨어 구조는 응용, ECU 그리고 시스템 레벨에서의 모드 관리를 조직하기 위해 방법들을 표준화해야 함
	Monitorability	Auto.Func.Mon.001 (RS_Main_00260)	자동차 소프트웨어 구조는 생산과 서비스 목적들을 위해 실행 시간에 진단 수단들을 제공해야 함
	Performance	Auto.Func.Per.001 (RS_Main_00200)	자동차 소프트웨어 구조는 자원 효율적인 구현을 허용해야만 함
	Physical context	Auto.Func.PhyCon.001 (RS_Main_00130)	자동차 소프트웨어 구조는 하드웨어로부터 추상화를 제공해야만 함
	Sensing	Auto.Func.Sens.001	자율주행 자동차의 센서들은 주변 차량을 정확히 감지할 수 있어야 함

2) 인간적(Human) 측면

자동차의 인간적 측면은 자동차 요소들이 사람과 관련된 항목들을 그룹화 한 것으로서 사람이 자동차와 어떻게 관련이 있는지를 나타낸다. 예를 들어 자동차 소프트웨어 구조에서는 개발 방법론이 개발자에 의해서 어떻게 사용될 수 있는지를 정의하고, 자율주행자동차는 운전자에게 여러 운행에 대한 정보를 어떻게 제공하는지에 대한 요구 사항을 정의 한다.

자동차의 인간적 측면의 세부 항목 도출을 위해서 AUTOSAR Classic 표준의 RS Features 문서의 3장 요구사항 추적 중에서 자동차 소프트웨어 구조 방법론(methodology)과 자율주행 자동차 관련 SW기술 동향 논문의 운전자 관련 기술들을 참조하였다. 인간적 측면의 요구사항은 총 7종이 도출되었다.

<표 4-3> 자동차의 Human 측면의 관심사들(Concerns)과 요구사항들

측면(Aspect)	관심사(Concern)	항목 ID (Autosar 표준 ID)	설명
Human	Human factors	Auto.Hum.HF.001 (RS_Main_00250)	자동차 소프트웨어 구조 방법론은 일공유(work-share) 모델에서 전형적인 역할들과 활동들을 미리정의한 것들을 제공해야 함
		Auto.Hum.HF.002 (RS_Main_00251)	자동차 소프트웨어 구조 방법론은 work-share 모델에서 역할과 권리들을 제공해야만 함
		Auto.Hum.HF.003 (RS_Main_00300)	자동차 소프트웨어 구조는 큰 회사 개발 그룹들간 혹은 그룹내에서 work-share를 지원하기 위해 데이터 교환 포맷들을 제공해야만 함
		Auto.Hum.HF.004	자율주행 자동차는 HVI(Human Vehicle Interface)를 통해 운전자에게 경고 및 정보를 제공할 수 있어야 함
		Auto.Hum.HF.005	자율주행 자동차는 HVI를 통해 운전자로부터 명령을 입력받을 수 있어야 함
	Usability	Auto.Hum.Usa.001	자율주행 자동차는 HVI를 통해 운전자에게 운행하는 경로에 대한 정보를 명확하게 제공해야 함
		Auto.Hum.Usa.002	자율주행 자동차는 차량 내부에 장착된 운전자의 상태를 감시하는 영상 센서 또는 통합 센서로서 운전자의 졸음, 주의 태만, 음주 등의 상태를 실시간으로 확인하여 경보 장치로 알려줘야 함

3) 신뢰성(Trustworthiness) 측면

자동차의 신뢰성 측면은 개인정보 보호(Privacy), 신뢰성이 있는 시스템과 어떤 문제에도 빨리 회복하는 것(Resilience, 복원력), 주변 재해 등으로부터 보호하는 안전 관련된 사항(Safety), 보안이 철저한 접근(Security)을 제공하는 특징들로 그룹 되어 진다.

자동차의 신뢰성 측면의 세부 항목 도출을 위해서 자동차 소프트웨어 구조에서는 AUTOSAR Classic 표준의 RS Features 문서의 3장 요구사항 추적 중에서 개인정보(Privacy), 신뢰성(Reliability), 안전(Safety), 보안(Security) 관련 항목을 참조하였다. 복원력(Resilience)의 경우 4.2절 Operating System의 세부항목(RS_BRF_01248)을 참조하였다. 또한, 자율주행 자동

차의 신뢰성의 경우, 자동차 관련 SW기술 동향 논문의 능동 차량 제어 기술에서의 요구사항을 참조하였다. 신뢰성 측면의 요구사항은 총 12종이 도출되었다.

<표 4-4> 자동차의 Trustworthiness 측면의 관심사들(Concerns)과 요구사항들

측면(Aspect)	관심사(Concern)	항목 ID	설명
Trustworthiness	Privacy	Auto.Trust.Priv.001 (RS_Main_00180)	자동차 소프트웨어 구조는 하나의 공유 개발 프로세스에서 지적 자산을 보호하기 위한 메커니즘을 제공해야만 함
	Reliability	Auto.Trust.Rel.001 (RS_Main_00011)	자동차 소프트웨어 구조는 안정성 있는 시스템들의 개발을 지원해야만 함
		Auto.Trust.Rel.002	자율주행 자동차의 능동 차량 제어 기술에서 고신뢰성 이중 안전 센서 및 액츄에이터 개발이 필요함
	Resilience	Auto.Trust.Res.001 (RS_BRF_01248)	자동차 소프트웨어 구조 OS는 OS 응용의 종료와 재시작을 지원해야만 함
	Safety	Auto.Trust.Safe.001 (RS_Main_00010)	자동차 소프트웨어 구조는 안전 연관된 시스템들의 개발을 지원해야 함
		Auto.Trust.Safe.002 (RS_Main_00030)	자동차 소프트웨어 구조는 안전 연관된 시스템들의 개발 프로세스를 지원해야 함
		Auto.Trust.Safe.003 (RS_Main_00350)	자동차 소프트웨어 구조 표준들은 안전 연관된 특성들의 달성을 묘사하기 위해 분석되어지고 방법에 따라 지원해야 함
		Auto.Trust.Safe.004	자율주행 자동차의 능동 차량 제어 기술에서 기능 안전성 요구 사항을 기반으로 한 설계를 해야 함
		Auto.Trust.Safe.005	자율주행 자동차의 능동 차량 제어 기술에서 위험 분석 및 검증을 통한 고장 안전 기술이 필요함
	Security	Auto.Trust.Secu.001 (RS_Main_00170)	자동차 소프트웨어 구조는 ECU에 안전한 접근을 제공해야 함
		Auto.Trust.Secu.002 (RS_Main_00510)	자동차 소프트웨어 구조는 안전한 온보드 통신을 지원해야 함
		Auto.Trust.Secu.003 (RS_Main_00514)	자동차 소프트웨어 구조는 안전한 시스템들의 개발을 지원해야 함

4) 실시간(Timing) 측면

자동차의 실시간 측면은 자동차 소프트웨어 구조의 시간관련 요구사항과 그리고 자율주행 시스템이 어떤 기능을 수행하는데 있어서 실시간 통신을 지원하는데 필요한 항목들이나 지연 시간 요구사항들과 관련된 것들이다.

자동차 ECU 내부 혹은 ECU간 통신 요구사항은 AUTOSAR Classic 표준의 RS Features 문서의 4.2절 운영체제(Operating System), 4.4절 서비스(Services), 4.6절 버스를 통한 통신(Communication via Bus)파트를 참조하였고, 그 밖의 V2X[138]와 같은 자율주행을 위한 차량과 다른 시스템간의 실시간 통신 요구사항은 자율주행 자동차 관련 SW기술 동향 논문과 한국 전자 통신 연구원의 WAVE 기반 차량 안전 및 C-ITS 기술 동향 문서를 참조하였다. 실시간 측면의 요구사항은 총 8종이 도출되었다.

<표 4-5> 자동차의 Timing 측면의 관심사들(Concerns)과 요구사항들

측면(Aspect)	관심사(Concern)	항목 ID	설명
Timing	Managing timing and latency	Auto.Timing.Mng.001	자율주행 자동차의 차량 내 통신과 V2X통신에 있어서 메시지는 정해진 시간 내에 전송되어야 함
	Synchronization	Auto.Timing.Sync.001 (RS_BRF_01216)	자동차 소프트웨어 구조 OS는 외부 시간 소스로 스케줄 테이블을 동기화할 수 있는 인터페이스를 제공해야만 함
	Time Awareness	Auto.Timing.Aware.001	자율주행 자동차의 자율주행시스템은 실시간의 정확한 센서 정보를 바탕으로 경로를 생성해야함
		Auto.Timing.Aware.002	자율주행 자동차의 실시간 데이터 검색/수집/가공 기술을 지원해야 함
		Auto.Timing.Aware.003 (RS_BRF_01272)	자동차 소프트웨어 구조 OS는 소프트웨어 컴포넌트들의 시간 측정을 허용하는 기능을 제공해야 함
		Auto.Timing.Aware.004 (RS_BRF_01432)	자동차 소프트웨어 구조는 시스템 시간 서비스를 제공해야만 함
		Auto.Timing.Aware.005 (RS_BRF_01468)	자동차 소프트웨어 구조 서비스들은 상대 시간 측정을 위한 서비스를 제공해야만 함
	Time-interval and latency	Auto.Timing.Interval.001 (RS_BRF_01600)	자동차 소프트웨어 구조는 time-out처리를 지원해야만 함

138) 한국전자통신연구원(2014), WAVE 기반 차량 안전 및 C-ITS 기술 동향, 자동차IT융합연구실 융합 기술연구부문.

5) 데이터(Data) 측면

자동차의 데이터 측면은 자동차 소프트웨어의 데이터 포맷, 파일 관리, 데이터 동작(읽기, 쓰기, 생성, 삭제 등) 및 분류 관리 등에 관한 것이다.

자동차의 데이터 측면의 세부 항목 도출을 위해서 AUTOSAR Classic 표준의 RS Features 문서의 3장 요구사항 추적과 4.1절의 시스템과 구조(System and Architectures), 4.3절의 Runtime Environment(RTE) 그리고 4.8절의 메모리 스택(Memory Stack)을 참조하였다. 데이터 측면의 요구사항은 총 7종이 도출되었다.

<표 4-6> 자동차의 Data 측면의 관심사들(Concerns)과 요구사항들

측면(Aspect)	관심사(Concern)	항목 ID	설명
Data	Data semantic	Auto.Data.Sem.001 (RS_Main_00320)	자동차 소프트웨어 구조는 하나의 ECU에 있는 응용 소프트웨어를 통합하기 위해 모든 필요한 측면을 명시하기 위한 포맷들을 제공해야 함
		Auto.Data.Sem.002 (RS_Main_00300)	자동차 소프트웨어 구조는 거대한 회사 개발 그룹들 간 혹은 내에서 일공유를 지원하기 위한 데이터 교환 포맷들을 제공해야 함
	Operations on data	Auto.Data.OP.001 (RS_BRF_01816)	자동차 소프트웨어 구조 비휘발성 메모리 기능성은 논리적인 메모리 블록들에 기반하여 지속적인 데이터를 구성해야 함
		Auto.Data.OP.002 (RS_BRF_01808)	자동차 소프트웨어 구조 비휘발성 메모리 처리는 다른 종류의 메모리 하드웨어를 지원해야만 함
		Auto.Data.OP.003 (RS_BRF_01824)	자동차 소프트웨어 구조 비휘발성 메모리 기능성은 비휘발성 메모리에서 랜덤 액세스 메모리(RAM)로 매핑을 제공해야만 함
		Auto.Data.OP.004 (RS_BRF_01120)	자동차 소프트웨어 구조는 설정된 BSW(Basic Software) 데이터의 재플래쉬(re-flashing)을 지원해야만 함
		Auto.Data.OP.005 (RS_BRF_01352)	자동차 소프트웨어 구조 RTE는 직접 읽기/쓰기 데이터 접근과 실행가능한 파일이 호출되기전 미리 읽어오는 것과 실행가능한 파일이 리턴된 후 이후에 쓰는 것도 제공해야 함

6) 조립(Composition) 측면

자동차의 구성 측면은 자동차 소프트웨어의 계층적 구조 등 ECU와 같은 여러 자동차 요소들이 어떻게 연결되고 상호 동작될 수 있는지를 다룬다.

자동차의 구성 측면에 대한 세부항목 및 요구사항을 도출하기 위해서 AUTOSAR Classic 표준의 RS Features 문서의 3장 요구사항 추적을 참조하였다. 구성 측면의 요구사항은 총 7종이 도출되었다.

<표 4-7> 자동차의 Composition 측면의 관심사들(Concerns)과 요구사항들

측면(Aspect)	관심사(Concern)	항목 ID	설명
Composition	Adaptability	Auto.Comp.Adapt.001 (RS_Main_00270)	자동차 소프트웨어 구조는 새로운 릴리즈 버전들을 향하는 완화 전략을 제공해야 함
	Complexity	Auto.Comp.Complex.001 (RS_Main_00160)	자동차 소프트웨어 구조는 전체 시스템의 인터페이스들을 설명하는 수단들을 제공해야 함
		Auto.Comp.Complex.002 (RS_Main_00190)	자동차 소프트웨어 구조는 같은 ECU에서 다른 소프트웨어 구조와 상호 동작을 지원해야 함
	Constructivity	Auto.Comp.Const.001 (RS_Main_00080)	자동차 소프트웨어 구조는 응용 소프트웨어들을 위한 구성 모델을 묘사하는 수단을 제공해야 함
		Auto.Comp.Const.002 (RS_Main_00150)	자동차 소프트웨어 구조는 응용 소프트웨어의 전개 및 재배치를 지원해야 함
		Auto.Comp.Const.003 (RS_Main_00310)	자동차 소프트웨어 구조는 계층적인 응용 소프트웨어 설계 방법들을 지원해야 함
		Auto.Comp.Const.004 (RS_Main_00400)	자동차 소프트웨어는 계층적인 소프트웨어 구조를 제공해야 함

7) 자동차 사용자 요구사항에 대한 분석

기능적 측면에서 CPS의 특성을 나타내고 있는 관심사(Concern)는 Actuation, Controllability, Measurability, Sensing이다. 사이버 시스템이 물리 시스템을 제어하기 위해서는 물리 시스템을 측정하고, 물리시스템에 조작을 행하여 제어하는 기능이 필요하기 때문이다. 인간적 측면에서는 Usability 관심사로 분류되는 자율주행차와 운전자의 상호작용에 주의를 두어야 한다. 자율주행차는 상세한 시스템 기능을 분석하지 않았기 때문에 자율주행차에 대한 데이터 측면에 요구사항이 많이 도출되지 않았으나, 자율주행 기능과 안전성, 보안성 등의 특성을 보장하기 위해서는 데이터 측면에서 고려되어야 할 점이 많다고 판단된다.

제2절 항공기 사용자 요구사항

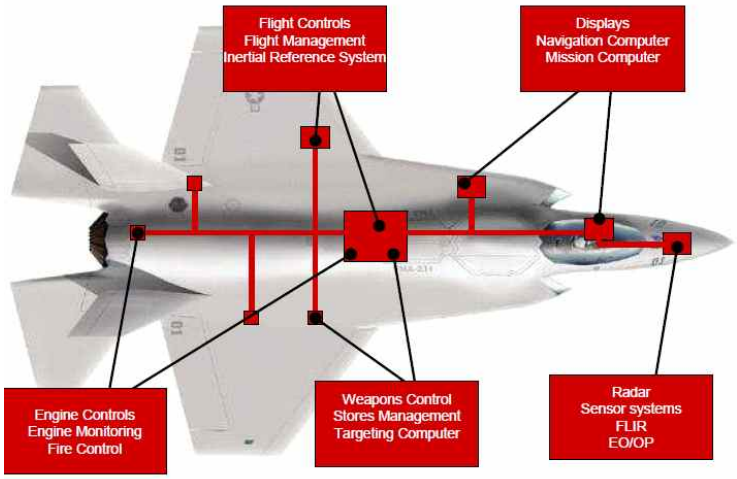
본 절에서는 NIST CPS 프레임워크를 재정의한 CPS 프레임워크를 이용하여 여러 CPS 도메인 중 하나인 항공기의 측면(Aspect), 관심사들(Concerns)과 요구사항들에 대해서 다룬다. CPS 프레임워크의 6개의 측면(Aspect)들과 각 측면(Aspect)마다 여러 관심사(Concern)들을 이용하여 항공기 요구사항 분석을 수행한다. 이 절에서는 기존 항공기의 소프트웨어 표준인 ARINC(Aeronautical Radio, Incorporated) 653[139] 중심으로 반영이 되었다.

1. 항공기 요구사항 분석 대상

항공기 요구사항 분석 대상으로 항공 소프트웨어 표준인 ARINC 653과 무인항공기 소프트웨어를 선정했다.

ARINC 653 플랫폼은 항공 전자에서 소프트웨어 증가로 인한 복잡성과 늘어나는 시스템으로 인한 SWap(Size, Weight and Power) 문제를 해결하기 위해 분산된 시스템들을 표준화하여, 하나의 시스템에서 비행제어, 엔진제어, 무기제어, 통신 등 여러 응용을 수행할 수 있는 통합 모듈형 구조의 항공 전자 시스템의 기술을 제공한다. 기존 구조는 각각의 기능을 하나의 컴퓨터시스템으로 구성하였다.

[그림 4-3] 통합 모듈형 구조



자료 : IEEE-CS Seminar(2008), ARINC 653, Courtesy of Wind River Inc., 2008.6.4.

139) An Avionics Standard for Safe, Partitioned Systems

즉, 항공 전자기기의 운영체제와 응용 프로그램 간의 인터페이스를 제공한다. ARINC 653은 여러 응용프로그램 간 오류의 전달을 방지하고 독립적인 수행을 보장하기 위한 파티셔닝(partitioning)[140]과, 오류가 발생했을 때 미리 정해진 오류 처리 루틴을 수행하기 위한 기능을 제공한다. 또한 신뢰성을 높이기 위한 51개의 기본 API를 비롯해 다양한 운영체제의 기능들을 제시하고 있다.[141]

2. CPS 프레임워크의 측면 관점에서 항공기 요구사항 명세

ARINC 653 플랫폼에서 제공하는 여러 요소들을 각 측면(Aspect)에 맞춰 관심사(Concern)들에 기준하여 분류했다. 또한, 일반적인 무인항공기의 특성들을 고려하며, 그 특성들에 맞춰서 해당되는 요소들을 ARINC 플랫폼과 마찬가지로, 측면(Aspect)들과 관심사(Concern)들로 분류하였다. 그 외에도 항공기 관련의 빅데이터와 항공기 시스템 엔지니어링을 검증하는 플랫폼의 특성도 반영하였다.

Functional 측면(Aspect)은 무인항공기[142]와 ARINC 653[143] 플랫폼에서 설명하는 항공기가 갖추어야 할 기능을 서술하고 있다. 다음으로 Human 측면(Aspect)은 항공기와 사람 사이의 상호 작용을 위해 갖추어야 할 기능을 서술하고 있다. Trustworthiness 측면(Aspect)은 신뢰성 있는 동작을 위해 갖추어야 할 기능에 대해 서술하고 있다. Timing 측면(Aspect)은 항공기의 기능을 수행하는데 요구되는 시간과 관련이 있는 기능에 대해 서술하였다. Data 측면(Aspect)은 항공기에서 송・수신되는 데이터들이 갖추어야 할 기능에 대해서 서술하고 있다. 마지막으로 composition 측면(Aspect)은 CPS에 있는 구성요소들 간의 원활한 동작을 위해 필요한 기능에 대해 서술하였다.

항공기의 사용자 요구사항은 기능적 측면 37종, 인간적 측면 5종, 신뢰성 측면 14종, 실시간 측면 6종, 데이터 측면 12종, 그리고 구성 측면의 9종이 도출되어 총 83종이 도출되었다.

항목 ID는 Aero.XXXX.YYY.###로 표현된다. XXXX는 해당 측면(Aspect), YYY는 관심사(Concern), ###은 일렬 번호를 나타낸다.

---

140) 파티션(Partition)은 시간적, 공간적으로 다른 파티션들과 분리. 시간적 파티션은 시스템 설정 시간에 정적으로 정해진 파티션 수행 시간 동안 다른 파티션의 방해를 받지 않는 것이 보장되는 것. 공간적인 파티셔닝은 파티션 내의 프로세스들이 수행되는 메모리 영역 보호

141) 손동환(2014), 항공기 시스템의 SW를 위한 표준, TTA 저널 Vol 154, 2014.7.8.

142) http://mysnu.org/community/newtechnology.php?search_order=&search_part=&c_cate1=&mode=v&idx=11050&thisPageNum=

143) 손동환 (2014), 항공기 시스템의 SW를 위한 표준, TTA 저널 Vol 154, chapter2, 2014.7.8.

1) 기능적(Functional) 측면

항공기의 Functional 측면(Aspect)은 항공기 관련으로 무인항공기와 ARINC 653 플랫폼 등의 실질적인 기능들인 제어, 센싱, 통신 등의 특성들을 반영하였다. 기능적 측면의 요구사항은 총 39종이 도출되었다.

<표 4-8> 항공기의 Functional 측면들(Aspects)과 관심사들(Concerns)

측면(Aspect)	관심사(Concern)	항목 ID	설명
Functional	Actuation	Aero.Func.Act.001	무인항공기는 각 로터의 회전수를 다르게 제어함으로 기체의 회전을 발생시켜 추력 방향을 vectoring함
		Aero.Func.Act.002	무인항공기는 추력의 크기를 조정하여 원하는 방향으로 비행할 수 있도록 고안됨
	Communication	Aero.Func.Comm.001	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 파티션간 데이터를 교환하기 위한 통신기능을 보유
		Aero.Func.Comm.002	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 메시지들의 교환으로 이루어짐
		Aero.Func.Comm.003	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 파티션 간의 논리적인 통신 링크의 채널을 보유
		Aero.Func.Comm.004	항공기 소프트웨어 표준 플랫폼(ARINC 653)는 특정 채널로 파티션에서 메시지를 보내거나, 받는데 필요한 자원을 제공하는 포트가 있음
		Aero.Func.Comm.005	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 크게 메시지 교환과 동기화를 위한 기능들로 구성
	Controllability	Aero.Func.Ctrl.001	무인항공기는 제어 입력 결정을 위해, 일반적으로 저가의 온보드 센서 만을 이용하여 이러한 상태 추정을 정확하게 함
	Functionality	Aero.Func.Func.001	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 응용 간 오류의 전달을 방지함
		Aero.Func.Func.002	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 독립적인 응용 수행을 보장하기 위해 파티셔닝 기술 제공
		Aero.Func.Func.003	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 미리 정해진 오류 처리 루틴을 수행하기 위한 인프라인 헬스모니터링 기능을 제공
		Aero.Func.Func.004	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 도메인과 OS 독립적으로 기능을 제공
	Manageability	Aero.Func.Mng.001	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 시간적, 공간적으로 파티션으로 분류함

		Aero.Func.Mng.002	항공기 소프트웨어 표준 플랫폼(ARINC 653)의 시간적 파티셔닝으로 다른 파티션의 방해를 받지 않도록 보장함
		Aero.Func.Mng.003	항공기 소프트웨어 표준 플랫폼(ARINC 653)의 공간적 파티셔닝은 파티션 내의 메모리 영역을 정적으로 설정함
		Aero.Func.Mng.004	항공기 소프트웨어 표준 플랫폼(ARINC 653)의 공간적 파티셔닝은 독립적인 수행을 보장함
		Aero.Func.Mng.005	항공기 소프트웨어 표준 플랫폼(ARINC 653)의 파티션은 파티션 내의 관련된 기능을 동적으로 수행하기 위한 하나 이상의 프로세스로 구성됨
		Aero.Func.Mng.006	항공기 소프트웨어 표준 플랫폼(ARINC 653)의 프로세스 관리는 FIFO 스케줄링을 기반으로 구현
		Aero.Func.Mng.007	항공기 소프트웨어 표준 플랫폼(ARINC 653)의 헬스 모니터링이 오류를 모듈 레벨, 파티션 레벨과 프로세스 레벨로 나눠서 관리함
		Aero.Func.Mng.008	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 오류 처리 프로세스를 생성하기 위한 인터페이스를 제공함
		Aero.Func.Mng.009	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 SW오류와 HW오류 같은 에러들을 처리함
		Aero.Func.Mng.010	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 여러 에러들을 설정 테이블에 정의된 헬스 모니터링 콜백 함수들에 의해 처리
		Aero.Func.Mng.011	항공기 빅데이터 처리로 적절한 시점에 예방적인 유지보수 활동을 진행할 수 있게 됨
	Measurability	Aero.Func.Meas.001	무인항공기는 각속도계와 가속도계로 구성된 IMU(inertial measurement unit)과 지자기 센서를 사용하여, 가속도계로 중력방향을 추정함
		Aero.Func.Meas.002	무인항공기는 지자기 센서로 남북방향을 추정하여 무인항공기의 공간상 회전을 추정하는 방법이 있음
	Monitorability	Aero.Func.Mon.001	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 오류 처리하는 기능을 운영체제에서 제공하는 인프라인 헬스모니터링이 있음
		Aero.Func.Mon.002	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 오류를 격리하고, 오류의 확산을 방지함
	Performance	Aero.Func.Per.001	항공기 빅데이터에서 MRO(Maintenance, Repair, and Overhaul)를 수행하여 항공기 데이터 처리 성능을 향상시켜야 함
		Aero.Func.Per.002	항공기 빅데이터에서 필터링한 각 데이터에 북마크나 하이퍼링크 등의 참조 기술을 활용하여

			데이터 관리 성능을 향상시켜야 함
		Aero.Func.Per.003	항공기 빅데이터로 인한 예방적 활동은 항공기의 정지시간을 줄여줄 수 있어, 사용자에게 서비스 할 수 있는 비행기 대수를 늘려야 함
	Physical	Aero.Func.Phy.001	무인항공기가 바람이나 외란[144]에 강인한 협업 무인항공기 제어 기술이 필요
	Physical context	Aero.Func.Phycon.001	무인항공기의 회전과 추력 제어가 되면 무선 조종이 가능하게 됨
		Aero.Func.Phycon.002	무인항공기는 위성측위시스템(GNSS:GPS, GLONASS 등) 방법을 사용함
	Sensing	Aero.Func.Sens.001	항공기 빅데이터 처리로 항공사들은 다양한 기체 부품의 결함을 감지하고 예측할 수 있음
	Uncertainty	Aero.Func.Uncert.001	무인항공기의 배터리 기술의 비약적인 발전 필요
		Aero.Func.Uncert.002	무인항공기의 새로운 에너지원의 개발 필요
		Aero.Func.Uncert.003	무인항공기의 새로운 디자인의 신개념 무인항공기 연구가 필요

2) 인간적(Human) 측면

항공기의 Human 측면(Aspect)은 항공기와 사람들과의 상호 작용에 관해서 정의하였다. 인간적 측면의 요구사항은 총 7종이 도출되었다.

<표 4-9> 항공기의 Human 측면들(Aspects)과 관심사들(Concerns)

측면(Aspect)	관심사(Concern)	항목 ID	설명
Human	Human factors	Aero.Hum.HF.001	무인항공기 촬영영상은 방송용 촬영의 고가의 크레인 카메라를 대체함
		Aero.Hum.HF.002	무인항공기를 활용한 셀피 기능이나 피사체 추종 촬영 등이 있음
		Aero.Hum.HF.003	무인항공기는 다수의 공중촬영영상을 붙여서 건설 현장의 진척 상황을 보여주는 3차원 환경 지도를 작성
		Aero.Hum.HF.004	무인항공기는 공중촬영영상을 분석하여 농업 현장의 병충해 등을 평가함
		Aero.Hum.HF.005	무인항공기를 이용한 배송 서비스는 무겁지 않으면서, 중요하거나 시급한 물품 배송을 중심으로 진행됨

144) 자동 제어계의 상태를 바꾸려고 하는 외적 작용. 자동 조종된 비행체에 가해지는 바람의 영향은 이 좋은 예.

3) 신뢰성(Trustworthiness) 측면

항공기의 Trustworthiness 측면(Aspect)은 항공기의 신뢰성, 보안성 등의 관련된 특성들을 반영하였다. 신뢰적 측면의 요구사항은 총 11종이 도출되었다.

<표 4-10> 항공기의 Trustworthiness 측면들(Aspects)과 관심사들(Concerns)

측면(Aspect)	관심사(Concern)	항목 ID	설명
Trustworthiness	Privacy	Aero.Trust.Priv.001	무인항공기의 영상 촬영은 기존의 개인정보법체계나 프라이버시 보호체계 내에서 무인항공기에 대한 규제를 시도함
	Reliability	Aero.Trust.Rel.001	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 오류를 모듈레벨, 파티션레벨, 프로세스 레벨로 3가지로 나눠 관리
	Resilience	Aero.Trust.Res.001	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 시스템 내에서 오류 발생 시에 오류를 자동적으로 처리하는 기능이 있음
		Aero.Trust.Res.002	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 비상 모드 파티션 스케줄링을 통해 시스템의 자동복구나 회귀를 수행할 수 있음
	Safety	Aero.Trust.Safe.001	항공기 소프트웨어 인증 표준(DO 178)은 항공기 수준의 위험성 및 안전성 분석을 수행
		Aero.Trust.Safe.002	항공기 소프트웨어 인증 표준(DO 178)은 발생 가능한 위험과 그로 인한 손실을 예측하여 분석을 수행
		Aero.Trust.Safe.003	항공기 소프트웨어 인증 표준(DO 178)은 분석된 결과를 바탕으로 체계설계를 보완하여 위험성을 완화함
		Aero.Trust.Safe.004	무인항공기 비행 중 일어날 수 있는 안전사고에 대한 안정성 확보 기술이 필요
		Aero.Trust.Safe.005	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 비행제어 응용은 오류가 발생하면, 비행기가 추락할 수 있어 바로 사람의 생명에 영향이 미침
		Aero.Trust.Safe.006	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 오류 없이, 독립적으로 수행 할 수 있는 파티셔닝 기술 보유
	Security	Aero.Trust.Secu.001	무인항공기의 정보 보안기술이 필요
		Aero.Trust.Secu.002	무인항공기의 물리 보안기술이 필요

		Aero.Trust.Secu.003	무인항공기의 융합 보안 기술이 필요
		Aero.Trust.Secu.004	무인항공기 서버 간 통신 보안 기술이 필요

4)실시간(Timing) 측면

항공기의 Timing 측면(Aspect)은 항공기에서 시간과 빈도 등에 관한 요소들을 정의하였다. 실시간 측면의 요구사항은 총 6종이 도출되었다.

<표 4-11> 항공기의 Timing 측면들(Aspects)과 관심사들(Concerns)

측면(Aspect)	관심사(Concern)	항목 ID	설명
Timing	Synchronization	Aero.Timing.Sync.001	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 동기화를 위해 세마포어와 이벤트를 제공함
		Aero.Timing.Sync.002	무인항공기에서 미디어 서버 간 동기화가 필요
	Time awareness	Aero.Timing.Aware.001	항공기 소프트웨어 표준 플랫폼(ARINC 653)의 시간적 파티셔닝은 다른 파티션의 방해를 받지 않도록 보장함
		Aero.Timing.Aware.002	항공기 소프트웨어 표준 플랫폼(ARINC 653)의 파티션은 최소 단위의 파티션 스케줄링 순서인 major time frame을 사용함
		Aero.Timing.Aware.003	항공기 빅데이터를 통해, 운항 지연 횟수와 비행 취소 건수를 획기적으로 줄일 수 있음
	Time-interval and latency	Aero.Timing.interval.001	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 각 파일시스템 서비스들의 지연시간을 명시해야 함

5) 데이터(Data) 측면

항공기의 Data 측면(Aspect)은 데이터 생성, 관리, 상호 작용 등에 관한 부분이다. 데이터 측면의 요구사항은 총 12종이 도출되었다.

<표 4-12> 항공기의 Data 측면들(Aspects)과 관심사들(Concerns)

측면(Aspect)	관심사(Concern)	항목 ID	설명
Data	Identity	Aero.Data.Id.001	항공기 선박 자동 식별시스템은 넓은 해상에서 운항 중인 여러 선박의 식별신호를 하늘에서 동시에 수신
		Aero.Data.Id.002	항공기 선박 자동 식별시스템은 항공기의 위치정보와 카메라의 지향각 등을 계산해 촬영 중인 선박의 식별정보를 확인할 수 있음

		Aero.Data.Id.003	항공기 선박 자동 식별시스템의 카메라를 특정 선박으로 향하는 것만으로도 해당 선박의 실시간 영상과 식별정보를 함께 파악할 수 있음
	Operations on data	Aero.Data.OP.001	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 파티션 간 데이터를 교환하기 위해 메시지들의 교환으로 이루어짐
		Aero.Data.OP.002	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 일반 파일시스템과 마찬가지로 저장 공간을 파일 형태로 관리(읽기, 쓰기, 생성, 삭제, 닫기, 열기 등)하는 기능이 있음
		Aero.Data.OP.003	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 Queueing 포트를 지원함
		Aero.Data.OP.004	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 Sampling 포트를 지원함
	Relationship between data	Aero.Data.Relat.001	항공기 빅데이터에서 항공기 센서 등으로부터 수집된 데이터는 중앙화 된 센터 등을 통해 일괄적으로 관리되어야 함
	Data velocity	Aero.Data.Vel.001	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 RAM의 버퍼에 일차적으로 저장되어 응용들이 빠르게 저장할 수 있도록 함
		Aero.Data.Vel.002	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 RAM 버퍼의 메시지들을 비휘발성 메모리로 저장하는 로그북 시스템 기능이 있음
	Data volume	Aero.Data.Vol.001	항공기 엔진으로부터 발생하는 데이터는 항공기 한 대당 연간 약 25억 페타바이트 발생
		Aero.Data.Vol.002	데이터 전송 과정에 있어 속도나 전송 오류, 전송 용량 한계 등이 해결해야 되어 공유가 되어야 함

6) 조립(Composition) 측면

항공기의 Composition 측면(Aspect)은 구성 요소의 속성들과 관련된 특성을 반영하였다. 구성 측면의 요구사항은 총 10종이 도출되었다.

<표 4-13> 항공기의 Composition 측면들(Aspects)과 관심사들(Concerns)

측면(Aspect)	관심사(Concern)	항목 ID	설명
Composition	Adaptability	Aero.Comp.Adapt.001	무인항공기는 공중영상촬영으로 사람의 시야를 하늘로 확장
		Aero.Comp.Adapt.002	무인항공기는 배송 서비스로 응용 분야를 확장
	Complexity	Aero.Comp.Complex.001	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 소프트웨어 및 하드웨어 모듈을 위한 표준 데이터 구조를 정의함
		Aero.Comp.Complex.002	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 불필요한 파라미터 형식의 변동을 줄임
		Aero.Comp.Complex.003	항공기 소프트웨어 표준 플랫폼(ARINC 653)은 통상적인 I/O 프로세싱을 줄일 수 있음
	Discoverability	Aero.Comp.Discover.001	항공기 빅데이터 분석을 통해 고객과 항공사 모두에게 근본적인 이익이 될 수 있는 전략을 짤 수 있게 해야 함
		Aero.Comp.Discover.002	무인항공기와 외부 환경을 모두 '협력 러닝모델'로 활용해 기능 효율성을 높임
		Aero.Comp.Discover.003	무인항공기는 AI로 연산 과제를 분류·배분하는 드론과 지상 컴퓨팅 환경을 연결해 수집 정보를 나눠 학습하는 '스플릿 러닝' 기술 필요
		Aero.Comp.Discover.004	상대적으로 손쉬운 연산 과제는 무인항공기가 직접 처리하게 하고, 어려운 것은 외부에 전송·처리해야 함

7) 항공기 사용자 요구사항에 대한 분석

항공기도 자동차와 유사하게 기능적 측면에서 CPS의 특성을 극명하게 반영하는 관심사(Concern)는 Actuation, Controllability, Measurability로 주로 무인항공기의 기능으로 도출되었다. 신뢰성 측면에서는 특이한 점은 무인항공기의 영상 촬영이 개인의 정보나 사생활을 침범하지 않도록 하는 면이 도출되었다. 항공기에서 실시간성 측면은 항공기 안전을 위해 필수적인 요구사항이며, 안전이 중요 시 되는 CPS에서는 반드시 고려되어야 하는 측면이다.

제3절 스마트공장 사용자 요구사항

본 절에서 정의하는 스마트공장의 기능들도 CPS 프레임워크의 6개의 측면(Aspect)들과 각 측면(Aspect)마다 여러 관심사(Concern)들로을 이용하여 분석하였다. 모든 측면(Aspect)들은 CPS의 특징을 잘 보여주고 있는데 이는 사물인터넷, 빅데이터, 인공지능의 특징을 포함하는 industrie 4.0의 Reference Architecture Model Industrie 4.0(RAMI4.0)을 기준으로 작성하였다.

특히 RAMI4.0이 구성하는 다양한 표준 중에서 IEC 62890 생명주기(Life-Cycle Status), IEC 61784 제조 통신(Industrial Communication Networks), IEC 62541 사물인터넷(OPC UA), 그리고 VDMA24582 모니터링 (Condition Monitoring)[145]를 참고하여 스마트공장이 가져야 하는 생산 제품에 대한 관리, 구성 기기 또는 이종 기기 간의 통신, 그리고 구성 기기 또는 생산 제품에 대한 지속적인 모니터링에 대해 참고하였다.

스마트공장이 대규모화되면서 단순히 현장기술자의 제어에 의해 동작하는 방법에서 벗어나 시스템 스스로가 관찰하고 판단하며 대처하는 기능과 사람과의 의사소통을 통해 최적의 판단을 도출하는 방향으로 발전하고 있다.

1. 스마트공장 요구사항 분석 대상

스마트공장 요구사항 분석 대상을 RAMI 4.0으로 결정하고, 자료 분석과 전문가 인터뷰를 통해 사용자 요구사항을 도출하였다.

RAMI 4.0(Reference Architectural Model Industrie 4.0)은 독일 Industry 4.0의 관련 기술을 적용한 참조모형으로써 3차원적으로 Industry 4.0에 관련된 기술을 체계적으로 분류하고 정의한 것으로 기존 생산 자동화 관련 표준인 IEC62264 제어 시스템 통합 (enterprise-control system integration), IEC 61512와 IEC62890 생명주기(Life-Cycle Status)에 네트워크, 장치 등을 더해 Industry 4.0의 범위를 확장시킨 것이다.

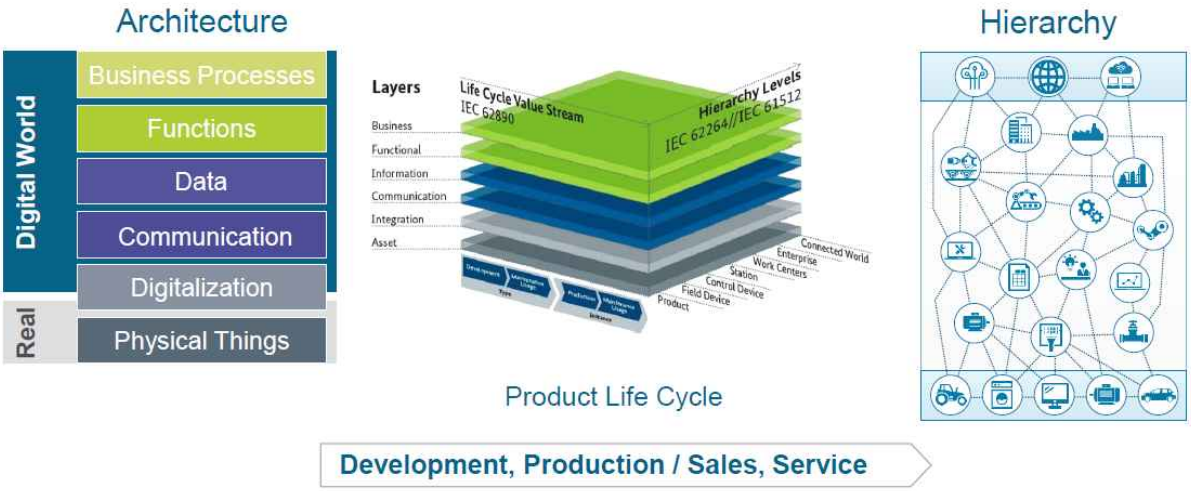
다음 그림은 RAMI 4.0의 3차원 모형을 나타낸다. 첫 번째 축인 Hierarchy 구조는 연결된 세상, 스마트공장, 스마트 제품으로 구성되어 있다. 유연한 시스템과 기계로 구성되어 있는 스마트공장은 네트워크를 통한 분산된 기능을 가지고 있다. 두 번째 축은 제품 생명주기 축으로 제품 개발과 유지보수로 구성되어 있다.

RAMI의 세로 층은 6개의 층으로 구성되어 있다. Asset층은 실제 세계의 물리적 물건을 포

145) Fieldbus Neutral Reference Architecture for Condition Monitoring in Factory Automation , 2014.4.

함하고, Integration층은 실제 세계를 디지털 세상으로 이행하고, Communication층은 정보를 접속하게 하며, Information층은 필요한 정보를 포함한다. Functional층층 Asset의 기능들의 집합이며, Business층은 조직과 사업 프로세스를 나타낸다.[146)]

[그림 4-4] Reference Architectural Model Industrie 4.0 (RAMI 4.0)



자료 : Dr. Karsten Schweichhart, Reference Architectural Model Industrie 4.0 (RAMI 4.0) An introduction

2. CPS 프레임워크의 측면 관점에서 스마트공장 요구사항 명세

CPS 프레임워크 6개의 측면(Aspect)에 대한 관심사들(Concerns)을 기준으로 사용자 요구사항을 도출하였다. 각 측면(Aspect)은 CPS가 가져야 할 소프트웨어 기능 중심으로 서술하였고, 일반화하기 어려운 실제 기기 등의 하드웨어나 부가가치를 창출해내는 특수한 기술들에 대해서는 요구사항 도출에 한계점이 있어 중요하게 다루어지지는 않았다.

CPS의 Functional 측면(Aspect)은 스마트공장을 구성하는 기기들과 기기들이 수행하는 공정, 기기들이 만드는 제품 그리고 스마트공장의 환경에 대해 CPS가 갖추어야 기능을 서술하고 있다. 다음으로 Human 측면(Aspect)은 스마트공장과 현장기술자 사이의 원활한 의사소통을 위해 갖추어야 할 기능을 서술하고 있다. Trustworthiness 측면(Aspect)은 스마트팩토리에서 신뢰성이 있는 동작을 위해 갖추어야 할 기능에 대해 서술하고 있다. Timing 측면(Aspect)은 스마트공장의 기능을 수행하는데 요구되는 시간과 관련이 있는 기능에 대해 서술하였다. Data 측면(Aspect)은 스마트팩토리에서 송·수신되는 데이터들이 갖추어야 할 기

146) Dr. Karsten Schweichhart(2016), Reference Architectural Model Industrie 4.0 (RAMI 4.0) An introduction

능에 대해서 서술하고 있다. 마지막으로 composition 측면(Aspect)은 CPS에 있는 구성요소들 간의 원활한 동작을 위해 필요한 기능에 대해 서술하였다.

스마트공장의 사용자 요구사항은 기능적 측면 35종, 인간적 측면 7종 신뢰성 측면 15종, 실시간 측면 11종, 데이터 측면 17종 그리고 구성 측면의 11종이 도출되어 총 96총이 도출되었다.

항목 ID는 SF.XXXX.YYY.###로 표현된다. XXXX는 해당 측면(Aspect), YYY는 관심사(Concern), ###은 일렬 번호를 나타낸다.

1) 기능적(Functional) 측면

스마트공장의 구성 기기들이 필수적으로 수행해야하는 공정, 통신, 모니터링에 대한 기능에 대해 다루고 있다. 주로 스마트공장의 구성 기기가 갖추어야할 기능을 CPS 프레임워크에 따라 도출하였다. 기능적 측면의 관심사들(Concerns)은 총 35종이 도출되었다.

<표 4-14> 스마트공장의 Functional 측면의 관심사들(Concerns)과 요구사항들

측면(Aspect)	관심사(Concern)	항목 ID	설명
Functional	Actuation	SF.Func.Act.001	스마트공장의 기기들은 설정된 공정에 따라서 생산 라인을 구성함
		SF.Func.Act.002	스마트공장의 기기들은 상호 연결을 통해 동작함
		SF.Func.Act.003	스마트공장의 기기들은 설정된 공정에 따라서 제품을 생산함
		SF.Func.Act.004	스마트공장은 생산중인 제품의 상태를 추적할 수 있음
		SF.Func.Act.005	스마트공장은 제품의 생산 요구에 따라 다른 동작을 할 수 있음
	Communication	SF.Func.Comm.001	스마트공장의 기기들은 OPC/UA, PROFINET, MODBUS TCP/IP, POWERLINK 등의 다양한 산업용 통신 기술 지원함
		SF.Func.Comm.002	스마트공장의 기기들은 산업용 통신기술을 통해 메시지 송수신
		SF.Func.Comm.003	스마트공장의 기기들은 산업용 통신 기술을 통해 관리자로부터 제어 메시지 수신
		SF.Func.Comm.004	스마트공장의 기기들은 산업용 통신 기술을 통해 상태 정보를 서버로 전송
	Controllability	SF.Func.Ctrl.001	스마트공장의 환경(예. 공장 내 온도, 습도 등)을 제어할 수 있음

		SF.Func.Ctrl.002	스마트공장의 공정 상태를 제어할 수 있음
		SF.Func.Ctrl.003	스마트공장의 기기들의 동작 상태를 제어할 수 있음
		SF.Func.Ctrl.004	스마트공장의 기기들은 원격으로 제어될 수 있음
	Functionality	SF.Func.Func.001	스마트공장은 제품을 효율적으로 생산하기 위한 공정의 재구성 지원
		SF.Func.Func.002	스마트공장은 공정을 최적화 하여 시간과 비용을 절약 가능
		SF.Func.Func.003	스마트공장은 일부에 결함이 있는 상태더라도 정상적으로 작동이 가능함
		SF.Func.Func.004	모델링&시뮬레이션(이하 M&S)를 통하여 스마트공장의 진단 및 상태 예측 가능
	Manageability	SF.Func.Mng.001	스마트공장에서 기기들의 복잡한 연동 구조 관리 가능
		SF.Func.Mng.002	스마트공장에서 작동하는 기기들의 시간축을 관리할 수 있음
		SF.Func.Mng.003	스마트공장에서 기기들의 작동 기능에 대하여 관리 가능
	Measurability	SF.Func.Meas.001	스마트공장에서 생산하는 제품이 요구사항을 만족시키는지 확인 가능
		SF.Func.Meas.002	스마트공장의 공정이 요구사항대로 설정되었는지 확인 할 수 있음
		SF.Func.Meas.003	스마트공장의 네트워크가 요구사항을 만족시킬 수 있도록 설정되었는지 확인할 수 있음
		SF.Func.Meas.004	스마트공장의 내부 환경 설정이 적합하게 되었는지 확인할 수 있음
		SF.Func.Meas.005	스마트공장 구성요소의 모델링과 시뮬레이션을 통하여 실제 상태를 확인할 수 있음
	Monitorability	SF.Func.Mon.001	스마트공장의 구성요소들이 적절하게 작동하는지 확인할 수 있음
		SF.Func.Mon.002	스마트공장의 공정들이 적절하게 설정되는지 확인할 수 있음
	Performance	SF.Func.Perform.001	스마트공장의 구성요소들은 설정된 공정이 요구하는 성능을 만족해야 함
	Physical	SF.Func.Phy.001	스마트공장의 기기들은 기계요소들로 구성
	Physical context	SF.Func.Phycont.001	스마트공장의 공정 설정에 있어서 기기들의 실제 위치도 고려함
	Sensing	SF.Func.Sens.001	스마트공장의 기기들은 생산중인 제품의 상태에 대하여 인지

		SF.Func.Sens.002	스마트공장의 기기들은 내부 상태에 대하여 인지
		SF.Func.Sens.003	스마트공장은 공장 내부의 온도, 습도 등의 환경 정보를 인지할 수 있음
	States	SF.Func.State.001	스마트공장의 기기들은 상황에 따라 동작 상태의 변동이 이루어 짐
	Uncertainty	SF.Func.Uncert.001	스마트공장의 기기들은 예측하지 못한 오작동 등이 발생할 수 있음

2) 인간적(Human) 측면

현장기술자가 스마트공장을 제어하는데 스마트공장이 제공해야하는 기능에 대해 설명하고 있다. 주로 스마트공장이 현장기술자와 의사소통하기 위한 기능 총 7총을 도출하였다.

<표 4-15> 스마트공장의 Human 측면의 관심사들(Concerns)과 요구사항들

측면(Aspect)	관심사(Concern)	항목 ID	설명
Human	Human factors	SF.Hum.HF.001	스마트공장의 공정 구성을 위하여 기기들은 관리자에 의해 제어됨
		SF.Hum.HF.002	스마트공장 기기의 상태 정보들은 관리자가 원격 모니터링을 할 수 있도록 사용됨
		SF.Hum.HF.003	스마트공장 기기들은 현장 기술자에 의해서 조작될 수 있음
	Usability	SF.Hum.Usa.001	스마트공장 모니터링을 통하여 공장 전체의 상태를 실시간으로 확인할 수 있음
		SF.Hum.Usa.002	스마트공장의 모니터링 및 원격 제어를 통하여 이상 상황에 대처할 수 있음
		SF.Hum.Usa.003	제품의 요구사항에 따라 스마트공장 공정을 유연하게 구성할 수 있음
		SF.Hum.Usa.004	공장 상태에 따라 공정을 유연하게 구성할 수 있음

3) 신뢰성(Trustworthiness) 측면

스마트공장에서 구성된 기기들이 신뢰성 있는 동작을 수행하기 위해 갖추어야 할 관심사들(Concerns)은 총 15종을 도출하였다.

<표 4-16> 스마트공장의 Trustworthiness 측면의 관심사들(Concerns)과 요구사항들

측면(Aspect)	관심사(Concern)	항목 ID	설명
Trustworthiness	Privacy	SF.Trust.Priv.001	서버에 저장된 스마트공장의 기기 정보에 접근 할 수 있는 권한에 관한 보안 체계를 제공해야 함
		SF.Trust.Priv.002	스마트공장의 제어 네트워크에 접근할 수 있는 권한에 관한 보안 체계를 제공해야 함
	Reliability	SF.Trust.Rel.001	스마트공장의 기기들은 공정 설정에 따라서 정확하게 설정됨
		SF.Trust.Rel.002	스마트공장의 기기들은 공정 설정에 따라 배정된 작업들을 정확하게 수행함
		SF.Trust.Rel.003	스마트공장의 기기들은 원격으로부터의 설정을 정확하게 반영함
	Resilience	SF.Trust.Res.001	스마트공장의 기기 결함 발생 상황에서 빠르고 유연하게 대처할 수 있음
		SF.Trust.Res.002	스마트공장의 공정 이상 발생 상황에서 빠르고 유연하게 대처할 수 있음
		SF.Trust.Res.003	스마트공장의 기기들의 연결에 문제가 발생할 경우 네트워크 재설정을 통하여 빠르게 대처할 수 있음
		SF.Trust.Res.004	스마트공장의 기기들의 이상 상황에서 M&S를 통하여 이상 상황의 대처법을 시뮬레이션 해 볼 수 있음
	Safety	SF.Trust.Safe.001	스마트공장은 환경을 조절하여 작업 상황에서의 결함 발생을 방지할 수 있어야 함
		SF.Trust.Safe.002	스마트공장의 기기들은 작업 상황에서 안전사고 발생을 예방할 수 있어야 함
		SF.Trust.Safe.003	스마트공장의 기기들은 작업자의 조작을 감안하여 안전사고 발생을 예방할 수 있어야 함
		SF.Trust.Safe.004	스마트공장의 기기들은 상호 연동을 통하여 안정적인 상태를 유지할 수 있어야 함
	Security	SF.Trust.Sec.001	스마트공장의 기기들과 장비들의 보호를 위한 보안 체계가 있어야 함
		SF.Trust.Sec.002	스마트공장에서 발생되는 메시지에 대한 보안 체계가 있어야 함

4) 실시간(Timing) 측면

스마트공장이 수행하는 동작들의 시간적인 특성에 대해 서술하고 있다. 스마트공장의 소프트웨어와 하드웨어에 걸쳐 전반적으로 이루어진 기능에 갖추어야할 기능을 도출하였다. 스마트공장에서는 실시간성을 구현하기 이벤트의 스케쥴링이 중요하다. 실시간 측면의 관심사들(Concerns)은 총 11종이 도출되었다.

<표 4-17> 스마트공장의 Timing 측면의 관심사들(Concerns)과 요구사항들

측면(Aspect)	관심사(Concern)	항목 ID	설명
Timing	Logical time	SF.Time.Logic.001	스마트공장은 모델을 통하여 이벤트의 발생 순서를 표현할 수 있음
		SF.Time.Logic.002	스마트공장은 모델을 이용하여 이벤트의 발생 시간을 측정 가능
	Synchronization	SF.Time.Sync.001	스마트공장의 각 노드들은 서로 시간동기화 필요
		SF.Time.Sync.002	스마트공장의 각 노드들은 주파수가 동기화 필요
		SF.Time.Sync.003	스마트공장의 각 노드들은 Phase의 동기화 필요
		SF.Time.Sync.004	스마트공장은 노드 간 동기화에 대한 정보를 이용하여 지능적으로 동기화 실행
		SF.Time.Sync.005	스마트공장의 구성 요소간의 분산 M&S 시 동기화된 데이터로 시뮬레이션 수행
	Time awareness	SF.Time.Aware.001	스마트공장은 모델을 설계할 때 시간적 정확성을 고려하여 설계하여야 함
	Time-interval and latency	SF.Time.Interval.001	스마트공장의 이벤트 간에 걸리는 시간은 설정된 요구사항보다 작아야 함
		SF.Time.Interval.002	스마트공장의 이벤트 간 걸리는 시간이 그 요구시간을 초과했을 때, 에러를 발생시키는 기준이 되는 시간을 정의해야 함
		SF.Time.Interval.003	특정 스마트공장의 이벤트들은 국제 표준을 준수해야 함

5) 데이터(Data) 측면

스마트공장에서 송·수신되는 데이터들이 스마트공장 내에서 원활하게 이용되기 위한 기능에 대해 설명하고 있다. 주로 스마트공장의 데이터들 사이의 관계에 대한 내용을 도출하여, 데이터 측면 관심사들(Concerns)은 총 18종이 도출하였다.

<표 4-18> 스마트공장의 Data 측면의 관심사들(Concerns)과 요구사항들

측면(Aspect)	관심사(Concern)	항목 ID	설명
Data	Data semantic	SF.Data.Sem.001	스마트공장 내에 있는 데이터들은 서로 공유될 수 있어야 함
		SF.Data.Sem.002	스마트공장 내에 공유된 데이터들은 같은 의미를 가져야 함
	Identity	SF.Data.Id.001	스마트공장과 관계있는 구성요소들은 그들 고유의 식별자를 가지고 있어야 함
		SF.Data.Id.002	스마트공장은 관계있는 구성요소들을 정확하게 인식할 수 있어야 함
	Operations on data	SF.Data.OP.001	스마트공장은 데이터를 생성/읽기/업데이트/삭제하는 기능을 제공해야 함
		SF.Data.OP.002	스마트공장은 데이터 무결성을 제공해야함
		SF.Data.OP.003	스마트공장은 생성된 데이터를 이용하여 학습할 수 있어야 함
		SF.Data.OP.004	스마트공장은 생성된 데이터를 이용하여 가상화할수 있어야 함
		SF.Data.OP.005	스마트공장은 데이터에 따라 적합한 엑세스 수준을 제공해야 함
	Relationship between data	SF.Data.Relat.001	스마트공장은 서로 영향을 주는 데이터끼리 그룹화할 수 있어야 함
		SF.Data.Relat.002	스마트공장은 데이터를 서로 합치거나 분리해서 새로운 데이터를 만들 수 있어야 함
		SF.Data.Relat.003	스마트공장은 생성된 데이터의 우선순위를 정할 수 있어야 함
		SF.Data.Relat.004	스마트공장은 측정된 데이터를 이용하여 상황을 판단할 수 있어야 함
	Data velocity	SF.Data.Vel.001	스마트공장을 설계할 때 데이터 전송지연을 고려해야 함
		SF.Data.Vel.002	스마트공장은 데이터를 지연없이 실시간으로 전송해야 함
		SF.Data.Vel.003	스마트공장은 전송 지연에 대한 상황에 대처할 수 있어야 함
	Data volume	SF.Data.Vol.001	스마트팩토리는 데이터저장에 대한 정책을 가지고 있어야 함

6) 조립(Composition) 측면

스마트공장의 구성 요소들이 가져야 하는 특징에 대해 설명하고 있다. 주로 스마트공장의 다른 구성요소 사이의 관계에 대한 내용을 도출하였다. 특히 조립 측면에서는 스마트공장을 구성하는 요소들 간의 원활한 의사소통이 중요하며, 관심사들(Concerns)은 총 11종을 도출하였다.

<표 4-19> 스마트공장의 Composition 측면의 관심사들(Concerns)과 요구사항들

측면(Aspect)	관심사(Concern)	항목 ID	설명
Compsition	Adaptability	SF.Com.Adapt.001	스마트공장은 상황에 적절하게 S/W를 추가 또는 제어할 수 있어야 함
		SF.Com.Adapt.002	스마트공장은 상황에 적절하게 H/W를 추가 또는 제어할 수 있어야 함
	Complexity	SF.Com.Complex.001	스마트공장은 가지고 있는 구성요소들의 구성을 직관적으로 표현할 수 있어야 함
		SF.Com.Complex.002	스마트공장은 컴포넌트간의 복잡성으로 인한 손실을 예측할 수 있어야 함
		SF.Com.Complex.003	스마트공장은 복잡성을 낮추기 위한 최적을 상태를 도출할 수 있어야 함
		SF.Com.Complex.004	스마트공장은 다른 컴포넌트를 추가할 때 복잡성을 줄일 수 있는 설계를 도출할 수 있어야 함.
	Constructivity	SF.Com.Const.001	스마트공장은 스마트공장의 모듈 구성요소를 합칠 수 있다.
		SF.Com.Const.002	스마트공장은 사용자의 요구사항을 만족시킬 수 있도록 최적의 형태로 구성될 수 있어야 함
	Discoverability	SF.Com.Discover.001	스마트공장은 구성요소의 고유식별자를 이용하여 특정 구성요소를 찾을 수 있어야 함
		SF.Com.Discover.002	스마트공장은 구성요소의 고유식별자를 이용하여 특정 구성요소의 기능을 찾을 수 있어야 함
		SF.Com.Discover.003	스마트공장의 분산 모델링과 시뮬레이션을 할 때 다른 모델들을 찾을수 있어야 함

7) 스마트공장 사용자 요구사항에 대한 분석

스마트공장은 생산하는 제품이 요구사항을 만족시키는 지 CPS 상에서 확인할 수 있는 기능이 있으며, 이는 기능적 측면의 Measurability에 분류된다. 자동차나 항공과 유사하게 스마트 공장도 제어를 위해서는 Measurability, Sensing의 관심사(Concern)에 속하는 기능이 지원되고, State 관심사에 속하는 요구사항에 의하여 Actuation 관심사에 속하는 기능들이 다르게 수행된다. Uncertainty 관심사는 스마트공장의 기기들에 기능에 중요하게 다루어져야 하는 부분이다.

스마트공장에서는 인간적 측면의 Usability 관심사가 작업자가 공장에서 작업을 하는데 필수적으로 고려되어야 한다. 스마트 공장의 신뢰성 부분에서는 Resilience, Safety 관심사가 공장 운영에 필수적으로 고려되어야 한다.

스마트공장은 여러 기기들이 통합적으로 작동하기 때문에 시간 측면의 Synchronization, Time-interval and Latency 관심사가 중요하게 다루어져야 한다. 또한 구성 측면의 Complexity 관심사에서 공장 구축 및 운영에 필수적으로 고려해야 할 요구사항이 도출된다.

제4절 사용자 요구사항 분석 결과

미국의 NIST(National Institute of Standards and Technology)의 CPS 프레임워크를 활용하여, CPS 분석의 대상 도메인으로 선정한 자동차, 항공기, 스마트공장 관련 소프트웨어 표준 및 관련 자료를 분석하고, 전문가 인터뷰를 통하여 다음과 같은 결과를 얻었다.

<표 4-20> 자동차, 항공기, 스마트공장 사용자 요구사항 개수

측면(Aspect)	자동차	항공기	스마트공장
Functional	25	37	35
Human	7	5	7
Trustworthiness	12	14	15
Timing	8	6	11
Data	7	12	17
Composition	7	9	11
합계	66	83	96

기능적인(Functional) 측면은 자동차, 항공기, 스마트공장의 기능이 아니라 소프트웨어 측면에서만 분석하였기 때문에 소프트웨어와 직접적인 관련이 없는 기능은 제외하였다. Actuation, Controllability, Measurability, Sensing 등은 자율주행자동차, 무인항공기 등의 요구사항을 나타내 특히 CPS에서 더욱 요구되는 사항이라 볼 수 있었다.

인간적(Human) 측면은 자동차의 경우는 개발 시의 역할, 자율주행차의 경우는 운전자와의 관계를 나타낸다. 스마트공장의 경우는 관리자에게 정보를 제공하고, 관리자의 조작관련 사항이다.

신뢰성(Trustworthiness) 측면은 숫자로 보아도 알 수 있듯이 CPS에서 중요하게 다루어져야 한다. 안전, 보안, 복구, 신뢰성 등 CPS를 구현하기 위해서 반듯이 제공되어야 하는 기능이다.

시간(Timing) 관련해서는 실시간성, 시간 동기화, 정확성, 시간 간격에 대한 조건과 지연 시 처리 방법이 지원되어야 한다.

데이터(Data)는 조작이 가능해야 하고 처리속도와 양도 고려해야 하며, 여러 구성요소에서 의미와 형식면에서 식별이 가능해야 한다. 본 장에서는 자율주행차, 무인항공기에 한정한 데이터 조작은 조사에서 제외되었다. 데이터 관련 요구사항은 향후 연구에서 추가할 필요가 있다. 데이터에 대한 일반적인 조작은 스마트공장에서 정리되었다.

마지막으로 조립(Composition) 측면에서는 모듈화, 모형화, 계층화되어 변경 없이 쉽게 복잡한 시스템이 서로 연동이 가능해야 한다.

다음은 각 측면별 도출하지 않은 관심사(Concern)에 대해 정리하였다

<표 4-21> 자동차, 항공기, 스마트공장 제외된 관심사들(Concerns)

측면(Aspect)	자동차	항공기	스마트공장
Functional	Physical, States,Uncertainty	-	-
Human	-	-	-
Trustworthiness	-	-	-
Timing	Logical time	Logical time	-
Data	Identity, Relationship between data, Velocity, Volume	-	-
Composition	Discoverability	-	-

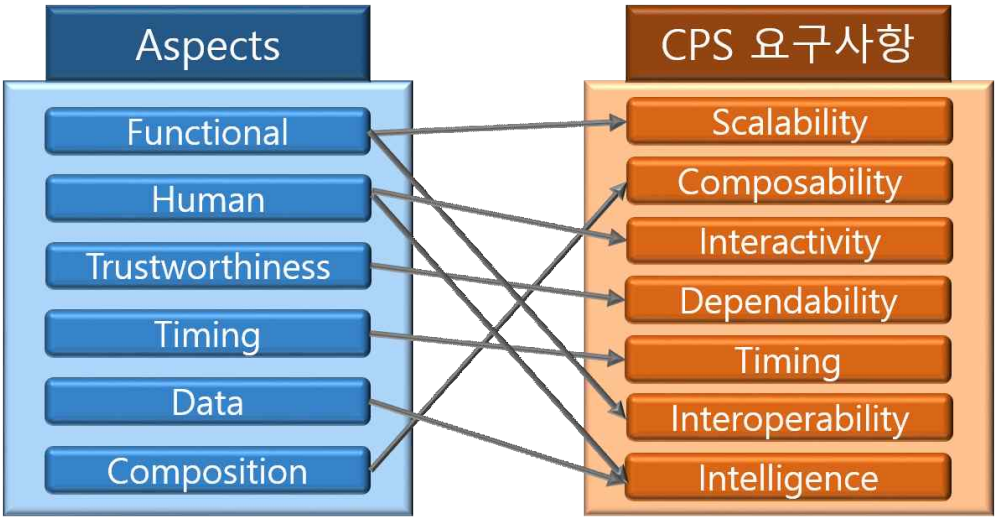
자동차에 관해서는 Autosar Classic 표준을 기반으로 분석했으며, 향후 자율주행을 지원하는 Autosar Adaptive Platform에 대해 분석할 필요가 있다.

제5장 CPS 시스템 요구사항 분석 및 도출

CPS는 2장에서 살펴본 것처럼 단순 물리 시스템에서 확장성(Scalability), 융복합성(Composability), 상호작용성(Interactivity), 고신뢰성(Dependability), 실시간성(Timing), 상호운용성(Interoperability), 지능성(Intelligence)이 강화된 복합시스템이다. 이 장에서는 위의 특성 7가지로 시스템 요구사항을 분류하여 분석하고자 한다.

이 절에서는 3장에서 살펴본 CPS 프레임워크의 측면(Aspect) 기반 분석 방법론을 통해 도출된 사용자 요구사항에 기반을 두고 CPS 시스템 요구사항을 도출한다. 그림에서 보는 것처럼 CPS 프레임워크의 측면들은 그 내용에 따라 CPS의 7가지 특성의 시스템 요구 사항과 일대일, 다대일, 일대다 관계로 매핑 될 수 있다.

[그림 5-1] 측면들(Aspects)와 CPS 요구사항과 매핑 관계



다음의 표는 7가지 CPS 특성에 대한 정의를 정리하였다.

<표 5-1> 7가지 CPS 특성 목록과 설명

구분	설명
Scalability	시스템의 기능적・성능적・물리적 규모의 확장성 요구사항
Composability	CPS들 혹은 서브 시스템들 간의 융복합성에 대한 요구사항
Interactivity	CPS들 간 혹은 CPS와 휴먼 간의 유연한 상호작용성에 대한 요구

	사항
Dependability	신뢰성(Reliability), 회복성(Resilience), 유지보수성(Sustainability), 안전성(Safety) 및 보안성(Security) 등에 대한 요구사항
Timing	CPS 운용 시 요구되는 실시간성(Realtime) 및 동기화(Synchronization) 기능에 대한 요구사항
Interoperability	대규모 CPS에서 발생되는 이종성을 극복하기 위한 데이터의 상호 호환성 제공에 대한 요구사항
Intelligence	CPS의 지능형 서비스 제공을 위한 응용차원의 요구사항

제1절 CPS 관점의 자동차 요구사항 정의

본 절에서는 3장에서 정의한 CPS 프레임워크의 6개 측면(Aspect)과 관심사(Concern)로 분류된 사용자 요구사항을 이용하여, 2장에서 도출한 CPS 특성을 자동차 시스템에 적용하여 자동차가 가져야할 CPS 시스템 요구사항을 도출하였다. [그림 5-1]을 참조하면 CPS 프레임워크의 측면(Aspect)들은 그 내용에 따라 CPS의 7가지 특성과 일대일, 다대일, 일대다 관계로 매핑되는 것을 확인할 수 있다.

본 절에서의 자동차는 요구사항 조사의 범위를 한정짓기 위해 자동차의 ECU 및 자동차를 구성하는 하드웨어에 적용되는 자동차 소프트웨어 구조와 자율 주행을 수행하는 소프트웨어에 대해서만 요구사항을 도출하기로 한다.

시스템 확장성(Scalability) 관련 요구사항은 자동차 구조가 내·외부 기기와 연동하여 그 규모를 적합하게 확장하거나 축소하는데 필요한 요구사항을 정의하였다. 융복합성(Composability) 관련 요구사항은 자동차 구성 요소들을 관리하는데 있어서 필요한 요구사항을 정리하였다. 상호연동성(Interactivity) 관련 요구사항은 자동차를 구성하는 요소들과 운전자 사이의 상호작용하기 위해 필요한 요구사항을 정의하였다. 고신뢰성(Dependability) 관련 요구사항은 자동차가 고신뢰적인 동작을 하는데 필요한 요구사항을 정의하였다. 실시간성(Timing) 관련 요구사항은 자동차 내부 요소들 간 동작하는데 실시간성이 필요한 요소 및 동기화에 대해 정의하였다. 상호운용성(Interoperability) 관련 요구사항은 자동차 내에서 기기들 사이의 의사소통하기 위해 필요한 요구사항 및 원활한 동작을 위한 요구사항을 정의하였다. 마지막으로 지능성(Intelligence) 관련 요구사항은 자동차가 지능적인 동작을 기능별로 나누어 요구사항을 정의하였다.

각 항목별로 시스템 확장성 6종, 융복합성 17종, 상호작용성 30종, 고신뢰성 27종, 실

시간성 13종, 상호운용성 8종 그리고 지능성 7종이 도출되어 총 108종의 자동차 시스템 요구사항이 도출되었다. 도출된 요구사항은 Auto-XXX-Req-### 형태의 아이디(ID)를 가지게 된다. XXX는 확장성(Scalability), 융복합성(Composability), 상호연동성(Interactivity) 등 특성의 앞 3개의 알파벳을 나타낸다. ###는 일렬 번호이다. 요구사항 번호는 요구사항 추적 관리를 위해 사용된다.

1. 요구사항 명세

1) 시스템 확장성(Scalability)

자동차의 시스템 확장성은 자동차가 물리적(하드웨어)으로나 소프트웨어적으로 여러 개의 컴포넌트가 추가 되어 확장될 수 있는 것을 말한다.

자동차의 경우 자동차 소프트웨어 구조 표준인 오토사(Autosar)에 의하면 여러 개의 내·외부의 주변장치 연결과 소프트웨어 컴포넌트로 지원해야함을 명시함으로 시스템 확장성을 제공해야 함을 나타내고 있다.

아래의 표는 요구사항과 요구사항 ID에 대한 설명 그리고 관련 관심사(Concern) ID로 구성되며, 각각의 요구사항 ID는 'CPS-요구사항-요구사항ID' 형식으로 작성하였다. 또한, 각 요구사항별로 표시된 관심사(Concern) id항목은 2장에서 측면(Aspect)별로 분석한 세부 항목들(관심사(Concern))에 해당하는 것이고, 이 요구사항과 연관된 관심사(Concern) id를 뜻한다. 이를 통해 어떻게 CPS 프레임워크 기반으로 분석한 측면(Aspect)이 사용자 요구사항 분류를 통해 CPS 요구사항과 매핑되고 세부 요구사항을 도출해낼 수 있는지 알 수 있다.

자동차의 시스템 확장성 요구사항은 주로 자동차의 Functional 측면(Aspect) 관련 관심사(Concern)들과 연관되어 도출된다. 총 6종의 확장성 요구사항이 도출되었다.

<표 5-2> 자동차의 시스템 확장성 요구사항

요구사항 구분	요구사항 ID	설명	관심사(Concern) ID
Scalability	Autom-Scal-Req-001	자동차 소프트웨어 구조는 멀티 코어 MCU들을 지원해야 함	Auto.Func.Func.005 Auto.Func.Mng.002
	Autom-Scal-Req-002	자동차 소프트웨어 구조의 RTE(Run-Time Environment)는 BSW들의 하나 그리고 여러 개의 인스턴스를 지원하고 처리해야 함	Auto.Func.Comm.001 Auto.Func.Mng.002

	Autom-Scal-Req-003	자동차 소프트웨어 구조 통신은 여러 개의 PDU(Packet Data Unit)을 하나의 PDU로 결합시키는 것을 지원해야함	Auto.Func.Comm.004
	Autom-Scal-Req-004	자동차 소프트웨어 구조는 기본 소프트웨어 모듈들의 여러 인스턴스생성을 허용해야 함	Auto.Func.Func.001 Auto.Func.Mng.002
	Autom-Scal-Req-005	자동차 소프트웨어 구조는 MCU들과 연결되는 장치들 증가를 위해 MCU에 직접 연결되거나 I/O 버스를 통해서 내부와 외부 주변 장치들을 지원해야 함	Auto.Func.Func.005 Auto.Func.Func.007
	Autom-Scal-Req-006	자동차 소프트웨어 구조는 멀티코어 MCU들에 BSW분산을 지원해야함	Auto.Func.Func.001 Auto.Func.Func.005

2) 융복합성(Composability)

자동차의 융복합성은 시스템을 구성하는 여러 컴포넌트들이 융합되거나 복합적으로 구성될 수 있는 특징이 있음을 의미한다.

자동차 소프트웨어 구조 표준인 AUTOSAR 표준에서는 차량의 응용이나 액츄에이터, 센서 등을 위한 다양한 소프트웨어 컴포넌트들이 융복합되어 ECU 하드웨어 위에서 동작함으로서 자동차 융복합성을 제공한다. 또한, 자동차 소프트웨어 구조 표준은 계층적이고 모듈화된 구조를 통해서 융복합성을 용이하게 된다.

자동차 융복합성 요구사항은 주로 자동차의 Functional 측면(Aspect)와 Composition 측면(Aspect) 관련 관심사들(Concerns)과 연관되어 다음의 표와 같이 연결될 수 있다. 자동차 도메인에서는 17가지의 융복합성 요구사항이 도출되었다.

<표 5-3> 자동차의 시스템 융복합성 요구사항

요구사항 구분	요구사항 ID	설명	관심사(Concern) ID
Composability	Autom-Comp-Req-001	자동차 소프트웨어 구조는 계층화된 소프트웨어 구조를 제공해야 함	Auto.Func.Func.002 Auto.Func.Func.003 Auto.Comp.Const.003 Auto.Comp.Const.004
	Autom-Comp-Req-002	자동차 소프트웨어 구조는 마이크로 컨트롤러 독립적인 계층과 의존적인 계층에서 하드웨어 의존적인 계층을 구성해야 함	Auto.Func.Func.002 Auto.Func.Func.003 Auto.Func.Func.004 Auto.Func.Func.005 Auto.Func.Func.007 Auto.Comp.Const.003 Auto.Comp.Const.004 Auto.Func.Func.001
	Autom-Comp-Req-003	자동차 소프트웨어 구조의 BSW(기본 소프트웨어)는 상위 계층의 모듈을 접근하기 위해 콜백 함수들을 제공해야 함	Auto.Func.Func.001 Auto.Func.Func.003 Auto.Func.Func.004 Auto.Comp.Const.004

		Auto.Func.Func.008
Autom-Comp-Req-004	자동차 소프트웨어 구조는 RTE(Run-Time Environment)와 BSW를 사용하기 위해 존재하는 모든 구조적 제약들을 기록해야 함	Auto.Func.Func.001 Auto.Comp.Const.004 Auto.Func.Func.008
Autom-Comp-Req-005	자동차 소프트웨어 구조는 각 기본 소프트웨어 계층에서 접근할 수 있는 서비스 레이어를 제공해야 함	Auto.Func.Func.001 Auto.Func.Func.003 Auto.Comp.Const.004 Auto.Func.Func.008 Auto.Comp.Complex.001
Autom-Comp-Req-006	자동차 소프트웨어 구조는 하드웨어 의존적인 계층과 독립적인 계층에서 BSW를 구성해야 함	Auto.Func.Func.001 Auto.Func.Func.002 Auto.Func.Func.003 Auto.Func.Func.004 Auto.Func.Func.008
Autom-Comp-Req-006	자동차 소프트웨어 구조는 소프트웨어 계층들 내에서 모듈 설계를 제공해야 함	Auto.Func.Func.002 Auto.Func.Func.003 Auto.Func.Func.004 Auto.Func.Func.008
Autom-Comp-Req-007	자동차 소프트웨어 모듈들은 소스와 오브젝트 레벨에서 모듈을 구별하기 위해 메타 데이터 정보를 제공해야 함	Auto.Func.Func.001
Autom-Comp-Req-008	자동차 소프트웨어 모듈 설계는 멀티태스킹 환경에서 협력하기 위해 모듈들을 지원할 수 있어야 함	Auto.Func.Func.001
Autom-Comp-Req-009	자동차 소프트웨어 구조의 BSW 모듈들은 표준 인터페이스들을 제공해야 함	Auto.Func.Func.001 Auto.Func.Func.004 Auto.Func.Func.008
Autom-Comp-Req-010	자동차 소프트웨어 구조는 BSW 모듈이 초기화되기 전에도 소프트웨어 컴포넌트들이 시작될 수 있는 것을 허용해야 함	Auto.Func.Func.001 Auto.Func.Func.008
Autom-Comp-Req-011	자동차 소프트웨어 구조는 BSW 모듈의 제한된 동적인 재설정을 지원해야 함	Auto.Func.Func.001
Autom-Comp-Req-012	자동차 소프트웨어 구조에서 RTE는 소프트웨어 컴포넌트와 BSW 사이의 유일한 인터페이싱 계층이 되어야 함	Auto.Func.Func.001 Auto.Func.Func.002
Autom-Comp-Req-013	자동차 소프트웨어 구조 RTE는 소프트웨어 컴포넌트들 사이와 소프트웨어 컴포넌트들과 BSW사이에서 외부 인터페이스들을 제공해야 함	Auto.Func.Func.003 Auto.Func.Comm.001 Auto.Func.Func.008
Autom-Comp-Req-014	자동차 소프트웨어 구조 RTE 인터페이스들은 주소들에 독립적이어야 함	Auto.Func.Comm.001
Autom-Comp-Req-015	자동차 소프트웨어 구조 RTE는 소프트웨어 컴포넌트들의 하나와 여러 개의 인스턴스 생성을 지원 하고 처리해야 함	Auto.Func.Comm.001 Auto.Func.Mng.002
Autom-Comp-Req-016	자동차 소프트웨어 구조는 각 BSW 계층으로부터 접근할 수 있는 서비스 계층을 제공해야 함	Auto.Func.Func.001 Auto.Func.Func.002 Auto.Func.Comm.001
Autom-Comp-Req-017	자동차 소프트웨어 구조 비휘발성 메모리 기능성은 하드웨어 의존과 독립 계층으로 나눠져야 함	Auto.Func.PhyCon.001, Auto.Comp.Const.004, Auto.Func.Func.006

3) 상호작용성(Interactivity)

자동차의 상호작용성은 사람(운전자)과 자동차 간의 인터페이스를 통한 상호작용과 자동차 내 ECU내에서 운영체제를 통한 응용 소프트웨어 간이나 자동차 내 하나의 ECU와 다른 ECU간 다양한 통신 프로토콜을 통한 유연한 통신 및 상호 작용이 요구됨을 뜻한다. 또한, 자율 주행 자동차의 경우, V2X 통신을 지원하기 위해 다양한 무선 통신 프로토콜을 이용한 상호작용이 요구된다.

자동차의 상호작용성 요구사항은 주로 Functional 측면(Aspect)와 연관됨을 알 수 있다. 총 30가지의 요구사항이 도출되었다.

<표 5-4> 자동차의 시스템 상호작용성 요구사항

요구사항 구분	요구사항 ID	설명	관심사(Concern) ID
Interactivity	Autom-Inter-Req-001	자율 주행 자동차 SW는 운전자가 인터페이스를 통해서 자동차의 액츄에이터를 동작시킬 수 있어야 함	Auto.Func.Act.001 Auto.Hum.HF.005
	Autom-Inter-Req-002	자율 주행 자동차 SW는 센싱을 통해서 운전자에게 경보 혹은 상황정보를 줄 수 있어야 함	Auto.Func.Act.002 Auto.Hum.HF.004 Auto.Hum.Usa.001 Auto.Hum.Usa.002
	Autom-Inter-Req-003	자율 주행 자동차 SW는 외부 인프라를 통해 차량의 운행을 지원받을 수 있어야 함	Auto.Func.Act.003
	Autom-Inter-Req-004	자동차 소프트웨어 OS는 OS 응용소프트웨어간의 통신을 지원해야 함	Auto.Func.Func.002 Auto.Func.Comm.001 Auto.Func.Comm.002 Auto.Data.Sem.001
	Autom-Inter-Req-005	자동차 소프트웨어 구조 통신은 CAN(Controller Area Network) 통신 버스를 지원해야 함	Auto.Func.Comm.001 Auto.Func.Comm.004
	Autom-Inter-Req-006	자동차 소프트웨어 구조 통신은 FlexRay[147]를 지원해야 함	Auto.Func.Comm.001 Auto.Func.Comm.004
	Autom-Inter-Req-007	자동차 소프트웨어 구조 통신은 LIN(Local Interconnect Network)을 지원해야 함	Auto.Func.Comm.001 Auto.Func.Comm.004
	Autom-Inter-Req-008	자동차 소프트웨어 구조 통신은 Ethernet을 지원해야 함	Auto.Func.Comm.001 Auto.Func.Comm.004
	Autom-Inter-Req-009	자동차 소프트웨어 구조는 높은 수준의 응용 통신 필요들을 표현을 허용하는 인터페이스를 제공해야 함	Auto.Func.Func.002 Auto.Func.Func.003
	Autom-Inter-Req-010	자동차 소프트웨어 구조는 ECU들의 시작과 종료를 지원해야 함	Auto.Func.Mng.002
	Autom-Inter-Req-011	자동차 소프트웨어 구조는 ECU들과 버스들의 sleep과 wakeup을 지원해야 함	Auto.Func.Mng.002
	Autom-Inter-Req-012	자동차 소프트웨어 구조는 그것과 상호작용하기 위한	Auto.Func.Mon.001

	외부 부트로더를 허용하는 인터페이스를 제공해야 함	
Autom-Inter-Req-013	자동차 소프트웨어 구조는 전력을 낭비하지 않기 위해 만약 아무 태스크와 인터럽트가 없다면 코어들을 끄는 것들을 지원할 수 있어야 함	Auto.Func.Func.001 Auto.Func.Per.001
Autom-Inter-Req-014	자동차 소프트웨어 구조 RTE는 브로드캐스트 신을 지원해야 함	Auto.Func.Comm.001 Auto.Func.Comm.002
Autom-Inter-Req-015	자동차 소프트웨어 구조 RTE는 프로시져 콜(클라이언트 서버 콜, 원격 프로시져 콜 등)을 지원해야 함	Auto.Func.Func.004 Auto.Func.Comm.001 Auto.Func.Comm.002
Autom-Inter-Req-016	자동차 소프트웨어 구조 통신은 SOME/IP(Scalable service-Oriented MiddlewarE over IP)를 지원해야 함	Auto.Func.Comm.004
Autom-Inter-Req-017	자동차 소프트웨어 구조 RTE는 소프트웨어 컴포넌트와 BSW 모듈들을 스케줄해야 함	Auto.Func.Func.001 Auto.Func.Comm.004
Autom-Inter-Req-018	자동차 소프트웨어 구조 서비스들은 통신 서비스들을 지원해야 함	Auto.Func.Func.002 Auto.Func.Comm.004
Autom-Inter-Req-019	자동차 소프트웨어 구조 통신은 시그널들을 필터링하는 것을 지원해야 함	Auto.Func.Comm.004
Autom-Inter-Req-020	자동차 소프트웨어 구조 통신은 시그널들을 위한 초기 값들을 지원해야 함	Auto.Func.Comm.004
Autom-Inter-Req-021	자동차 소프트웨어 구조 통신은 시그널들의 그룹들의 데이터 일관성을 지원해야 함	Auto.Func.Comm.004
Autom-Inter-Req-022	자동차 소프트웨어 구조 통신은 전송과 수신 취소를 지원해야 함	Auto.Func.Comm.004
Autom-Inter-Req-023	자동차 소프트웨어 구조 통신은 버스의 최대 전송 단위보다 큰 데이터 크기들의 전송을 지원해야 함	Auto.Func.Comm.004
Autom-Inter-Req-024	자동차 소프트웨어 구조 통신은 전용의 최적화된 모듈에서 크고 동적인 데이터의 통신을 지원해야 함	Auto.Func.Comm.004
Autom-Inter-Req-025	자동차 소프트웨어 구조 통신은 자동차에서 테스트를 위해 널리 사용되는 XCP 프로토콜을 지원해야 함	Auto.Func.Comm.004
Autom-Inter-Req-026	자동차 소프트웨어 구조 통신은 다른 망들 사이에서 글로벌 타임의 분산과 동기화를 지원해야 함	Auto.Timing.Aware.003 Auto.Timing.Aware.004 Auto.Timing.Aware.005 Auto.Func.Comm.004
Autom-Inter-Req-027	자동차 소프트웨어 구조 통신은 버스들의 상태 관리를 지원해야 함	Auto.Func.Comm.004, Auto.Func.Mng.002
Autom-Inter-Req-028	자동차 소프트웨어 구조 통신 상태 관리는 동적 버스 접근 제한을 지원해야 함	Auto.Func.Comm.004, Auto.Func.Mng.002
Autom-Inter-Req-029	자동차 소프트웨어 구조 통신 은 버스 통신이 활성화된 상태에서 노드들의 선택적인 정지를 지원해야 함	Auto.Func.Func.002 Auto.Func.Comm.004, Auto.Func.Mng.002
Autom-Inter-Req-030	자동차 소프트웨어 구조 SW 개발시 협력을 통한 개발이 될 수 있도록 표준화된 데이터 포맷과 절차를 거쳐야 함	Auto.Hum.HF.001 Auto.Hum.HF.002 Auto.Hum.HF.003 Auto.Trust.Priv.001 Auto.Data.Sem.002

4) 고신뢰성(Dependability)

자동차의 고신뢰성 요구사항은 자동차의 하드웨어 혹은 소프트웨어 컴포넌트가 어떤 시스템 결함이 있을 것을 대비해서 이중화되어 결함이 있더라도 복구시간이 없이 동작되게 하거나 시스템이 결함이나 손상이 있다하더라도 빠른 시간 혹은 시스템이나 응용에서 요구하는 시간 내에 복구하는 것을 말한다.

AUTOSAR 표준에서 정의된 자동차 소프트웨어 구조는 통신의 신뢰성을 위한 여분의 통신 링크를 지원해야하고, 가능한 정도의 모든 지역 에러 복구를 수행할 수 있어야 한다.

자동차의 고신뢰성 요구사항은 주로 Trustworthiness 측면(Aspect)와 연관된다. 27가지의 자동차 고신뢰 요구사항이 도출되었다.

<표 5-5> 자동차의 시스템 고신뢰성 요구사항

요구사항 구분	요구사항 ID	설명	관심사(Concern) ID
Dependability	Autom-Dep-Req-001	자동차 소프트웨어 구조 비휘발성 메모리 기능성은 하드웨어 신뢰성을 향상시키기 위해서 메커니즘을 제공해야 함	Auto.Trust.Rel.001 Auto.Data.OP.001 Auto.Data.OP.002 Auto.Data.OP.003
	Autom-Dep-Req-002	자동차 소프트웨어 구조는 하드웨어 통신 데이터 무결성 메커니즘을 사용해야 함	Auto.Trust.Safe.001 Auto.Trust.Safe.002 Auto.Trust.Safe.003
	Autom-Dep-Req-003	자동차 소프트웨어 구조는 여분의 여러 통신 링크들을 지원해야 함	Auto.Trust.Safe.001, Auto.Trust.Rel.001 Auto.Trust.Rel.002
	Autom-Dep-Req-004	자동차 소프트웨어 에러 핸들링은 의도한 행동으로부터 정의된 비정상적인 동작 상태들과 예기치 않은 예외들을 구별해야 함	Auto.Trust.Rel.001
	Autom-Dep-Req-005	자동차 소프트웨어 구조 진단 프로그램은 결함 상태의 발생을 기록하기 위해 중앙 저장 장치를 제공해야 함	Auto.Func.Mon.001
	Autom-Dep-Req-006	자동차 소프트웨어 구조의 기본 소프트웨어는 가능한 정도까지 모든 지역 에러 복구를 수행할 수 있어야 함	Auto.Trust.Rel.001 Auto.Trust.Res.001
	Autom-Dep-Req-007	자동차 소프트웨어 구조의 BSW와 RTE는 메모리 쓰기 보호를 지원해야 함	Auto.Trust.Safe.001 Auto.Data.OP.004 Auto.Data.OP.005
	Autom-Dep-Req-008	자동차 소프트웨어 구조 OS는 응용은 응용 소프트웨어와 BSW의 분리 및 보호를 지원해야 함	Auto.Trust.Safe.001, Auto.Func.Func.001

147) FlexRay 통신 버스는 자동차 제조업체와 주요 공급업체가 협력하여 개발된 결정성있는 내고장성(fault-tolerant) 고속 버스 시스템

	Autom-Dep-Req-009	자동차 소프트웨어 구조 OS는 BSW 모듈들 간의 분리 및 보호를 지원해야 함	Auto.Trust.Safe.001, Auto.Func.Func.001
	Autom-Dep-Req-010	자동차 소프트웨어 구조 OS는 응용에서의 에러를 만날 경우 만약 원한다면 응용 소프트웨어들의 종료와 재시작을 지원해야 함	Auto.Trust.Safe.001
	Autom-Dep-Req-011	자동차 소프트웨어 OS는 멀티코어 교착상태(deadlock) 프리 상호 배제를 지원해야 함	Auto.Func.Func.001
	Autom-Dep-Req-012	자동차 소프트웨어 구조 서비스들은 시스템 진단 기능성을 제공해야 함	Auto.Func.Mon.001 Auto.Trust.Rel.001
	Autom-Dep-Req-013	자동차 소프트웨어 구조 서비스들은 시스템 범위 암호 기능성을 제공해야 함	Auto.Trust.Secu.001
	Autom-Dep-Req-014	자동차 소프트웨어 구조 통신은 진단을 위해CAN을 통한 표준 전송 프로토콜을 지원해야 함	Auto.Func.Comm.004 Auto.Func.Mon.001
	Autom-Dep-Req-015	자동차 소프트웨어 구조 통신은 진단을 위해IP(인터넷 프로토콜)를 통한 표준 전송 프로토콜을 지원해야 함	Auto.Func.Comm.004 Auto.Func.Mon.001
	Autom-Dep-Req-016	자동차 소프트웨어 구조 비휘발성 메모리 기능성은 메모리 블록들의 무결성을 확보해야 함	Auto.Trust.Rel.001, Auto.Func.Func.006
	Autom-Dep-Req-017	자동차 소프트웨어 구조 추상화는 암호 하드웨어에 접근하는 것을 제공해야 함	Auto.Trust.Rel.001
	Autom-Dep-Req-018	자동차 소프트웨어 구조 IO 하드웨어 추상화는 불법적인 동작으로부터 하드웨어를 보호해야 함	Auto.Trust.Rel.001
	Autom-Dep-Req-019	자동차 소프트웨어 구조는 비인가된 읽기 접근으로부터 시스템을 보호하는 메커니즘을 제공해야 함	Auto.Trust.Secu.001
	Autom-Dep-Req-020	자동차 소프트웨어 구조는 비인가된 수정으로부터 시스템을 보호하는 메커니즘을 제공해야 함	Auto.Trust.Secu.001
	Autom-Dep-Req-021	자동차 소프트웨어 구조는 비인가된 사용으로부터 시스템을 보호하는 메커니즘을 제공해야 함	Auto.Trust.Secu.001
	Autom-Dep-Req-022	자동차 소프트웨어 구조는 소프트웨어 혹은 하드웨어에 의해 구현된 암호 솔루션들에의 획일적인 접근을 제공해야 함	Auto.Trust.Secu.001, Auto.Trust.Secu.003
	Autom-Dep-Req-023	자동차 소프트웨어 구조는 메시지 데이터 인증을 제공해야 함	Auto.Trust.Secu.002
	Autom-Dep-Req-024	자동차 소프트웨어 구조는 메시지의 신선도(Freshness) 인증을 지원해야 함	Auto.Trust.Secu.002
	Autom-Dep-Req-025	자동차 소프트웨어 구조는 메시지 데이터 무결성 인증을 제공해야 함	Auto.Trust.Secu.002
	Autom-Dep-Req-026	자동차 소프트웨어 구조 BSW와 RTE는 데이터 일관성을 보장해야 함	Auto.Trust.Secu.001
	Autom-Dep-Req-027	자동차 소프트웨어 구조는 안전을 증대시키기 위해 하드웨어 메모리 보호 기능들의 사용을 지원해야 함	Auto.Trust.Safe.001

5) 실시간성(Timing)

자동차의 실시간성 요구사항은 자동차의 구성 요소들이 동작하거나 통신하는 상황에서 실시간으로 동작되어야하는 응용이나 통신의 요구사항을 말한다.

자동차 소프트웨어 구조는 엄격한 시간을 요구하는 응용 지원 및 차량의 ECU간 실시간 연동할 수 있는 통신 기능이 요구되며 태스크의 우선순위에 따른 처리 기능과 동기화 기능이 제공되어야 한다.

자동차의 실시간성 요구사항은 주로 자동차의 Functional 측면(Aspect)와 Timing 측면(Aspect)의 관심사(Concern)들과 관련된다. 실시간성 요구사항 13종이 도출되었다.

<표 5-6> 자동차의 시스템 실시간성 요구사항

요구사항 구분	요구사항 ID	설명	관심사(Concern) ID
Timing	Autom-Timing-Req-001	자동차 소프트웨어 구조 통신은 TTCAN(Time Triggered Controller Area Network)을 지원해야 함	Auto.Func.Comm.004
	Autom-Timing-Req-002	자동차 소프트웨어 구조 서비스는 표준화된 와치독(Watchdog)의 처리를 지원해야 함	Auto.Trust.Rel.001
	Autom-Timing-Req-003	자동차 소프트웨어 구조 서비스는 상대적인 시간 측정을 위한 시간 서비스들을 지원해야 함	Auto.Func.Func.001, Auto.Func.PhyCon.001 Auto.Timing.Aware.005
	Autom-Timing-Req-004	자동차 소프트웨어 구조 통신은 time-out handling을 지원해야 함	Auto.Trust.Rel.001, Auto.Func.Comm.004 Auto.Timing.Interval.001
	Autom-Timing-Req-005	자동차 V2X 통신에서 차량 안전 서비스를 위한 상황정보는 100ms 주기의 차량 정보와 위치 정보를 보내줘야 함	Auto.Func.Comm.005 Auto.Timing.Mng.001
	Autom-Timing-Req-006	자동차 V2X 통신에서 협력 주행 제어를 위한 차량 정보는 10m 주기의 차량 정보와 위치 정보를 보내줘야 함	Auto.Func.Comm.005 Auto.Timing.Mng.001
	Autom-Timing-Req-007	자동차 V2I통신에서 차량은 도로 안전/협력주행제어를 위한 도로 상황 정보를 100ms 주기의 도로상황 MAP, 위치 보정정보를 수신하여야 함	Auto.Func.Comm.005
	Autom-Timing-Req-008	자동차 소프트웨어 구조는 실행 시간에 대한 인터럽트 응답시간의 트레이드를 위한 설정 파라미터들을 지원해야 함	Auto.Func.Func.001
	Autom-Timing-Req-009	자동차 소프트웨어 구조 OS는 엄격한 시간을 요하는 시스템을 위해 테스크들의 리스트들을 시작하는 것을	Auto.Func.Func.001

		지원해야 함	
	Autom-Timing-Req-010	자동차 소프트웨어 구조 OS는 스케줄 테이블들을 외부 시간 소스로 동기화하는 것을 지원해야 함	Auto.Func.Func.001 Auto.Timing.Sync.001
	Autom-Timing-Req-011	자동차 소프트웨어 구조 OS는 소프트웨어 컴포넌트들의 시간 측정을 허용하는 기능성을 제공해야 함	Auto.Func.Func.001
	Autom-Timing-Req-012	자동차 소프트웨어 구조 서비스들은 시스템 시간 서비스들을 지원해야 함	Auto.Comp.Const.002, Auto.Func.Func.005
	Autom-Timing-Req-013	자동차 소프트웨어 구조 비휘발성 메모리 기능성은 비휘발성 메모리 접근하는데 딜레이를 최소화하기 위해서 태스크들의 우선순위 화와 동기화를 지원해야 함	Auto.Trust.Rel.001

6) 상호운용성(Interoperability)

자동차의 상호운용성 요구사항은 현재의 자동차의 컴포넌트와 이전 버전이나 혹은 다른 표준을 사용하는 컴포넌트라고 할지라도 서로 호환이 되어야 하는 것이다.

AUTOSAR 표준의 경우 비표준 자동차 소프트웨어의 BSW를 지원하고 AUTOSAR OS는 OSEK OS와 하위호환성을 가진다. 자동차의 상호운용성 요구사항은 8종이 도출되었다.

<표 5-7> 자동차의 시스템 상호운용성 요구사항

요구사항 구분	요구사항 ID	설명	관심사(Concern) ID
Interoperability	Autom-Operate-Req-001	자동차 소프트웨어 구조 통신은 빅엔디안 데이터 표기와 리틀 엔디안 표기 사이의 데이터 변환을 지원해야 함	Auto.Func.Comm.004
	Autom-Operate-Req-002	자동차 소프트웨어 구조 RTE는 자동 재척도화(Re-scaling)과 포트 데이터 요소들의 변환을 제공해야 함	Auto.Func.Comm.001
	Autom-Operate-Req-003	자동차 소프트웨어 구조 통신은 2개의 CAN 버스 사이에서 IPDU(Inter-PDU) 게이트웨이를 지원해야 함	Auto.Func.Comm.004
	Autom-Operate-Req-004	자동차 소프트웨어 구조는 비표준 자동차 소프트웨어의 기본 소프트웨어(BSW) 모듈을 지원해야 함	Auto.Func.Func.001 Auto.Func.Comm.003 Auto.Comp.Complex.002
	Autom-Operate-Req-005	자동차 소프트웨어 구조는 다른 자동차 버전에서 같은 ECU를 가지고 사용하는 기본 소프트웨어 데이터의 리플레쉬를 지원해야 함	Auto.Func.Func.001 Auto.Func.Mng.001 Auto.Comp.Adapt.001 Auto.Comp.Complex.002
	Autom-Operate-Req-006	자동차 소프트웨어 구조는 소프트웨어 기능을 낮추기 위해 다른 표준화된 방법들을 지원해야 함	Auto.Func.Per.001, Auto.Func.Mng.002 Auto.Func.Comm.003 Auto.Comp.Adapt.001

	Autom-Operate-Req-007	자동차 소프트웨어 구조 SW와 OS는 이전버전 SW나 OSEK OS와 호환될 수 있도록 하위 호환성을 가져야 함	Auto.Func.Per.001, Auto.Func.Mng.002 Auto.Comp.Adapt.001.
	Autom-Operate-Req-008	자동차 소프트웨어 구조 RTE는 소프트웨어 컴포넌트들에 투명한 데이터 변환을 지원해야 함	Auto.Func.Comm.001

7) 지능성(Intelligence)

자동차의 지능성이란 시스템이 과거의 데이터나 최근의 이벤트를 통해서 시스템의 문제나 변화로 지능적으로 인식해서 그에 따라 적절하게 대처할 수 있는 능력을 요구하는 것을 말한다.

특히 자동차의 지능성은 차량 관리 시스템에서 ECU정보를 기반으로 현재 차량의 결함을 지능적으로 예측하거나 자율 주행 시스템에서는 현재 주변 도로나 운행 상황에 따라 지능적으로 운전에 도움을 주어야 한다.

자동차의 지능성 요구사항은 주로 자동차의 Functional 측면(Aspect)과 관련이 있다. 7종의 지능성 요구사항이 도출되었다.

<표 5-8> 자동차 CPS의 시스템 지능성 요구사항

요구사항 구분	요구사항 ID	설명	관심사(Concern) ID
Intelligence	Autom-Int-Req-001	자율 주행 시스템은 다양한 차량 센서들로 부터 상황 정보를 인지하여 그에 따른 동작에의 변동이 이루어져야 함	Auto.Func.Act.002 Auto.Func.Comm.004 Auto.Func.Ctrl.001 Auto.Func.Meas.001 Auto.Func.Sens.001 Auto.Trust.Safe.004 Auto.Trust.Safe.005 Auto.Timing.Aware.001
	Autom-Int-Req-002	차량 관리 시스템은 현재 차량의 ECU 정보들을 기반으로 차량의 결함을 지능적으로 예측해낼 수 있어야 함	Auto.Func.Comm.004
	Autom-Int-Req-003	자율 주행 시스템은 V2X시스템 등 통신 시스템을 기반으로 실시간 교통상황을 반영하여 지능적으로 주행할 수 있어야 함	Auto.Func.Comm.005 Auto.Func.Ctrl.001 Auto.Func.Ctrl.002 Auto.Timing.Aware.002
	Autom-Int-Req-004	자율 주행 시스템은 주변에 보행자가 있을 경우 센서로 정확히 감지하고 회피할 수 있도록 차량의 동작 변동이 이루어져야 함	Auto.Func.Act.002 Auto.Func.Act.003 Auto.Func.Ctrl.001 Auto.Trust.Safe.004 Auto.Trust.Safe.005
	Autom-Int-Req-005	자율 주행 시스템은 차량 운행 도중 앞에 나타나는 장애물이 있을 때 감지하고 경보하고 위험한 상황	Auto.Func.Act.001 Auto.Func.Act.002 Auto.Func.Act.003

		인 경우 직접적인 차량 동작을 변경시켜야 함	Auto.Func.Ctrl.001 Auto.Func.Meas.001 Auto.Hum.HF.004 Auto.Hum.Usa.002
	Autom-Int-Req-006	자율 주행 시스템은 운행 중 속도 제한 표지판이 있는 경우 이에 따라 경보하고 필요시 차량 속도를 제한할 수 있어야 함	Auto.Func.Act.002 Auto.Hum.Usa.002
	Autom-Int-Req-007	자율 주행 시스템은 각종 신호등과 표지판을 읽어서 운전자에게 잘 전달해주고 필요시 차량의 동작을 제어할 수 있어야 함	Auto.Hum.HF.004 Auto.Hum.HF.005 Auto.Hum.Usa.001 Auto.Func.Act.002

2. 자동차에서 CPS의 역할

자율주행 자동차를 지원하기 위한 자동차 소프트웨어 구조는 필수적으로 안전과 안정성을 위해 고신뢰성을 제공해야 하며 여러 기능을 확장성과 융복합적으로 제공할 뿐 아니라 차량내의 ECU들이 실시간으로 유연하게 연동되고, V2X를 통한 차량 외부와의 통신을 지원해야하고 지능제어 SW의 역할이 크게 요구된다.

1) CPS 프레임워크의 측면과 CPS 특성과의 관계

CPS 특성인 시스템 확장성(Scalability), 융복합성(Composability), 상호작용성(Interactivity), 고신뢰성(Dependability), 실시간성(Timing), 상호운용성(Interoperability), 지능성(Intelligence)과 CPS 프레임워크의 기능적 측면에 속하는 관심사들(Concerns)은 전체적으로 연관이 되어 있다. CPS 특성을 구현하기 위해서는 기능적 요소들이 지원되어야하기 때문이다. 특히 시스템 확장성(Scalability)은 기능적 측면들과 연결성이 가장 많은 CPS의 특성이다.

융복합성은 CPS 프레임워크의 조립(Composition) 측면의 관심사들과 연관이 지어진다. 상호작용성을 구현하기 위해서는 실시간(Timing) 측면에 대한 고려가 있어야 하며, 특히 자율자동화의 상호작용성을 구현하기 위해서는 운전자와의 상호작용을 고려하는 CPS 프레임워크의 인간적(Human) 측면을 고려해야 한다.

고신뢰성을 구현하기 위해서는 CPS 프레임워크의 신뢰성(Trustworthiness), 데이터(Data) 측면의 사용자 요구사항을 구현해야 한다. 실시간성은 CPS 프레임워크의 실시간(Timing)측면과 직접적인 연관이 있다.

상호운용성을 구현하기 위해서는 CPS 프레임워크의 조립(Composition) 측면과 기능적(Functional) 측면의 통신(Communication)에 대한 관심사와 관련된 요구사항을 고려해야 한다.

지능성은 주로 자율주행차와 연관된 특성이며, CPS 프레임워크의 신뢰성(Trustworthiness),

실시간(Timing), 인간적(Human) 측면의 요구사항들을 구현함으로써 나타난다.

2) 자동차에서 시스템 요구사항

다음은 자동차 시스템 요구사항과 그에 따라 적용된 예를 나타낸다.

(1) 확장성

자동차 소프트웨어 구조의 여러 개의 내·외부의 주변장치 연결과 소프트웨어 컴포넌트 지원 및 자동차 소프트웨어 구조 통신의 여러 차량 동시 접속 지원을 통한 확장성 확보된다.

예) AUTOSAR 표준에서는 MCU와 직접 연결되거나 I/O버스를 통해서 여러 개의 내외부 주변장치를 지원하고, 소프트웨어 컴포넌트(BSW)들의 여러 개의 인스턴스를 지원하고 처리한다.

(2) 융복합성

차량의 다양한 소프트웨어 컴포넌트들이 융복합하여 계층적인 자동차 소프트웨어 구조를 통해 ECU 하드웨어에 접근할 수 있도록 함으로서 기능 융복합을 가능케 한다.

예) AUTOSAR의 Runtime Environment(RTE)는 ECU 하드웨어 위에 있는 BSW계층과 응용 레벨의 소프트웨어 컴포넌트들 사이에 있는 계층으로서 인터페이스를 통해서 접근이 가능하다.

(3) 상호작용성

자동차의 하나의 ECU 내에서 OS를 통한 응용 간 그리고 차량의 하나의 ECU에서 다른 ECU간의 다양한 통신 프로토콜을 통한 유연한 통신 기능과 상호작용성이 요구되고 있다.

예) AUTOSAR의 통신은 ECU내 OS를 통한 응용 간 통신과 함께 CAN, FlexRay, LIN, Ethernet, SOME/IP를 통한 ECU간 혹은 ECU와 다른 디바이스 간 통신을 지원해야한다.

(4) 고신뢰성

자동차 소프트웨어 구조는 통신의 신뢰성을 위한 여분의 통신 링크를 지원해야하고, 가능한 정도의 모든 지역 에러 복구를 수행할 수 있어야 하며, 결함 발생이 생긴 경우 이를 기

록해야하며, 시스템 진단 기능성 제공과 진단을 위한 통신 프로토콜을 지원해야 한다.

예) AUTOSAR는 여분의 통신 링크를 지원하며, 진단 프로그램은 중앙 저장 장치를 통한 결함 상태 발생을 기록하고, AUTOSAR OS는 응용 소프트웨어와 BSW의 분리 및 보호를 지원하며 진단을 위한 CAN 및 IP를 통한 표준 전송 프로토콜을 지원해야 한다.

(5) 실시간성

자동차 소프트웨어 구조는 엄격한 시간을 요구하는 응용 지원 및 차량의 ECU간 실시간 연동할 수 있는 통신 기능이 요구되며 테스크의 우선순위에 따른 처리 기능과 동기화 기능이 제공되어야 한다.

예) AUTOSAR OS는 엄격한 시간을 요구하는 시스템을 위해 테스크들의 리스트들을 시작하는 것을 지원하고, 테스크들의 우선 순위화와 동기화를 지원해야 한다.

(6) 상호운용성

자동차 소프트웨어 구조는 비표준 소프트웨어의 기본 소프트웨어를 지원해야하며 자동차 소프트웨어 OS는 하위 호환성을 가져야 한다.

예) AUTOSAR는 비표준 자동차 소프트웨어의 BSW를 지원하고 AUTOSAR OS는 OSEK OS와 하위 호환성을 가진다.

(7) 지능성

차량 관리 시스템은 ECU 정보를 기반으로 현재 차량의 결함을 지능적으로 예측해낼 수 있어야 하고 자율 주행시스템은 차량 운전을 학습하여 지능적으로 운전할 수 있어야 한다.

예) AUTOSAR의 Adaptive 표준은 자율주행, V2X 네트워킹, 멀티미디어 응용들 그리고 over-the-air 업데이트를 제공하는 응용을 위한 새로운 표준이다.

제2절 CPS 관점의 항공기 요구사항 정의

이 절에서의 항공기 CPS 시스템은 요구사항 조사의 범위를 한정짓기 위해 무인항공기와 항공기 SW 플랫폼에 대해서만 요구사항을 도출하기로 한다.

시스템 확장성(Scalability) 요구사항은 항공기의 하나의 응용에서 서로 다른 응용을 구현하는데 필요한 요구사항을 정의하였다. 융복합성(Composability) 요구사항은 항공기 구성 요소들을 관리하는데 있어서 필요한 요구사항을 정리하였다. 상호연동성(Interactivity) 요구사항은 항공기를 구성하는 요소들과 조종사 사이에서 상호작용하기 위해 필요한 요구사항을 정의하였다. 고신뢰성(Dependability) 요구사항은 항공기가 고신뢰적인 동작을 하는데 필요한 요구사항을 정의하였다. 실시간성(Timing) 관련 요구사항은 항공기 시스템 내부 요소들 간 동작하는데 실시간성이 필요한 요소 및 동기화에 대해 정의하였다. 상호운용성(Interoperability) 관련 요구사항은 항공기 시스템 내에서 기기들 사이의 의사소통하기 위해 필요한 요구사항 및 원활한 동작을 위한 요구사항을 정의하였다. 마지막으로 지능성(Intellegence) 관련 요구사항은 항공기 시스템의 지능적인 동작을 기능 별로 나누어 요구사항을 정의하였다.

각 항목별로 시스템 확장성 8종, 융복합성 8종, 상호작용성 10종, 고신뢰성 18종, 실시간성 7종, 상호운용성 5종 그리고 지능성 6종이 도출되어 총 62종의 항공기 시스템 요구사항이 도출되었다. 도출된 요구사항은 자동차 아이디와 유사하게 Aero-XXX-Req-### 형태의 아이디를 가지게 된다.

1. 요구사항 명세

1) 시스템 확장성(Scalability)

항공기의 시스템 확장성은 하나의 응용에서 다양한 서로 다른 응용 기능들을 구현 할 수 있는 파티션 기능을 통해 제공한다.

현재 무인항공기와 자동차 분야 등으로 RTOS(Real-time OS)가 필요한 분야에서 ARINC 653 플랫폼이 이미 많이 사용 하고 있고, 계획되고 있다. 항공기 분야에서 8종의 시스템 확장성 요구사항이 도출되었다.

<표 5-9> 항공기의 시스템 확장성 요구사항

요구사항 구분	요구사항 ID	설명	관심사(Concern) ID
Scalability	Aero-Scal-Req-001	무인항공기의 제한된 가반하중의 한계점을 개선할 기술이 필요함	Aero.Hum.HF.005
	Aero-Scal-Req-002	항공기 소프트웨어(ARINC 653) 플랫폼은 하나의 컴퓨터에서 여러 응용을 다룰 수 있어야 함	Aero.Func.Func.002 Aero.Func.Mng.001 Aero.Trust.Safe.006
	Aero-Scal-Req-003	항공기 소프트웨어(ARINC 653) 플랫폼은 무인 항공기 도메인 등, 다른 종류의 항공기에도 적용이 가능해야 함	Aero.Func.Func.004
	Aero-Scal-Req-004	항공기 소프트웨어(ARINC 653) 플랫폼은 자동차 분야 등의 여러 도메인에서 사용 하도록 되어야 함	Aero.Func.Func.004
	Aero-Scal-Req-005	항공기의 IMA 아키텍처에서는 여러 어플리케이션이 동일한 컴퓨팅 리소스를 공유하고 재사용 할 수 있어야 함	Aero.Data.Relat.001
	Aero-Scal-Req-006	항공기 소프트웨어(ARINC 653) 플랫폼은 다른 상용화 OS에서의 적용이 가능하도록 해야 함	Aero.Comp.Complex.001
	Aero-Scal-Req-007	항공기 소프트웨어(ARINC 653) 플랫폼은 대부분의 서로 다른 항공기체에서 사용될 수 있어야 함	Aero.Comp.Complex.001
	Aero-Scal-Req-008	항공기 소프트웨어(ARINC 653) 플랫폼은 많은 데이터량을 처리할 수 있어야 함	Aero.Data.Vol.001

2) 융복합성(Composability)

여러 다양한 환경이나 여러 상황에서 필요한 기술을 융복합하여 항공기가 유연하게 동작하기 위한 요구사항들을 정의했다. 항공기에 관련된 융복합성 요구사항은 7종이 도출되었다.

<표 5-10> 항공기의 시스템 융복합성 요구사항

요구사항 구분	요구사항 ID	설명	관심사(Concern) ID
Composability	Aero-Comp-Req-001	항공기 소프트웨어(ARINC 653) 플랫폼은 오류 처리를 위해 콜백 함수들을 제공해야 함	Aero.Func.Func.003 Aero.Func.Mng.007 Aero.Func.Mng.008 Aero.Func.Mng.009 Aero.Func.Mng.010 Aero.Func.Mon.001
	Aero-Comp-Req-002	무인항공기는 자동 호버링(Auto hovering)[148]으로부터 경유지점 비행, 궤적 추적 비행, 자동 귀환 등 자율 비행을 위해서는 무인항공기의 공간상 위치 추정이 필요함	Aero.Func.Phycon.002
	Aero-Comp-Req-003	무인항공기는 원하는 방향으로 비행하거나 추락을 피하기 위해서는 무인항공기의 회전과 제어기술에 대해 알아야 함	Aero.Func.Meas.001 Aero.Func.Meas.002 Aero.Func.Phycont.001 Aero.Func.Phy.001
	Aero-Comp-Req-004	무인항공기의 IT 기술과 타 산업간 융복합 시에 발생되는 보안 위협을 해결하기 위한 보안 기술 필요	Aero.Trust.Secu.003

	Aero-Comp-Req-005	항공기 소프트웨어(ARINC 653) 플랫폼은 추상화 계층을 제공해서 POSIX를 지원하는 OS뿐이 아닌, 다양한 OS에서도 적용되어야 함	Aero.Func.Func.004
	Aero-Comp-Req-006	무인항공기는 고객의 요구에 따라 제품을 만들기 위해서 하드웨어 모듈을 쉽게 추가, 제거할 수 있어야 함	Aero.Comp.Complex.001
	Aero-Comp-Req-007	무인항공기는 하드웨어 모듈의 추가 또는 제거가 제대로 이루어지지 않고 동작하고자 할 때, 그에 맞는 조치를 할 수 있어야 함	Aero.Func.Mng.009
	Aero-Comp-Req-008	무인항공기의 여러 상황에서 필요한 기술들의 연구가 지속되어야 함	Aero.Func.Uncert.001 Aero.Func.Uncert.002 Aero.Func.Uncert.003

3) 상호작용성(Interactivity)

무인항공기와 사람간의 통신이 이루어져 사람이 이동하기에 제약이 있는 곳을 갈 수 있게 되고, 항공기와 다른 사물간의 유연한 통신기능을 제공하기 위한 정밀한 SW기능이 요구된다. 카메라 기능을 가진 드론이 실제로 사람이 가기 어려운 환경에서 사용 되고, 현재 무인항공택시 개발하는 기업들이 있는데, 이를 개발하기 위해서 주변 장애물들에 대한 정보들을 얻고 반응할 수 있는 SW가 필요하다. 9종의 상호작용성 요구사항이 도출되었다.

<표 5-11> 항공기의 시스템 상호작용성 요구사항

요구사항 구분	요구사항 ID	설명	관심사(Concern) ID
Interactivity	Aero-Inter-Req-001	무인항공기와 지상 컴퓨팅 환경(외부 환경)을 연결해 수집 정보를 교환하는 기술이 필요함	Aero.Timing.Sync.002 Aero.Comp.Discover.002 Aero.Comp.Discover.003 Aero.Comp.Discover.004 Aero.Data.Id.001
	Aero-Inter-Req-002	항공기 소프트웨어(ARINC 653) 플랫폼은 파티션 간 통신이 이루어져야 함	Aero.Func.Comm.001 Aero.Func.Comm.002 Aero.Func.Comm.003 Aero.Func.Comm.004 Aero.Data.OP.001 Aero.Data.OP.003 Aero.Data.OP.004
	Aero-Inter-Req-003	항공기 소프트웨어(ARINC 653) 플랫폼은 파티션 내 통신이 이루어져야 함	Aero.Func.Comm.005 Aero.Func.Mng.005 Aero.Func.Mng.006 Aero.Timing.Sync.001
	Aero-Inter-Req-004	무인항공기 서버 간 통신 보안 기술 필요	Aero.Trust.Secu.004
	Aero-Inter-Req-005	항공기 소프트웨어의 서비스들은 통신 서비스들을 지원해야 함	Aero.Func.Comm.001 Aero.Func.Comm.002 Aero.Data.OP.002

148) 사용자가 조정기에서 손을 떼도 자동으로 공중에 떠 있는 기능

	Aero-Inter-Req-006	항공기 소프트웨어의 시그널들을 필터링하는 것을 지원해야 함	Aero.Func.Comm.001
	Aero-Inter-Req-007	항공기 소프트웨어의 통신은 시그널들을 위한 초기 값들을 지원해야 함	Aero.Func.Comm.001
	Aero-Inter-Req-008	항공기 소프트웨어의 통신은 시그널들의 그룹들의 데이터 일관성을 지원해야 함	Aero.Func.Comm.001
	Aero-Inter-Req-009	항공기 소프트웨어의 통신은 전송과 수신 취소를 지원해야 함	Aero.Func.Comm.001
	Aero-Inter-Req-010	무인항공기는 사용자 편의를 위한 응용서비스를 지원가능해야 함	Aero.Comp.Adapt.001 Aero.Comp.Adapt.002

4) 고신뢰성(Dependability)

항공기는 하드웨어, 운영체제와 응용프로그램에서 오류가 발생했을 때, 자동적으로 회복시켜주는, 오류를 처리 해주는 인프라 기술인 헬스모니터링을 보유하여 오류를 해결할 수 있어야 한다. 17종의 고신뢰성 요구사항이 도출되었다.

<표 5-12> 항공기의 시스템 고신뢰성 요구사항

요구사항 구분	요구사항 ID	설명	관심사(Concern) ID
Dependability	Aero-Dep-Req-001	항공기 소프트웨어(ARINC 653) 플랫폼은 비휘발성 메모리 기능성은 하드웨어 신뢰성을 향상시키기 위해서 메커니즘을 제공해야 함	Aero.Data.Vel.001 Aero.Data.Vel.002
	Aero-Dep-Req-002	항공기 소프트웨어(ARINC 653) 플랫폼의 에러 핸들링은 의도한 행동으로부터 정의된 비정상적인 동작 상태들과 예기치 않은 예외들을 구별해야 함	Aero.Func.Func.003 Aero.Func.Mng.007 Aero.Func.Mng.008 Aero.Func.Mng.010 Aero.Func.Mon.001 Aero.Func.Mon.002 Aero.Trust.Rel.001
	Aero-Dep-Req-003	항공기 소프트웨어(ARINC 653) 플랫폼은 시스템 설정 시간에 정적으로 정해진 파티션 수행 시간 동안 다른 파티션의 방해를 받지 않도록 보장되어야 함	Aero.Func.Func.001 Aero.Func.Func.002 Aero.Func.Mng.001 Aero.Func.Mng.002 Aero.Func.Usa.002 Aero.Timing.Aware.001 Aero.Timing.Aware.002
	Aero-Dep-Req-004	항공기 소프트웨어(ARINC 653) 플랫폼은 파티션 내의 프로세스들이 수행되는 메모리 영역을 다른 파티션들이 침범하지 못하도록 보장해야 함	Aero.Func.Func.001 Aero.Func.Func.002 Aero.Func.Mng.001 Aero.Func.Mng.003 Aero.Func.Mng.004 Aero.Func.Usa.002
	Aero-Dep-Req-005	무인항공기로 영상촬영 할 때, 카메라 인식이 잘 작동 되어야 함	Aero.Hum.HF.001 Aero.Hum.HF.002 Aero.Hum.HF.003 Aero.Hum.HF.004 Aero.Trust.Priv.001 Aero.Data.Id.002 Aero.Data.Id.003

			Aero.Comp.Adapt.001
	Aero-Dep-Req-006	무인항공기의 컴퓨터 또는 네트워크상의 정보의 훼손, 변조, 유출 등을 방지하기 위한 보안 기술 필요	Aero.Trust.Secu.001 Aero.Trust.Secu.004
	Aero-Dep-Req-007	항공기 소프트웨어 OS는 응용에서의 에러를 만날 경우 만약을 대비해 응용 소프트웨어들의 종료와 재시작을 지원해야 함	Aero.Trust.Safe.001
	Aero-Dep-Req-008	항공기 빅데이터 관리는 서버로부터 받아온 제어 메시지를 각각에 적합하게 해석할 수 있어야 함	Aero.Func.Sens.001 Aero.Func.Mng.011 Aero.Timing.Aware.003 Aero.Data.Relat.001
	Aero-Dep-Req-009	항공기 소프트웨어(ARINC 653) 플랫폼의 비휘발성 메모리 기능성은 메모리 블록들의 무결성을 확보해야 함	Aero.Trust.Rel.001
	Aero-Dep-Req-010	항공기 소프트웨어의 추상화는 암호 하드웨어에 접근하는 것을 제공해야 함	Aero.Trust.Rel.001
	Aero-Dep-Req-011	항공기 소프트웨어의 I/O 하드웨어 추상화는 불법적인 동작으로부터 하드웨어를 보호해야 함	Aero.Trust.Rel.001
	Aero-Dep-Req-012	항공기 소프트웨어는 미검증된 접근으로부터 시스템을 보호하는 메커니즘을 제공해야 함	Aero.Trust.Secu.001 Aero.Trust.Secu.004
	Aero-Dep-Req-013	항공기 소프트웨어는 미검증된 수정으로부터 시스템을 보호하는 메커니즘을 제공해야 함	Aero.Trust.Secu.001
	Aero-Dep-Req-014	항공기 소프트웨어는 미검증된 사용으로부터 시스템을 보호하는 메커니즘을 제공해야 함	Aero.Trust.Secu.001
	Aero-Dep-Req-015	항공기 소프트웨어 구조는 메시지 데이터 무결성 인증을 제공해야 함	Aero.Func.Ctrl.001
	Aero-Dep-Req-016	항공기 소프트웨어 구조는 메시지 데이터 인증을 제공해야 함	Aero.Func.Ctrl.001
	Aero-Dep-Req-017	항공기 소프트웨어 구조는 안전을 증대시키기 위해 하드웨어 메모리 보호 기능들의 사용을 지원해야 함	Aero.Trust.Safe.004
	Aero-Dep-Req-018	항공기 소프트웨어 플랫폼은 오류를 효율적으로 관리하기 위한 기술을 지원해야 함	Aero.Trust.Rel.001 Aero.Trust.Res.001 Aero.Trust.Res.002

5) 실시간성(Timing)

항공기의 제어 기능과 기체 상태확인 등을 실시간 연동할 수 있는 기능이 필요하고, 원격 제어에도 실시간 연동 기능이 제공 되어야 하기에, 실시간 운영체제와 시간 동기화 기술 등이 제공 되어야 한다. 7종의 실시간성 요구사항이 도출되었다.

<표 5-13> 항공기의 시스템 실시간성 요구사항

요구사항 구분	요구사항 ID	설명	관심사(Concern) ID

Timing	Aero-Timing-Req-001	항공기 소프트웨어(ARINC 653) 플랫폼은 표준화된 와치독(Watchdog)의 처리를 지원해야함	Aero.Func.Func.003 Aero.Func.Mng.007 Aero.Func.Mng.010 Aero.Func.Mon.001
	Aero-Timing-Req-002	항공기 소프트웨어(ARINC 653) 플랫폼은 내부 테스트 파티션 스케줄링과 정상모드 파티션 스케줄링 두 가지 모드로 파티셔닝 스케줄링이 구현 될 필요가 있음	Aero.Timing.Aware.002
	Aero-Timing-Req-003	무인항공기의 영상 촬영은 실시간으로 처리 되어야 함	Aero.Data.Id.003
	Aero-Timing-Req-004	무인항공기에서 다중 프로젝션 지원을 위해서 이를 지원하는 미디어 서버와 이를 운용하는 서버간 동기화가 필요	Aero.Trust.Sync.002
	Aero-Timing-Req-005	항공기 소프트웨어(ARINC 653) 플랫폼은 파티션 내 통신에서, 동기화를 위해 세마포어와 이벤트를 제공해야 함	Aero.Timing.Sync.002
	Aero-Timing-Req-006	항공기의 IMA 아키텍처에서는 레거시 코드 재사용으로 개발 기간을 단축 할 수 있어야 함	Aero.Comp.Complex.001 Aero.Comp.Complex.002
	Aero-Timing-Req-007	항공기 빅데이터 분석을 통해 실시간으로 결함 상태 등의 현황을 알 수 있어야 함	Aero.Data.Relat.001 Aero.Comp.Discover.001

6) 상호운용성(Interoperability)

항공기는 일반적인 항공기 뿐 아니라 여러 다른 기종의 항공기에서도 사용이 가능해야 한다.

ARINC 653 플랫폼은 일반적인 항공기 뿐 아니라, 헬리콥터, 무인항공기와 자동차 분야 등에서도 사용 될 수 있다. 항공기의 상호운용성 요구사항은 5개가 도출되었다.

<표 5-14> 항공기의 시스템 상호운용성 요구사항

요구사항 구분	요구사항 ID	설명	관심사(Concern) ID
Interoperability	Aero-Operate-Req-001	항공 빅데이터는 MRO(Maintenance, Repair, and Overhaul)를 수행함에 있어 시스템은 어떠한 인터넷 환경에도 접속할 수 있는 웹 인터페이스를 지원할 수 있어야 함	Aero.Func.Per.001 Aero.Func.Per.002 Aero.Data.Vol.002
	Aero-Operate-Req-002	항공기 소프트웨어(ARINC 653) 플랫폼은 불필요한 파라미터 형식의 변동을 줄이고 통상적인 I/O 프로세싱을 줄여, 응용프로그램의 효율을 향상시키고, 프로그램 이식성을 높여야 함	Aero.Comp.Complex.001 Aero.Comp.Complex.002 Aero.Comp.Complex.003 Aero.Data.Vol.002
	Aero-Operate-Req-003	항공기 소프트웨어(ARINC 653) 플랫폼은 자동차 분야 등의 여러 도메인에서 사용 하도록 되어야 함	Aero.Func.Func.004
	Aero-Operate-Req-004	항공기 소프트웨어(ARINC 653) 플랫폼은 다른 상용화 RTOS에서의 적용이 가능하도록 해야 함	Aero.Func.Func.004

	Aero-Operate-Req-005	항공기 소프트웨어(ARINC 653) 플랫폼은 여러 다른 기체에서의 서브시스템에 적용 가능해야 함	Aero.Func.Func.004

7) 지능성(Intelligence)

항공기는 다양한 결함 상태를 빅데이터 처리와 분석으로 예측할 수 있어야 하고, 인공지능 기술을 사용하여 무인항공기에서 주변 상황 인지를 실시간으로 적용 할 수 있는 기능이 요구된다. 6종의 항공기 지능성 요구사항이 도출되었다.

<표 5-15> 항공기의 시스템 지능성 요구사항

요구사항 구분	요구사항 ID	설명	관심사(Concern) ID
Intelligence	Aero-Int-Req-001	항공기 소프트웨어(ARINC 653) 플랫폼은 오류 발생 시, 지능적으로 처리할 수 있는 기능 필요함	Aero.Func.Func.003 Aero.Func.Mng.007 Aero.Func.Mng.010 Aero.Func.Mon.001 Aero.Trust.Safe.005 Aero.Trust.Res.001 Aero.Trust.Res.002
	Aero-Int-Req-002	항공기 빅데이터 분석으로 다양한 기체 부품의 결함을 예측할 수 있어야 함	Aero.Func.Sens.001 Aero.Func.Mng.011 Aero.Timing.Aware.003 Aero.Func.Per.003 Aero.Comp.Discover.001
	Aero-Int-Req-003	항공기 소프트웨어 인증 표준(DO-178)에서 항공기가 다양한 상황에서 발생 가능한 위험과 그로인한 손실을 예측하며 분석을 수행할 수 있어야 함	Aero.Trust.Safe.002 Aero.Trust.Safe.003
	Aero-Int-Req-004	항공기 빅데이터 관리에서 필요한 데이터 소스가 발생했을 때, 지능적으로 업데이트 될 수 있는 전송망을 갖춰야 함	Aero.Data.Relat.001 Aero.Func.Per.001 Aero.Func.Per.002
	Aero-Int-Req-005	무인항공기는 각 센서들의 정보들을 통해 주변 환경을 인지해야 함	Aero.Data.Id.002 Aero.Hum.HF.002 Aero.Hum.HF.003 Aero.Hum.HF.004
	Aero-Int-Req-006	무인항공기는 주변 환경을 인지하고 지능적으로 상황에 맞춰 변화를 줄 수 있어야 함	Aero.Func.Act.001 Aero.Func.Act.002

2. 항공기에서 CPS의 역할

원활한 편리성과 안정성을 지원하기 위한 항공 분야에서는 다양한 응용 기능들이 실시간으로 연동되고, 지능적으로 물리시스템을 인지하고, 제어해야 하는 복합 지능제어SW의 역할이 요구된다.

1) CPS 프레임워크의 측면과 CPS 특성과의 관계

자동차에서와 마찬가지로 CPS 특성은 CPS 프레임워크의 기능적 측면에 속하는 관심사들(Concerns)은 전체적으로 연관이 되어 있다. CPS 특성을 구현하기 위해서는 기능적 요소들이 지원되어야하기 때문이다.

그러나 자동차 시스템 요구사항에서와는 다르게 항공 시스템 요구사항 중 시스템 확장성(Scalability)에 속하는 요구사항은 CPS 프레임워크의 데이터(Data), 조립(Composition) 측면과 연관이 있다. 이는 항공 표준인 ARINC 653이 지원하는 파디셔닝(partitioning, 시간과 공간 분할) 기능을 구현하기 위해서는 데이터 측면의 공유가 필요하기 때문이다. 또한 ARINC 653으로 구현된 플랫폼은 서로 다른 항공기체에서 사용되기 때문에 조립 측면도 고려가 되어야 한다.

융복합성(Composability)은 자동차와 동일하게 CPS 프레임워크의 조립(Composition) 측면의 관심사들(Concerns)과 연관이 지어진다. 상호작용성(Interactivity)을 구현하기 위해서는 실시간(Timing) 측면에 대한 고려가 있어야 하며, 파디셔닝 기능으로 인해 데이터 측면에 대한 고려도 필요하다.

고신뢰성(Dependability)을 구현하기 위해서는 CPS 프레임워크의 신뢰성(Trustworthiness), 데이터(Data) 측면의 사용자 요구사항을 구현해야 한다. 무인항공기의 경우는 인간적 측면도 고려가 되어야 한다. 실시간성(Timing)은 CPS 프레임워크의 실시간(Timing)측면과 직접적인 연관이 있으며, 무인항공기의 경우는 데이터 측면도 고려대상이다.

상호운용성(Interoperability)을 구현하기 위해서는 CPS 프레임워크의 조립(Composition) 측면과 관련된 요구사항을 고려해야 한다.

지능성(Intelligence)은 빅데이터 분석과 무인항공기와 연관된 특성이며, CPS 프레임워크의 신뢰성(Trustworthiness), 인간적(Human), 데이터(Data) 측면의 요구사항들을 구현함으로써 나타난다.

2) 항공기에서 시스템 요구사항

다음은 항공기 요구사항과 그에 따라 적용된 예를 나타낸다.

(1) 확장성

하나의 응용에서 다양한 서로 다른 응용 기능들을 구현 할 수 있는 파티션 기능을 제공하므로, 확장성을 보장한다.

예) 현재 무인항공기와 자동차 분야 등으로 RTOS가 필요한 분야에서 ARINC 653 플랫폼이 많은 적용 및 연구되고 있다.

(2) 융복합성

다양한 분야와 상황 등에서 항공기를 융복합하여 사용하기에 많은 제약이 있었던 부분에서, 소프트웨어 기반으로 다양한 요소들을 융복합을 가능하게 한다.

(3) 상호작용성

무인항공기와 사람간의 통신이 이루어져 사람이 이동하기에 제약이 있는 곳을 갈 수 있게 되고, 항공기와 다른 사물간의 유연한 통신기능을 제공하기 위한 정밀한 소프트웨어 기능이 요구된다.

예) 카메라 기능을 가진 드론이 실제로 사람이 가기 어려운 환경에서 사용 되고, 현재 무인항공택시 개발하는 기업들이 있는데, 이를 개발하기 위해서 주변 장애물들에 대한 정보들을 얻고 반응할 수 있는 소프트웨어가 필요하다.

(4) 고신뢰성

하드웨어, 운영체제와 응용프로그램에서 오류가 발생했을 때, 자동적으로 회복시켜주는, 오류를 처리 해주는 인프라 기술인 헬스모니터링을 보유하여 오류를 해결할 수 있다.

예) ARINC 653의 헬스 모니터링은 오류를 격리하고 오류의 확산을 방지하고, 오류를 3개의 레벨로 분류하여 처리 및 관리한다.

(5) 실시간성

항공기의 제어 기능과 기체 상태확인 등을 실시간 연동할 수 있는 기능이 필요하고, 원격 제어에도 실시간 연동 기능이 제공 되어야 하기에, 실시간 운영체제와 시간 동기화 기술 등이 제공 되어야 한다.

(6) 상호운용성

ARINC 653 플랫폼은 일반적인 항공기 뿐 아니라, 헬리콥터, 무인항공기와 자동차 분야 등에서도 사용 될 수 있다.

예) ARINC 653 기반인 윈드리버 기업에서 개발한 VxWorks 653 플랫폼을 에어버스의 헬리오닉스(헬리콥터)에 적용하였다.

(7) 지능성

다양한 결함 상태를 빅데이터 처리와 분석으로 예측할 수 있어야 하고, 인공지능 기술을 사용하여 무인항공기에서 주변 상황 인지를 실시간으로 적용 할 수 있는 기능이 요구된다.

예) 유비파이 기업에서는 장착된 카메라로부터 영상 기반의 실시간 실내 위치 인식 기술을 통해 영상 내 물체를 인지 및 추적하거나 실내외에서 지정된 위치로 이동하는 등, 기존에 불가능했던 기능을 연구 중에 있다.

제3절 스마트공장 요구사항 정의

이 절에서는 2장에서 정의한 CPS 프레임워크의 6개 측면들(Aspect), 관심사들(Concerns) 로 분류된 사용자 요구사항을 이용하여 CPS 특성별로 스마트공장이 가져야할 CPS 시스템 요구사항을 도출하였다.

시스템 확장성(Scalability)으로 분류된 요구사항은 스마트공장이 내·외부 기기와 연동하여 그 규모를 적합하게 확장하거나 축소하는데 필요한 요구사항을 정의하였다. 융복합성(Composability)에 속하는 요구사항은 스마트공장 내의 구성 요소들을 관리하는데 있어서 필요한 요구사항을 정리하였다. 상호연동성(Interactivity) 관련 요구사항은 스마트공장 내에 있는 기기들과 현장기술자 사이의 의사소통하기 위해 필요한 요구사항을 정의하였다. 고신뢰성(Dependability) 관련 요구사항은 스마트공장이 고신뢰적인 동작을 하는데 필요한 요구사항을 정의하였다. 실시간성(Timing) 관련 요구사항은 스마트공장 간 동작하는데 실시간성이 필요한 요소 및 동기화에 대해 정의하였다. 상호운용성(Interoperability) 관련 요구사항은 스마트공장에서 기기들 사이의 의사소통하기 위해 필요한 요구사항 및 원활한 동작을 위한 요구사항을 정의하였다. 마지막으로 지능성(Intellgence) 관련 요구사항은 스마트공장이 지능적인 동작을 기능 별로 나누어 요구사항을 정의하였다.

각 항목별로 시스템 확장성 13종, 융복합성 20종, 상호작용성 17종, 고신뢰성 20종, 실시간성 12종, 상호운용성 12종 그리고 지능성 8종이 도출되어 총 102종의 시스템 요구사항이 도출되었다. 도출된 요구사항은 SF-XXX-Req-### 형태의 아이디를 가지게 된다.

1. 요구사항 명세

1) 시스템 확장성(Scalability)

시스템 확장성은 스마트공장에 추가되거나 제거되는 기기와의 연동 및 유연한 변화를 위한 요구사항을 정의하였다.

CPS 프레임워크의 Functional 측면(Aspect)이 반영되어 시스템 확장성이 정의되었다. 13종의 스마트공장 시스템 확장성 요구사항이 정의되었다.

<표 5-16> 스마트공장의 시스템 확장성 요구사항

요구사항 구분	요구사항 ID	설명	관련 관심사(Concern) ID
Scalability	SF-Scal-Req-001	스마트공장에서 현재 공장의 기기 설비들의 연동 구조를 실시간으로 모니터링 할 수 있어야 함	SF.Func.Act.002 SF.Func.Mng.001 SF.Func.Mon.001 SF.Func.Mon.002
	SF-Scal-Req-002	스마트공장 시스템에 기기가 추가되거나 제거 또는 변경될 때 다른 기기들 간의 연결에는 영향을 주지 않아야 함	SF.Func.Act.002 SF.Func.Phy.001
	SF-Scal-Req-003	스마트공장 시스템은 기존에 사용되던 클라이언트/서버 통신 기술을 지원해야 함	SF.Func.Comm.001 SF.Func.Comm.002
	SF-Scal-Req-004	스마트공장 시스템은 Pub/Sub 모델의 통신 기술을 지원해야 함	SF.Func.Comm.001 SF.Func.Comm.002
	SF-Scal-Req-005	새로이 추가된 기기들은 Pub/Sub 모델의 통신 기술을 통하여 다른 시스템의 변경 없이 데이터를 보낼 수 있어야 함	SF.Func.Comm.001 SF.Func.Comm.002 SF.Func.Comm.003 SF.Func.Comm.004
	SF-Scal-Req-006	새로이 추가된 기기들은 Pub/Sub 모델의 통신 기술을 통하여 다른 시스템의 변경 없이 데이터를 받을 수 있어야 함	SF.Func.Comm.001 SF.Func.Comm.002 SF.Func.Comm.003 SF.Func.Comm.004
	SF-Scal-Req-007	스마트공장의 네트워크 연결 구조는 전체 팩토리의 변동에 따라 적합하게 변동될 수 있어야 함	SF.Func.Func.001 SF.Func.Func.002 SF.Func.Func.003
	SF-Scal-Req-008	스마트공장에서 제어 시스템의 구조 및 기능은 전체 팩토리의 변동에 따라 적합하게 변동될 수 있어야 함	SF.Func.Func.001 SF.Func.Func.002 SF.Func.Mng.003 SF.Func.Perform.001
	SF-Scal-Req-009	스마트공장 정보 시스템의 구조 및 기능은 전체 팩토리의 변동에 따라 적합하게 변동될 수 있어야 함	SF.Func.Func.001 SF.Func.Func.002 SF.Func.Mng.003
	SF-Scal-Req-010	스마트공장의 제어 메시지는 모든 기기에서 동일한 방법으로 해석될 수 있어야 함	SF.Func.Comm.003
	SF-Scal-Req-011	스마트공장에서 기기 설비들의 상태를 고려하여 공정을 설정할 수 있어야 함	SF.Func.Act.003 SF.Func.Ctrl.003 SF.Func.Ctrl.004 SF.Func.Func.001 SF.Func.Mon.001 SF.Func.Mng.002 SF.Func.Meas.002
	SF-Scal-Req-012	스마트공장에서 공정 설정이 변경되어 기기 간 연동 구조가 바뀔 수 있어야 함	SF.Func.Func.001 SF.Func.Mng.001
	SF-Scal-Req-013	스마트공장 구성요소의 추가/제거가 용이하도록 분산 M&S 환경이 구성되어야 함	SF.Func.Func.004 SF.Func.Meas.005

2) 융복합성(Composability)

다양한 이기종 기기가 포함되어 있는 스마트공장이 유연하게 동작하기 위해 필요한 기능들을 융복합성 요구사항을 통하여 정의하였다.

CPS 프레임워크의 Composition 측면(Aspect)이 반영되어 20종의 융복합성 요구사항이 정의되었다.

<표 5-17> 스마트공장의 시스템 융복합성 요구사항

요구사항 구분	요구사항 ID	설명	관련 관심사(Concern) ID
Composability	SF-Com-Req-001	스마트공장은 원격으로 S/W업그레이드를 할 수 있어야 함	SF.Com.Adapt.001
	SF-Com-Req-002	스마트공장은 특정 관리자에게만 S/W업그레이드의 권한을 줄 수 있어야 함	SF.Data.OP.005
	SF-Com-Req-003	스마트공장은 제품을 만들면서 S/W 업그레이드를 할 수 있어야 함	SF.Com.Adapt.001
	SF-Com-Req-004	스마트공장은 실시간으로 최신 S/W버전의 유무를 판단하고 이를 알려줄 수 있어야 함	SF.Com.Adapt.001
	SF-Com-Req-005	스마트공장은 S/W업그레이드 후 변경된 사항을 사용자에게 알려줄 수 있어야 함	SF.Com.Adapt.001
	SF-Com-Req-006	스마트공장은 S/W업그레이드가 필요한 구성요소들을 직관적으로 사용자에게 알려줄 수 있어야 함	SF.Com.Adapt.001 SF.Com.Complex.001
	SF-Com-Req-007	스마트공장은 S/W업그레이드를 하기위해 소요되는 시간을 측정할 수 있고 이를 사용자에게 알려줄 수 있어야 함	SF.Com.Adapt.002
	SF-Com-Req-008	스마트공장은 고객의 요구에 따라 제품을 만들기 위해서 H/W모듈을 쉽게 추가 또는 제거할 수 있어야 함	SF.Com.Adapt.002
	SF-Com-Req-009	스마트공장은 H/W를 추가 또는 제거할 때 완벽하게 추가 또는 제거되었는지 사용자에게 알려줘야 함	SF.Com.Adapt.002 SF.Com.Const.001. SF.Trust.Safe.002
	SF-Com-Req-010	스마트공장은 H/W의 추가 또는 제거 제대로 이루어지지 않고 동작할려고 할 때, 그에 맞는 조치를 할 수 있어야 함	SF.Com.Adapt.002 SF.Com.Const.001 SF.Trust.Safe.002
	SF-Com-Req-011	스마트공장은 특정 제품을 제작할 때 필요한 H/W모듈을 예상하고 알려줄 수 있어야 함	SF.Com.Adapt.002 SF.Com.Const.001.

	SF-Com-Req-012	스마트공장은 사용자가 새로운 공정환경에 적응할 수 있도록 도울 수 있어야 함	SF.Com.Complex.001
	SF-Com-Req-013	스마트공장은 현재 포함하고 있는 구성요소들을 사용자가 한눈에 관찰할 수 있게 제공해야 함	SF.Com.Complex.001
	SF-Com-Req-014	스마트공장은 구성요소가 추가 또는 제거되었을 때 변하는 복잡성을 계산하고 이를 사용자에게 알려줄 수 있어야 함	SF.Com.Complex.002
	SF-Com-Req-015	스마트공장은 지능적으로 복잡성을 최소로 낮추기 위한 최적의 설계를 사용자에게 제공할 수 있어야 함	SF.Com.Complex.004
	SF-Com-Req-016	스마트공장은 시뮬레이션을 통해 시스템의 무결정성을 검증할 수 있어야 함	SF.Data.OP.004
	SF-Com-Req-017	스마트공장은 고객의 요구를 도출할 수 있는 구성요소의 구조를 도출할 수 있어야 함	SF.Com.Const.002
	SF-Com-Req-018	스마트공장은 구성요소에 부여된 고유식별자를 이용하여 제품 제작에 필요한 구성요소와 그 기능을 식별할 수 있어야 함	SF.Com.Discover.001 SF.Com.Discover.002
	SF-Com-Req-019	스마트공장은 개발 및 검증을 위하여 구성요소의 하이브리드 M&S를 지원해야 함	SF.Data.OP.004 SF.Com.Const.002
	SF-Com-Req-020	스마트공장은 개발 및 검증을 위하여 구성요소의 분산 M&S를 지원해야 함	SF.Data.OP.004 SF.Com.Const.002 SF.Com.Discover.003

3)상호작용성(Interactivity)

상호작용성 요구사항은 스마트공장을 제어하는 현장기술자가 원활한 제어를 위해 스마트공장이 제공해야하는 기능에 대하여 정의하였다.

CPS 프레임워크의 Human 측면(Aspect)이 반영되어 17종의 상호작용성 요구사항이 정의되었다.

<표 5-18> 스마트공장의 시스템 상호작용성 요구사항

요구사항 구분	요구사항 ID	설명	관련 관심사(Concern) ID
Interactivity	SF-Inter-Req-001	스마트공장의 관리 서버는 각각의 기기들에게 제어 메시지를 보낼 수 있어야 함	SF.Hum.HF.001 SF.Hum.Usa.002 SF.Hum.Usa.003
	SF-Inter-Req-002	스마트공장의 기기들은 관리 서버로부터 제어 메시지를 받아 올 수 있어야 함	SF.Hum.HF.001 SF.Hum.Usa.002 SF.Hum.Usa.003

	SF-Inter-Req-003	스마트공장의 기기들은 관리 서버로부터 받아온 제어 메시지를 각각에 적합하게 해석할 수 있어야 함	SF.Hum.HF.001 SF.Hum.Usa.002 SF.Hum.Usa.003
	SF-Inter-Req-004	스마트공장의 기기들은 관리 서버로부터 받아온 제어 메시지를 표시할 수 있어야 함	SF.Hum.HF.001 SF.Hum.Usa.002
	SF-Inter-Req-005	스마트공장의 기기들은 자신의 상태 정보를 관리 서버에 전달할 수 있어야 함	SF.Hum.HF.002 SF.Hum.Usa.001
	SF-Inter-Req-006	스마트공장의 기기들은 자신의 상태 정보를 표시할 수 있어야 함	SF.Hum.HF.002 SF.Hum.Usa.001
	SF-Inter-Req-007	스마트공장의 관리 서버는 수집된 기기들의 상태 정보를 표시할 수 있어야 함	SF.Hum.HF.002 SF.Hum.Usa.001
	SF-Inter-Req-008	스마트공장의 기기들은 현장 기술자에 의하여 조작될 수 있어야 함	SF.Hum.HF.003
	SF-Inter-Req-009	스마트공장의 기기들은 현장 기술자의 조작을 정확히 반영할 수 있어야 함	SF.Hum.HF.003
	SF-Inter-Req-010	스마트공장의 기기들은 현장 기술자의 조작에 대하여 관리 서버에 전달할 수 있어야 함	SF.Hum.HF.003 SF.Hum.Usa.001
	SF-Inter-Req-011	스마트공장의 기기들은 이상 상황에 대하여 서버에 알림을 보낼 수 있어야 함	SF.Hum.Usa.001 SF.Hum.Usa.002
	SF-Inter-Req-012	스마트공장의 기기들은 이상 상황에 대하여 현장 기술자에게 알림을 보낼 수 있어야 함	SF.Hum.Usa.001 SF.Hum.Usa.002
	SF-Inter-Req-013	스마트공장의 기기들은 생산중인 제품에 대한 정보를 서버로부터 받아올 수 있어야 함	SF.Hum.Usa.003
	SF-Inter-Req-014	스마트공장의 기기들은 생산중인 제품에 대한 정보를 갱신할 수 있어야 함	SF.Hum.Usa.003
	SF-Inter-Req-015	스마트공장의 관리 서버는 공장 상태에 따라서 공정 정보를 설정할 수 있어야 함	SF.Hum.Usa.004
	SF-Inter-Req-016	스마트공장의 기기들은 현재 공정에 대한 정보를 서버로부터 받아올 수 있어야 함	SF.Hum.Usa.004
	SF-Inter-Req-017	스마트공장의 기기들은 공정 설정에 따라서 작업을 달리할 수 있어야 함	SF.Hum.Usa.004

4) 고신뢰성(Dependability)

고신뢰성 요구사항은 스마트공장의 안정적이고 원활한 동작을 위해 필요한 요구사항의 정의하였다. CPS 프레임워크의 Trustworthiness 측면(Aspect)이 반영되어 20종의 고신뢰성 요구사항이 정의되었다.

<표 5-19> 스마트공장의 시스템 고신뢰성 요구사항

요구사항 구분	요구사항 ID	설명	관련 관심사(Concern) ID
Dependability	SF-Depend-Req-001	스마트공장의 서버에 있는 기기 정보에 접근에 관한 보안 체계가 구성되어야 함	SF.Trust.Priv.001 SF.Trust.Sec.001
	SF-Depend-Req-002	스마트공장의 제어 네트워크에 접근에 관한 보안 체계가 구성되어야 함	SF.Trust.Priv.002 SF.Trust.Sec.001
	SF-Depend-Req-003	스마트공장의 관리 서버는 공정에 따라서 기기별로 제어 로직을 정확하게 설정할 수 있어야 함	SF.Trust.Rel.001
	SF-Depend-Req-004	스마트공장의 기기들은 설정된 제어 로직을 정확하게 실행할 수 있어야 함	SF.Trust.Rel.001 SF.Trust.Rel.002
	SF-Depend-Req-005	스마트공장의 기기들은 네트워크를 통하여 정확하게 설정 될 수 있어야 함	SF.Trust.Rel.003
	SF-Depend-Req-006	스마트공장 시스템 이상 상황 발생을 지능적으로 예측할 수 있어야 함	SF.Trust.Res.001 SF.Trust.Res.002 SF.Trust.Res.004
	SF-Depend-Req-007	스마트공장 시스템은 기기 결함 상황의 발생을 실시간으로 확인할 수 있어야 함	SF.Trust.Res.001
	SF-Depend-Req-008	스마트공장 시스템은 기기 결함 상황을 공정 변경 등의 대처를 통하여 대처할 수 있어야 함	SF.Trust.Res.001
	SF-Depend-Req-009	스마트공장 시스템은 공정 이상 상황의 발생을 실시간으로 확인할 수 있어야 함	SF.Trust.Res.002
	SF-Depend-Req-010	스마트공장 시스템은 공정 이상 상황의 발생 시 서버에서의 공정 재 설정 등과 같은 대처를 통하여 빠르게 대처할 수 있어야 함	SF.Trust.Res.002
	SF-Depend-Req-011	스마트공장 시스템은 네트워크 이상 상황을 감지할 수 있어야 함	SF.Trust.Res.003
	SF-Depend-Req-012	스마트공장 시스템은 네트워크 이상 상황 시에 네트워크 재설정을 통하여 빠르게 대처할 수 있어야 함	SF.Trust.Res.003
	SF-Depend-Req-013	스마트공장 시스템은 작업 환경을 실시간으로 확인할 수 있어야 함	SF.Trust.Safe.001 SF.Trust.Safe.002
	SF-Depend-Req-014	스마트공장 시스템은 작업 환경을 적합하게 조절할	SF.Trust.Safe.001 SF.Trust.Safe.002

		수 있어야 함	SF.Trust.Safe.003
	SF-Depend-Req-015	스마트공장의 기기들은 상호 연동을 통하여 안정적인 상태를 유지할 수 있어야 함	SF.Trust.Safe.001 SF.Trust.Safe.004
	SF-Depend-Req-016	스마트공장에서 발생되는 메시지에 대한 보안 체계가 제공되어야 함	SF.Trust.Sec.002
	SF-Depend-Req-017	스마트공장의 구성요소의 상태를 M&S를 통하여 예측할 수 있어야 함	SF.Trust.Res.004
	SF-Depend-Req-018	스마트공장의 전체 구성을 분산 연동 M&S를 통하여 검증 할 수 있어야 함	SF.Trust.Res.004
	SF-Depend-Req-019	스마트공장 이상 상황시 분산 M&S를 통하여 여러 시나리오를 시뮬레이션 및 평가할 수 있어야 함	SF.Trust.Res.004
	SF-Depend-Req-020	스마트공장의 분산 M&S의 결과를 실제 기기들에 적용할 수 있어야 함	SF.Trust.Res.004

5) 실시간성(Timing)

실시간성 요구사항은 스마트공장의 서브시스템 동작하는 기능에 대한 실시간성에 대하여 정의하였다.

CPS 프레임워크의 Timing 측면(Aspect)의 사용자 요구사항을 만족시킬 수 있도록 스마트공장의 실시간성 요구사항이 정의되었다.

<표 5-20> 스마트공장의 시스템 실시간성 요구사항

요구사항 구분	요구사항 ID	설명	관련 관심사(Concern) ID
Timing	SF-Timing-Req-001	스마트공장은 고객이 요구하는 제품 공정의 순서를 설계하고 그 시간을 예측 및 측정할 수 있어야 함	SF.TIme.Logic.001 SF.TIme.Logic.002
	SF-Timing-Req-002	스마트공장은 컴포넌트 또는 모듈의 데이터를 이용하여 이기종 노드 간 지능적으로 동기화를 할 수 있어야 함	SF.TIme.Sync.001 SF.TIme.Sync.002 SF.TIme.Sync.003 SF.TIme.Sync.004
	SF-Timing-Req-003	스마트공장은 모듈 간 걸리는 시간을 측정하고 학습하여 모듈을 이용한 새로운 모델을 만들 때 모델 내의 공정 시간을 예측할 수 있어야 함	SF.TIme.Sync.004 SF.Time.Aware.001 SF.Time.Aware.001 SF.Data.OP.003
	SF-Timing-Req-004	스마트공장 내에서 발생하는 신호들은 설정된 시간 내에 도달해야하며 그 시간을 초과했을 때 에러신호가 사용자에게 알려주는 시간을 설정된 시간 내에 도달해야 함	SF.Time.Interval.001 SF.Time.Interval.002
	SF-Timing-Req-005	다른 스마트공장과 통신을 위해 특정 이벤트는 국제 표준을 준수해야 함	SF.Time.Interval.003

	SF-Timing-Req-006	스마트공장은 실시간으로 다른 노드 간 시간동기화를 확인할 수 있어야 함	SF.TIme.Sync.001
	SF-Timing-Req-007	스마트공장은 실시간으로 다른 노드 간 주파수동기화를 확인할 수 있어야 함	SF.TIme.Sync.002
	SF-Timing-Req-008	스마트공장은 실시간으로 다른 노드 간 Pharse동기화를 확인할 수 있어야 함	SF.TIme.Sync.003
	SF-Timing-Req-009	스마트공장은 실시간으로 다른 노드 간 Pharse동기화를 확인할 수 있어야 함	SF.TIme.Sync.003
	SF-Timing-Req-010	스마트공장은 새로운 모델을 생성할 때 시뮬레이션 통해 측정된 모델의 성능과 실제 성능이 비슷해야 함	SF.Time.Aware.001
	SF-Timing-Req-011	스마트공장은 모델을 통해 동작 과정을 실시간으로 사용자에게 알려 줘야함	SF.TIme.Logic.001
	SF-Timing-Req-012	스마트공장의 정확한 분산 M&S를 위하여 시뮬레이션 발생 데이터들의 시뮬레이션 시간 동기화 기능이 제공되어야 함	SF.Time.Sync.005

6) 상호운용성(Interoperability)

상호운용성 요구사항은 스마트공장 내에 있는 기기들 사이의 통신, 제어, 그리고 모니터링 등의 기능에 대해 정의하였다.

CPS 프레임워크의 Interoperability 측면(Aspect)이 반영되어 상호운용성 요구사항이 정의되었다.

<표 5-21> 스마트공장의 시스템 상호운용성 요구사항

요구사항 구분	요구사항 ID	설명	관련 관심사(Concern) ID
Interoperability	SF-Operate-Req-001	스마트공장은 다양한 센서로부터 데이터를 받기 위해 산업용 통신기술을 이용하여 메시지를 주고받는다.	SF.Func.Comm.001 SF.Func.Comm.002
	SF-Operate-Req-002	스마트공장은 다양한 산업용 통신기술을 이용하여 사용자로부터 제어메세지를 받고 공정 상태를 제어할 수 있음.	SF.Func.Comm.002 SF.Func.Ctrl.002
	SF-Operate-Req-003	스마트공장은 센서를 이용하여 예기치 않은 동작을 발견하고 지능적으로 대처할 수 있음	SF.Func.Ctrl.002 SF.Func.Uncert.001
	SF-Operate-Req-004	스마트공장은 내부에서 다양한 모듈들과 통신하기 위해서 요구사항에 만족되는 네트워크 설정을 유지해야 함	SF.Func.Meas.001 SF.Func.Meas.002

	SF-Operate-Req-005	스마트공장은 기기들의 실제 위치를 고려하여 효율적으로 공정 과정을 설정함	SF.Func.Func.001 SF.Func.Phy.001 SF.Func.Phycont.001
	SF-Operate-Req-006	스마트공장 내에 발생하는 결함은 모든 기기가 공유하여 적절한 대처할 할 수 있게 환경을 만들 수 있음.	SF.Func.Func.003
	SF-Operate-Req-007	스마트공장은 주변 환경을 센서로 측정하고 주변 환경에 따라 공정 상태를 제어할 수 있음	SF.Func.Ctrl.001 SF.Func.Ctrl.002
	SF-Operate-Req-008	스마트공장은 제품의 주문량에 따라 유연하게 공정속도를 조절할 수 있음	SF.Func.Func.002 SF.Func.Sens.002
	SF-Operate-Req-009	스마트공장은 설정된 공정에 따라 생산 라인을 구성해야하고 기기 간 유연한 동작을 위해서 요구사항에 부합하면 설계를 해야 함	SF.Func.Act.001 SF.Func.Act.005 SF.Func.Mng.002
	SF-Operate-Req-010	스마트공장의 기기들은 그들의 설정들을 서버로 산업용 통신기술을 통하여 전송할 수 있음	SF.Func.Meas.003 SF.Func.Meas.004
	SF-Operate-Req-011	스마트공장은 서버로부터 기기들의 설정을 전달받고 복잡한 연동 구조를 유연하게 재구성 할 수 있음	SF.Func.Mng.001
	SF-Operate-Req-012	스마트공장의 분산 M&S 환경은 이종 기기 및 모델들이 연동될 수 있어야 함	SF.Func.Func.004

7) 지능성(Intelligence)

지능성 요구사항은 스마트공장에서 지능성이 어떤 기능에 필요한지 정의하였다.

CPS 프레임워크의 Data 측면(Aspect)와 Functional 측면(Aspect)가 반영되어 지능성 요구사항이 정의되었다.

<표 5-22> 스마트공장의 시스템 지능성 요구사항

요구사항 구분	요구사항 ID	설명	관련 관심사(Concern) ID
Intelligence	SF-Int-Req-001	스마트공장의 기기들은 환경과 공정 상황을 인지하고 그에 맞추어 동작 상태의 변동이 이루어질 수 있음	SF.Func.Control.002 SF.Func.Control.003 SF.Func.Func.001 SF.Func.Func.002 SF.Func.State.001 SF.Data.Id.002 SF.Data.Relat.003
	SF-Int-Req-002	스마트공장 시스템은 공장 내부의 현재 환경 상황을 인지하고 최적의 상태가 되도록 조절할 수 있음	SF.Func.Control.001 SF.Func.Func.002 SF.Func.Measure.004 SF.Func.Sens.001 SF.Func.Sens.003 SF.Data.Id.002 SF.Data.Relat.004
	SF-Int-Req-003	스마트공장 시스템은 공정에서의 결함이나 기기	SF.Data.OP.003

		의 결함을 지능적으로 예측해 낼 수 있음	SF.Data.OP.004
	SF-Int-Req-004	스마트공장 시스템은 빅 데이터 처리 기술에 기반하여 결함 상황에서의 회복을 할 수 있음	SF.Func.Func.003 SF.Data.OP.003 SF.Data.Relat.001 SF.Data.Relat.002 SF.Data.Relat.005
	SF-Int-Req-005	스마트공장 시스템은 네트워크 설정을 빅 데이터 처리 기술에 기반을 두어 지능적으로 수행할 수 있음	SF.Data.OP.003
	SF-Int-Req-006	스마트공장의 기기들은 생산중인 제품의 상태에 대하여 인지할 수 있음	SF.Func.Act.004 SF.Func.Measure.001 SF.Data.Sem.001 SF.Data.OP.001 SF.Data.OP.002 SF.Data.OP.003 SF.Data.OP.004 SF.Data.Sem.002 SF.Data.Vol.001
	SF-Int-Req-007	스마트공장의 기기들은 자신 혹은 다른 기기에 대한 정보를 인지할 수 있음	SF.Data.Id.001 SF.Data.Id.002 SF.Data.Sem.002 SF.Data.OP.002
	SF-Int-Req-008	스마트공장의 기기들은 데이터 전송 지연에 대해 대처할 수 있음	SF.Data.Vel.001 SF.Data.Vel.002 SF.Data.Vel.003

2. 스마트공장에서 CPS의 역할

도래하는 개인화생산 체계를 지원하기 위한 스마트 공장은 기능적・성능적・인터페이스적 측면에서 하드웨어보다는 소프트웨어에 의존적이며, 많은 공장 구성요소들이 실시간으로 유연하게 연동되고, 지능적으로 물리시스템을 제어해야 하는 복합 지능제어 소프트웨어의 역할이 크게 요구된다.

1) CPS 프레임워크의 측면과 CPS 특성과의 관계

자동차, 항공기에서와 마찬가지로 CPS 특성은 CPS 프레임워크의 기능적 측면에 속하는 관심사들(Concerns)은 전체적으로 연관이 되어 있다. CPS 특성을 구현하기 위해서는 기능적 요소들이 지원되어야하기 때문이다.

융복합성(Composability)은 자동차, 항공기와 동일하게 CPS 프레임워크의 조립(Composition) 측면의 관심사들(Concerns)과 연관이 지어진다. 스마트공장에서 상호작용성(Interactivity)은 CPS 프레임워크의 인간적(Human) 측면의 요구사항이 대부분 도출되었다.

고신뢰성(Dependability)을 구현하기 위해서는 CPS 프레임워크의 신뢰성(Trustworthiness) 측면의 사용자 요구사항을 구현해야 하며, 실시간성(Timing)은 CPS 프레임워크의 실시간(Timing)측면과 직접적인 연관이 있다.

지능성(Intelligence)은 항공과 유사하게 빅데이터 분석 연관된 특성이며, CPS 프레임워크의 기능적(Functional), 데이터(Data) 측면의 요구사항들을 구현함으로써 나타난다.

2) 스마트공장의 시스템 요구사항

다음은 스마트공장 요구사항과 그에 따라 적용된 예를 나타낸다.

(1) 확장성

기존의 고정적인 PLC 중심 제어시스템 구조와 클라이언트/서버모델의 공정관리시스템에서 유연성이 높은 데이터 중심의 Pub/Sub 모델로의 전환을 통해 고도 확장성을 확보할 수 있다.

예) SDX(Software Defined Everything) 개념의 도입으로 산업네트워크, 공장제어시스템 및 정보시스템들의 구조 및 기능 변경이 용이해지고 있다.

(2) 융복합성

다양한 이종 센서/액츄에이터/제어기 등을 융복합하여 새로운 기능과 서비스로 재구성하는 방법이 기존의 하드웨어 중심 접근에서는 불가능하였으나, SW 플랫폼 기반에서는 이러한 다양성을 투명화하여 기능 융복합을 가능하게 하였다.

예) OPC UA가 표준화되어 이종 PLC, PC, IoT 디바이스로 부터의 데이터를 쉽게 수집할 수 있도록 지원하여 이종 시스템들간 상호 데이터를 공유할 수 있는 환경을 제공한다.

(3) 상호작용성

생산로봇과 사람간의 협업형 생산활동이 가능하게 되고, Factory-thing들간의 유연한 통신기능과 상호작용성을 제공하기 위한 SW기능이 요구된다.

예) 슈트트가르트대학내 ARENA2036 프로그램에서는 다양한 생산주체조합을 생산 목적 중심으로 설계하고 다양한 Factory-thing들간의 상호작용을 가능하게 하는 실험을 진행 중에 있다.

(4) 고신뢰성

디지털 트윈(Digital Twin)과 같은 가상시스템을 활용할 경우 모든 물리시스템들과 프로세스들을 가상화하여 통합 검증하여 생산시스템과 공정관리시스템에 대한 개별 검증 시 나타날 수 있는 부정합의 문제점들을 해결할 수 있다.

예) 디지털 트윈은 실세계 물리시스템의 디지털 파트너 소프트웨어로서, 물리시스템과 항상 인터넷으로 연결되어 있어 실시간으로 모니터링하고, 미래 상태를 예측할 수 있어 최적 제어할 수 있는 자율형 지능제어 소프트웨어이다.

(5) 실시간성

스마트 공장의 센서/액츄에이터/제어기 등을 실시간 연동할 수 있는 기능이 요구되며, 이를 만족시키기 위해서는 실시간 운영체제, 실시간 네트워크 및 초정밀 시간 동기화 기술 등이 제공되어야 한다.

예) OPC UA에서 최근 개발 중인 표준 기술로 IEEE TSN(Time Sensitive Network)와 OMG DDS(Data Distribution Service) 및 SDN(Software Defined Network) 등과의 연계성 제공이며, 이들 기술들은 실시간성을 보장하기 위한 최신 기반 기술이다.

(6) 상호운용성

다양한 스마트 공장 정보시스템들간 데이터 및 네트워크 호환성을 제공하기 위한 표준 기술로 OPC 그룹의 OPC UA(Unified Architecture) 플랫폼이 개발되어 운용 중에 있다.

7) 지능성

인공지능 및 빅데이터 처리에 의해 하드웨어 및 공정 특성을 학습하고 실시간 적용을 통한 오류검출과 예지보전을 수행함 예) MS AZURE, GE PREDIX, SIMENS MINDSPHERE 등의 IoT 클라우드에서는 OPC UA를 통해 센서 데이터 수집 후 인공지능 엔진과 빅데이터 서비스를 기본적으로 제공하여 스마트 공장의 지능성을 극대화시키고 있다.

제4절 CPS 공통 시스템 요구사항

이 절에서는 1절에서 3절을 통하여 도출된 시스템 요구사항을 통하여 CPS 시스템 요구사항의 공통점을 분석한다. 자동차, 항공기, 스마트공장 도메인의 CPS 시스템 요구사항들을 종합하여 CPS 도메인들이 공통적으로 필요로 하는 공통 시스템 요구사항을 도출하였다.

세 가지 도메인의 요구사항에서 모두 드러나는 요구사항을 공통 요구사항으로 도출하는 것을 원칙으로 하되, 특정 도메인에만 드러나지만 CPS에서 보편적으로 필요로 하는 요구사항들도 마찬가지로 공통 요구사항으로 도출하였다.

각 항목별로 시스템 확장성 5종, 융복합성 5종, 상호작용성 10종, 고신뢰성 5종, 실시간성 4종, 상호운용성 3종 그리고 지능성 3종이 도출되어 총 35종의 공통 시스템 요구사항이 도출되었다. 도출된 요구사항은 CPS-XXX-Req-### 형태의 아이디를 가지며 이것들을 만족시킬 수 있는 CPS 공통 플랫폼 기능을 도출하는데 활용된다.

1. 공통 요구사항 명세

1) 시스템 확장성(Scalability)

시스템 확장성 요구사항은 CPS가 유연한 시스템의 변경 등의 확장성을 가지기 위하여 공통적으로 필요로 하는 사항들에 대하여 정의한다. 공통 요구사항으로는 총 5종의 요구사항이 도출되었다.

<표 5-23> CPS 공통 시스템 확장성 요구사항

요구사항 구분	요구사항 ID	설명	관련 요구사항 ID
Scalability	CPS-Scal-Req-001	CPS의 시스템 연동은 시스템의 변동에 대처가 가능한 유연한 구조를 가져야 함	Autom-Scal-Req-002 SF-Scal-Req-002
	CPS-Scal-Req-002	CPS는 여러 기능 모듈 인스턴스의 생성 및 연결을 허용해야 함	Autom-Scal-Req-002 Autom-Scal-Req-004 Aero-Scal-Req-003 SF-Scal-Req-008
	CPS-Scal-Req-003	CPS는 각 시스템에 적합한 통신 기술을 지원할 수 있어야 함	Autom-Scal-Req-003 Autom-Scal-Req-005 SF-Scal-Req-003 SF-Scal-Req-004 SF-Scal-Req-005 SF-Scal-Req-006

	CPS-Scal-Req-004	CPS는 유연하고 확장성 있는 통신이 가능해야 함	Autom-Scal-Req-003 SF-Scal-Req-007
	CPS-Scal-Req-005	CPS는 목적에 따른 기능 수행을 위한 연동 구조 변경 및 태스크 스케줄링이 가능해야 함	Autom-Scal-Req-006 Aero-Scal-Req-006 SF-Scal-Req-011 SF-Scal-Req-012

2) 융복합성(Composability)

융복합성 요구사항은 CPS 자체 혹은 CPS를 구성하는 서브시스템간의 융복합성에 대하여 CPS가 공통적으로 필요로 하는 요소들을 정의한 것으로 5종의 공통 요구사항이 도출되었다.

<표 5-24> CPS 공통 시스템 융복합성 요구사항

요구사항 구분	요구사항 ID	설명	관련 요구사항 ID
Composability	CPS-Com-Req-001	CPS를 구성하는 모듈들의 연동 구조를 확인할 수 있어야 함	Autom-Comp-Req-001 SF-Com-Req-013
	CPS-Com-Req-002	CPS는 구성하고 있는 서브시스템의 추가/제거 등의 변동을 쉽게 할 수 있어야 함	Aero-Comp-Req-006 SF-Com-Req-008 SF-Com-Req-009
	CPS-Com-Req-003	CPS를 구성하는 모듈의 상태 및 변동을 추적할 수 있어야 함	Autom-Comp-Req-003 Autom-Comp-Req-004 Aero-Comp-Req-001 Aero-Comp-Req-002 Aero-Comp-Req-003 SF-Com-Req-004 SF-Com-Req-006 SF-Com-Req-013
	CPS-Com-Req-004	CPS는 구성 모듈의 추가 또는 제거가 제대로 이루어지지 않고 동작할 때, 적절한 조치를 취할 수 있어야 함	Autom-Comp-Req-010 Autom-Comp-Req-011 Aero-Comp-Req-007 SF-Com-Req-010
	CPS-Com-Req-005	CPS는 여러 기능 모듈들의 협력을 지원해야 함	Autom-Comp-Req-008 Aero-Comp-Req-004 SF-Com-Req-015 SF-Com-Req-017

3) 상호작용성(Interactivity)

상호작용성은 CPS들 사이 혹은 CPS와 휴먼 사이의 유연한 상호작용성에 대한 요구사항을 정의한 것으로 CPS 공통 상호작용성 요구사항으로는 총 10가지가 도출되었다.

<표 5-25> CPS 공통 시스템 상호작용성 요구사항

요구사항 구분	요구사항 ID	설명	관련 요구사항 ID
Interactivity	CPS-Inter-Req-001	CPS를 구성하는 서브시스템간의 통신을 지원해야 함	Autom-Inter-Req-001 Aero-Inter-Req-002 SF-Inter-Req-001 SF-Inter-Req-002 SF-Inter-Req-005
	CPS-Inter-Req-002	CPS를 구성하는 소프트웨어간의 통신을 지원해야 함	Autom-Inter-Req-004 Aero-Inter-Req-005
	CPS-Inter-Req-003	CPS는 이종의 외부 환경과의 통신을 지원해야 함	Autom-Inter-Req-003 Aero-Inter-Req-001 SF-Inter-Req-001 SF-Inter-Req-002
	CPS-Inter-Req-004	CPS의 통신은 서브시스템간의 시그널 메시지를 위한 모니터링 등의 서비스 지원이 필요	Autom-Inter-Req-002 Autom-Inter-Req-004 Autom-Inter-Req-019 Autom-Inter-Req-020 Aero-Inter-Req-006 Aero-Inter-Req-007 SF-Inter-Req-011 SF-Inter-Req-012
	CPS-Inter-Req-005	CPS의 통신은 주고 받는 데이터의 해석에서 일관성을 지원해야 함	Autom-Inter-Req-021 Aero-Inter-Req-008 SF-Inter-Req-003 SF-Inter-Req-003
	CPS-Inter-Req-006	CPS는 각각의 시스템에 최적화 된 통신 기술들을 지원해야 함	Autom-Inter-Req-005 ~ Autom-Inter-Req-008 Autom-Inter-Req-014 Autom-Inter-Req-015 Autom-Inter-Req-016 Aero-Inter-Req-004 SF-Inter-Req-005
	CPS-Inter-Req-007	CPS는 통신 과정에서 전송과 수신 취소를 지원해야 함	Autom-Inter-Req-022 Aero-Inter-Req-009
	CPS-Inter-Req-008	CPS는 통신을 통하여 구성하고 있는 서브시스템을 제어할 수 있어야 함	Autom-Inter-Req-010 Autom-Inter-Req-011 Autom-Inter-Req-013 Autom-Inter-Req-029 Aero-Inter-Req-002 Aero-Inter-Req-003 SF-Inter-Req-001 SF-Inter-Req-002 SF-Inter-Req-003 SF-Inter-Req-004
	CPS-Inter-Req-009	CPS는 각각 모듈들이 정확히 수행할 수 있도록 요구사항을 만족시키는 통신 환경을 제공해야 함	Autom-Inter-Req-026 SF-Inter-Req-009
	CPS-Inter-Req-010	CPS는 통신 상태를 모니터링하고 관리할 수 있어야 함	Autom-Inter-Req-027 Autom-Inter-Req-028 SF-Inter-Req-005 SF-Inter-Req-006 SF-Inter-Req-007

4) 고신뢰성(Dependability)

고신뢰성 요구사항은 안정성, 회복성, 유지보수성, 안전성 및 보안성을 만족시키하기 위하여 CPS가 공통적으로 필요로 하는 요구사항에 대하여 정의하였다. 총 5가지 공통 요구사항이 도출되었다.

<표 5-26> CPS 공통 시스템 고신뢰성 요구사항

요구사항 구분	요구사항 ID	설명	관련 요구사항 ID
Dependability	CPS-Depend-Req-001	CPS는 구성 요소들이 주고받는 정보의 훼손, 변조, 유출 등을 방지하기 위한 보안 기술이 필요함	Autom-Dep-Req-022 Aero-Dep-Req-006 SF-Depend-Req-001 SF-Depend-Req-002
	CPS-Depend-Req-002	CPS는 작동 중에 예상하지 못한 비정상적인 동작 상태를 인지할 수 있어야 함	Autom-Dep-Req-004 Autom-Dep-Req-005 Aero-Dep-Req-002 SF-Depend-Req-006 SF-Depend-Req-007 SF-Depend-Req-009 SF-Depend-Req-011
	CPS-Depend-Req-003	CPS는 비정상적인 동작 상황에서 시스템을 안정화 시킬 수 있어야 함	Autom-Dep-Req-006 Aero-Dep-Req-007 SF-Depend-Req-008 SF-Depend-Req-010 SF-Depend-Req-012
	CPS-Depend-Req-004	CPS의 소프트웨어들은 안정적 작동을 위하여 수행될 메모리 영역을 보장해주어야 함	Autom-Dep-Req-016 Aero-Dep-Req-004
	CPS-Depend-Req-005	CPS는 메시지 데이터의 무결성 인증을 제공해야 함	Autom-Dep-Req-025 Aero-Dep-Req-015

5) 실시간성(Timing)

CPS 공통 요구사항 중 실시간성 요구사항은 CPS와 CPS 서브시스템의 동작 기능에 대한 실시간성에 대하여 정의되어 있다. 총 4가지 실시간성 요구사항이 공통적으로 도출되었다.

<표 5-27> CPS 공통 시스템 실시간성 요구사항

요구사항 구분	요구사항 ID	설명	관련 요구사항 ID
Timing	CPS-Timing-Req-001	CPS를 구성하고 있는 이기종 노드간의 시간 동기화를 할 수 있어야 함	Autom-Timing-Req-010 Aero-Timing-Req-004

			SF-Timing-Req-002
	CPS-Timing-Req-002	CPS의 태스크들의 시간적 요구사항을 엄격히 만족시킬 수 있어야 함	Autom-Timing-Req-008 Autom-Timing-Req-009 Autom-Timing-Req-013
	CPS-Timing-Req-003	CPS에서 주고받는 시그널 등의 메시지 전송은 실시간성을 만족해야 함	Aero-Timing-Req-007 SF-Timing-Req-004
	CPS-Timing-Req-004	CPS는 외부 시스템과 실시간 상호작용이 일어날 수 있어야 함	Autom-Timing-Req-005 Autom-Timing-Req-006 Autom-Timing-Req-007 Aero-Timing-Req-003 SF-Timing-Req-011

6) 상호운용성(Interoperability)

상호운용성 요구사항은 CPS 내에 있는 서브시스템들 사이의 통신, 제어, 그리고 모니터링 등의 기능에 대하여 정의하였고 공통 요구사항으로는 총 3가지 요구사항이 도출되었다.

<표 5-28> CPS 공통 시스템 상호운용성 요구사항

요구사항 구분	요구사항 ID	설명	관련 요구사항 ID
Interoperability	CPS-Operate-Req-001	CPS는 이종 노드들의 연동을 위하여 게이트웨이나 인터페이스 등의 기능을 지원해야 함	Autom-Operate-Req-001 Autom-Operate-Req-003 Aero-Operate-Req-001 SF-Operate-Req-001
	CPS-Operate-Req-002	CPS의 소프트웨어는 이종 노드에의 적용을 위하여 표준을 준수한 이식성을 가질수 있어야 함	Autom-Operate-Req-005 Autom-Operate-Req-006 Aero-Operate-Req-002 ~ Aero-Operate-Req-005
	CPS-Operate-Req-003	CPS에서 발생되는 데이터는 이종 노드간에 공유될 수 있어야 함	Autom-Operate-Req-008 Aero-Operate-Req-001 SF-Operate-Req-002

7) 지능성(Intelligence)

지능성 요구사항은 CPS의 지능적 기능을 제공하기 위하여 필요로 하는 요구사항으로 총 3가지 요구사항이 공통적으로 도출되었다.

<표 5-29> CPS 공통 시스템 지능성 요구사항

요구사항 구분	요구사항 ID	설명	관련 요구사항 ID
Intelligence	CPS-Int-Req-001	CPS는 시스템이 놓인 상황을 지능적으로 인지하여 그에 따른 동작에의 변동이 이루어질 수 있어야 함	Autom-Int-Req-001 Aero-Int-Req-005 Aero-Int-Req-006 SF-Int-Req-001 SF-Int-Req-002 SF-Int-Req-006 SF-Int-Req-007
	CPS-Int-Req-002	CPS는 시스템의 이상 혹은 결함 상황을 지능적으로 예측 및 진단할 수 있어야 함	Autom-Int-Req-002 Aero-Int-Req-002 Aero-Int-Req-003
	CPS-Int-Req-003	CPS는 이상 및 결함 상황에서 상황 정보들을 인지하여 지능적으로 대처할 수 있어야 함	Autom-Int-Req-004 Autom-Int-Req-005 Aero-Int-Req-004 SF-Int-Req-004 SF-Int-Req-005 SF-Int-Req-008 SF-Int-Req-009

2. 분석

시스템 확장성에 속하는 공통 시스템 요구사항은 통신 관련 기능이 대부분이다. 스마트공장의 경우는 공장 도메인 기능에 속하는 공정 관련 사항과 공장 구성요소 변경에 관한 요구사항이 많아 공통 기능에는 포함하지 않았다.

융복합성, 고신뢰성, 상호운용성에 속하는 자동차 요구사항은 자동차 표준인 AUTOSAR에 관련된 것이 많았으며, 스마트공장의 경우도 도메인에 관련된 요구사항이 많았다. 지능성에 포함된 요구사항은 도메인에 속하는 요구사항이 많이 공통의 요구사항으로 도출하기는 어려운 면이 있다.

상호작용성의 경우는 통신에 관련된 일반적인 요구사항이 많아 공통요구사항으로 도출되었다.

제5절 CPS 시스템 요구사항 분석결과

1. 각 도메인별 시스템 요구사항

CPS 특성별 시스템 요구사항은 다음과 같이 도출되었다.

<표 5-30> CPS 특성과 응용 도메인별 시스템 요구사항수

CPS 특성	자동차	항공기	스마트공장	CPS
시스템 확장성(Scalability)	6	8	13	5
융복합성(Composability)	17	8	20	5
상호작용성(Interactivity)	30	10	17	10
고신뢰성(Dependability)	27	18	20	5
실시간성(Timing)	13	7	12	4
상호운용성(Interoperability)	8	5	12	3
지능성(Intelligence)	7	6	8	3
합 계	108	62	102	35

2. 비기능 시스템 요구사항

자동차, 항공, 스마트공장과 공통적 CPS 도메인의 요구사항들을 기능과 비기능 요구사항으로 분류한다. 비기능 요구사항의 경우 시스템 사용 시 보이는 기능이 아니기 때문에 사용자들이 개발 시 무시할 수 있으나, 성능, 보안, 가용성과 안전과 같은 시스템에 필요한 특성으로 비기능 요구사항을 만족하지 않을 경우는 시스템을 사용하는데 문제가 될 수 있는 중요한 요구사항이다.

2장의 이론적 배경에서 설명한 것과 같이 기능 요구사항은 시스템이 제공해야 하는 서비스에 대한 내용이며, 비기능 요구사항은 특정 기능보다는 시간적 제약, 시스템 구현에 대한 제약 사항, 개발 프로세스에 대한 표준 준수와 이식성, 효율성, 신뢰성, 재사용성 등의 특성을 포함한다. 그러므로 도출된 기능 요구사항을 통해서는 각 도메인별 CPS시스템에서 반드시 구현되어야 할 필수적인 작업과 동작에 대해 파악 할 수 있다. 그리고 비기능 요구사항

을 통해서는 개발하고자 하는 응용도메인 CPS에 따라 구체적인 필요한 제약사항과 품질 요건은 다르겠지만 전반적으로 요구되는 제약사항과 품질 요건 등을 기술함으로써 응용도메인에 맞는 시스템 개발 시 활용할 수 있다.

위에 설명한 기준에 따라 전체 시스템 요구사항 307종 가운데 총 35종을 비기능 요구사항으로 구분하였다. 응용 도메인별로는 자동차는 총 108종 중 8종, 항공기는 총 62종 중 13종, 스마트공장은 총 102종 중 5종, CPS 공통 요구사항은 총 35종 중 9종의 요구사항이 비기능 요구사항으로 구분되었다.

<표 5-31> CPS 특성과 응용 도메인별 비기능 시스템 요구사항 수

CPS 특성	자동차	항공	스마트공장	CPS	합계
시스템확장성	0	4	0	1	**5**
융복합성	0	1	0	0	**1**
상호작용성	3	0	0	0	**3**
고신뢰성	2	1	3	1	**7**
실시간성	3	2	2	4	**11**
상호운용성	0	4	0	1	**5**
지능성	0	1	0	2	**3**
합계	**8**	**13**	**5**	**9**	**35**

아래 표에서 비기능 요구사항으로 구분한 요구사항을 정리하고 구분한 이유에 대해서 설명하였다. 단, Autom-Dep-Req-001가 비기능 요구사항이 되기 위해서는 신뢰성에 대한 수치적 표현이 필요하다.

<표 5-32> 자동차의 비기능 요구사항

CPS 특성	요구사항 ID	설명	이유
상호 작용성	Autom-Inter-Req-013	자동차 소프트웨어 구조는 전력을 낭비하지 않기 위해 만약 아무 태스크와 인터럽트가 없다면 코어들을 끄는 것들을 지원할 수 있어야 함	전력의 효율성
	Autom-Inter-	자동차 소프트웨어 구조 통신은 버스의 최대 전송	최대 지원 데이터 크기

	Req-023	단위보다 큰 데이터 크기들의 전송을 지원해야 함	제한
	Autom-Inter-Req-030	자동차 소프트웨어 구조 SW 개발 시 협력을 통한 개발이 될 수 있도록 표준화된 데이터 포맷과 절차를 거쳐야 함	데이터 포맷 및 절차 표준화
고신뢰성	Autom-Dep-Req-001	자동차 소프트웨어 구조 중 비휘발성 메모리 기능성은 하드웨어 신뢰성을 향상시키기 위해서 메커니즘을 제공해야 함	신뢰성 제고 (신뢰성 X% 향상 등과 같이 수치로 표현)
	Autom-Dep-Req-003	자동차 소프트웨어 구조는 여분의 여러 통신 링크들을 지원해야 함	서비스의 품질 제고
실시간성	Autom-Timing-Req-005	자동차 V2X 통신에서 차량 안전 서비스를 위한 상황정보는 100ms 주기의 차량 정보와 위치 정보를 보내줘야 함	시간 제약사항이 구체적으로 명시
	Autom-Timing-Req-006	자동차 V2X 통신에서 협력 주행 제어를 위한 차량 정보는 10ms 주기의 차량 정보와 위치 정보를 보내줘야 함	시간 제약사항이 구체적으로 명시
	Autom-Timing-Req-007	자동차 V2I통신에서 차량은 도로 안전/협력주행제어를 위한 도로 상황 정보를 100ms 주기의 도로상황 MAP, 위치 보정정보를 수신하여야 함	시간 제약사항이 구체적으로 명시

항공기 소프트웨어 플랫폼인 ARINC 653은 시스템확장성과 상호운영성 면이 강조되어, 비기능 요구사항 중 호환성 및 이식성에 대한 사항이 많았다. 지능성 요구사항은 기능 요구사항이나 비기능 요구사항 어느 쪽으로도 포함가능하다. 실시간성 요구사항은 비기능 요구사항으로써 완전해지려면 시간 제약이 포함되어야 한다.

<표 5-33> 항공기의 비기능 요구사항

CPS 특성	요구사항 ID	설명	이유
시스템 확장성	Aero-Scal-Req-004	항공기 소프트웨어(ARINC 653) 플랫폼은 무인 항공기 도메인 등, 다른 종류의 항공기에도 적용이 가능해야 함	호환성
	Aero-Scal-Req-005	항공기 소프트웨어(ARINC 653) 플랫폼은 자동차 분야 등의 여러 도메인에서 사용 하도록 되어야 함	타 도메인으로의 호환성(이식성)
	Aero-Scal-Req-007	항공기 소프트웨어(ARINC 653) 플랫폼은 다른 상용화 OS에서의 적용이 가능하도록 해야 함	타 OS에서의 적용 가능성(이식성)
	Aero-Scal-Req-008	항공기 소프트웨어(ARINC 653) 플랫폼은 많은 데이터량을 처리할 수 있어야 함	처리할 수 있는 데이터의 용량 설정(수 TB 등)
융복합성	Aero-Comp-Req-005	항공기 소프트웨어(ARINC 653) 플랫폼은 추상화 계층을 제공해서 POSIX를 지원하는 OS뿐이 아닌, 다양한 OS에서도 적용되어야 함	타 OS에서의 적용 가능성(이식성)
고신뢰성	Aero-Dep-Req-001	항공기 소프트웨어(ARINC 653) 플랫폼은 비휘발성 메모리 기능성은 하드웨어 신뢰성을 향상시키기 위해서 메커니즘을 제공해야 함	신뢰성
실시간성	Aero-Timing-Req-003	무인항공기의 영상 촬영은 실시간으로 처리 되어야 함	영상 촬영 처리 소요시간(수 ms이내 처리)
	Aero-Timing-Req-007	항공기 빅데이터 분석을 통해 실시간으로 결함 상태 등의 현황을 알 수 있어야 함	상태 확인에 소요되는 시간(수 ms이내 응답)
상호	Aero-Opera	항공기 소프트웨어(ARINC 653) 플랫폼은 불필요한	효율성 및 이식성

운용성	te-Req-002	파라미터 형식의 변동을 줄이고 통상적인 I/O 프로세싱을 줄여, 응용프로그램의 효율을 향상시키고, 프로그램 이식성을 높여야 함	
	Aero-Operate-Req-003	항공기 소프트웨어(ARINC 653) 플랫폼은 자동차 분야 등의 여러 도메인에서 사용 하도록 되어야 함	여러 도메인에서의 사용 가능성(이식성)
	Aero-Operate-Req-004	항공기 소프트웨어(ARINC 653) 플랫폼은 다른 상용화 RTOS에서의 적용이 가능하도록 해야 함	다른 OS에서의 적용 가능성(이식성)
	Aero-Operate-Req-005	항공기 소프트웨어(ARINC 653) 플랫폼은 다른 상용화 RTOS에서의 적용이 가능하도록 해야 함	다른 기체에서의 적용 가능성(이식성)
지능성	Aero-Int-Req-002	항공기 빅데이터 분석으로 다양한 기체 부품의 결함을 예측할 수 있어야 함	예측 정확도(X% 이상 예측)

스마트공장 요구사항의 경우는 고신뢰성과 실시간성 요구사항에 수치적 요구사항을 규정하여 비기능 요구사항으로 정리될 수 있다.

<표 5-34> 스마트공장의 비기능 요구사항

CPS 특성	요구사항 ID	설명	이유
고신뢰성	SF-Depend-Req-006	스마트공장 시스템 이상 상황 발생을 지능적으로 예측할 수 있어야 함	예측 정확도(X% 이상 예측)
	SF-Depend-Req-010	스마트공장 시스템은 공정 이상 상황의 발생 시 서버에서의 공정 재설정 등과 같은 대처를 통하여 빠르게 대처할 수 있어야 함	대처 속도(수 초 이내 응답)
	SF-Depend-Req-012	스마트공장 시스템은 네트워크 이상 상황 시에 네트워크 재설정을 통하여 빠르게 대처할 수 있어야 함	재설정 대처 속도(수 초 이내 응답)
실시간성	SF-Timing-Req-004	스마트공장 내에서 발생하는 신호들은 설정된 시간 내에 도달해야하며 그 시간을 초과했을 때 에러신호가 사용자에게 알려주는 시간을 설정된 시간 내에 도달해야 함	설정된 시간적 요구사항을 시스템에 맞게 구체적으로 설정해야함 (수 ms내 응답)
	SF-Timing-Req-010	스마트공장은 새로운 모델을 생성할 때 시뮬레이션 통해 측정된 모델의 성능과 실제 성능이 비슷해야 함	시뮬레이션 모델의 성능 모사도(X% 유사)

CPS 공통 요구사항은 항공 비기능 요구사항과 특성이 비슷하여, 이 연구에서는 항공 항공기 소프트웨어 플랫폼인 ARINC 653 만으로도 CPS 비기능 요구사항이 정리됨을 알 수 있다.

<표 5-35> CPS 공통 비기능 요구사항

CPS 특성	요구사항 ID	설명	이유
시스템 확장성	CPS-Scal-Req-005	CPS는 목적에 따른 기능 수행을 위한 연동 구조 변경 및 태스크 스케줄링이 가능해야 함	태스크 스케줄링의 정교성

고신뢰성	CPS-Depend-Req-002	CPS는 작동 중에 예상하지 못한 비정상적인 동작 상태를 인지할 수 있어야 함	이상 상황에 대한 인지율 (X % 이상 인지)
실시간성	CPS-Timing-Req-001	CPS를 구성하고 있는 이기종 노드간의 시간 동기화를 할 수 있어야 함	시간 동기화 수준(수 ms 등)
	CPS-Timing-Req-002	CPS의 태스크들의 시간적 요구사항을 엄격히 만족시킬 수 있어야 함	태스크들의 수행 시간 (수 m/s 내 응답 등)
	CPS-Timing-Req-003	CPS에서 주고받는 시그널 등의 메시지 전송은 실시간성을 만족해야 함	메시지 전송의 실시간성(수 m/s 이내 전송 완료 등)
	CPS-Timing-Req-004	CPS는 외부 시스템과 실시간 상호작용이 일어날 수 있어야 함	상호작용의 실시간성(수 m/s 내 응답 등)
상호 운용성	CPS-Operate-Req-002	CPS의 소프트웨어는 이종 노드에의 적용을 위하여 표준을 준수한 이식성을 가질 수 있어야 함	표준을 준수한 이종 노드에서의 이식성
지능성	CPS-Int-Req-001	CPS는 시스템이 놓인 상황을 지능적으로 인지하여 그에 따른 동작에의 변동이 이루어질 수 있어야 함	시스템 상태에 대한 인지율(X % 이상 인지)
	CPS-Int-Req-002	CPS는 시스템의 이상 혹은 결함 상황을 지능적으로 예측 및 진단할 수 있어야 함	결함 상황(이상 상황)에 대한 예측 및 진단 정확도(X % 이상 예측)

비기능 요구사항을 도출한 결과를 살펴보면 가장 많이 도출된 비기능 요구사항은 실시간성과 관련된 것으로 수 m/s내 응답이 가능한지를 확인하는 것이 11종을 차지하였다. CPS에서는 실시간성이 중요하며, 비기능 요구사항으로 시간에 대한 제약 사항에 대한 고려가 필요함을 시사한다.

두 번째는 상호운용성 관련 특성으로 타 OS나 이종 시스템으로의 이식성에 관련된 비기능 요구사항이 8종으로 구분되었다. CPS는 여러 서브시스템으로 구성되어 있으며, 여러 시스템이 추가되거나, 제거되는 일이 빈번할 것이고 호환성이나 이식성 이슈는 CPS에서 계속 발생할 것으로 예상된다.

세 번째는 고신뢰성 관련 특성으로 이상 상황에 대한 인지율이나 신뢰성을 높이기 위해 이상 상황에 대한 대응 속도 등의 내용이 포함되었다. CPS의 특성 중 고신뢰성은 시스템이 활용되기 위한 중요한 특성이며, 신뢰성을 높이기 위한 지능성은 지금까지 활용되던 시스템과는 다른 새로운 특성으로 기능 요구사항이나 비기능 요구사항으로 어떻게 분류하던지 요구사항 분석 및 구현에 반드시 고려되어야할 특성이다.

제6장 IoT 플랫폼 검증을 통한 IoT 플랫폼 기반 CPS 플랫폼 구성

IoT(Internet of Things, 사물인터넷)는 센서를 통하여 사물들에서 정보를 수집하고, 사물들을 연결하여 새로운 서비스를 제공한다. CPS는 물리 세계에서 수집한 정보를 분석하고, 인공지능을 이용하여 결정된 사항으로 물리 세계를 제어하여 유연하고, 신뢰성 있고, 실시간으로 지능성을 제공하는 서비스를 생성한다. CPS는 물리 세계에서 정보를 수집하고 다른 시스템과 통신하기 위한 수단으로 IoT를 활용할 수 있다. CPS 플랫폼에 비해 IoT 플랫폼은 표준화되고 활성화되어 활용되고 있어, CPS 플랫폼의 정보 수집과 통신 기능을 IoT 플랫폼을 이용하여 어느 정도 구현이 가능한지 검증해보고자 한다.

이 장에서는 IoT 플랫폼에서 제공되는 기능 중 CPS에서 필요로 하는 기능을 정리하여, 그 기능이 CPS에서 사용가능한 지 검증한다. 이를 통하여 CPS 플랫폼의 구조를 정의한다.

제1절 개요

이 장은 검증항목, IoT 플랫폼 검증, 대규모 CPS 지원 플랫폼 구성이 포함되어 있다. IoT 플랫폼 요구사항과 CPS 시스템 요구사항 중 검증할 대상을 선정한다. IoT 플랫폼 검증을 위해 테스트베드를 구성하고, 테스트케이스를 설계하여 검증을 시행한다. 검증한 결과를 기반으로 대규모 CPS 지원 플랫폼 구조를 제안한다.

IoT 플랫폼이 어느 정도 CPS 플랫폼 기능을 지원하는 지 검증하기 위해 IoT 플랫폼 중 oneM2M 표준을 준수하는 Mobius를 선정했다. 그 기능 중 CPS에서 필요한 기능을 선별하여 IoT 플랫폼 검증 항목으로 선정한다. 자동차, 항공, 스마트공장에서 추출한 CPS 시스템 요구사항 중 IoT 플랫폼이 지원할 수 있는 CPS의 기능을 선별하여 CPS 시스템 요구사항 검증항목으로 선정한다.

검증 테스트베드 구현을 위한 요구사항을 정의하고, 테스트 케이스를 설계한다. 각각의 테스트 케이스는 검증항목으로 선정된 IoT 플랫폼의 기능을 검증이

가능하도록 설계한다. IoT 플랫폼이 제공하지 않은 CPS 플랫폼의 기능은 검증이 불가능하여 이 보고서의 검증 대상에서 제외한다.

IoT 플랫폼이 제공하지 않은 CPS 플랫폼의 기능은 스마트공장과 같이 특정 도메인에 연관된 기능이 다수 포함되어 있다. 예를 들면 공정 설정 기능, 모형화와 시뮬레이션 기능 등이다. 또한 IoT 플랫폼은 실시간성, 고신뢰성, 지능성을 지원하지 않는다.

검증한 결과를 기반으로 IoT 플랫폼이 제공하지 않는 CPS 플랫폼이 가져야 기능에 대해 분석하고, IoT 플랫폼 기반의 CPS 플랫폼 구조에 대해 제안한다.

제2절 검증 항목

1. IoT 플랫폼 요구사항 검증 항목

CPS 플랫폼 분석을 위한 IoT 플랫폼으로 oneM2M 표준을 준수하는 Mobius를 사용하고자 한다. oneM2M 표준은 에너지, 국방, 공공서비스 등 각기 운영되던 서비스 플랫폼 형식에서 벗어나, 인프라 환경을 통합하고 공유한 사물인터넷 표준이다. oneM2M은 유럽, 미국, 한국 등 표준 기준들이 참여해 공신력이 높고, 요구사항, 시스템 구조, 프로토콜, 보안 등 해결해야 하는 과제를 구조화하여 수행하고 있어, 요구사항에 관련된 연구인 이 연구에서 사용하기에 적당하다고 판단되었다.

oneM2M 표준에서 제시되어 있는 요구사항에 대한 설명과 각 요구사항이 oneM2M의 어느 Release에 정의되어 있는지에 대하여 정리한다. 또한 각 요구사항을 유사한 CPS 요구사항과 매핑하여 IoT 플랫폼을 활용한 검증을 구성하는데 활용한다.

oneM2M 요구사항은 시스템 요구사항 98종, 관리 요구사항 19종, 의미 요구사항 36종, 보안 요구사항 63종, 요금청구 요구사항 6종, 운영 요구사항 10종, 통신 요구사항 15종으로 구성되어 있다.

활용된 oneM2M 요구사항은 247종의 기능요구사항 중 공통 CPS 요구사항 및 스마트공장 요구사항과 관련된 기능인 88종이다.

<표 6-1> oneM2M 표준 요구사항 검증 항목 식별 표

요구사항 구분	설명	검증 항목 수
Overall System	전체적인 시스템 관점에서의 oneM2M 표준 요구사항에 대해 Overall System Requirements에서 정의	37
Management	oneM2M 표준에서 시스템 네트워크 차원의 관리에 대한 요구사항을 정의	9
Semantics	시멘틱 요구사항에서는 온톨로지, 시멘틱 어노테이션에 대한 oneM2M 표준에서의 필요사항을 정의	10
Security	전체적인 oneM2M 시스템 보안 관련 요구사항을 정의	20
Operational	oneM2M 시스템의 동작 상태 확인 및 관리 관련 요구사항을 정의	5
Communication Management	전체 시스템 상의 통신 관리에 대한 요구사항을 정의	7

247종 중 oneM2M 표준 규격에 반영되지 않은 기능요구사항들은 제외되었다. 또한 통신사 및 네트워크 운용사 입장에서의 요금 징수 등과 같은 요금청구 요구사항, 소프트웨어와 직접적인 관련이 없는 네트워크 인프라 구축을 위한 기능요구사항 등 공통 CPS 요구사항 및 스마트공장 요구사항과 관련되지 않은 요구사항들은 제외하였다.

1) oneM2M Overall System 요구사항

전체적인 시스템 관점에서의 oneM2M 표준 요구사항에 대해 Overall System Requirements에서 정의하고 있다.

<표 6-2> oneM2M 표준의 Overall System Requirements

요구사항 ID	설명	Release	CPS 요구사항 ID
OSR-001	oneM2M 시스템은 IP 액세스(IP access)를 기반으로 하는 다중 통신 수단을 사용하여 M2M 애플리케이션 간 통신을 허용해야 함	Rel-1	CPS-Scal-Req-003 CPS-Scal-Req-004 CPS-Operate-Req-001 SF-Operate-Req-001 SF-Operate-Req-010
OSR-007	oneM2M 시스템은 M2M 응용 프로그램이 다른 M2M 서비스 공급자가 관리하는 응용 프로그램 및 데이터 /	Rel-1	CPS-Inter-Req-004 CPS-Inter-Req-006

	정보와 상호 작용할 수 있는 메커니즘을 제공하며, 적절하게 사용권한을 부여 함		SF-Com-Req-002 SF-Com-Req-013
OSR-008	oneM2M 시스템은 M2M 응용 프로그램이 M2M 장치의 네트워크 기술 및 특정 통신 프로토콜을 인식 할 필요 없이 M2M 응용 프로그램이 M2M 장치 (즉 장치의 응용 프로그램)와 통신 할 수 있는 기능을 제공 함	Rel-1 (see note 11)	CPS-Scal-Req-003 CPS-Scal-Req-004 SF-Scal-Req-003 SF-Scal-Req-004 SF-Operate-Req-001 SF-Operate-Req-002
OSR-009	oneM2M 시스템은 단일 또는 다중 M2M 응용 프로그램이 단일 또는 다중 M2M 장치 / 게이트웨이 (장치 / 게이트웨이의 응용 프로그램)와 상호 작용할 수 있는 기능을 지원해야 함	Rel-1	CPS-Operate-Req-001 CPS-Operate-Req-003 SF-Operate-Req-004
OSR-010	oneM2M 시스템은 주어진 시간 간격 내에 메시지의 오류를 감지하기 위해 안정적인 전달을 요구하는 M2M 응용 프로그램의 수신자에게 메시지를 확인에 대한 메커니즘을 지원해야 함	Rel-1	CPS-Inter-Req-010 SF-Inter-Req-003
OSR-013	oneM2M 시스템은 M2M 응용 프로그램이 수용 할 수 있는 지연 허용(delay tolerance) 오차를 알고 있어야하며, 정책 기준에 따라 적절하게 통신을 계획하거나 기본 네트워크에 계획한 통신을 요청해야 함	Rel-1	CPS-Inter-Req-006 CPS-Inter-Req-009 SF-Operate-Req-004
OSR-014	oneM2M 시스템은 이기종(heterogeneous) M2M 영역 네트워크를 지원하는 M2M 게이트웨이 뒤에 M2M 장치와 통신 할 수 있어야 함	Rel-1	CPS-Operate-Req-001 SF-Operate-Req-004
OSR-016	oneM2M 시스템은 M2M 응용 프로그램 / M2M 장치 / 게이트웨이에 대한 사용 가능한 M2M 응용 프로그램 / 관리 정보의 가용성 및 변경 사항을 M2M 응용 프로그램에 알리는 기능을 제공함 (M2M 영역 네트워크 변경 포함)	Rel-1	CPS-Inter-Req-004 CPS-Inter-Req-010 SF-Com-Req-014
OSR-017	oneM2M 시스템은 M2M 응용 프로그램 제공자에게 다른 M2M 서비스 세트에 대한 액세스를 제공 할 수 있어야함 최소 서비스 세트는 다음과 같음 - 연결 관리 - 장치 관리(서비스 수준 관리) - 응용 프로그램 데이터 관리 다른 배포 시나리오를 가능하게하기 위해 이러한 서비스는 oneM2M 시스템에서 개별적으로 하위 집합 또는 전체 서비스 집합으로 제공되어야 함	Rel-1	CPS-Inter-Req-004 CPS-Operate-Req-003 SF-Inter-Req-015 SF-Operate-Req-010
OSR-020	oneM2M 시스템은 데이터 / 정보의 저장 및 검색 측면에 관한 정책 및 관리를 지원할 수 있어야 함	Rel-1	CPS-Inter-Req-004

OSR-021	oneM2M 시스템은 여러 M2M 애플리케이션 간에 데이터를 공유 할 수 있는 메커니즘을 제공 할 수 있어야 함	Rel-1	CPS-Inter-Req-002
OSR-022	M2M 솔루션의 일부 구성 요소 (예 : WAN 연결 끊김)를 사용할 수 없는 경우 oneM2M 시스템은 M2M 솔루션 구성 요소의 정상 작동을 지원할 수 있어야 함	Rel-1	CPS-Com-Req-004
OSR-023	oneM2M 시스템은 M2M 서비스 가입에 의해 사용될 M2M 서비스를 식별 할 수 있어야 함	Rel-1	SF-Scal-Req-001
OSR-024	oneM2M 시스템은 M2M 서비스 가입에 사용되는 M2M 장치를 식별 할 수 있어야 함	Rel-1	SF-Scal-Req-001
OSR-025	oneM2M 시스템은 M2M 서비스 가입에 사용되는 M2M 응용 프로그램을 식별 할 수 있어야 함	Rel-1	SF-Scal-Req-001
OSR-026	기본 네트워크가 제공하는 경우 oneM2M 시스템은 M2M 서비스 가입에서 사용하는 M2M 장치를 기본 네트워크 및 장치에서 제공하는 장치 식별자와 연결할 수 있음	Rel-1	CPS-Inter-Req-008
OSR-029	oneM2M 시스템은 그룹을 통해 각 액추에이터 또는 센서에 공통 명령을 보내는 것을 지원할 수 있어야 함	Rel-1	CPS-Inter-Req-008
OSR-030	oneM2M 시스템은 그룹 구성원의 관리 (추가, 제거, 검색 및 업데이트)를 지원할 수 있어야 함	Rel-1	SF-Inter-Req-013 SF-Inter-Req-016 SF-Com-Req-018
OSR-031	oneM2M 시스템은 그룹을 다른 그룹의 구성원으로 지원할 수 있어야 함	Rel-1	SF-Inter-Req-013 SF-Inter-Req-016 SF-Com-Req-018
OSR-032	oneM2M 시스템은 해당 데이터를 수집, 저장 및 보고할 때 M2M 응용 프로그램의 데이터와 관련된 이벤트 범주 (예 : 일반, 긴급)를 지원할 수 있어야 함	Rel-1	CPS-Depend-Req-005
OSR-043	oneM2M 시스템은 그룹 구성원이 공통 기능 집합을 지원하는지 확인할 수 있어야 함	Rel-1	SF-Com-Req-013 SF-Com-Req-018
OSR-044	oneM2M 시스템은 예측 불가능(unpredictable)하고 자발적(spontaneous)으로 도달 할 수 있는 M2M 장치 뿐만 아니라 정의된 시간 계획 (예 : 주기적)을 기반으로 도달 할 수 있는 M2M 장치와의 통신을 지원해야 함	Rel-1	CPS-Timing-Req-002 CPS-Timing-Req-003
OSR-047	oneM2M 시스템은 M2M 장치 및 / 또는 게이트웨이가 지리적 위치 정보를 M2M 응용 프로그램에 보고 할 수 있는 메커니즘을 지원할 수 있어야 함	Rel-1	SF-Inter-Req-015 SF-Inter-Req-016
OSR-048	oneM2M 시스템은 M2M 장치 및 / 또는 게이트웨이가 자신 또는 다른 M2M 장치의 지리적 위치 정보를 공유할 수 있게 해주는 M2M 서비스를 제공해야 함	Rel-1	SF-Inter-Req-017

OSR-051	기본 네트워크가 제공하는 적합한 인터페이스의 가용성에 따라 oneM2M 시스템은 지정된 영역의 M2M 장치 그룹에 broadcast/ multicast data에 대한 기본 네트워크를 요청할 수 있어야 함	Rel-1	CPS-Inter-Req-006 CPS-Inter-Req-009
OSR-054	oneM2M 시스템은 다른 M2M 응용 프로그램, M2M 장치 또는 M2M 게이트웨이(gateway)의 자원에 대한 액세스 권한을 얻기 위해 M2M 응용 프로그램, M2M 장치 또는 M2M 게이트웨이를 지원할 수 있어야 함	Rel-1	CPS-Inter-Req-006 CPS-Inter-Req-009
OSR-055	oneM2M 시스템은 미리 알려지지 않은 하나 또는 하나 이상의 허가된 M2M 애플리케이션과 데이터를 교환하기 위해 M2M 애플리케이션의 기능을 제공 할 수 있어야 함	Rel-1 (see note 20)	CPS-Scal-Req-004
OSR-056	oneM2M 시스템은 M2M 게이트웨이 또는 M2M 장치에서 사용 가능한 M2M 응용 프로그램을 검색 할 수 있어야 함	Rel-1	SF-Com-Req-013
OSR-057	oneM2M 시스템은 데이터 교환을 위해 M2M 애플리케이션에서 사용할 수 있는 M2M 게이트웨이 및 M2M 장치를 검색 할 수 있어야 함	Rel-1	CPS-Inter-Req-004 CPS-Inter-Req-010
OSR-063	oneM2M 시스템은 인프라 구조 도메인과 M2M 장치 / 게이트웨이 간의 M2M 서비스 계층 연결 및 메시징 스케줄링을 관리 할 수 있어야 함	Rel-1	CPS-Scal-Req-005 CPS-Operate-Req-001
OSR-064	oneM2M 시스템은 메시지 지연 허용치 및/또는 카테고리에 따라 메시지를 수집 할 수 있어야 함	Rel-1	CPS-Depend-Req-005
OSR-066	oneM2M 시스템, M2M 응용 서비스 제공자에 의해 요청된 기준에 따라 선택 M2M 노드에 M2M 응용 프로그램의 배치 및 운영을 지원할 수 있는 액세스 권한을 받아야 함	Rel-1	CPS-Com-Req-002
OSR-067	oneM2M 시스템은 M2M 애플리케이션이 요청한대로 운영 및 관리 조치를 취할 수 있어야 함	Rel-1	CPS-Inter-Req-010
OSR-071	oneM2M 시스템은 허가된 M2M 응용 프로그램이 M2M 장치의 M2M 서비스 관리 상태를 설정할 수 있어야 함	Rel-1	CPS-Inter-Req-010
OSR-075	oneM2M 시스템은 Time Series 데이터를 수집하고 저장할 수 있어야 함	Rel-2	CPS-Timing-Req-001
OSR-076	oneM2M 시스템은 누락된 데이터를 Time Series로 탐지하고 보고 할 수 있어야 함	Rel-2	CPS-Timing-Req-001
OSR-094	oneM2M 시스템은 서로 다른 장치 / 응용 프로그램 간의 상호운용성(interoperability)을 지원하기 위한 정보 모델을 제공해야 함	Rel-2	CPS-Operate-Req-001 CPS-Operate-Req-003

2) oneM2M Management Requirements

oneM2M 표준에서 시스템 네트워크 차원의 관리에 대한 요구사항을 정의하고 있다.

<표 6-3> oneM2M 표준의 Management Requirements

요구사항 ID	설명	Release	CPS 요구사항 ID
MGR-001	oneM2M 시스템은 자원이 제한된 M2M 장치를 포함하여 M2M 게이트웨이 / 장치의 관리 및 구성을 지원할 수 있어야 함	Rel-1	CPS-Scal-Req-002 CPS-Com-Req-002
MGR-002	oneM2M 시스템은 해당 네트워크의 장치 및 해당 네트워크의 매개변수(예 : topology, protocol)에 대한 정보를 포함하여 M2M 영역 네트워크를 검색 할 수 있는 기능을 제공해야 함	Rel-1	CPS-Inter-Req-006 CPS-Inter-Req-010
MGR-003	oneM2M 시스템은 M2M 영역 네트워크의 장치 및 매개 변수(예 : topology, protocol)의 관리 정보 모델을 유지하고 설명 할 수 있는 기능을 제공 할 수 있어야 함	Rel-1	CPS-Inter-Req-004 CPS-Inter-Req-009
MGR-008	oneM2M 시스템은 M2M 영역 네트워크의 장치를 소프트웨어로 관리할 수 있는 기능을 제공해야 함	Rel-1	CPS-Inter-Req-002
MGR-009	oneM2M 시스템은 M2M 게이트웨이 / 장치 및 M2M 영역 네트워크의 다른 장치를 재부팅 및 / 또는 재설정 할 수 있는 기능을 제공해야함	Rel-1	CPS-Depend-Req-002
MGR-010	oneM2M 시스템은 장치가 M2M 영역 네트워크에 액세스하도록 권한을 부여하는 기능을 제공해야 함	Rel-1	SF-Depend-Req-001 SF-Depend-Req-002
MGR-011	oneM2M 시스템은 M2M 영역 네트워크의 장치 topology를 수정할 수 있는 기능을 제공해야하며, M2M 영역 네트워크 정책에 따라 제한을 받음	Rel-1	CPS-Scal-Req-003 CPS-Inter-Req-008 CPS-Inter-Req-009
MGR-014	oneM2M 시스템은 M2M 게이트웨이 / 장치 및 M2M 지역 네트워크의 다른 장치가 기록한 이벤트 및 정보를 검색 할 수 있어야 함	Rel-1	SF-Com-Req-013
MGR-015	oneM2M 시스템은 M2M 게이트웨이 / 장치 및 M2M 영역 네트워크의 다른 장치에 대한 펌웨어 관리(예 : 업데이트)를 지원할 수 있어야함	Rel-1	SF-Com-Req-003 SF-Com-Req-004 SF-Com-Req-006

3) oneM2M Semantics(의미) Requirements

시멘틱 요구사항에서는 온톨로지, 의미, 표현에 대한 oneM2M 표준에서의 필요사항을 정의하고 있다.

(1) oneM2M Ontology[149] Related Requirements

oneM2M 시스템에서의 온톨로지 사용 관련 요구사항을 정의하고 있다.

149) 공유된 개념에 대한 정형화되고 명시적인 명세(formal and explicit specification). 온톨로지는 단어와 관계들로 구성된 일종의 사전으로서 생각할 수 있으며, 그 속에는 특정 도메인에 관련된 단어들이 계층적으로 표현되어 있고, 추가적으로 이를 확장할 수 있는 추론 규칙이 포함

<표 6-4> oneM2M 표준의 Ontology Related Requirements

요구사항 ID	설명	Release	CPS 요구사항 ID
ONT-002	M2M 시스템은 ontologies를 사용하여 사물의 의미론적 기술(모델 간의 관계 포함) 모델링을 지원해야 함	Rel-2	SF-Scal-Req-010 SF-Com-Req-018
ONT-003	M2M 시스템은 ontologies를 위한 공통 모델링 언어를 지원해야함(예 : OWL)	Rel-2	SF-Scal-Req-010 SF-Com-Req-018
ONT-008	M2M 시스템은 M2M 시스템의 자원으로 표현되지 않는 측면(예 : 방)을 나타내는 개념을 포함하는 ontologies를 사용할 수 있어야 함	Rel-2	SF-Scal-Req-010 SF-Com-Req-018
ONT-010	M2M 시스템은 동일한 M2M 자원에 대한 다중 ontologies의 동시 사용을 지원할 수 있어야 함	Rel-2	SF-Com-Req-015 SF-Com-Req-018
ONT-015	M2M 시스템은 M2M 시스템 외부에서 사용할 수 있는 ontologies(예 : HGI 장치 템플릿)를 기반으로 장치를 모델링 할 수 있어야 함	Rel-2	SF-Com-Req-015 SF-Com-Req-018

(2) oneM2M Semantic Annotation Requirements

oneM2M 시스템의 의미 및 주석 정의 관련 요구사항을 정의하고 있다.

<표 6-5> oneM2M 표준의 Semantics Annotation Requirements

요구사항 ID	설명	Release	CPS 요구사항 ID
ANN-001	oneM2M 시스템은 oneM2M 자원에 대한 의미 정보를 관리하는 기능을 제공해야 함, e.g, create, retrieve, update, delete, associate/link.	Rel-2	SF-Com-Req-015 SF-Com-Req-018
ANN-002	oneM2M 시스템은 semantic description(의미론적 설명)을 위한 공통 언어를 지원해야 함, e.g. RDF.	Rel-2	SF-Scal-Req-010 SF-Com-Req-018
ANN-003	oneM2M 시스템은 컨테이너에 포함 된 응용 프로그램 관련 데이터와 같은 oneM2M 자원의 semantic annotation을 지원해야 함	Rel-2	SF-Com-Req-015 SF-Com-Req-018 SF-Com-Req-013
ANN-004	oneM2M 시스템은 관련 온톨로지를 기반으로 semantic annotation을 지원해야 함	Rel-2	SF-Com-Req-015 SF-Com-Req-018 SF-Com-Req-013
ANN-005	oneM2M 시스템은 M2M 시스템에서 semantic descriptions을 사용할 수 있는 기능을 제공해야 함, e.g. announcement	Rel-2	SF-Com-Req-015 SF-Com-Req-018 SF-Com-Req-013

4) oneM2M Security Requirement

전체적인 oneM2M 시스템 보안 관련 요구사항을 정의하고 있다.

<표 6-6> oneM2M 표준의 Security Requirements

요구사항 ID	설명	Release	CPS 요구사항 ID
SER-002	oneM2M 시스템은 데이터의 기밀성을 보장 할 수 있어야 함	Rel-1	CPS-Depend-Req-001
SER-003	oneM2M 시스템은 데이터의 무결성을 보장 함	Rel-1	CPS-Inter-Req-005 CPS-Depend-Req-001 CPS-Depend-Req-005
SER-008	oneM2M 시스템은 승인되지 않은 M2M 서비스 및 M2M 응용 프로그램 서비스 연결에 대비한 대응책을 지원해야 함	Rel-1	CPS-Depend-Req-002 CPS-Depend-Req-003
SER-009	oneM2M 시스템은 Underlying Networks, M2M 서비스 및 M2M 응용 프로그램 서비스와의 상호 작용을 위해 상호 인증을 지원할 수 있어야 함	Rel-1	CPS-Operate-Req-001 CPS-Operate-Req-002 CPS-Operate-Req-003
SER-010	oneM2M 시스템은 Security credentials의 오용, 복제, 대체 또는 절도로부터 보호하기 위한 메커니즘을 지원할 수 있어야 함	Rel-1	CPS-Depend-Req-001 CPS-Depend-Req-005
SER-014	oneM2M 시스템은 M2M 게이트웨이 / 장치에서 인증되고 허가 된 M2M 응용 프로그램에 구성 데이터(configuration data)를 제공 할 수 있어야 함	Rel-1	CPS-Depend-Req-005
SER-022	oneM2M 시스템은 M2M 애플리케이션 서비스 제공 업체가 지원하는 엔티티(예 : 장치 / 게이트웨이 / 서비스 인프라)에서 M2M 애플리케이션과 관련된 상호 작용을 인증 할 수 있게 함	Rel-1	CPS-Depend-Req-001 CPS-Operate-Req-003
SER-027	M2M 시스템은 특정 자원에 대해 동일한 액세스 제어 권한을 가진 M2M 응용 프로그램의 그룹화를 지원하여 M2M 응용 프로그램이 특정 그룹의 구성원인지 여부를 확인함으로써 액세스 제어 유효성 검사를 수행 할 수 있도록 함	Rel-2	CPS-Depend-Req-005
SER-028	oneM2M 시스템은 보안 프로토콜 종단점(Security protocol end-points)이 개별 응용 프로그램에서 생성 한 데이터의 일부를 보호하여 데이터를 전달하는 중간 엔터티 (신뢰할 수 있는지 여부를 불문하고)가 일반 텍스트로 데이터의 보호 된 부분에 액세스 할 수 없도록 함	Rel-2	CPS-Depend-Req-005
SER-029	oneM2M 시스템은 보안 프로토콜 종단점(Security protocol end-points)이 개별 응용 프로그램에서 생성한 데이터의 일부를 보호 할 수 있도록 하여 보안 프로토콜 종단점이 데이터를 전달하는 중간 서비스 계층 엔터티(신뢰할 수 있는지 여부를 불문하고)의 수정을 포함하여 수정 사항을 탐지 할 수 있도록 함	Rel-2	CPS-Depend-Req-002
SER-030	oneM2M 시스템은 보안 프로토콜 종단점이 개별 oneM2M 메시지의 부분을 보호하여 메시지를 전달하는 중간 엔티티(신뢰할 수 있는지 여부를 불문하고)는 일반 텍스트 메시지의 보호 부분에 액세스 할 수 없도록 함	Rel-2	CPS-Depend-Req-005
SER-031	oneM2M 시스템은 보안 프로토콜 종단점이 개별 oneM2M 메시지의	Rel-2	CPS-Depend-Req-002

	일부를 보호 할 수 있도록 하여 보안 프로토콜 종단점이 메시지를 전달하는 중간 서비스 계층 엔티티(신뢰할 수 있는지 여부를 불문하고)의 수정을 포함하여 수정 사항을 탐지 할 수 있도록 함		
SER-032	oneM2M 시스템은 보안 프로토콜 종단점이 하나 이상의 oneM2M 메시지 부분을 보호하는 데 사용되는 보안 세션을 설정하여 메시지를 전달하는 중간 엔터티 (신뢰할 수 있는지 여부를 불문하고)가 해당 메시지의 보호 된 부분에 액세스 할 수 없도록 함	Rel-2	CPS-Depend-Req-001 CPS-Depend-Req-005
SER-033	oneM2M 시스템은 보안 프로토콜 종단점이 하나 이상의 oneM2M 메시지의 부분을 보호하기 위해 보안 세션을 설정하여 보안 프로토콜 종단점이 메시지를 전달하는 중간 서비스 계층 엔티티 (신뢰할 수 있는지 여부를 불문하고)에 의한 수정을 포함하여 수정 사항을 감지 할 수 있도록 함	Rel-2	CPS-Depend-Req-001 CPS-Depend-Req-002
SER-047	oneM2M 시스템은 기밀 보호를 위해 개별 응용 프로그램 데이터 (예 : 호스팅 된 데이터)의 일부를 보호 할 수 있어야 함	Rel-2	CPS-Depend-Req-001 CPS-Depend-Req-005
SER-048	oneM2M 시스템은 기밀성, 무결성, 및 변조 방지를 위해 종단 간 데이터 자격증명(end-to-end data Credentials)의 보호를 보장해야 함	Rel-2	CPS-Depend-Req-005
SER-050	oneM2M 시스템은 사전 정의 된 조건을 무단 수정으로부터 보호 할 수 있어야 함	Rel-2	CPS-Depend-Req-001 CPS-Depend-Req-003
SER-051	oneM2M 시스템은 인가 된 엔티티의 요청에 따라 M2M 장치 / 게이트웨이가 생성 / 저장 한 M2M 데이터를 삭제할 수 있어야 함	Rel-2	CPS-Depend-Req-003
SER-053	oneM2M 시스템은 법적 요구 사항, 제조업체 및 데이터 주체의 상태를 관리하기 위해 다양한 수준의 privacy profiles를 지원 함	Rel-2	CPS-Depend-Req-005
SER-059	oneM2M 시스템은 위임 된 액세스 권한의 진위성, 무결성 및 기밀성을 보호해야 함	Rel-2	CPS-Depend-Req-005

5) oneM2M Operational Requirements

oneM2M 시스템의 동작 상태 확인 및 관리 관련 요구사항을 정의하고 있다.

<표 6-7> oneM2M 표준의 Operational Requirements

요구사항 ID	설명	Release	CPS 요구사항 ID
OPR-001	oneM2M 시스템은 M2M 응용 프로그램의 모니터링 및 진단 기능을 제공해야 함	Rel-1	SF-Scal-Req-001
OPR-002	oneM2M 시스템은 M2M 응용 프로그램의 소프트웨어 관리 기능을 제공해야 함	Rel-1	SF-Com-Req-006 SF-Inter-Req-001
OPR-003	oneM2M 시스템은 M2M 응용 프로그램 (시작, 중지, 재시작)의 실행 상태를 구성 할 수 있어야 함	Rel-1	SF-Com-Req-014
OPR-005	oneM2M 시스템은 M2M 응용 프로그램에 의해 M2M 장치 또는 M2M 게이트웨이의 사용 및 트래픽 특성과 관련하여 M2M 응용 프로그램과 정보를 교환 할 수 있어야 함	Rel-2	CPS-Operate-Req-001 CPS-Operate-Req-003

	여기에는 다음과 같은 3GPP 기능에 대한 지원이 포함되어야 함: “Time controlled“ (see note).		
OPR-006	기본 네트워크가 제공하는 적합한 인터페이스의 가용성에 따라 oneM2M 시스템은 M2M 장치 또는 M2M 게이트웨이의 사용 및 트래픽 특성과 관련된 정보를 기본 네트워크에 제공 할 수 있어야 함	Rel-2	CPS-Inter-Req-004 CPS-Inter-Req-010 CPS-Operate-Req-001

6) oneM2M Communication Management Requirements

전체 시스템 상의 통신 관리에 대한 요구사항을 정의하고 있다.

<표 6-8> oneM2M 표준의 Communication Management Requirements

요구사항 ID	설명	Release	CPS 요구사항 ID
CMR-001	oneM2M 시스템은 M2M Gateway / Device / Infrastructure Domain으로 또는 M2M Gateway / Device / Infrastructure Domain으로부터 메시지를 버퍼링하는 통신 서비스를 M2M Applications에 제공 함	Rel-1	CPS-Inter-Req-006
CMR-002	oneM2M 시스템은 통신 정책과 버퍼링 된 메시지와 관련된 서비스 환경 설정에 따라 버퍼 된 메시지 전달을 지원할 수 있어야 함	Rel-1	CPS-Inter-Req-007
CMR-003	oneM2M 시스템은 M2M 응용 프로그램이 다음 서비스 환경 설정을 사용하여 통신 요청을 보낼 수 있게 함: 데이터 전달을 시작하기 위한 지연 허용 오차를 포함한 QoS 매개 변수; 다른 수준의 우선 순위 또는 QoS 클래스로의 통신 요청 분류;	Rel-1	CPS-Inter-Req-009
CMR-004	oneM2M 시스템은 규정 된 통신 정책을 준수하면서 메시지와 관련된 서비스 선호도에 대한 인지와 함께 다른 출처에서 M2M 게이트웨이 및 / 또는 M2M 장치 내에서 메시지의 동시 처리를 지원할 수 있어야 함	Rel-1	CPS-Inter-Req-003 CPS-Inter-Req-010 CPS-Timing-Req-002 CPS-Timing-Req-003
CMR-006	oneM2M 시스템은 응용 프로그램이 요청 된 통신(우선 순위, 중요도 등)을 분류 할 수 있도록 지원하여 oneM2M 시스템이 이 categorization를 고려하여 실제 통신(일정 계획, 집계, 압축 등)을 조정할 수 있도록 함	Rel-1	CPS-Inter-Req-009 CPS-Timing-Req-001 CPS-Timing-Req-005
CMR-008	oneM2M 시스템은 M2M Gateway / Device / Infrastructure Domain 간에 데이터를 교환 할 때 통신 정책을 기반으로 데이터 집계를 지원 함	Rel-1	CPS-Inter-Req-003
CMR-015	oneM2M 시스템은 일련의 데이터 (예 : 시계열 데이터)를 식별하고 이 시리즈에 속하는 개별 데이터를 나타낼 수 있어야 함	Rel-2	CPS-Timing-Req-004

2. CPS 시스템 요구사항 검증항목

5장에서 정의된 스마트공장 시스템 요구사항과 CPS 공통 요구사항 항목을 정리하여 IoT 플랫폼 표준인 oneM2M과 대조하여 검증을 진행한다. oneM2M 기반 플랫폼 중 가장 광범위

하게 사용되고 있는 오픈 IoT 플랫폼인 Mobius 플랫폼을 검증 플랫폼으로 선정하였고, 이를 통하여 기능 지원 여부를 확인할 정성적 검증 대상을 식별하였다. 다음 7개의의 표들은 각각의 요구사항 항목에 따른 스마트공장 요구사항과 CPS 공통 요구사항을 기반으로 검증 내용을 설명한다. 단 스마트공장 요구사항과, CPS 공통 요구사항이 유사한 요구사항을 명시할 경우 하나의 항목으로 묶어서 검증내용을 도출한다.

검증내용 항목을 통하여 Mobius를 통하여 정성적 검증이 가능한 요구사항인지 아닌지를 판별하였다. IoT 플랫폼을 통하여 제공되어야 할 기능들은 검증 테스트케이스를 명시하여 검증 가능 여부를 표기하였고, oneM2M 표준에서 전혀 고려되지 않는 CPS 요구사항은 검증 불가 항목으로 표기하여 각 항목별로 검증 불사 사유를 설명한다. 각 요구사항의 검증 여부는 표의 가장 오른쪽 검증 열의 O(검증 진행 대상이 맞음)/X(검증 진행 대상이 아님)를 통하여 확인할 수 있다.

각 요구사항 항목별로 확장성 7종, 융복합성 6종, 상호작용성 12종, 고신뢰성 3종, 실시간성 1종 그리고 상호운용성 8종이 도출되어 총 37종의 요구사항이 검증 대상으로 식별되었고 지능적 요구사항은 AI와 같은 지능형 서비스와 관련된 요구사항들이 많아 IoT 플랫폼을 통한 검증에는 어려움이 있어 검증 대상에서 배제되었다.

<표 6-9> CPS 검증 대상 식별 요약

요구사항 분류	대상 요구사항 수
확장성(Scalability)	7
융복합성(Composability)	6
상호작용성(Interactivity)	12
고신뢰성(dependability)	3
실시간성(Timing)	1
상호운용성(Interoperability)	8
지능성(Intelligence)	0

1) 시스템 확장성(Scalability)

oneM2M 표준은 스마트공장 관련 공정설계나 관리 등의 요구 사항은 지원하지 않기 때문에 이에 관련된 시스템 확장성은 지원하지 않는다.

<표 6-10> CPS 확장성 요구사항 식별표

CPS 요구사항 ID	검증 상세	검증
SF-Scal-Req-001	공장 기기 설비들의 연동 구조 실시간 모니터링 검증 테스트케이스 : LifeCycle 테스트케이스	O
SF-Scal-Req-002 CPS-Scal-Req-001	기기의 추가, 제거 및 변동 시 다른 기기들 간 연결 영향 검증 불가 사유 : Mobius 플랫폼은 기기들의 추가/제거 및 변동 시 다른 기기에 영향성 확인 기능을 제공하지 않음	X
SF-Scal-Req-003 CPS-Scal-Req-003	클라이언트/서버 통신 기술 지원 검증 테스트케이스 : Transport 테스트케이스	O
SF-Scal-Req-004 CPS-Scal-Req-003	Pub/Sub 모델의 통신 기술 지원 검증 테스트케이스 : Transport 테스트케이스	O
SF-Scal-Req-005 CPS-Scal-Req-003	Pub/Sub 모델 통신 기술을 통한 데이터 송신 검증 테스트케이스 : Transport 테스트케이스	O
SF-Scal-Req-006 CPS-Scal-Req-003	Pub/Sub 모델 통신 기술을 통한 데이터 수신 검증 테스트케이스 : Transport 테스트케이스	O
SF-Scal-Req-007 CPS-Scal-Req-004	전체 시스템의 변동에 따른 네트워크 구조의 유연한 변경 기능 지원 검증 불가 사유 : Mobius 플랫폼은 팩토리 변동에 따라 네트워크 연결 구조를 변경할 수 없음	X
SF-Scal-Req-008	전체 시스템의 변동에 따른 서브시스템의 작업 스케줄링, 연동 구조 등이 적합하게 설정될 수 있는 기능의 지원 검증 불가 사유 : Mobius 플랫폼은 팩토리 변동에 따라 제어 시스템의 구조 및 기능을 변경할 수 없음	X
SF-Scal-Req-009	정보 시스템의 구조 및 기능의 전체 팩토리 변동에 따른 적합한 변동 기능의 지원 검증 불가 사유 : Mobius 플랫폼은 팩토리 변동에 따라 정보 시스템의 구조 및 기능을 변경할 수 없음	X
SF-Scal-Req-010	모든 기기에서 제어 메시지 해석의 동일성 기능 지원 검증 테스트케이스 : Semantic 테스트케이스	O
SF-Scal-Req-011 CPS-Scal-Req-005	기기 설비들의 상태를 고려한 공정 설정이 가능한 기능 지원 검증 불가 사유 : Mobius 플랫폼은 공정 설정 기능을 제공하지 않음 (인력/재료/공정/설비 모델링)	X
SF-Scal-Req-012 CPS-Scal-Req-005	공정 설정 변경에 따른 기기 간 연동 구조 변경 기능 지원 검증 불가 사유 : Mobius 플랫폼은 공정 설정 기능을 제공하지 않음	X
SF-Scal-Req-013	스마트공장 구성을 돕기 위한 분산 M&S 환경 제공되어야 함 검증 불가 사유 : Mobius 플랫폼은 분산 모델링&시뮬레이션 엔진을 제공하지 않음	X
CPS-Scal-Req-002	기능 모듈 인스턴스 생성 및 연결 기능 지원 검증 테스트케이스 : S/W & H/W Manager	O

2) 융복합성(Composability)

oneM2M 표준은 하드웨어의 자유로운 추가와 제거, 소프트웨어 업그레이드 관련 작업 등의 기능은 지원하지 않는다.

<표 6-11> CPS 융복합성 요구사항 식별표

CPS 요구사항 ID	검증 상세	검증
SF-Com-Req-001	원격 S/W업그레이드 기능 지원 검증 테스트케이스 : S/W & H/W Manager	O
SF-Com-Req-002	S/W업그레이드에 대한 권한 관리 기능 지원 검증 테스트케이스 : Security 테스트케이스	O
SF-Com-Req-003	제품 생산 중 S/W업그레이드 지원 검증 테스트케이스 : Mobius 플랫폼은 프로세스 실행 중 업그레이드를 지원하지 않음	X
SF-Com-Req-004 CPS-Com-Req-003	실시간으로 서브시스템의 최신 S/W버전 유무 판단 및 알림 기능 지원 검증 불가 사유 : Mobius 플랫폼은 S/W버전 판단은 가능하지만 실시간성을 보장하지 못함	X
SF-Com-Req-005	S/W업그레이드 후의 변경 사항 알림 기능 지원 검증 테스트케이스 : S/W & H/W Manager	O
SF-Com-Req-006 CPS-Com-Req-003	사용자에 직관적인 S/W업그레이드 필요 구성요소 알림 가능 여부 검증 불가 사유 : Mobius 플랫폼은 S/W 업그레이드가 필요한 구성요소를 판별 불가능	X
SF-Com-Req-007	S/W업그레이드를 위한 소요 시간 측정 및 알림 검증 불가 사유 : Mobius 플랫폼 외 업그레이드 소요 시간 측정을 위한 어플리케이션 필요	X
SF-Com-Req-008 CPS-Com-Req-002	고객 요구에 따른 제품 생산을 위한 H/W모듈 추가/제거 가능 여부 검증 테스트케이스 : Mobius 플랫폼은 고객 요구에 따른 제품 생산 판단이 불가능	X
SF-Com-Req-009 CPS-Com-Req-002	H/W 추가/제거의 알림 기능 검증 테스트케이스 : S/W & H/W Manager	O
SF-Com-Req-010 CPS-Com-Req-004	H/W 추가/제거 이상 동작 시 조치 기능 검증 불가 사유 : Mobius 플랫폼은 H/W 추가/제거로 인한 이상 동작 판단 불가능	X
SF-Com-Req-011	특정 제품 제작 시 필요한 H/W모듈 예상 및 알림 검증 불가 사유 : Mobius 플랫폼은 특정 제품 제작 시 필요한 H/W 모듈의 판단 불가능	X
SF-Com-Req-012	사용자의 새로운 공정 환경 적응에 도움 지원 검증 불가 사유 : Mobius 플랫폼은 Smart Factory의 공정 환경 판단 기능을 지원하지 않음	X
SF-Com-Req-013	사용자의 팩토리 구성 요소들 관찰 가능 여부	O

CPS-Com-Req-001	검증 테스트케이스 : S/W & H/W Manager	
SF-Com-Req-014	구성요소 추가/제거 시 변하는 복잡성 계산 및 알림 가능 여부 검증 불가 사유 : Mobius 플랫폼은 Smart Factory 구성요소 추가/제거 시 변동에 따른 복잡성 계산 기능을 제공하지 않음	X
SF-Com-Req-015 CPS-Com-Req-005	지능적으로 복잡성을 최소로 낮추기 위한 최적 설계 제공 여부 검증 불가 사유 : Mobius 플랫폼은 지능적인 최적 설계 제안 불가능	X
SF-Com-Req-016	시뮬레이션을 통한 시스템 변동의 무결성에 대한 검증 기능 검증 불가 사유 : Mobius 플랫폼은 확장성 검증을 위한 시뮬레이션 기능을 제공하지 않음	X
SF-Com-Req-017 CPS-Com-Req-005	사용자의의 요구를 만족시킬 수 있는 최적의 전체 시스템 배치 및 연동 구조 도출 검증 불가 사유 : Mobius 플랫폼은 고객의 요구를 인지하지 못함	X
SF-Com-Req-018 CPS-Com-Req-003	구성요소에 부여된 고유 식별자를 통한 제품 제작 구성요소 및 기능 식별 가능성 검증 테스트케이스 : S/W & H/W Manager	O
SF-Com-Req-019	시스템 개발 및 검증을 위하여 구성요소의 하이브리드 M&S를 지원해야 함 검증 불가 사유 : Mobius 플랫폼은 M&S기능을 포함하지 않고 있음	X
SF-Com-Req-020	시스템 개발 및 검증을 위하여 구성요소의 분산 M&S를 지원해야 함 검증 불가 사유 : Mobius 플랫폼은 M&S 기능을 포함하지 않고 있음	X

3) 상호작용성(Interactivity)

oneM2M 표준은 상호작용 시 실시간성을 지원할 수 없고, 현장 기술자들의 작업 지원이 어렵다.

<표 6-12> CPS 상호작용성 요구사항 식별표

CPS 요구사항 ID	검증 상세	검증
SF-Inter-Req-001 CPS-Inter-Req-001 CPS-Inter-Req-008	관리 서버의 각 기기들에 제어 메시지 송신 검증 테스트케이스 : Transport 테스트케이스	O
SF-Inter-Req-002 CPS-Inter-Req-001 CPS-Inter-Req-008	관리 서버의 각 기기들에 제어 메시지 수신 검증 테스트케이스 : Transport 테스트케이스	O
SF-Inter-Req-003 CPS-Inter-Req-005 CPS-Inter-Req-008	구성 기기들의 관리 서버 제어 메시지에 대한 적합한 해석 가능 여부 검증 테스트케이스 : Semantic 테스트케이스	O
SF-Inter-Req-004 CPS-Inter-Req-008	구성 기기들의 관리 서버로부터 받아온 제어 메시지 표시 가능 여부 검증 테스트케이스 : S/W & H/W Manager	O

SF-Inter-Req-005 CPS-Inter-Req-001	구성 기기들의 관리 서버로 상태 정보 전달 가능 여부 검증 불가 사유 : Mobius 플랫폼은 구성 기기들의 상태 정보를 확인할 수 있지만 실시간성을 보장하지 못함	X
SF-Inter-Req-006	구성 기기들의 상태 정보 표시 가능 여부 검증 불가 사유 : Mobius 플랫폼은 구성 기기들의 상태 정보를 확인할 수 있지만 실시간성을 보장하지 못함	X
SF-Inter-Req-007	관리 서버의 구성 기기 상태 정보 표시 가능 여부 검증 테스트케이스 : S/W & H/W Manager	O
SF-Inter-Req-008	현장 기술자의 구성 기기들 조작 지원 검증 불가 사유 : Mobius 플랫폼은 현장 기술자가 구성 기기로의 데이터 전송은 지원하지만 실제 조작이 제대로 이루어졌는지 판단 불가능	X
SF-Inter-Req-009	구성 기기들의 현장 기술자 조작 반영 여부 검증 불가 사유 : Mobius 플랫폼은 현장 기술자 조작에 따라 구성 기기들의 동작이 제대로 이루어졌는지 확인 불가능	X
SF-Inter-Req-010	구성 기기들의 현장 기술자 조작에 대해 관리 서버에 전달 가능 여부 Mobius 플랫폼은 구성 기기들의 동작이 제대로 이루어졌는지 판단이 불가능 하여 서버에 전달할 수 없음	X
SF-Inter-Req-011 CPS-Inter-Req-004	구성 기기들의 이상 상황 서버에 알림 검증 불가 사유 : Mobius 플랫폼은 구성 기기들의 이상 상황 판단 기능을 제공하지 않음	X
SF-Inter-Req-012 CPS-Inter-Req-004	구성 기기들의 이상 상황 현장 기술자에 알림 검증 불가 사유 : Mobius 플랫폼은 구성 기기들의 이상 상황 판단 기능을 제공하지 않음	X
SF-Inter-Req-013	구성 기기들의 서버로부터 생산 제품에 대한 정보 수신 가능 여부 검증 테스트케이스 : Lifecycle 테스트케이스	O
SF-Inter-Req-014	구성 기기들의 서버로부터 생산 제품에 대한 정보 갱신 가능 여부 검증 불가 사유 : Mobius 플랫폼에서 Container의 변경은 가능하나, AE의 변경에 따른 Container들의 갱신은 지원하지 않음	X
SF-Inter-Req-015	관리 서버의 공장 상태에 따른 공정 정보 설정 가능 여부 검증 테스트케이스 : Lifecycle 테스트케이스	O
SF-Inter-Req-016	구성 기기들의 서버로부터 현재 공정에 대한 정보 수신 가능 여부 검증 테스트케이스 : Lifecycle 테스트케이스	O
SF-Inter-Req-017	구성 기기들의 공정 설정에 따른 작업 지원 검증 테스트케이스 : Lifecycle 테스트케이스	O
CPS-Inter-Req-002	구성 소프트웨어간 통신 지원 검증 테스트케이스 : Transport 테스트케이스	O
CPS-Inter-Req-003	이종 외부 환경과의 통신 지원 검증 테스트케이스 : Transport 테스트케이스	O
CPS-Inter-Req-006	각 시스템에 최적화 된 통신 기술 지원	O

	검증 테스트케이스 : Transport 테스트케이스	
CPS-Inter-Req-007	구성 기기들 사이의 데이터 전송 시 치명적인 오류가 전송되었을 때 그 전송을 취소할 수 있는 기능 제공 검증 불가 사유 : 전송이 시작된 메시지를 다시 회수할 수 있는 기능이 현재 개발되어 있지 않음	X
CPS-Inter-Req-009	요구사항을 만족시키는 통신 환경 제공 검증 불가 사유 : 플랫폼 만으로는 요구사항을 만족시키는 통신 환경을 제공하기에는 어려움이 있음 SDN 등 추가 기술 필요	X

4) 고신뢰성(Dependability)

oneM2M 표준이 지원할 수 없는 실시간성, 보안, 스마트공장 공정 변경, 시뮬레이션 기능 등 때문에 고신뢰성에 대한 제약이 많이 발견되었다.

<표 6-13> CPS 고신뢰성 요구사항 식별표

CPS 요구사항 ID	검증 상세	검증
SF-Depend-Req-001 CPS-Depend-Req-001	서버에 있는 기기 정보 접근에 관한 보안 체계 검증 테스트케이스 : Security 테스트케이스	O
SF-Depend-Req-002 CPS-Depend-Req-001	제어 네트워크 접근에 관한 보안 체계 검증 불가 사유 : SF-Depend-Req-001과 유사하게 주고받는 정보에 대한 보안 기술의 필요성을 제시하고 있어 같은 공통 요구사항으로 분류되었지만 Mobius 플랫폼은 네트워크에 대한 보안 요소를 제공하지 않아 SF-Depend-Req-002는 검증이 불가능	X
SF-Depend-Req-003	관리 서버의 공정에 따른 기기별 제어 로직 설정 가능 여부 검증 불가 사유 : Mobius 플랫폼은 기기에 특화된 제어 로직의 설정을 지원하지 않음	X
SF-Depend-Req-004	구성 기기들의 설정된 제어 로직 실행 여부 검증 불가 사유 : Mobius 플랫폼은 구성 기기의 실행 상태에 대한 확인 기능을 지원하지 않음	X
SF-Depend-Req-005	구성 기기들의 네트워크를 통한 원격 설정 시 정확한 설정 지원 여부 검증 테스트케이스 : Lifecycle 테스트케이스	O
SF-Depend-Req-006	시스템 이상 상황 발생을 지능적으로 예측 검증 불가 사유 : Mobius 플랫폼은 도메인에 최적화 된 지능적 기능을 지원하지 않음	X
SF-Depend-Req-007 CPS-Depend-Req-002	기기 결함 상황의 발생을 실시간으로 확인 가능 여부 검증 불가 사유 : Mobius 플랫폼은 실시간 데이터 전송을 보장하지 않음	X
SF-Depend-Req-008 CPS-Depend-Req-003	기기 결함 상황을 공정 변경 등을 통한 대처 가능여부 검증 불가 사유 : Mobius 플랫폼은 공정 재설정 등의 기능을 지원하지	X

	않음	
SF-Depend-Req-009 CPS-Depend-Req-002	실시간으로 공정 이상 상황의 발생 확인 가능 여부 검증 불가 사유 : Mobius 플랫폼은 실시간 데이터 전송을 보장하지 않음	X
SF-Depend-Req-010 CPS-Depend-Req-003	공정 이상 상황의 발생 시 서버에서의 대처 가능 여부 검증 불가 사유 : Mobius 플랫폼은 공정 재설정 기능을 지원하지 않음	X
SF-Depend-Req-011 CPS-Depend-Req-002	네트워크 이상 상황 감지 검증 불가 사유 : Mobius 플랫폼은 네트워크의 이상 상황을 감지할 수 없음	X
SF-Depend-Req-012 CPS-Depend-Req-003	네트워크 이상 상황 발생 시 네트워크 재설정을 통한 대처 가능 여부 검증 불가 사유 : Mobius 플랫폼은 네트워크 재설정 기능을 지원하지 않음	X
SF-Depend-Req-013	작업 환경 실시간 확인 검증 불가 사유 : Mobius 플랫폼은 실시간 데이터 전송을 지원하지 않음	X
SF-Depend-Req-014	작업 환경 적합한 조절 가능 여부 검증 불가 사유 : Mobius 플랫폼은 각 시스템에게 ‘적합한’ 환경을 도출해 낼 수 없음	X
SF-Depend-Req-015	구성 기기들의 상호 연동을 통한 안정적인 상태 유지 가능 여부 검증 불가 사유 : Mobius 플랫폼은 자체적으로 상호 연동을 통하여 안정적인 상태를 도출해 낼 수 없음	X
SF-Depend-Req-016 CPS-Depend-Req-005	스마트공장에서 발생되는 메시지에 대한 보안 체계 제공 여부 검증 테스트케이스 : Semantic 테스트케이스	O
SF-Depend-Req-017	전체 CPS 혹은 서브시스템의 상태의 M&S를 통한 예측 검증 불가 사유 : Mobius 플랫폼은 M&S 기능을 지원하지 않음	X
SF-Depend-Req-018	전체 CPS의 연동과 같은 구성 상태의 분산 연동 M&S 활용한 검증 검증 불가 사유 : Mobius 플랫폼은 M&S 기능을 지원하지 않음	X
SF-Depend-Req-019	이상 상황 시 분산 M&S를 통하여 다양한 상태 시뮬레이션 및 검증 검증 불가 사유 : Mobius 플랫폼은 M&S 기능을 지원하지 않음	X
SF-Depend-Req-020	분산 M&S로부터 도출된 최적의 제어와 연동 구조 설정과 같은 메시지의 실제 기기에의 적용 검증 불가 사유 : Mobius 플랫폼은 M&S 기능을 지원하지 않음	X
CPS-Depend-Req-004	안정적 작동을 위한 메모리 영역 보장 검증 불가 사유 : Mobius 플랫폼이 메모리 영역 보장을 지원하지 않음(자원 통합 관리)	X

5) 실시간성(Timing)

실시간성 관련 특성은 거의 지원하지 않는다.

<표 6-14> CPS 실시간성 요구사항 식별표

CPS 요구사항 ID	검증 상세	검증
SF-Timing-Req-001	고객이 요구하는 제품 공정의 순서 설계 및 그 시간 예측/측정 가능 여부 검증 불가 사유 : Mobius 플랫폼은 도메인에 최적화 된 시간 예측 기능을 지원하지 않음	X
SF-Timing-Req-002 CPS-Timing-Req-001	컴포넌트/모듈의 데이터를 이용한 이기종 노드 간 지능적 동기화 검증 불가 사유 : Mobius 플랫폼은 지능적인 기능을 제공하지 않음	X
SF-Timing-Req-003	모듈 간 걸리는 시간 측정/학습을 통해 새로운 모델 생성 시 모델 내 공정 시간 예측 검증 불가 사유 : Mobius 플랫폼은 도메인에 최적화 된 시간 예측 기능 및 모델링&시뮬레이션 기능을 지원하지 않음	X
SF-Timing-Req-004 CPS-Timing-Req-003	스마트공장 내 발생하는 신호들의 설정된 시간 내 도달 보장 및 초과 시 생성되는 에러 신호의 전송 시간 보장 기능 확인 검증 불가 사유 : Mobius 플랫폼은 전송 시간의 확인 기능을 제공하지 않음	X
SF-Timing-Req-005	다른 스마트공장와 통신을 위한 특정 이벤트의 국제 표준 준수 검증 불가 사유 : Mobius 플랫폼은 도메인에 최적화 된 국제 표준을 지원하지 않음	X
SF-Timing-Req-006	실시간으로 다른 노드 간 시간동기화 확인 검증 테스트케이스 : NetTime 테스트케이스	O
SF-Timing-Req-007	실시간으로 다른 노드 간 주파수동기화 확인 검증 불가 사유 : Mobius 플랫폼에서는 각 노드의 주파수 수준의 확인이 불가능함	X
SF-Timing-Req-008	실시간으로 다른 노드 간 Phase 동기화 확인 검증 불가 사유 : Mobius 플랫폼에서는 각 노드의 Phase 수준의 확인이 불가능함	X
SF-Timing-Req-009	새로운 모델 생성 시 시뮬레이션을 통한 측정 모델 성능과 실제 성능의 유사성 검증 불가 사유 : Mobius 플랫폼은 모델링&시뮬레이션 기능을 제공하지 않음	X
SF-Timing-Req-010	사용자에 모델을 통해 동작 과정을 실시간으로 알림 가능 여부 검증 불가 사유 : Mobius 플랫폼은 모델링&시뮬레이션 기능을 제공하지 않음	X
SF-Timing-Req-011	정확한 분산 M&S를 위한 시뮬레이션 발생 데이터들의 Time sync 기능 제공 검증 불가 사유 : Mobius 플랫폼은 모델링&시뮬레이션 기능을 제공하지 않음	X
SF-Timing-Req-012	정확한 분산 M&S 환경 제공을 위한 시뮬레이션 데이터 사이의 동기화 기능 검증 불가 사유 : Mobius 플랫폼은 모델링&시뮬레이션 기능을 제공하지 않음	X
CPS-Timing-Req-002	태스크들의 시간적 요구사항 만족 여부 검증 불가 사유 : Mobius 플랫폼은 전송 시간의 확인 기능을 제공하지 않음	X
CPS-Timing-Req-004	외부 시스템과의 실시간 상호작용 검증 불가 사유 : Mobius 플랫폼은 실시간 통신을 지원하지 않음	X

6) 상호운용성(Interoperability)

IoT의 주요 관심사는 기기들 간의 통신이므로 대부분의 상호운용성 요구사항을 지원한다.

<표 6-15> CPS 상호운용성 요구사항 식별표

CPS 요구사항 ID	검증 상세	검증
SF-Operate-Req-001 CPS-Operate-Req-001	다양한 산업용 통신기술을 이용하여 메시지 통신 여부 검증 테스트케이스 : Transport 테스트케이스	O
SF-Operate-Req-002	다양한 산업용 통신기술을 이용하여 제어메시지의 수신 및 동작의 가능 여부 검증 테스트케이스 : Transport 테스트케이스	O
SF-Operate-Req-003	센서를 이용하여 기기들의 예기지 않은 동작을 발견하고 지능적으로 대처하는 기능 제공 검증 불가 사유 : 데이터 수집을 통하여 지능적으로 대처할 수 없음	X
SF-Operate-Req-004	다양한 모듈들과 통신하기 위해서 네트워크 요구사항을 만족하였는지 확인 기능 제공 검증 불가 사유 : 네트워크 요구사항을 확인하기 위해서는 별도의 응용에서 확인하거나, SDN기술을 적용하여 확인해야 함	X
SF-Operate-Req-005	기기의 실제 위치를 고려하여 공정 과정을 설정하는 기능의 가능 여부 검증 테스트케이스 : Lifecycle 테스트케이스	O
SF-Operate-Req-006	발생하는 모든 결함을 공장 내에서 공유하여 적절한 대처할 할 수 있는 기능의 가능 여부 검증 테스트케이스 : S/W & H/W Management	O
SF-Operate-Req-007	센서를 이용하여 주변 환경을 측정하고 이에 따라 공정 상태를 제어하는 기능의 가능 여부 검증 테스트케이스 : Lifecycle 테스트케이스	O
SF-Operate-Req-008	제품의 주문량에 따라 유연하게 공정 속도를 제어하는 기능의 가능 여부 검증 테스트케이스 : Lifecycle 테스트케이스	O
SF-Operate-Req-009	요구사항에 맞는 공정의 설계가 되었는지 확인하는 기능 제공 검증 불가 사유 : Mobius 플랫폼을 통해서 DB 차원에서의 확인은 가능하나, 실질적으로 정상적 설계 여부는 확인할 수 없음	X
SF-Operate-Req-010	산업용 통신기술을 이용하여 기기들의 설정을 서버로 전송 검증 테스트케이스 : Transport 테스트케이스	O
SF-Operate-Req-011	기기들 간의 복잡한 연동 구조를 유연하게 재구성 검증 불가 사유 : Mobius 플랫폼을 통해서 연동 구조 변경 알림은 가능하나, 공정 자체의 재구성은 검증이 불가능 함	X
CPS-Operate-Req-002	소프트웨어의 이종 노드 적용을 위한 표준을 준수한 이식성 여부 검증 테스트케이스 : S/W & H/W Management	O
CPS-Operate-Req-003	이종 노드간의 데이터 공유 검증 테스트케이스 : Transport 테스트케이스	O

7) 지능성(Intelligence)

oneM2M 표준은 지능성 요구사항을 지원하지 않는다.

<표 6-16> CPS 지능성 요구사항 식별표

CPS 요구사항 ID	검증 상세	검증
SF-Int-Req-001 CPS-Int-Req-001	구성 기기들의 공정 환경 및 상황 인지, 그에 따른 동작 상태 변동 기능 제공 여부 검증 불가 사유 : Mobius 플랫폼만으로는 지능적인 인지 및 변동성을 지원하지 않음	X
SF-Int-Req-002 CPS-Int-Req-001 CPS-Int-Req-002	공장 내부의 현재 상황 인지 및 최적의 상태를 조성 가능 여부 검증 불가 사유 : Mobius 플랫폼만으로는 지능적인 인지 및 변동성을 지원하지 않음	X
SF-Int-Req-003 CPS-Int-Req-002	공정 또는 기기의 결함을 지능적으로 예측 검증 불가 사유 : Mobius 플랫폼만으로는 지능적인 인지 및 변동성을 지원하지 않음	X
SF-Int-Req-004 CPS-Int-Req-003	빅데이터 처리 기술에 기반으로 한 결함 상황 회복 가능 여부 검증 불가 사유 : Mobius 플랫폼은 상황 정보를 인지 및 그에 대한 지능적인 대처 기능을 제공하지 않음	X
SF-Int-Req-005 CPS-Int-Req-003	빅데이터 처리 기술을 이용하여 네트워크 설정을 지능적으로 설정 확인 검증 불가 사유 : Mobius 플랫폼은 상황 정보를 인지 및 그에 대한 지능적인 대처 기능을 제공하지 않음	X
SF-Int-Req-006 CPS-Int-Req-001	구성 기기들의 생산중인 제품 상태 인지 가능 여부 검증 불가 사유 : Mobius 플랫폼은 상황 정보를 인지 및 그에 대한 지능적인 대처 기능을 제공하지 않음	X
SF-Int-Req-007 CPS-Int-Req-001	자신 혹은 관련 기기의 정보 인지 검증불가 사유 : Mobius 플랫폼은 데이터 공유에 초점이 맞추어져 있어 서브시스템이 스스로 어떠한 데이터를 주고받는지 인지할 수 있는 지능적 요소는 고려되지 않음	X
SF-Int-Req-008 CPS-Int-Req-003	데이터 전송 지연에 대한 유연한 대처 검증 불가 사유 : Mobius 플랫폼은 데이터 전송 지연 상황에 대한 인지와 그에 대한 대처에 대한 부분은 전혀 고려되어 있지 않음	X

제3절 대규모 CPS 지원을 위한 IoT 플랫폼 검증

1. 검증 테스트베드 설계 및 구현

1) 검증 테스트베드 사용자 요구사항

5장에서 정의된 CPS 요구사항을 검증할 수 있는 테스트베드의 사용자 요구사항을 도출해 냈다. CPS의 요구사항 지원 여부에 대한 검증을 진행하기 위해서 필요한 네트워크 시뮬레이션, 서버나 응용이 작동될 호스트, 프로그래밍 환경 등의 사용자에게 반드시 제공되어야 하는 기능적 요소들을 테스트베드 요구사항으로 도출하였다. 총 34개의 테스트베드 사용자 요구사항이 도출되었다. 테스트베드 사용자 요구사항은 <표 6-17>에서 확인할 수 있다.

<표 6-17> 검증 테스트베드 사용자 요구사항

요구사항 ID	설명
TB-UR-001	검증 테스트베드는 대규모 CPS 환경 검증을 위한 네트워크 시뮬레이션 기능을 제공하여야 함
TB-UR-002	네트워크 시뮬레이션에 사용될 가상 네트워크의 토폴로지 및 설정 상태를 확인할 수 있는 GUI 환경을 제공하여야 함
TB-UR-003	네트워크 시뮬레이션에 사용될 가상 네트워크의 설정을 할 수 있도록 CLI를 통한 시뮬레이션 세팅 환경을 제공하여야 함
TB-UR-004	네트워크 시뮬레이션의 설정을 용이하게 할 수 있도록 GUI 시뮬레이션 세팅 환경을 제공하여야 함
TB-UR-005	네트워크 시뮬레이션 구성을 위해 사전에 구성된 가상 네트워크를 활용할 수 있어야 함
TB-UR-006	네트워크 시뮬레이션에서 사용될 수 있는 사전에 정의된 가상 네트워크를 임의로 수정 및 변경 할 수 있어야 함
TB-UR-007	네트워크 시뮬레이션에서 발생하는 가상 네트워크 오류를 검출할 수 있어야 함
TB-UR-008	네트워크 시뮬레이션을 위하여 가상 네트워크를 사용자 임의대로 구성할 수 있어야 함
TB-UR-009	네트워크 시뮬레이션을 위하여 임의로 가상 네트워크의 상태 시나리오를 설정할 수 있어야 함
TB-UR-010	설정된 가상 네트워크의 저장 및 관리 기능이 제공되어야 함
TB-UR-011	네트워크 시뮬레이션에서 발생하는 네트워크 트래픽을 확인할 수 있어야 함
TB-UR-012	네트워크 연결 정책을 유연하게 설정할 수 있어야 함
TB-UR-013	네트워크를 구성하는 호스트의 QoS를 설정할 수 있어야 함

TB-UR-013	가상 네트워크에 연결되어 있는 가상 호스트에서 응용 프로그램을 작성할 수 있어야 함
TB-UR-014	가상 네트워크에 연결되어 있는 가상 호스트에서 응용 프로그램을 수정할 수 있어야 함
TB-UR-015	가상 네트워크에 연결되어 있는 가상 호스트는 응용 프로그램을 실행할 수 있어야 함
TB-UR-016	가상 네트워크에 연결되어 있는 가상 호스트는 동시에 여러 응용을 실행할 수 있어야 함
TB-UR-017	가상 호스트에서 실행중인 응용에 오류가 발생할 경우 검출할 수 있어야 함
TB-UR-018	가상 호스트에서 실행중인 응용은 가상 네트워크를 통하여 다른 응용과 통신할 수 있어야 함
TB-UR-019	엣지 디바이스의 검증을 위하여 실제 디바이스를 가상 네트워크에 연결시킬 수 있어야 함
TB-UR-020	엣지 디바이스에 연결된 실제 센서로 부터 수집된 데이터를 가상 네트워크를 통해 다른 호스트로 전달할 수 있어야 함
TB-UR-021	엣지 디바이스와 가상 네트워크 간의 연결을 제거할 수 있어야 함
TB-UR-022	엣지 디바이스 연결성을 검증하기 위하여 가상의 엣지 디바이스를 생성 및 설정할 수 있어야 함
TB-UR-023	가상의 엣지 디바이스에 가상 센서를 생성 및 연결할 수 있어야 함
TB-UR-024	가상의 엣지 디바이스가 네트워크를 통해 데이터를 주고받을 수 있도록 연결성이 제공되어야 함
TB-UR-025	가상의 엣지 디바이스에 연결된 가상 센서로 부터 수집된 데이터를 다른 호스트로 전달할 수 있어야 함
TB-UR-026	가상의 엣지 디바이스에 연결성을 제공하기 위한 IoT 플랫폼 환경이 제공되어야 함
TB-UR-027	IoT 플랫폼 환경에 대응하는 엣지 디바이스 또는 응용과의 연결성이 만족되는 서버 환경이 제공되어야 함
TB-UR-028	서버의 포팅 및 작동 될 수 있는 환경을 사용자 임의로 구축할 수 있어야 함
TB-UR-029	서버를 통하여 각각의 응용들이 데이터를 올리고 내려 받을 수 있는 환경이 제공되어야 함
TB-UR-030	하나의 서버에 동시에 다수의 응용들이 연결될 수 있어야 함
TB-UR-031	서버는 각각의 응용들이 원활히 작동될 수 있도록 실시간 연결성을 제공해야 함
TB-UR-032	각각 CPS 응용들의 원활한 작동을 위하여 여러 개의 응용 서버를 동시에 작동할 수 있는 환경이 제공되어야 함
TB-UR-033	서버는 각각의 응용들의 동작 상태를 모니터링 할 수 있는 기능을 제공하여야 함
TB-UR-034	서버는 각각의 응용들의 동작 상태를 로깅 할 수 있는 기능을 제공하여야 함

2) 검증 테스트베드 시스템 요구사항

<표 6-17>에서 정의된 테스트베드 사용자 요구사항을 만족시키기 위한 시스템 요구사항을 <표 6-18>에 정의하였다. 검증 테스트베드 시스템 요구사항을 통하여 시스템 요구사항과 그 기능, 그리고 그 기능을 필요로 하는 연관된 사용자 요구사항 ID를 확인할 수 있다.

<표 6-18> 검증 테스트베드 시스템 요구사항

요구사항 ID	설명	연관 사용자 요구사항 ID
TB-SR-001	네트워크 시뮬레이션을 위하여 mininet simulator를 실행할 수 있어야 함	TB-UR-001
TB-SR-002	CLI 환경을 통하여 mininet 시뮬레이터에서 사용될 가상 네트워크 생성 기능을 제공해야 함	TB-UR-003 TB-UR-008
TB-SR-003	CLI 환경을 통하여 가상 네트워크에 가상 호스트를 추가할 수 있어야 함	TB-UR-003 TB-UR-008
TB-SR-004	CLI 환경을 통하여 가상 네트워크로부터 가상 호스트를 제거할 수 있어야 함	TB-UR-003 TB-UR-008
TB-SR-005	CLI 환경을 통하여 가상 호스트의 설정을 변경할 수 있어야 함	TB-UR-003 TB-UR-008
TB-SR-006	CLI 환경을 통하여 가상 네트워크에 가상 네트워크 설비를 추가할 수 있어야 함	TB-UR-003 TB-UR-008
TB-SR-007	CLI 환경을 통하여 가상 네트워크로부터 가상 네트워크 설비를 제거 할 수 있어야 함	TB-UR-003 TB-UR-008
TB-SR-008	CLI 환경을 통하여 가상 네트워크 설비의 설정을 변경할 수 있어야 함	TB-UR-003 TB-UR-008
TB-SR-009	CLI 환경을 통하여 가상 네트워크를 구성하는 객체간의 연결을 추가할 수 있어야 함	TB-UR-003 TB-UR-008
TB-SR-010	CLI 환경을 통하여 가상 네트워크를 구성하는 객체간의 연결을 제거 할 수 있어야 함	TB-UR-003 TB-UR-008
TB-SR-011	GUI 환경을 통하여 가상 네트워크의 토폴로지 및 설정 상태를 시각적으로 확인할 수 있어야 함	TB-UR-002 TB-UR-008
TB-SR-012	GUI 환경을 통하여 가상 호스트의 연결 및 설정 상태를 시각적으로 확인할 수 있어야 함	TB-UR-002 TB-UR-008
TB-SR-013	GUI 환경을 통하여 가상 네트워크 설비의 연결 및 설정 상태를 시각적으로 확인할 수 있어야 함	TB-UR-002 TB-UR-008
TB-SR-014	GUI 환경을 통하여 mininet 시뮬레이터에서 사용될 가상 네트워크 생성 기능을 제공해야 함	TB-UR-004 TB-UR-008
TB-SR-015	GUI 환경을 통하여 가상 네트워크에 가상 호스트를 추가할 수 있어야 함	TB-UR-004 TB-UR-008
TB-SR-016	GUI 환경을 통하여 가상 네트워크로부터 가상 호스트를 제거할 수 있어야 함	TB-UR-004 TB-UR-008

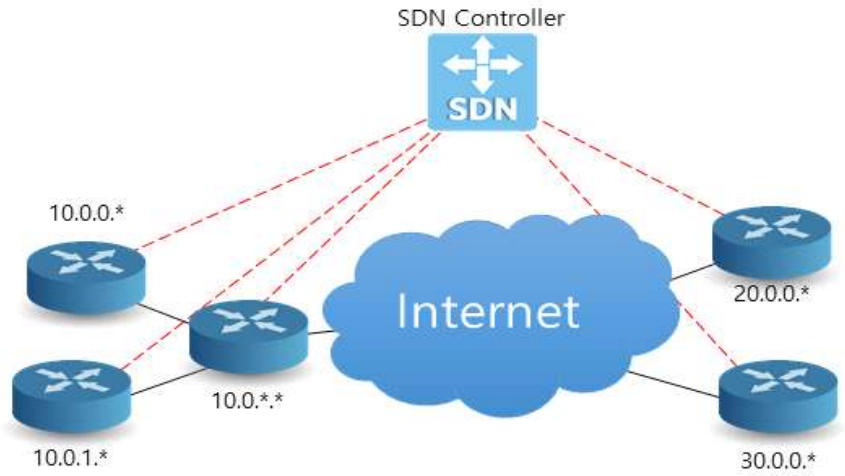
TB-SR-017	GUI 환경을 통하여 가상 호스트의 설정을 변경할 수 있어야 함	TB-UR-004 TB-UR-008
TB-SR-018	GUI 환경을 통하여 가상 네트워크에 가상 네트워크 설비를 추가할 수 있어야 함	TB-UR-004 TB-UR-008
TB-SR-019	GUI 환경을 통하여 가상 네트워크로부터 가상 네트워크 설비를 제거 할 수 있어야 함	TB-UR-004 TB-UR-008
TB-SR-020	GUI 환경을 통하여 가상 네트워크 설비의 설정을 변경할 수 있어야 함	TB-UR-004 TB-UR-008
TB-SR-021	GUI 환경을 통하여 가상 네트워크를 구성하는 객체간의 연결을 추가할 수 있어야 함	TB-UR-004 TB-UR-008
TB-SR-022	GUI 환경을 통하여 가상 네트워크를 구성하는 객체간의 연결을 제거 할 수 있어야 함	TB-UR-004 TB-UR-008
TB-SR-023	구성한 가상 네트워크를 python 파일 포맷(*.py)으로 저장할 수 있어야 함	TB-UR-005 TB-UR-010
TB-SR-024	python 파일 포맷(*.py)로 저장된 가상 네트워크를 불러올 수 있어야 함	TB-UR-005 TB-UR-010
TB-SR-025	python 파일 포맷(*.py)로 저장된 가상 네트워크를 수정할 수 있어야 함	TB-UR-005 TB-UR-006
TB-SR-026	네트워크 시뮬레이션 설정 과정에서 오류를 검출 할 수 있어야 함	TB-UR-007
TB-SR-027	네트워크 시뮬레이션 설정 과정에서 검출된 오류의 알림 기능을 제공해야 함	TB-UR-007
TB-SR-028	네트워크 시뮬레이션 실행 중 가상 네트워크에서 발생한 오류를 검출할 수 있어야 함	TB-UR-007
TB-SR-029	네트워크 시뮬레이션 실행 중 검출된 가상 네트워크의 오류를 알려줄 수 있어야 함	TB-UR-007
TB-SR-030	Iperf 등의 트래픽 생성기를 통하여 혼잡한 네트워크 시나리오를 구성할 수 있어야 함	TB-UR-009
TB-SR-031	트래픽 추적을 위하여 가상 네트워크 내에서 Wireshark를 구동할 수 있어야 함	TB-UR-011
TB-SR-032	SDN 기술을 활용할 수 있는 유연한 가상 네트워크를 구성하기 위하여 Openflow를 지원하는 스위치 장비를 설정할 수 있어야 함	TB-UR-012
TB-SR-033	SDN Controller를 활용하여 호스트 간 연결에서 유연하게 SDN의 연결 설정을 변경할 수 있어야 함	TB-UR-012
TB-SR-034	호스트에서 작동하는 응용에서 필요로 하는 QoS 수준을 설정할 수 있어야 함	TB-UR-013
TB-SR-035	SDN controller를 활용하여 호스트에서 작동하는 응용에서 설정된 QoS 수준을 보장할 수 있어야 함	TB-UR-012 TB-UR-013
TB-SR-036	가상 호스트에서 xterm을 통하여 C 프로그램을 작성 및 수정을 할 수 있어야 함	TB-UR-013 TB-UR-014
TB-SR-037	가상 호스트에서 xterm을 통하여 C++ 프로그램을 작성 및 수정을 할 수 있어야 함	TB-UR-013 TB-UR-014
TB-SR-038	가상 호스트에서 xterm을 통하여 javascript 프로그램을 작성 및 수정을 할 수 있어야 함	TB-UR-013 TB-UR-014
TB-SR-040	가상 호스트는 작성된 C 프로그램의 컴파일을 위하여 gcc compiler를 지원해야 함	TB-UR-013 TB-UR-015
TB-SR-041	가상 호스트는 작성된 C++ 프로그램의 컴파일을 위하여 g++ compiler를 지원해야 함	TB-UR-013 TB-UR-015

TB-SR-042	가상 호스트는 컴파일 된 응용 프로그램을 실행할 수 있는 환경을 제공해야 한다.	TB-UR-013 TB-UR-015
TB-SR-043	가상 호스트는 javascript 프로그램을 실행할 수 있도록 Node.js 환경을 제공해야 함	TB-UR-015
TB-SR-044	가상 호스트는 xterm을 통하여 동시에 여러 응용을 실행할 수 있는 환경을 제공해야 함	TB-UR-015 TB-UR-016
TB-SR-045	가상 호스트에서 발생하는 오류를 xterm의 CLI를 통하여 확인할 수 있어야 함	TB-UR-017
TB-SR-046	가상 호스트는 실행중인 응용이 다른 호스트의 응용과 통신할 수 있도록 네트워크 인터페이스를 제공해야 함	TB-UR-018
TB-SR-047	가상 호스트의 네트워크 인터페이스는 다른 호스트와의 TCP 통신을 지원해야 함	TB-UR-018
TB-SR-048	가상 호스트의 네트워크 인터페이스는 다른 호스트와의 HTTP 통신을 지원해야 함	TB-UR-018
TB-SR-049	가상 호스트의 네트워크 인터페이스는 다른 호스트와의 CoAP 통신을 지원해야 함	TB-UR-018
TB-SR-050	가상 호스트의 네트워크 인터페이스는 다른 호스트와의 MQTT 통신을 지원해야 함	TB-UR-018
TB-SR-051	실제 엣지 디바이스를 가상 네트워크와 연결할 수 있는 인터페이스가 제공되어야 함	TB-UR-019
TB-SR-052	실제 엣지 디바이스를 가상 네트워크로부터 분리할 수 있어야 함	TB-UR-021
TB-SR-053	실제 엣지 디바이스로부터 발생되는 데이터를 가상 네트워크를 통하여 다른 호스트로 전달할 수 있어야 함	TB-UR-020
TB-SR-054	실제 엣지 디바이스는 다른 가상 호스트로부터 데이터를 전달받을 수 있어야 함	TB-UR-020
TB-SR-055	가상 호스트에서 가상의 엣지 디바이스 환경을 구축할 수 있어야 함	TB-UR-022
TB-SR-056	가상의 엣지 디바이스에 가상 센서를 추가할 수 있어야 함	TB-UR-023
TB-SR-057	가상의 엣지 디바이스로부터 가상 센서를 제거할 수 있어야 함	TB-UR-023
TB-SR-058	가상의 엣지 디바이스는 가상 센서로부터 센싱 데이터를 전달받을 수 있어야 함	TB-UR-023
TB-SR-059	가상의 엣지 디바이스는 가상 네트워크를 통하여 다른 호스트로 데이터를 보낼 수 있어야 함	TB-UR-024 TB-UR-025
TB-SR-060	가상의 엣지 디바이스는 가상 네트워크를 통하여 다른 호스트로부터 데이터를 받을 수 있어야 함	TB-UR-024 TB-UR-025
TB-SR-061	엣지 디바이스는 oneM2M 표준을 준수하는 기반으로 하는 IoT 디바이스 플랫폼인 &Cube가 설치되어 있어야 함	TB-UR-026
TB-SR-062	엣지 디바이스의 IoT 디바이스 플랫폼 &Cube는 HTTP 통신을 지원해야 함	TB-UR-026
TB-SR-063	엣지 디바이스의 IoT 디바이스 플랫폼 &Cube는 CoAP 통신을 지원해야 함	TB-UR-026
TB-SR-064	엣지 디바이스의 IoT 디바이스 플랫폼 &Cube는 MQTT 프로토콜을 지원해야 함	TB-UR-026
TB-SR-065	엣지 디바이스는 IoT 디바이스 플랫폼 &Cube를 통하여 수집한 데이터를 Mobius 서버로 전송할 수 있어야 함	TB-UR-026
TB-SR-066	엣지 디바이스는 IoT 디바이스 플랫폼 &Cube를 통하여 IoT 서버로부터 데이터를 수신할 수 있어야 함	TB-UR-026
TB-SR-067	IoT 서버에는 IoT 서버 플랫폼인 Mobius 환경이 구축되어 있어야 함	TB-UR-027
TB-SR-068	Mobius를 통하여 구축된 IoT 서버는 HTTP 통신을 지원해야 함	TB-UR-027
TB-SR-069	Mobius를 통하여 구축된 IoT 서버는 CoAP 통신을 지원해야 함	TB-UR-027
TB-SR-070	Mobius를 통하여 구축된 IoT 서버는 MQTT 통신을 지원해야 함	TB-UR-027

TB-SR-071	Mobius를 통하여 구축된 IoT 서버는 Open API를 제공해야 함	TB-UR-027
TB-SR-072	응용 서버는 Node.js 환경이 구성되어 있어야 함	TB-UR-028
TB-SR-073	응용 서버는 응용의 요구사항에 따라 임의의 서버 환경을 구축할 수 있어야 함	TB-UR-028
TB-SR-074	응용 서버는 각 응용으로부터 접근할 수 있는 인터페이스를 제공해야 함	TB-UR-029
TB-SR-075	응용 서버는 연결되어있는 각 응용과 메시지를 주고 받을 수 있어야 함	TB-UR-029
TB-SR-076	Mobius 플랫폼은 다수의 IoT 디바이스 정보를 관리할 수 있어야 함	TB-UR-030
TB-SR-077	응용 서버는 동시에 여러 응용이 접근할 수 있는 인터페이스를 제공해야 함	TB-UR-030
TB-SR-078	응용 서버는 동시에 여러 응용이 접근할 경우 각 응용과의 연결을 관리할 수 있어야 함	TB-UR-030
TB-SR-079	응용 서버는 응용으로부터 전달받은 요청을 실시간으로 처리할 수 있어야 함	TB-UR-031
TB-SR-080	응용 서버는 응용으로 요청에 따른 응답 메시지를 실시간으로 보낼 수 있어야 함	TB-UR-031
TB-SR-081	서버 환경은 Mobius 서버와 다수의 응용 서버를 동시에 작동시킬 수 있어야 함	TB-UR-032
TB-SR-082	서버는 CLI를 통하여 연결되어 있는 응용들의 동작 상태를 확인할 수 있어야 함	TB-UR-033
TB-SR-083	서버는 파일 형태로 통하여 연결되어 있는 응용들의 동작 상태를 기록할 수 있어야 함	TB-UR-033

3) 검증 테스트베드 구조 설계

[그림 6-1] 테스트베드 네트워크 구조



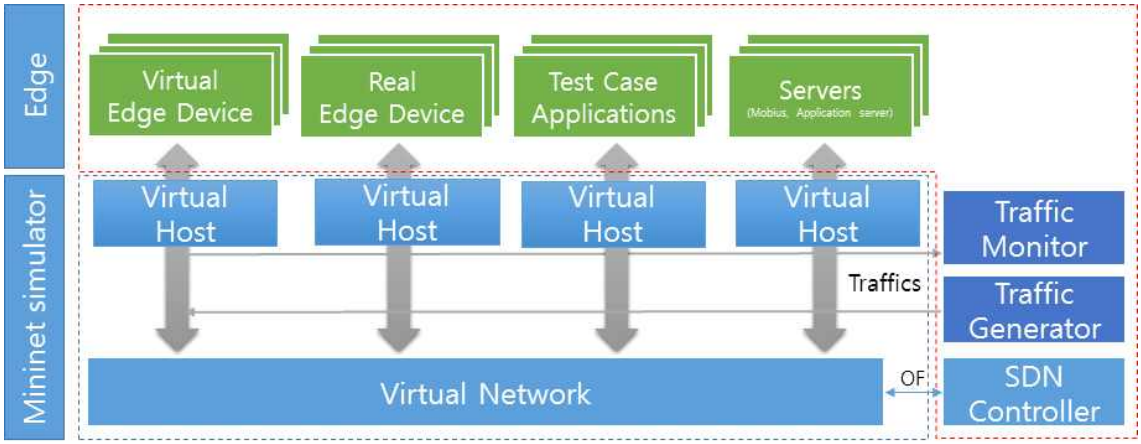
SDN : 소프트웨어 정의 네트워킹(Software defined networking)[150]

검증 테스트베드의 가상 네트워크는 LAN 환경과 WAN 환경을 모두 제공하여 다양한 시나리오에서의 검증을 돕는다. 각각의 스위치에 테스트 시나리오 별로 가상 호스트를 연결하여 테스트를 진행할 수 있다.

150) 개방형 API(Application Programming Interface, 응용 프로그래밍 인터페이스)를 통해 네트워크의 트래픽 전달 동작을 소프트웨어 기반 컨트롤러에서 제어/관리하는 접근방식

4) 검증 테스트베드 시스템 배치

[그림 6-2] 검증 테스트베드 시스템 배치도



검증 테스트베드를 구성하는 엣지 디바이스들과 응용, 서버는 [그림 6-2]에서 나타나듯이 시뮬레이터의 가상 호스트를 통하여 가상 네트워크와 연결된다. 네트워크 시뮬레이션의 시나리오 설정 및 네트워크 제어를 돕기 위하여 트래픽 모니터와 트래픽 발생기 그리고 SDN 컨트롤러를 활용한다. 테스트베드를 구성하는 각 항목에 대한 상세 항목은 아래 표에서 확인할 수 있다.

<표 6-19> 테스트베드 구성요소 상세

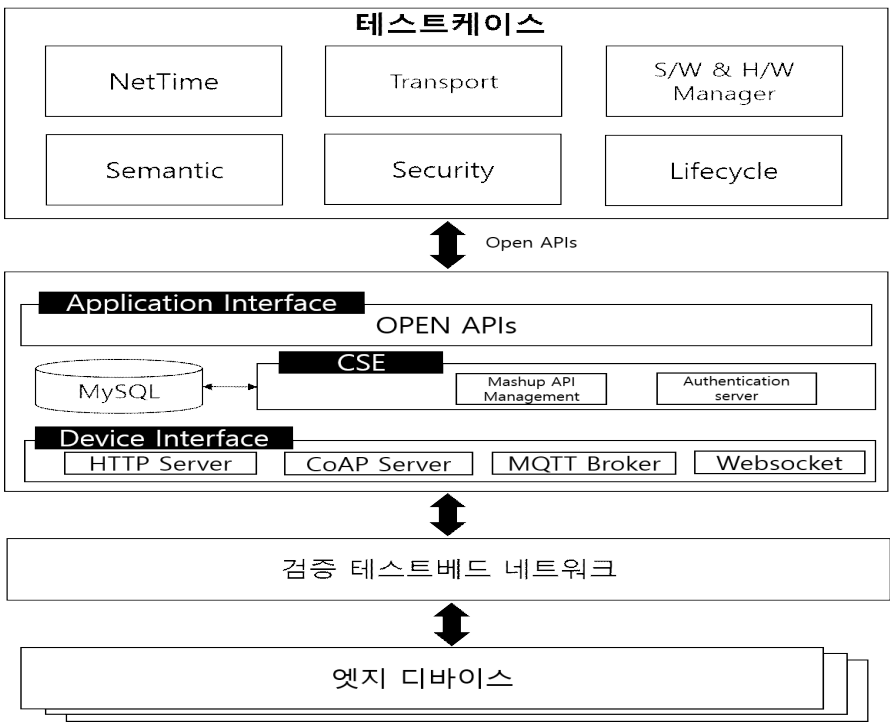
테스트베드 구성요소	구성 요소 기능	구성 방법
가상 네트워크	LAN 환경과 WAN 환경을 모두 테스트 할 수 있도록 가상 네트워크 토폴로지를 제공	Mininet 시뮬레이터를 통하여 SDN 기반의 LAN/WAN 복합 네트워크 토폴로지 구성
가상 호스트	가상 네트워크에 연결되어 있는 가상의 호스트들로 엣지 영역의 시스템들이 실행될 수 있는 환경과 가상 네트워크와의 인터페이스를 제공	가상 호스트의 CLI로 Xterm을 사용
가상 엣지 디바이스	가상 센서를 포함할 수 잇는 가상 엣지 디바이스로서 Mininet 시뮬레이터를 통하여 데이터를 응용과 서버 혹은 다른 디바이스로 전송	엣지 디바이스와 IoT 플랫폼 서버(Mobius)의 연결을 위하여 Mobius 플랫폼의 &Cube 포팅
실제 엣지 디바이스	실 센서를 포함할 수 있는 실제 엣지 디바이스로 Mininet 시뮬레이터에 연결되어 있는 가상 호스트와 연결되어 데이터를 응용과 서버 혹은 다른 엣지 디바이스로 전송	엣지 디바이스와 IoT 플랫폼 서버(Mobius)의 연결을 위하여 Mobius 플랫폼의 &Cube 포팅
테스트케이스	대규모 CPS의 여러 시나리오들을 구성하기 위한	응용의 기능에 따라 다양한 방법

응용	응용들, 응용 간 통신, 서버와의 통신, 엣지 디바이스의 통신 등 응용 목적에 따라 다양하게 구성	으로 구성될 수 있음
IoT 플랫폼 서버	엣지 디바이스나 응용과의 연결을 위한 IoT 플랫폼 서버 기능을 제공	IoT 플랫폼 서버는 Mobius 플랫폼의 Mobius Server를 포팅하여 구성
응용 서버	엣지 디바이스나 응용에의 서비스 제공을 위한 서버 기능을 제공	응용 서버의 기능과 목적에 따라 다양 방법으로 구성될 수 있음
트래픽 모니터링	네트워크 시뮬레이션 과정에서 발생하는 트래픽을 분석	발생하는 트래픽의 시각적 분석을 위하여 Wireshark를 사용
트래픽 발생기	네트워크 시뮬레이션에 사용될 다양한 네트워크 시나리오를 구성하기 위하여 네트워크 연결 상태를 인위적으로 변경하는데 사용	다양한 패턴의 트래픽을 쉽게 발생시킬 수 있도록 Iperf를 사용
SDN 컨트롤러	Mininet 시뮬레이터를 통하여 구성된 가상 네트워크를 제어하는데 사용	SDN 컨트롤러를 구성하기 위하여 Ryu 컨트롤러를 활용

2. 검증 테스트케이스 설계

1) 검증 응용 프로그램 구조 설계

[그림 6-3] 테스트 케이스 응용 구조



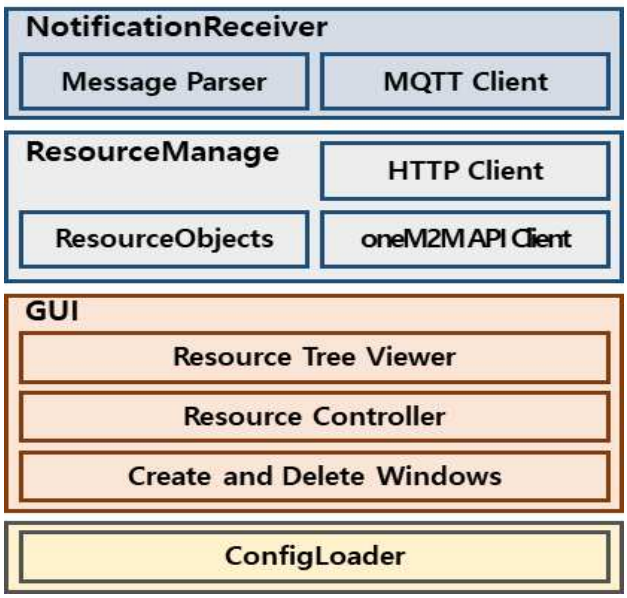
대규모 CPS 플랫폼을 지원하는 IoT 플랫폼인 Mobius 플랫폼을 검증하기 위해 7가지 어플리케이션들(NetTime, Software Manager, Hardware Manager, Transport, Semantic, Security, Lifecycle)을 구성한다. Mobius 플랫폼은 REST API[151]를 통해 resource를 관리하므로, 전체적인 검증은 oneM2M API testing 도구를 기반으로 한다.

2) 검증 테스트케이스 설계 및 구현

(1) oneM2M Browser

oneM2M Browser[152]는 Mobius 플랫폼에 있는 oneM2M resource를 모니터링 할 수 있는 도구로써 HTTP RESTful API와 MQTT 메시지를 기반으로 동작한다. oneM2M Browser는 Mobius 플랫폼과 MQTT(Message Queuing Telemetry Transport) Broker와 연결되어있다. Mobius 플랫폼은 내부의 구성요소(Resource)를 관리하는 공개 API를 제공하고 oneM2M Browser는 공개 API에 있는 Discovery API를 이용하여 구성요소를 트리형식으로 표현한다. oneM2M Browser가 제공하는 기능으로는 oneM2M 구성요소를 트리형식으로 표현, 구성요소의 정보를 탐색, 새로운 구성요소의 생성과 실시간으로 구성요소를 관리하는 것이다.

[그림 6-4] oneM2M Browser



151) REST(Representational State Transfer)는 월드 와이드 웹과 같은 분산 하이퍼미디어 시스템을 위한 소프트웨어 아키텍처의 한 형식.

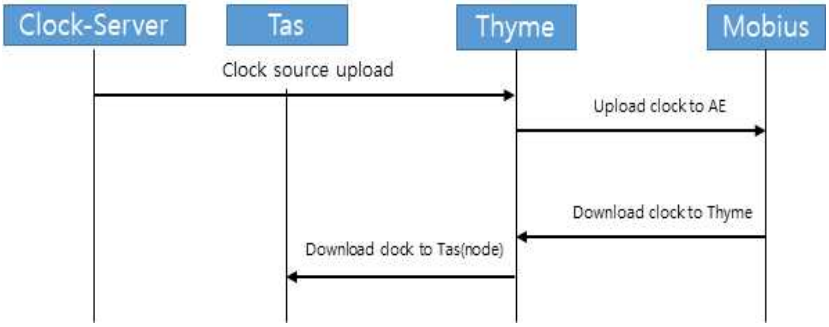
152) Chen Nan (2017), 『oneM2M Browser User Guide_v1.2_EN』, KETI.

(2) NetTime 응용프로그램

Mobius 플랫폼이 네트워크를 통해 연결된 플랫폼 내 구성노드 간의 시간동기화 지원과 이를 바탕으로 정확한 시간에 메시지를 전달할 수 있는 지 검증하기 위한 응용 프로그램이다.

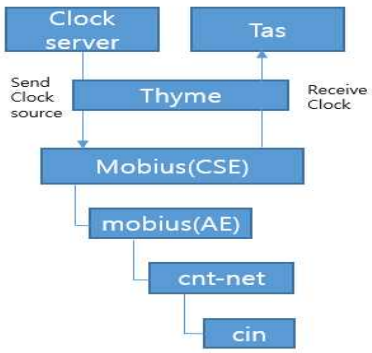
Clock 서버는 Mobius 플랫폼의 동기화될 시간을 제공해주는 서버이다.

[그림 6-5] Clock Server Time Flow



자신의 시간을 Mobius 서버에서 구성한 시간 데이터를 담을 수 있는 cnt-net container에 저장한다. 이론적 배경의 IoT oneM2M 부분에서 설명했듯이 container는 IoT 시스템에서의 자원을 전달할 수 있는 데이터 공간이고 그러한 자원 데이터들은 contentInstance 타입에 맞춰 전달된다. Clock source와 동기화될 필요가 있는 노드들(Tas)은 cnt-net에 저장된 시간 데이터를 Mobius 서버로부터 받아 현재 자신의 시간과 차이 값을 계산한다. 이 계산 값을 바탕으로 노드들은 서버와 노드 간 동기화 여부를 확인할 수 있다.

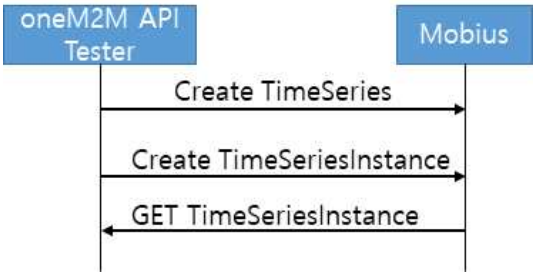
[그림 6-6] Clock Server Time Container



Mobius 플랫폼 내 구성된 노드들은 외부의 Clock source 서버와의 시간동기화 여부는 확

인할 수 있지만 Mobius 서버와 시간동기화 여부는 Mobius 서버의 설정 변경이 추가되어야만 가능하다. 반면 Mobius 서버의 현재 시간 data는 추가 설정 없이 얻을 수 있다. 우선, POSTMAN[153]을 이용하여 oneM2M API Tester의 TimeSeries API를 통해 TimeSeries container와 TimeSeriesInstance를 생성한다. 생성된 Container와 Instance를 통해 Mobius 서버의 시간 data를 Container와 Instance에 저장한다. Mobius 서버의 시간 data가 필요한 노드들은 서버에 Request를 통해 시간 정보를 가져올 수 있다.

[그림 6-7] Mobius Server Time Flow

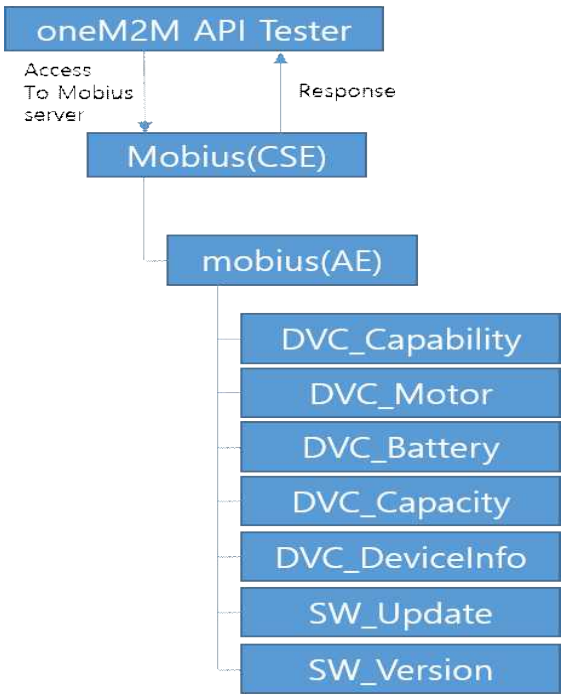


(3) S/W & H/W Manager 응용 프로그램

S/W & H/W Manager는 하드웨어와 소프트웨어가 플랫폼에 적용되거나 제거됐는지 확인하고 이러한 과정이 다른 모듈에 영향을 주는지 검증하는 응용 프로그램이다. 사용자는 Mobius 서버를 통해 하드웨어가 추가 또는 제거되거나 소프트웨어가 업데이트 되는 등의 상태를 확인할 수 있다.

153) Web, 응용 등의 API의 지원 여부를 테스트할 수 있는 환경을 제공하는 프로그램

[그림 6-8] H/W & S/W Manager Container



하드웨어의 경우 oneM2M API Tester를 통해 하드웨어의 상태 정보를 저장할 수 있는 Container인 DVC_Capability, DVC_Motor, DVC_Battery, DVC_Capacity, DVC_DeviceInfo를 생성한다. 생성된 Container를 통해 모터, 배터리, 캐패시터[154] 등의 하드웨어 상태 정보를 확인하고 설정할 수 있다.

Mobius 플랫폼의 구성된 하드웨어에서 구동되는 소프트웨어도 Mobius 서버의 연관된 container를 통해 상태 확인 및 설정이 가능하다. 이를 검증하기 위해 소프트웨어 데이터를 저장할 수 있는 SW_Update, SW_Version Container를 생성하였고 oneM2M API TESTER의 Discovery 및 ContainerInstance를 통해 소프트웨어의 상태 및 관리가 가능한지 확인 할 수 있다.

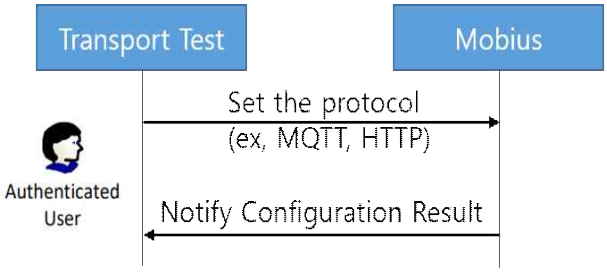
(4) Transport 응용 프로그램

Mobius 플랫폼이 적합한 통신 기술을 지원하고, 유연하고 확장성 있는 통신이 가능한지를 검증하기 위한 응용 프로그램이다. 인증된 유저(Authenticated User)는 말단 디바이스에서 Transport 응용프로그램을 이용하여 Mobius 서버와 어떤 프로토콜을 사용하여 연결할지를 설정할 수 있다. Mobius 서버는 인증된 유저가 설정한 데이터를 바탕으로 디바이스 및 다른

154) 축전기 혹은 콘덴서라고도 불리며 전기 회로에서 전기적 에너지를 저장하는 장치

서버들과 연결을 구성한다.

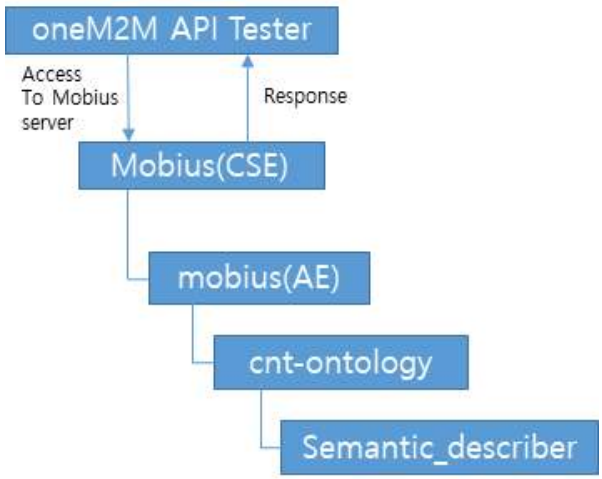
[그림 6-9] Transport Call flow



(5) Semantic 응용 프로그램

Mobius 플랫폼의 디바이스, 서버 등의 요소들의 데이터 및 제어 메시지들은 동일한 의미로서 해석될 수 있어야 한다. 이를 위해 Mobius는 지식 개념을 의미적으로 연결할 수 있는 시맨틱 웹 구현 도구인 온톨로지를 간단하게 구성할 수 있다. oneM2M API Tester의 Semantic Describer을 통해 Mobius 서버를 통해 주고받는 data container에 의미 서술(Semantic Description)을 생성한다. 생성된 서술에 추가적으로 온톨로지 정보를 설정함으로서 기기들 간의 동일한 의미 정보를 주고받을 수 있는지 검증할 수 있다.

[그림 6-10] Semantic Descriptor Container



(6) Security 응용 프로그램

Security는 oneM2M API Tester를 통해 Mobius의 허용된 사용자만 접근할 수 있는지 검증할 수 있는 응용 프로그램이다. Mobius는 허용된 사용자에게 ACP(Access Control

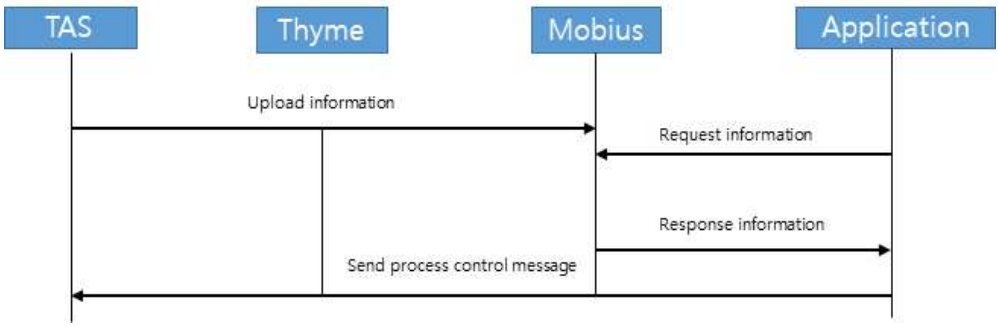
Policy)를 제공한다. 사용자는 제공된 ACP를 이용하여 Mobius 서버에 접근할 수 있는데 사용자의 ACP와 Mobius의 ACP가 같다면 접근 승인(ACCESS APPROVAL)을 사용자에게 전송해주면서 접근을 허용하고 다르다면 접근 불가(ACCEXX_DENIED)를 사용자에게 전송해주면서 접근을 막는다.

Mobius 서버는 인증된 유저인지를 판단하기 위하여 서버 내 admin_acp라는 인증 데이터 container를 가지고 있다. admin_acp는 랜덤으로 생성되며 인증받은 사용자는 이 admin_acp data를 바탕으로 Mobius 서버에 접근하여 Mobius 플랫폼의 resource들을 사용할 수 있다.

(7) Lifecycle 응용 프로그램

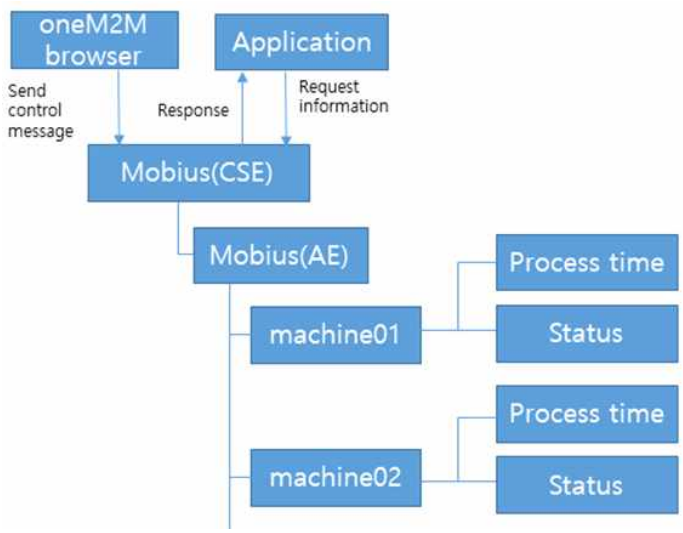
모듈이 실시간으로 정보를 서버로 전송하고 어플리케이션에서는 그 정보 데이터를 요청한다. 어플리케이션의 요청을 받은 서버는 어플리케이션으로 요청에 대한 정보를 전송해준다. 어플리케이션에서 공정속도의 설정 값을 모듈에 송신함으로써 공정 과정을 설정할 수 있다.

[그림 6-11] Lifecycle flow



Lifecycle은 Mobius 플랫폼이 제품의 정보 및 공정 정보를 사용자에게 표시해 주고 공정과정을 사용자가 설정할 수 있는지 검증해주는 어플리케이션이다. 각각의 모듈은 공정 속도의 값을 가지는 Process time와 공정 상태를 나타내주는 Status를 가지고 있다.

[그림 6-12] Lifecycle container



3) 검증 테스트케이스 상세 항목

앞에서 식별된 검증 대상 요구사항의 만족 여부를 검증하기 위한 검증 테스트케이스를 7가지로 구분하였다. 각 구분별로 상세 테스트케이스 항목을 정의하여 총 12종의 테스트케이스를 정의하였다. 하나의 상세 테스트케이스 항목을 통하여 유사성이 있거나 하나의 시나리오 내에서 검증될 수 있는 다양한 요구사항을 동시에 검증한다. 각 테스트케이스 별 상세 테스트케이스 항목 ID와 그와 관련된 요구사항들은 다음과 같다.

<표 6-20> 검증 테스트케이스 상세 항목

응용프로그램	테스트케이스 항목	관련요구사항
Transport	TC-TRP-01	SF-Scal-Req-003 CPS-Scal-Req-003 CPS-Inter-Req-006 SF-Operate-Req-001 CPS-Operate-Req-001 SF-Operate-Req-002 SF-Operate-Req-010
	TC-TRP-02	SF-Scal-Req-004 SF-Scal-Req-005 SF-Scal-Req-006 CPS-Scal-Req-003 CPS-Inter-Req-006 SF-Operate-Req-001 CPS-Operate-Req-001 SF-Operate-Req-002 SF-Operate-Req-010

	TC-TRP-03	SF-Inter-Req-001 SF-Inter-Req-002 CPS-Inter-Req-001 CPS-Inter-Req-008 CPS-Inter-Req-002
	TC-TRP-04	CPS-Inter-Req-003 CPS-Operate-Req-003
NetTime	TC-Net-01	SF-Timing-Req-006
S/W & H/W Manager	TC-MNG-01	SF-Com-Req-008 CPS-Com-Req-002 SF-Com-Req-013 CPS-Com-Req-001 SF-Com-Req-018 CPS-Com-Req-003 SF-Inter-Req-007
	TC-MNG-02	SF-Inter-Req-004 CPS-Inter-Req-008 SF-Operate-Req-006
	TC-MNG-03	CPS-Scal-Req-002 SF-Com-Req-001 SF-Com-Req-005 CPS-Operate-Req-002
Security	TC-SEC-01	SF-Depend-Req-001 CPS-Depend-Req-001 SF-Com-Req-002
semantic	TC-SEM-01	SF-Scal-Req-010 SF-Inter-Req-003 CPS-Inter-Req-005 CPS-Inter-Req-008 SF-Depend-Req-016 CPS-Depend-Req-005
Lifecycle	TC-LFC-01	SF-Scal-Req-001 SF-Inter-Req-013 SF-Inter-Req-016
	TC-LFC-02	SF-Inter-Req-015 SF-Inter-Req-017 SF-Depend-Req-005 SF-Operate-Req-005 SF-Operate-Req-007 SF-Operate-Req-008

3. 시험 검증

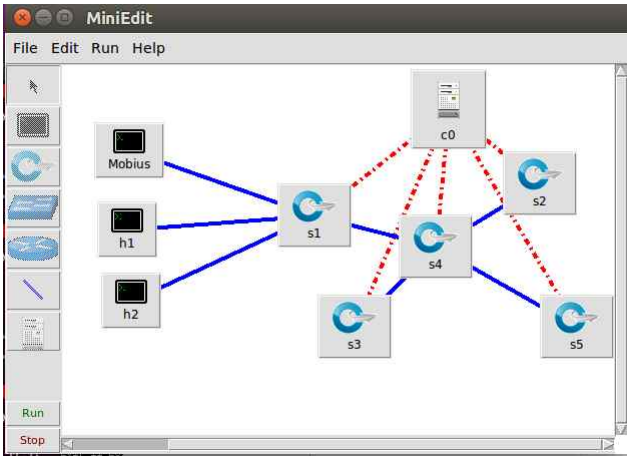
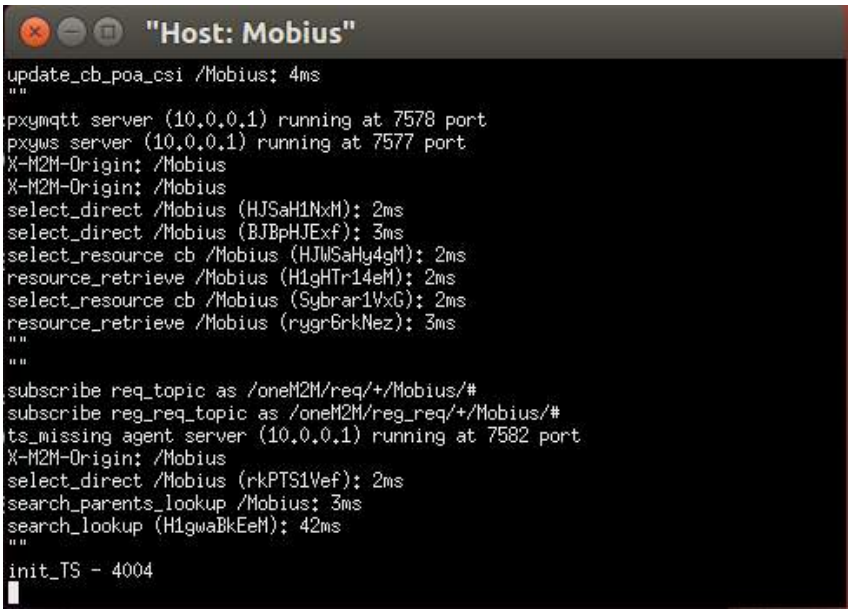
앞에서 설계 및 구현한 응용프로그램을 이용하여 각 상세 테스트케이스를 수행하여 검증 여부를 판단한다. 각 상세 테스트케이스 별로 시험 항목, 관련 요구사항, 시험 방법과 절차 판정 기준의 정의되고 실제로 진행된 검증 과정과 결과를 이미지화하여 표에 포함한다. 마지막으로 테스트케이스 수행 결과에 따라서 각 케이스별 판정을 내린다. 각 테스트케이스의 검증 대상이 되는 관련 요구사항 상세 설명은 스마트공장 요구사항을 중심으로 하여 설명하되 연관된 CPS 공통 요구사항을 같은 항목으로 명시한다. 시험결과 중 앞의 테스트케이스와 동일하거나 사소한 부분은 제외한다.

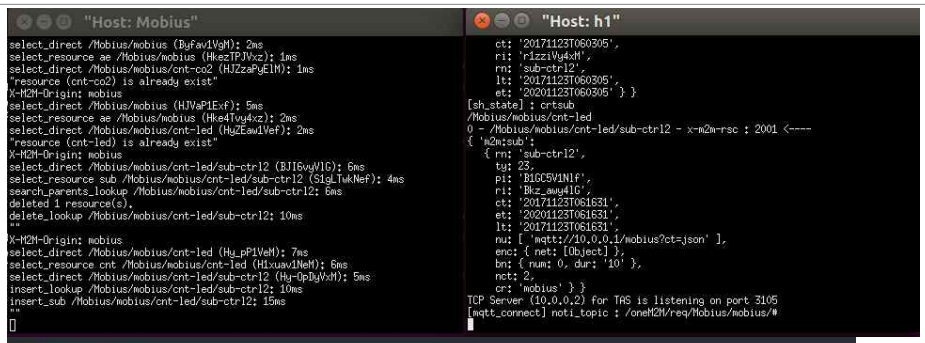
1) Transport 테스트케이스

이 테스트케이스는 IoT 플랫폼의 통신에 대한 CPS의 확장성, 상호작용성, 상호운영성 특성과 관련이 있다.

<표 6-21> Transport 테스트케이스 : TC-TRP-01

테스트케이스		
TC-TRP-01		
시험항목	Client/Server 통신 구조의 HTTP, CoAP 프로토콜 지원 여부 검증	
관련요구사항	SF-Scal-Req-003	클라이언트/서버 통신 기술 지원
	CPS-Scal-Req-003	
	CPS-Inter-Req-006	각 시스템에 최적화 된 통신 기술 지원
	SF-Operate-Req-001	다양한 산업용 통신기술을 이용하여 메시지 통신 여부
	CPS-Operate-Req-001	
	SF-Operate-Req-002	다양한 산업용 통신기술을 이용하여 제어메시지의 수신 및 동작의 가능 여부
	SF-Operate-Req-010	산업용 통신기술을 이용하여 기기들의 설정을 서버로 전송
시험 방법	Mininet simulator의 가상 네트워크를 구성하고, 하나의 가상 호스트에 Mobius 서버를 실행. 다른 호스트에 Thyme의 프로토콜을 변경하며 실행	
시험 절차	1. Mininet 시뮬레이터를 실행 2. 가상 호스트 Mobius에 Mobius 서버를 실행 3. Thyme의 conf.js를 수정하여 http로 설정 4. h1에서 thyme을 실행하여 Mobius 서버와 연결 확인 5. Thyme의 conf.js를 수정하여 coap으로 설정 6. h2에서 thyme을 실행하여 Mobius 서버와 연결 확인	
판정 기준	Client/Server 통신 구조를 가지는 프로토콜인 HTTP, CoAP 프로토콜로 Thyme	

	과 Mobius 사이의 연결 시도, 성공적으로 연결될 경우 “지원” 되지 않을 경우 “미지원”으로 판단
시험 결과	1. Mininet 시뮬레이터를 실행  2. 가상 호스트 Mobius에 Mobius 서버를 실행  3. thyme의 conf.js를 수정하여 http로 설정  4. h1에서 thyme을 실행하여 Mobius 서버와 연결 확인 5. Thyme의 conf.js를 수정하여 coap으로 설정




```
var conf = {};
var cse = {};
var ae = {};
var cnt_arr = [];
var sub_arr = [];
var acp = {};

conf.useprotocol = 'coap'; // select one for 'http' or 'mqtt' or 'coap' or 'ws'

// build cse
cse.host        = '10.0.0.1';
cse.port        = '7579';
cse.name        = 'Mobius';
cse.id          = '/Mobius';
cse.mqttport    = '1883';
cse.wsport      = '7577';
```

6.h2에서 thyme을 실행하여 Mobius 서버와 연결 확인

판정	지원

<표 6-22> Transport 테스트케이스 : TC-TRP-02

테스트케이스		
TC-TRP-02		
시험항목	Publish/Subscribe 통신 구조의 MQTT 프로토콜 지원 여부 검증	
관련요구사항	SF-Scal-Req-004	Pub/Sub 모델의 통신 기술 지원
	CPS-Scal-Req-003	
	SF-Scal-Req-005	Pub/Sub 모델 통신 기술을 통한 데이터 송신
	CPS-Scal-Req-003	
	SF-Scal-Req-006	Pub/Sub 모델 통신 기술을 통한 데이터 수신
	CPS-Scal-Req-003	
	CPS-Inter-Req-006	각 시스템에 최적화 된 통신 기술 지원
	SF-Operate-Req-001	다양한 산업용 통신기술을 이용하여 메시지 통신 여

<table>
<tr><td rowspan="3"></td><td>CPS-Operate-Req-001</td><td>부</td></tr>
<tr><td>SF-Operate-Req-002</td><td>다양한 산업용 통신기술을 이용하여 제어메시지의 수신 및 동작의 가능 여부</td></tr>
<tr><td>SF-Operate-Req-010</td><td>산업용 통신기술을 이용하여 기기들의 설정을 서버로 전송</td></tr>
<tr><td>시험 방법</td><td colspan="2">Mininet simulator의 가상 네트워크를 구성하고, 하나의 가상 호스트에 Mobius 서버를 실행. 다른 호스트에 Thyme의 프로토콜을 설정하여 실행</td></tr>
<tr><td>시험 절차</td><td colspan="2">1. Mininet 시뮬레이터를 실행
2. 가상 호스트 Mobius에 Mobius 서버를 실행
3. Thyme의 conf.js를 수정하여 mqtt로 설정
4. h1에서 thyme을 실행하여 Mobius 서버와 연결 확인</td></tr>
<tr><td>판정 기준</td><td colspan="2">Publish/Subscribe 통신 구조를 가지는 프로토콜인 MQTT로 설정하여 Thyme과 Mobius 사이의 연결 시도, 성공적으로 연결될 경우 “지원” 되지 않을 경우 “미지원”으로 판단</td></tr>
<tr><td>시험 결과</td><td colspan="2">1. Mininet 시뮬레이터를 실행

2. 가상 호스트 Mobius에 Mobius 서버를 실행

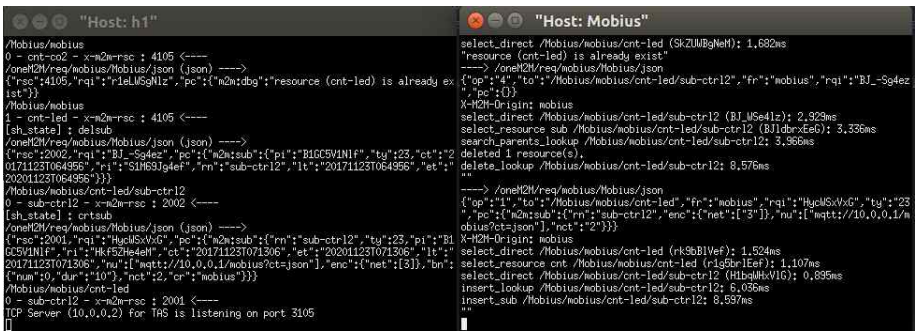
3. thyme의 conf.js를 수정하여 mqtt로 설정

4. h1에서 thyme을 실행하여 Mobius 서버와 연결 확인</td></tr>
<tr><td>판정</td><td colspan="2">지원</td></tr>
</table>

```
var conf = {};
var cse = {};
var ae = {};
var cnt_arr = [];
var sub_arr = [];
var acp = {};

conf.useprotocol = 'mqtt'; // select one for 'http' or 'mqtt' or 'coap' or 'ws'

// build cse
cse.host        = '10.0.0.1';
cse.port        = '7579';
cse.name        = 'Mobius';
cse.id          = '/Mobius';
cse.mqttport    = '1883';
cse.wsport      = '7577';
```

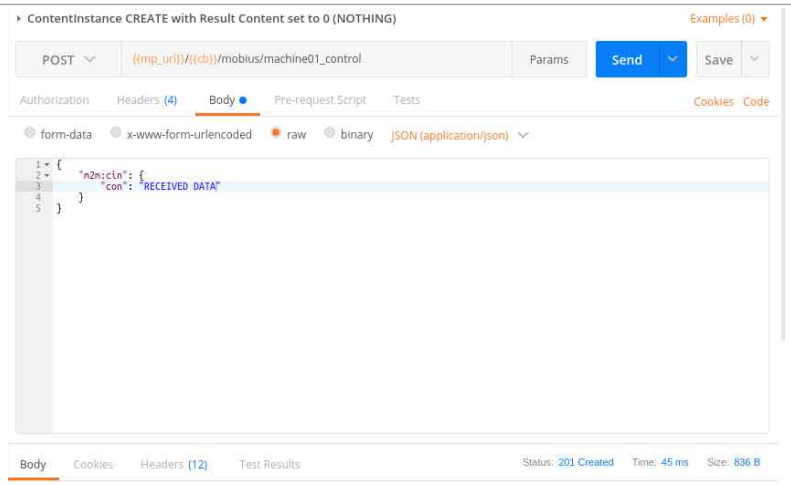


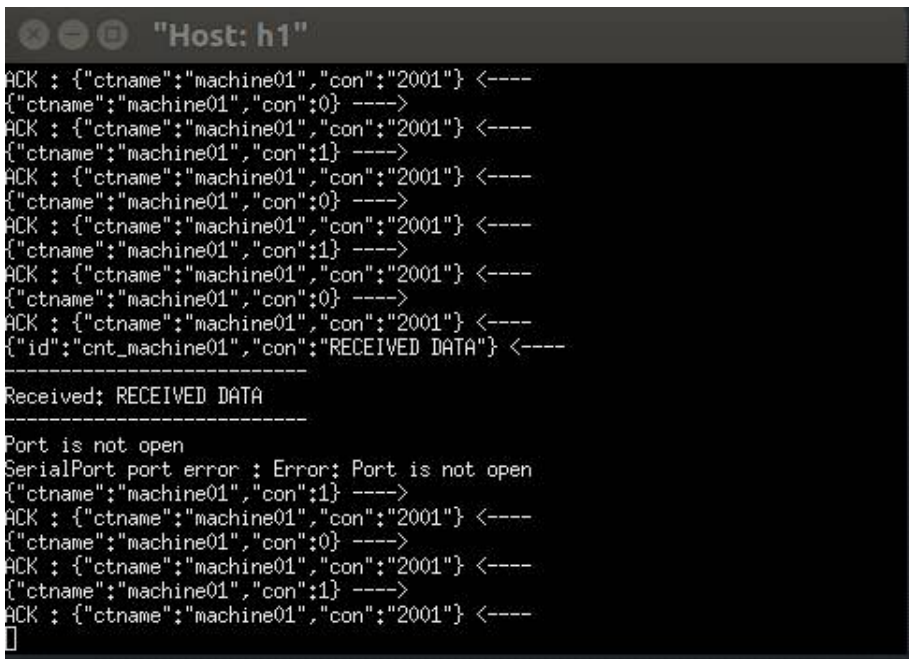
아래 테스트는 실험결과 기기 간의 통신은 되지 않고 서버를 통해서만 가능한 것으로 확인되었다.

<표 6-23> Transport 테스트케이스 : TC-TRP-03

<table>
<tr><th colspan="3">테스트케이스</th></tr>
<tr><td colspan="3">TC-TRP-03</td></tr>
<tr><td>시험항목</td><td colspan="2">Mobius 플랫폼을 이용한 데이터 송 수신 테스트</td></tr>
<tr><td rowspan="8">관련요구사항</td><td>SF-Inter-Req-001</td><td rowspan="3">관리 서버의 각 기기들에 제어 메시지 송신</td></tr>
<tr><td>CPS-Inter-Req-001</td></tr>
<tr><td>CPS-Inter-Req-008</td></tr>
<tr><td>SF-Inter-Req-002</td><td rowspan="3">관리 서버의 각 기기들에 제어 메시지 수신</td></tr>
<tr><td>CPS-Inter-Req-001</td></tr>
<tr><td>CPS-Inter-Req-008</td></tr>
<tr><td>CPS-Inter-Req-002</td><td>구성 소프트웨어 간 통신 지원</td></tr>
<tr><td>시험 방법</td><td colspan="2">Mininet simulator의 가상 네트워크를 구성하고, 하나의 가상 호스트에 Mobius 서버를 실행. 다른 호스트에 Thyme의 프로토콜을 설정하여 실행</td></tr>
<tr><td>시험 절차</td><td colspan="2">1. Mininet 시뮬레이터를 실행
2. 가상 호스트 Mobius에 Mobius 서버를 실행
3. h1에서 Thyme과 TAS를 실행하여 데이터 송신 테스트(기능1)
4. POSTMAN을 통하여 데이터를 전송하여 h1에서 Thyme과 TAS를 실행하여 데이터 수신 테스트
5. h2에서 thyme을 실행하여 h1과의 직접 통신 확인(기능2)
※ Thyme간의 직접적인 연결은 제공되지 않음, Thyme은 Mobius와 연결
표준 상 AE간 통신은 지원하지 않음</td></tr>
<tr><td>판정 기준</td><td colspan="2">Mobius 플랫폼과 기기들(기능1), 그리고 구성 소프트웨어간 직접 통신(기능2)이 모두 제공될 경우 ‘지원’, 둘 중 하나만 지원될 경우 ‘일부 지원(지원 기능)’, 모두 지원하지 않을 경우 미지원</td></tr>
<tr><td>시험 결과</td><td colspan="2">1. Mininet 시뮬레이터를 실행
2. 가상 호스트 Mobius에 Mobius 서버를 실행
3. h1에서 Thyme과 TAS를 실행하여 데이터 송신 테스트(기능1)
 

4. POSTMAN을 통하여 데이터를 전송하여 h1에서 Thyme과 TAS를 실행하여 데이터 수신 테스트</td></tr>
</table>

<table>
<tr><td></td><td> 

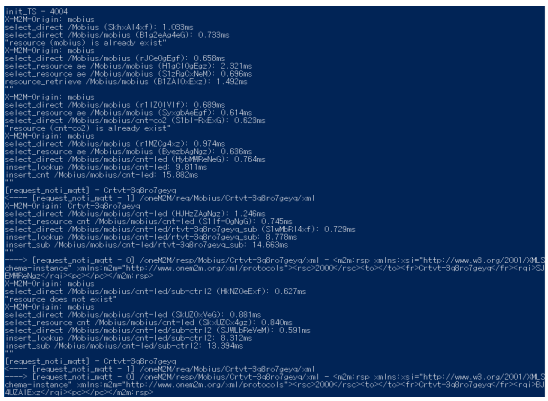
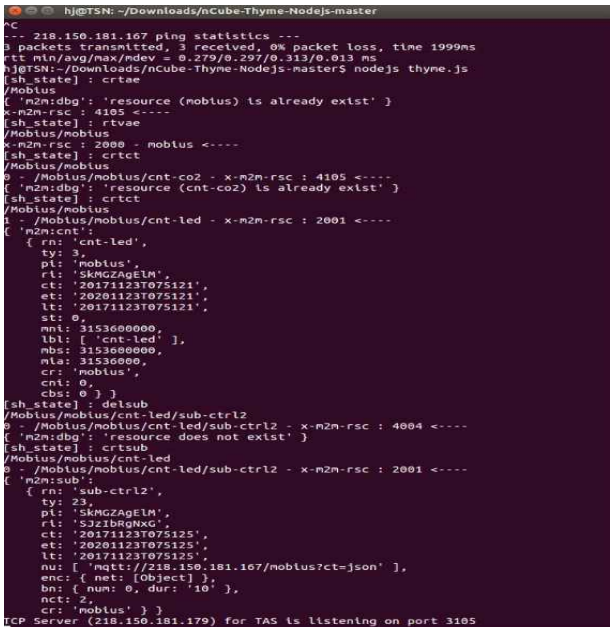


5. h2에서 thyme을 실행하여 h1과의 직접 통신 확인(기능2)
※ Thyme간의 직접적인 연결은 제공되지 않음, Thyme은 Mobius와 연결 표준 상 AE간 통신은 지원하지 않음</td></tr>
<tr><td>판정</td><td>일부 지원(기능1)</td></tr>
</table>

이 테스트는 OS가 다른 기기간의 통신 가능 여부에 관한 것이다.

<표 6-24> Transport 테스트케이스 : TC-TRP-04

테스트케이스		
TC-TRP-04		
시험항목	이종 환경과의 통신 지원	
관련요구사항	CPS-Inter-Req-003	이종 외부 환경과의 통신 지원
	CPS-Operate-Req-003	이종 노드간 데이터 공유

시험 방법	Windows 환경에서 Mobius를 실행하고, Linux 환경에서 Thyme을 실행하여 이종 환경간의 통신 지원 여부를 확인
시험 절차	1. Windows 환경인 PC 1에서 Mobius 서버 실행 2. Linux 환경인 PC 2에서 Thyme 실행
판정 기준	Windows 환경의 Mobius 서버와 Linux환경의 Thyme이 정상적으로 연결될 경우 ‘지원’, 아닐 경우 ‘미지원’
시험 결과	1. Windows 환경인 PC 1에서 Mobius 서버 실행  2. Linux 환경인 PC 2에서 Thyme 실행 
판정	지원

2) NetTime 테스트케이스

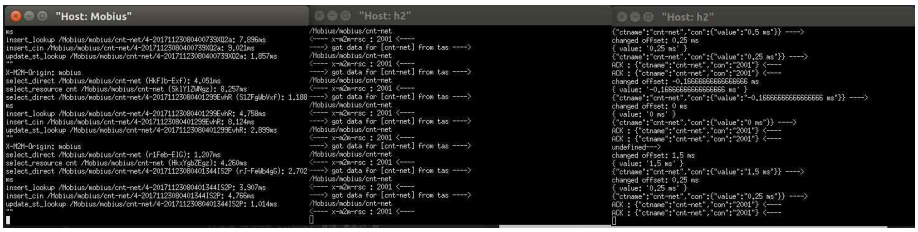
실시간성 관련 검증으로, oneM2M은 대부분의 실시간성 요구사항을 만족하지 못해 시간 동기화에 대한 부분만 검증했다.

<표 6-25> NetTime 테스트케이스 : TC-NET-01

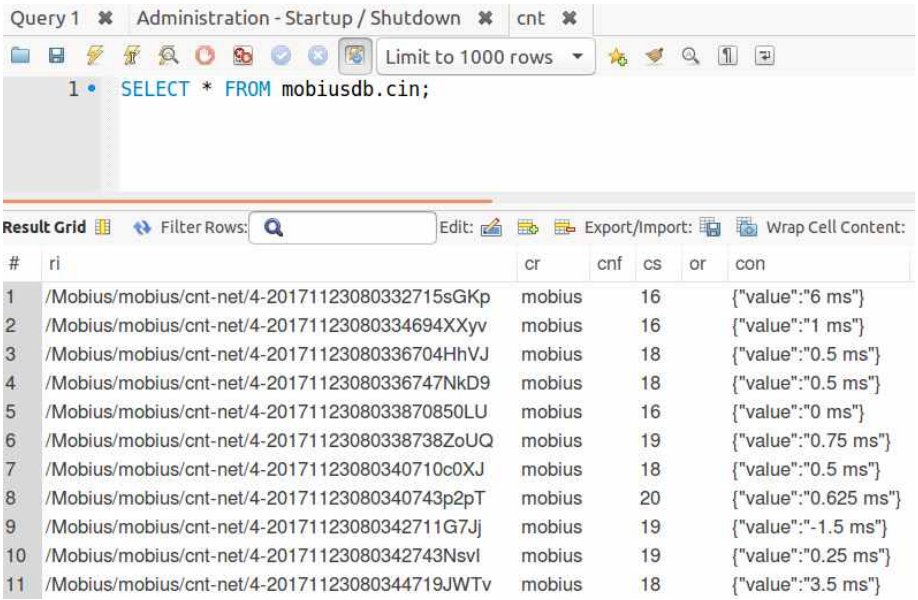
테스트케이스 TC-NET-01	
시험항목	Mobius 플랫폼을 통한 노드 간 실시간 시간 동기화 확인 시험
관련요구사항	SF-Timing-Req-006 실시간으로 다른 노드 간 시간동기화 확인
시험 방법	Mobius 플랫폼 내 노드 간 실시간 시간 동기화를 확인하기 위해 Time Source 서버와 시간 데이터를 전달할 Mobius 서버를 실행시키고, Time Source 서버의 시간을 Mobius 서버를 통해 받아 처리하기 위해 노드에서 Thyme과 TAS를 실행함
시험 절차	1. Mininet 시뮬레이터를 실행 2. 가상 호스트 Mobius에 Mobius 서버를 실행 3. Time Source 서버인 server.js 실행 4. Thyme이 시간 데이터를 받아올 수 있도록 설정 5. Thyme을 실행 6. TAS를 실행(app.js)시켜서 Time Source 서버와의 시간 차이 offset을 계산 7. 데이터 베이스 조회을 통해 Time source 서버와 노드 간 시간동기화를 확인
판정 기준	Time Source 서버가 Mobius 서버를 통해 시간 데이터 값을 노드로 전달하고, Time Source 서버와 Thyme 및 TAS가 동작되는 노드 간 시간 차이를 계산하여 그 결과 값을 확인할 수 있으면 “지원”인 것으로 판단하고, 그렇지 않으면 “미지원”으로 판단
시험 결과	1. Mininet 시뮬레이터를 실행 2. 가상 호스트 Mobius에 Mobius 서버를 실행 3. Time Source 서버인 server.js 실행  4. Thyme이 시간 offset을 받아올 수 있도록 cnt-net container 설정 5. Thyme.js로 Thyme을 실행

6. TAS인 app.js 실행

- Time Source 서버와의 시간 차이 offset 계산
- 계산 값을 cnt-net container를 이용하여 모비우스 서버로 전달



7. 결과 값을 통해 Time source 서버와 노드 간 시간동기화를 확인



#	ri	cr	cnf	cs	or	con
1	/Mobius/mobius/cnt-net/4-20171123080332715sGKp	mobius		16		{"value":"6 ms"}
2	/Mobius/mobius/cnt-net/4-20171123080334694XXyv	mobius		16		{"value":"1 ms"}
3	/Mobius/mobius/cnt-net/4-20171123080336704HhVJ	mobius		18		{"value":"0.5 ms"}
4	/Mobius/mobius/cnt-net/4-20171123080336747NkD9	mobius		18		{"value":"0.5 ms"}
5	/Mobius/mobius/cnt-net/4-20171123080338370850LU	mobius		16		{"value":"0 ms"}
6	/Mobius/mobius/cnt-net/4-20171123080338738ZoUQ	mobius		19		{"value":"0.75 ms"}
7	/Mobius/mobius/cnt-net/4-20171123080340710c0XJ	mobius		18		{"value":"0.5 ms"}
8	/Mobius/mobius/cnt-net/4-20171123080340743p2pT	mobius		20		{"value":"0.625 ms"}
9	/Mobius/mobius/cnt-net/4-20171123080342711G7Jj	mobius		19		{"value":"-1.5 ms"}
10	/Mobius/mobius/cnt-net/4-20171123080342743Nsvl	mobius		19		{"value":"0.25 ms"}
11	/Mobius/mobius/cnt-net/4-20171123080344719JWTv	mobius		18		{"value":"3.5 ms"}

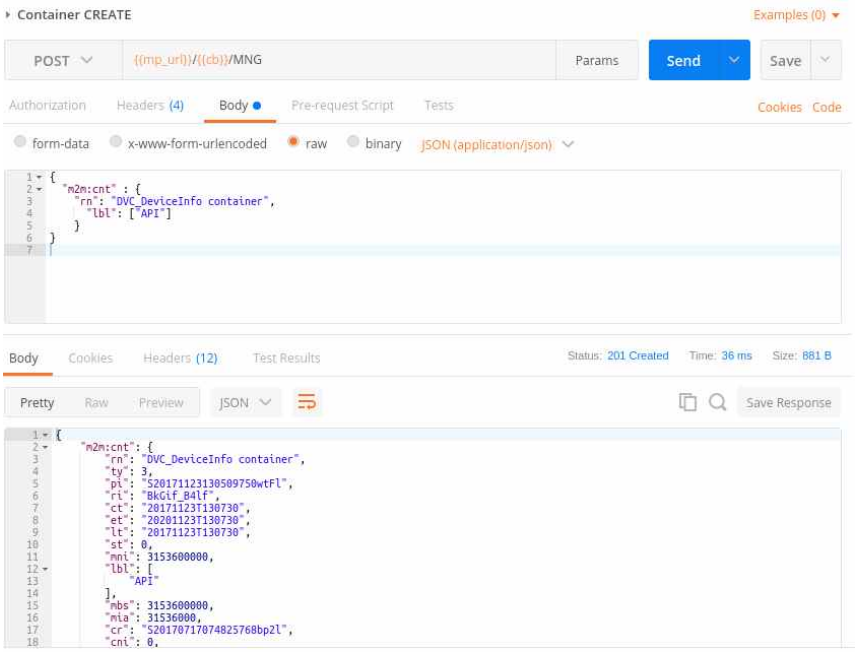
판정	지원

3) S/W&HW Manager 테스트케이스

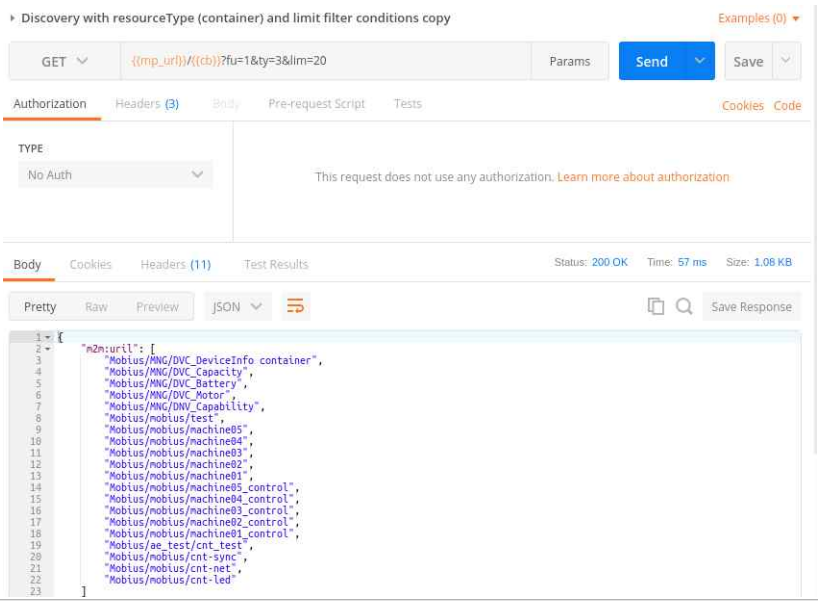
이 검증항목은 구성요소를 식별하고, 제어 메시지를 전달하고, 각 요소의 구성 변경을 확인하는 것이다.

<표 6-26> S/W&HW Manager 테스트케이스 : TC-MNG-01

테스트케이스		
TC-MNG-01		
시험항목	Mobius 플랫폼의 구성 기기(H/W, 생산제품) 관찰 및 식별 확인 시험	
관련요구사항	SF-Inter-Req-007	관리 서버의 구성 기기 상태 정보 표시 가능 여부
	SF-Com-Req-008	H/W 추가/제거의 알림
	CPS-Com-Req-002	
	SF-Com-Req-013	사용자의 팩토리 구성 요소들 관찰 가능 여부
	CPS-Com-Req-001	
	SF-Com-Req-018	구성요소에 부여된 고유 식별자를 통한 제품 제작 구성 요소 및 기능 식별 가능성
	CPS-Com-Req-003	
시험 방법	Mobius 플랫폼 내 구성 기기 관찰 및 생산 제품 식별 가능 여부를 확인하기 위해 Mobius 서버를 실행시키고, 서버에 구성되어 있는 기기 및 생산 제품 정보를 받아옴	
시험 절차	1. Mininet 시뮬레이터를 실행 2. 가상 호스트 Mobius에 Mobius 서버를 실행 3. oneM2M Testing API를 사용하기 위하여 h1에서 Postman을 실행 4. Mobius 서버에 구성 기기 및 생산 제품을 확인할 수 있는 container를 생성 5. Discovery resource를 통해 Mobius 플랫폼 구성기기들의 정보를 확인	
판정 기준	Mobius 플랫폼이 구성하고 있는 모든 기기 정보를 확인하고 생산 제품의 식별이 가능하면 “지원“, 그렇지 않으면 “미지원“로 판단	
시험 결과	1. Mininet 시뮬레이터를 실행 2. 가상 호스트 Mobius에 Mobius 서버를 실행 3. POSTMAN을 실행 4. Mobius 서버에 구성 기기 및 생산 제품을 확인할 수 있는 container를 생성 - DVC_Capability, DVC_Motor, DVC_Battery, DVC_Capacity, DVC_DeviceInfo container들을 생성(사진은 하나만 변경하여 반복)	



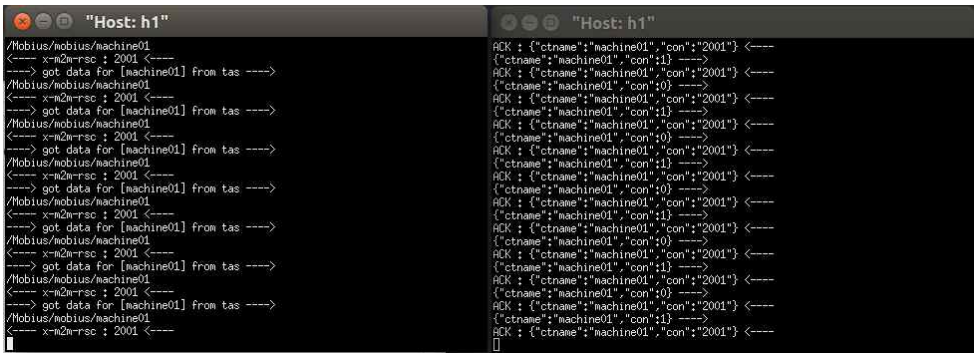
5. Discovery resource를 통해 Mobius 플랫폼 구성기기들의 정보를 확인

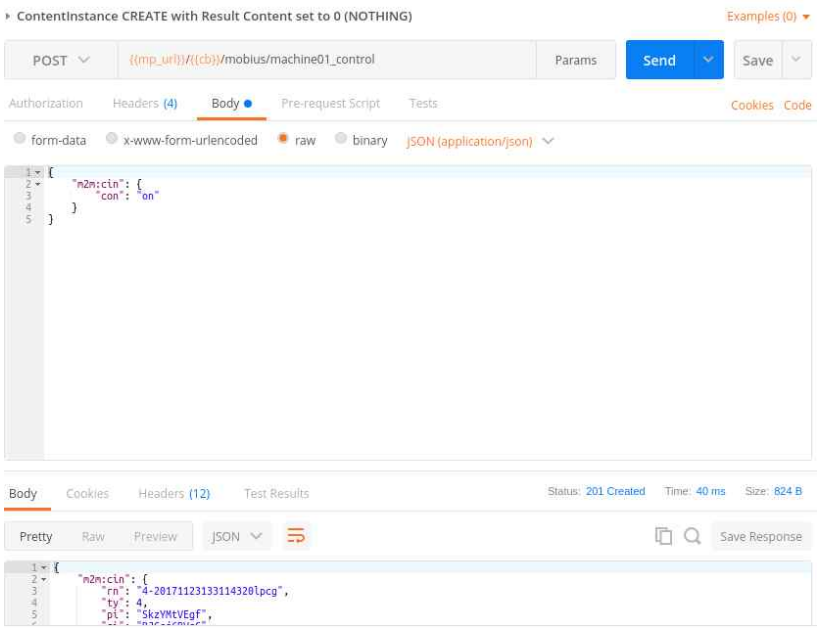


판정	지원

제어 메시지 관련 테스트이다.

<표 6-27> S/W&HW Manager 테스트케이스 : TC-MNG-02

테스트케이스		
TC-MNG-02		
시험항목	Mobius 플랫폼의 구성 기기들 관리 및 제어 메시지 전달 여부	
관련요구사항	SF-Inter-Req-004 CPS-Inter-Req-008	구성 기기들의 관리 서버로부터 받아온 제어 메시지 표시 가능 여부
	SF-Operate-Req-006	발생하는 모든 결함을 공장 내에서 공유하여 적절한 대처할 수 있는 기능의 가능 여부
시험 방법	Mobius 플랫폼 내 구성 기기들 관리 및 제어 기능을 확인하기 위해 Mobius 서버를 실행시키고, 관리 및 제어 메시지를 전달 또는 갱신	
시험 절차	1. Mininet 시뮬레이터를 실행 2. 가상 호스트 Mobius에 Mobius 서버를 실행 3. 가상 호스트 h1에 Thyme과 TAS 실행 4. oneM2M API Tester를 활용하기 위하여 POSTMAN 실행 5. 구성 기기로 관리 및 제어 메시지를 전달 6. 관리 및 제어 메시지 갱신 여부를 확인	
판정 기준	사용자가 Mobius 플랫폼 내 구성 기기들을 관리할 수 있고, 구성 기기들간의 제어 메시지를 전달 및 갱신할 수 있으면 “가능“, 그렇지 않으면 “불가능“으로 판단	
시험 결과	1. Mininet 시뮬레이터를 실행 2. 가상 호스트 Mobius에 Mobius 서버를 실행 3. 가상 호스트 h1에 Thyme과 TAS 실행  4. oneM2M API Tester를 활용하기 위하여 POSTMAN 실행 5. POSTMAN을 통하여 구성 기기로 관리 및 제어 메시지를 전달	

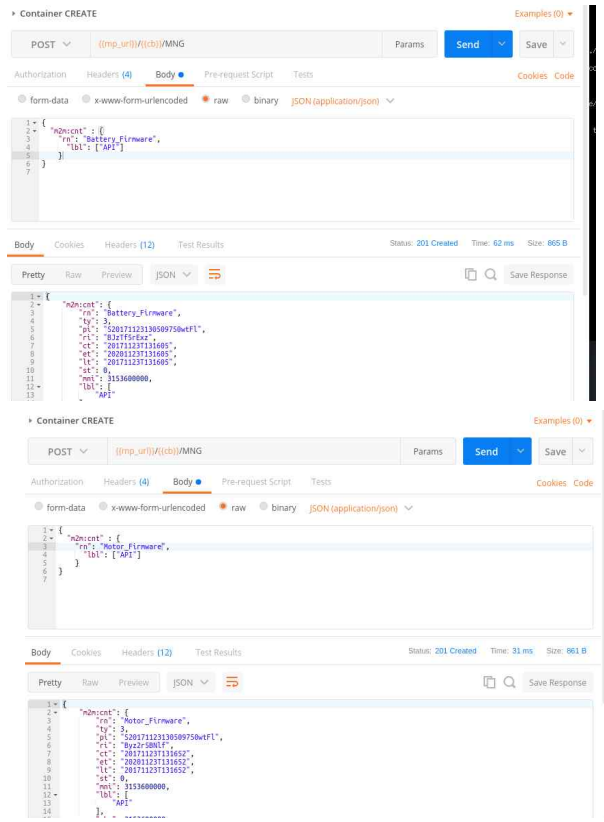
<table>
<tr><td></td><td> 

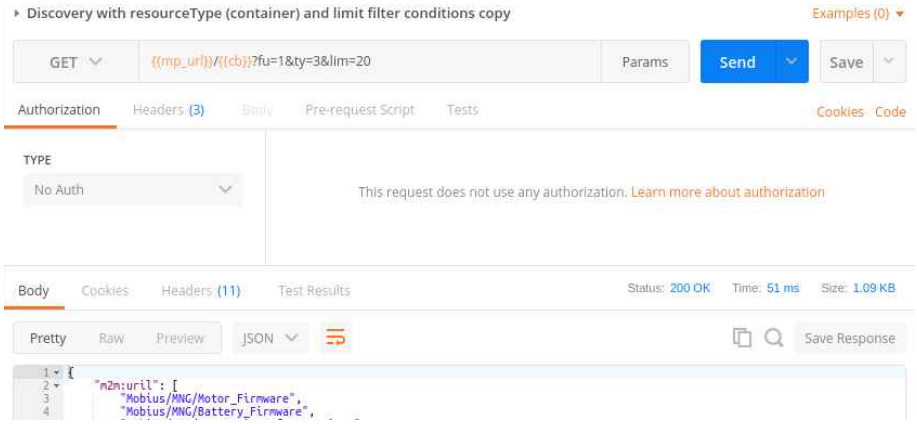
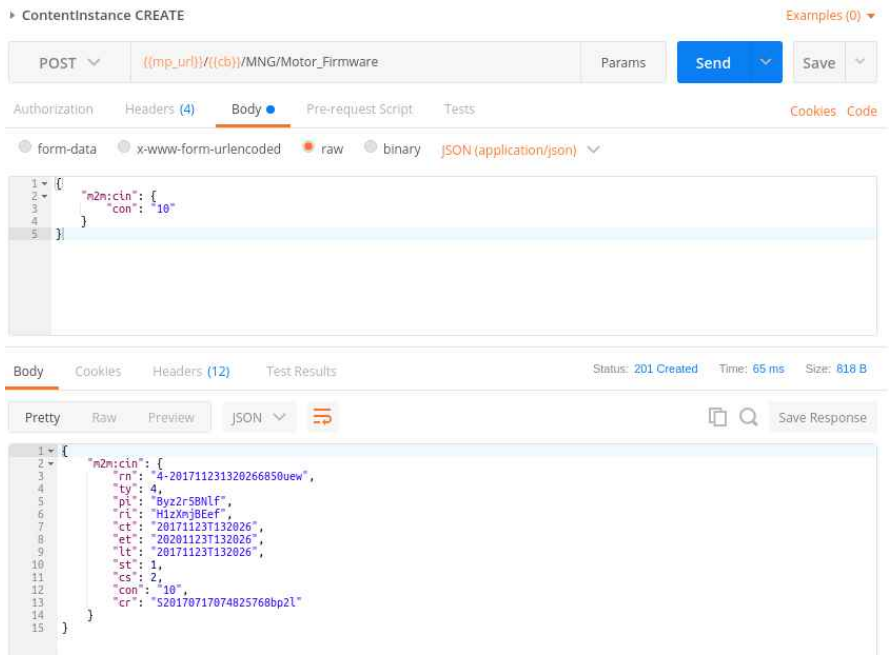
6. 관리 및 제어 메시지 갱신 여부 확인
 
</td></tr>
<tr><td>판정</td><td>지원</td></tr>
</table>

구성변경 관련 테스트이다.

<표 6-28> S/W&HW Manager 테스트케이스 : TC-MNG-03

테스트케이스	
TC-MNG-03	
시험항목	Mobius 플랫폼의 구성 기기들의 소프트웨어 및 기능 모듈 업그레이드(생성) 관리 및 변경 기능 확인 시험
관련요구사항	CPS-Scal-Req-002 기능 모듈 인스턴스 생성 및 연결 SF-Com-Req-001 원격 S/W업그레이드 SF-Com-Req-005 사용자에 S/W업그레이드 후의 변경 사항 알림 CPS-Operate-Req-002 소프트웨어의 이종 노드 적용을 위한 표준을 준수한 이식성 여부
시험 방법	Mobius 플랫폼 내 구성 기기들의 소프트웨어 및 기능 모듈 업그레이드 관리

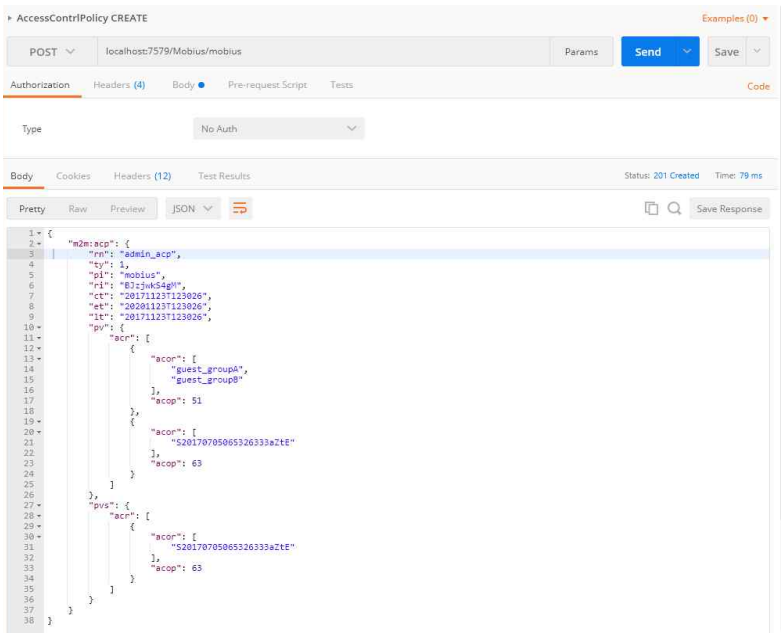
	및 변경 기능을 확인하기 위해 Mobius 서버를 실행시키고, 구성 기기들의 소프트웨어(기능 모듈)를 관리 및 변경할 수 있는 container를 생성하고 정보를 갱신 또는 받아옴
시험 절차	1. Mininet 시뮬레이터를 실행 2. 가상 호스트 Mobius에 Mobius 서버를 실행 3. oneM2M Testing API를 사용하기 위하여 h1에서 Postman을 실행 4. Mobius 서버에 소프트웨어(기능 모듈)을 확인할 수 있는 container를 생성 5. Discovery resource를 통해 Mobius 플랫폼 구성기기들의 소프트웨어(기능 모듈) 정보를 확인 6. POSTMAN Container UPDATE를 통해 소프트웨어(기능 모듈) 갱신 확인
판정 기준	Mobius 플랫폼 내 구성 기기들의 소프트웨어 및 기능 모듈 업그레이드 관리 및 변경 기능을 확인하기 위해 Mobius 서버를 실행시키고, 구성 기기들의 소프트웨어(기능 모듈)를 관리 및 변경할 수 있는 container를 생성하고 정보를 갱신 또는 받아올 수 있으면 “지원”인 것으로 판단하고, 그렇지 않으면 “미지원”으로 판단한다.
시험 결과	1. Mininet 시뮬레이터를 실행 2. 가상 호스트 Mobius에 Mobius 서버를 실행 3. oneM2M Testing API를 사용하기 위하여 h1에서 Postman을 실행 4. Mobius 서버에 소프트웨어(기능 모듈)을 확인할 수 있는 container를 생성 - Battery_Firmware, Motor_Firmware 

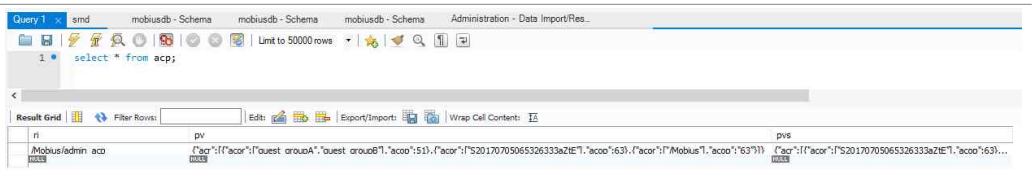
	5. Discovery resource를 통해 Mobius 플랫폼 구성기기들의 소프트웨어(기능 모듈) 정보를 확인  6. POSTMAN Container UPDATE를 통해 소프트웨어(기능 모듈) 갱신 확인 
판정	지원

4) Security 테스트케이스

다음은 보안관련 검증항목이다.

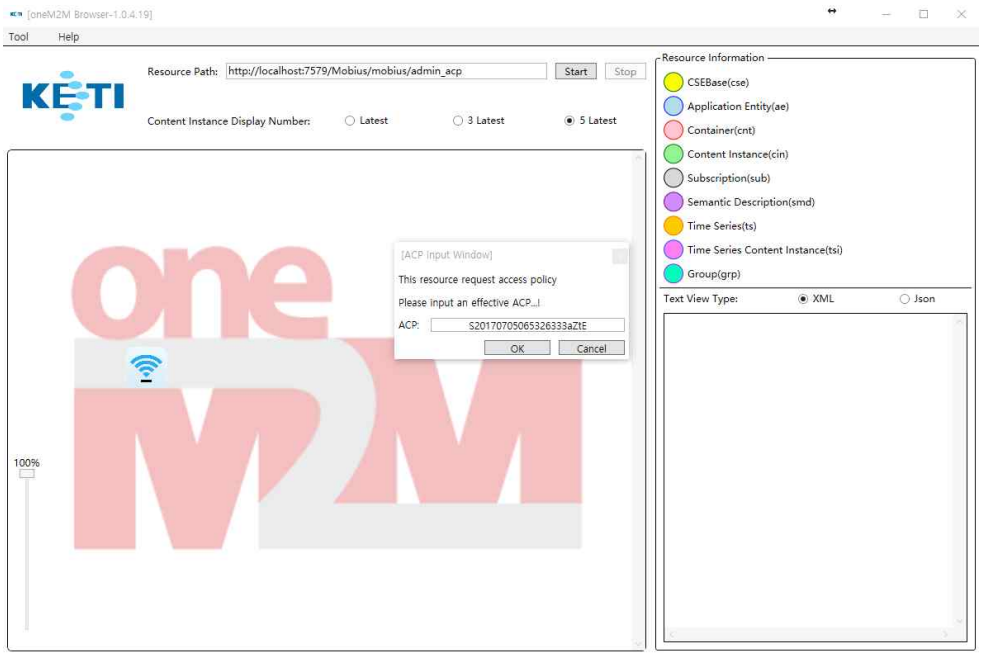
<표 6-29> Security 테스트케이스 : TC-SEC-01

테스트케이스	
TC-SEC-01	
시험항목	Mobius 플랫폼의 구성 기기 정보 보안 지원 확인
관련요구사항	SF-Depend-Req-001 서버에 있는 기기 정보 접근에 관한 보안 체계 CPS-Depend-Req-001 SF-Com-Req-002 S/W업그레이드에 대한 권한 관리
시험 방법	사용자는 Mobius 플랫폼이 구성 기기들의 정보 접근 및 권한에 관한 보안을 지원하는지 확인하기 위해 Mobius 서버를 실행시키고, 서버에 구성되어 있는 기기 노드들의 정보에 접근 Security 테스트케이스는 Monitoring Agent인 oneM2M Browser를 사용하기 위하여 Windows 환경에서 진행
시험 절차	1. oneM2M Testing API를 사용하기 위하여 windows 환경에서 Postman을 실행 2. Mobius 서버에 AccessControlPolicy를 생성 3. MySQL Workbench를 실행 4. Mobius 서버에 접근하기 위해 ACP를 입력 5. 인증 받은 사용자는 접근이 가능하고 인증 받지 않은 사용자(잘못된 ACP 입력)는 ACCESS_DENIED되는 것을 확인
판정 기준	사용자가 Mobius 플랫폼의 서버가 구성하고 있는 기기 정보에 접근할 때 인증 받은 사용자의 경우에만 접근을 할 수 있으면 '지원', 인증 받지 않은 사용자도 접근할 수 있으면 '지원하지 않음'으로 판단
시험 결과	1. POSTMAN을 실행 2. Mobius 서버에 ACP(AccessControlPolicy)를 생성 

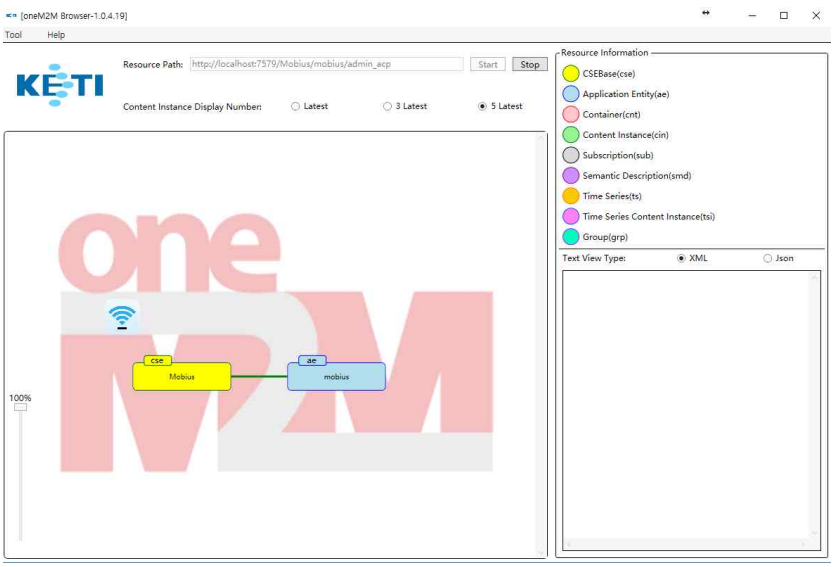


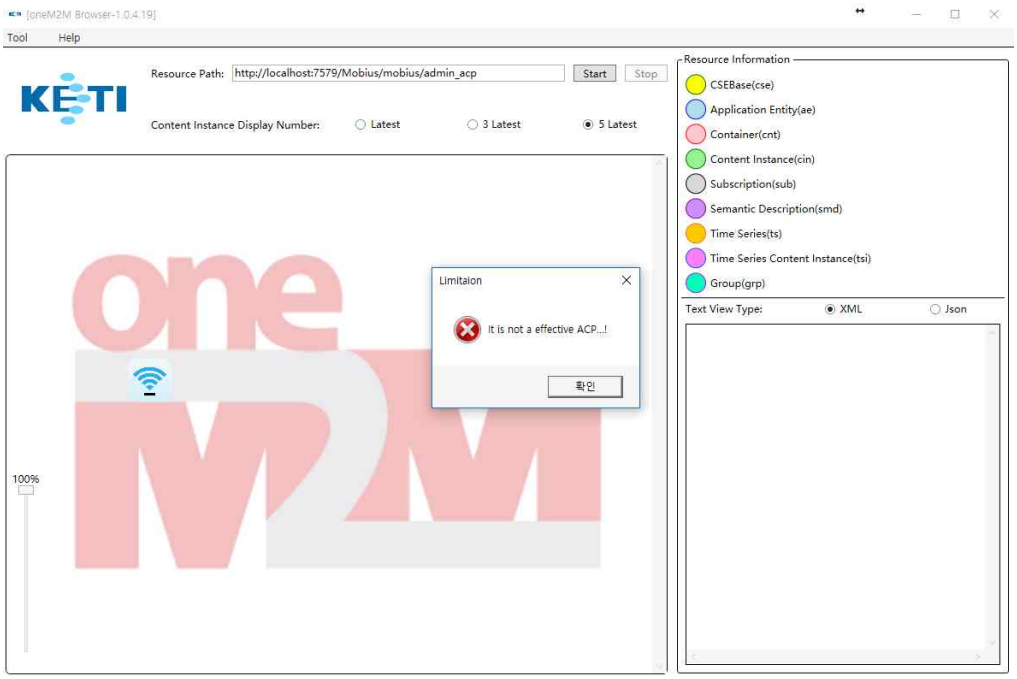
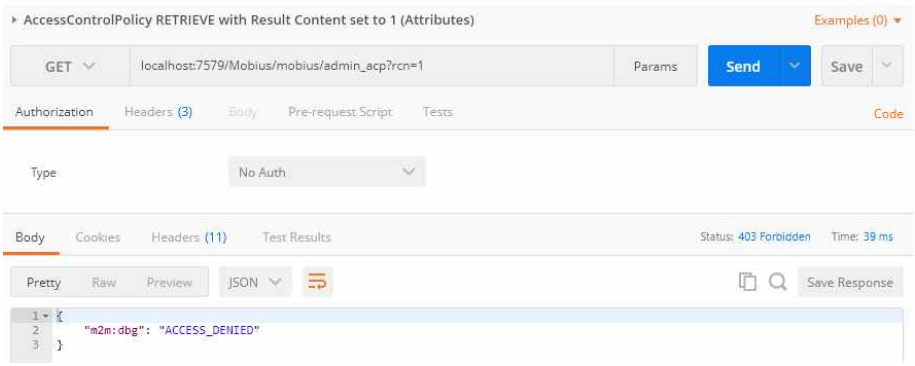
3. oneM2M Browser를 실행

4. Mobius 서버에 접근하기 위해 ACP를 입력



5. 인증 받은 사용자는 접근이 가능하고 인증 받지 않은 사용자(잘못된 ACP 입력)는 ACCESS_DENIED되는 것을 확인
(연결)

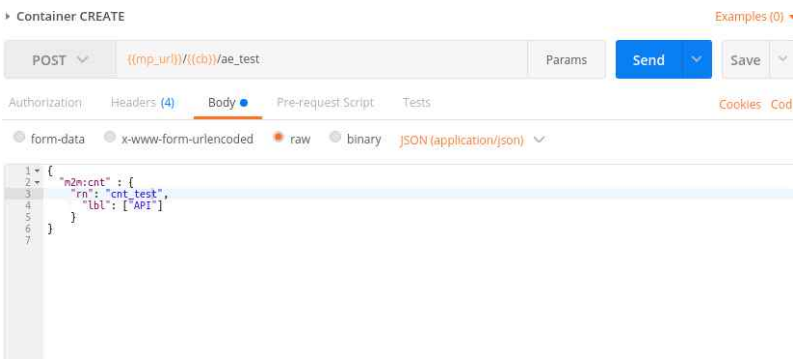


	(접근 제한)  
판정	지원

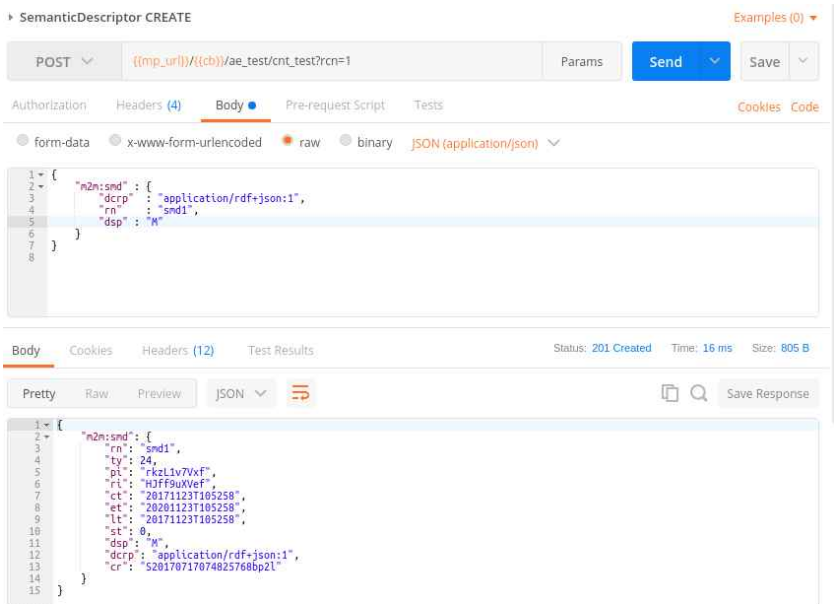
5) Semantic 테스트케이스

메시지 의미 해석에 대한 부분이다.

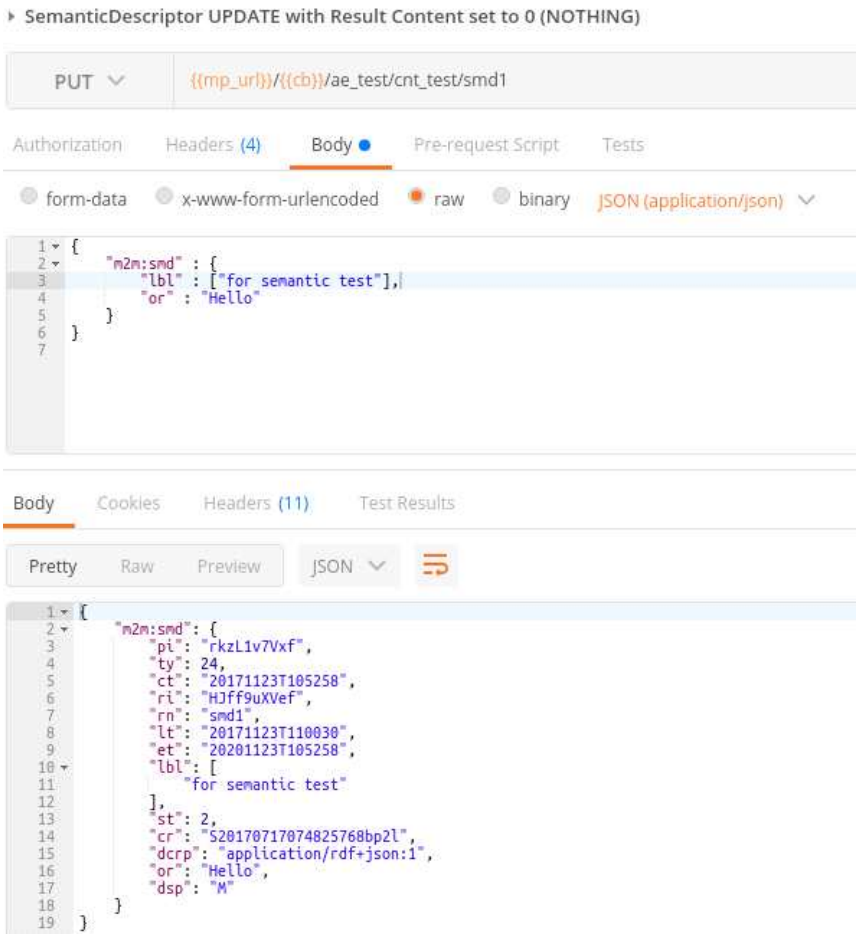
<표 6-30> Semantic 테스트케이스 : TC-SEM-01

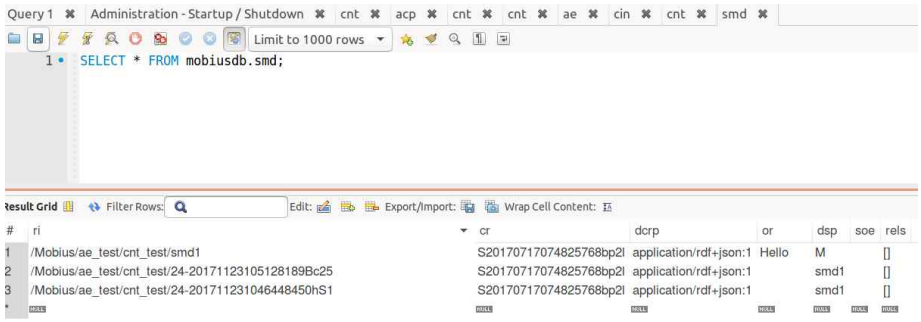
<table>
<tr><th colspan="3">테스트케이스</th></tr>
<tr><td colspan="3">TC-SEM-01</td></tr>
<tr><td>시험항목</td><td colspan="2">Mobius 플랫폼 내 발생하는 메시지의 동일 해석 및 보안 체계 지원 여부</td></tr>
<tr><td rowspan="6">관련요구사항</td><td>SF-Scal-Req-010</td><td>모든 기기에서 제어 메시지 해석의 동일성</td></tr>
<tr><td>SF-Inter-Req-003</td><td rowspan="3">구성 기기들의 관리 서버 제어 메시지에 대한 적합한 해석 가능 여부</td></tr>
<tr><td>CPS-Inter-Req-005</td></tr>
<tr><td>CPS-Inter-Req-008</td></tr>
<tr><td>SF-Depend-Req-016</td><td rowspan="2">스마트공장에서 발생되는 메시지에 대한 보안 체계 제공 여부</td></tr>
<tr><td>CPS-Depend-Req-005</td></tr>
<tr><td>시험 방법</td><td colspan="2">Mobius 플랫폼이 구성기기들의 제어 메시지 해석 및 보안을 위한 ontology를 지원하는지 확인하기 위해 Mobius 서버를 실행시키고 메시지에 ontology Reference resource를 추가</td></tr>
<tr><td>시험 절차</td><td colspan="2">1. Mininet simulator 실행
2. 가상 호스트 Mobius에 Mobius 서버를 실행
3. oneM2M Testing API를 사용하기 위하여 h1에 Postman을 실행
4. Mobius 서버에 구성 기기들의 메시지 전달을 위한 container를 생성
5. 생성된 container 안에 SemanticDescriptor를 생성
6. SemanticDescriptor에 container 설명 등록
7. container 내부에 SemanticDescriptor 정보를 확인</td></tr>
<tr><td>판정 기준</td><td colspan="2">oneM2M API testing application을 통해 ontology 정보를 전달할 수 있는 container를 생성 및 제거할 수 있고 그 내부에 SemanticDescriptor를 통해 메시지를 구분할 수 있으면 “지원”, 그렇지 않으면 “미지원”으로 판단</td></tr>
<tr><td>시험 결과</td><td colspan="2">1. Mininet simulator 실행
2. 가상 호스트 Mobius에 Mobius 서버를 실행
3. oneM2M Testing API를 사용하기 위하여 h1에 Postman을 실행
4. Mobius 서버에 구성 기기들의 메시지 전달을 위한 container를 생성
 
</td></tr>
</table>

5. 생성된 container 안에 SemanticDescriptor를 생성



6. SemanticDescriptor에 container 설명 등록



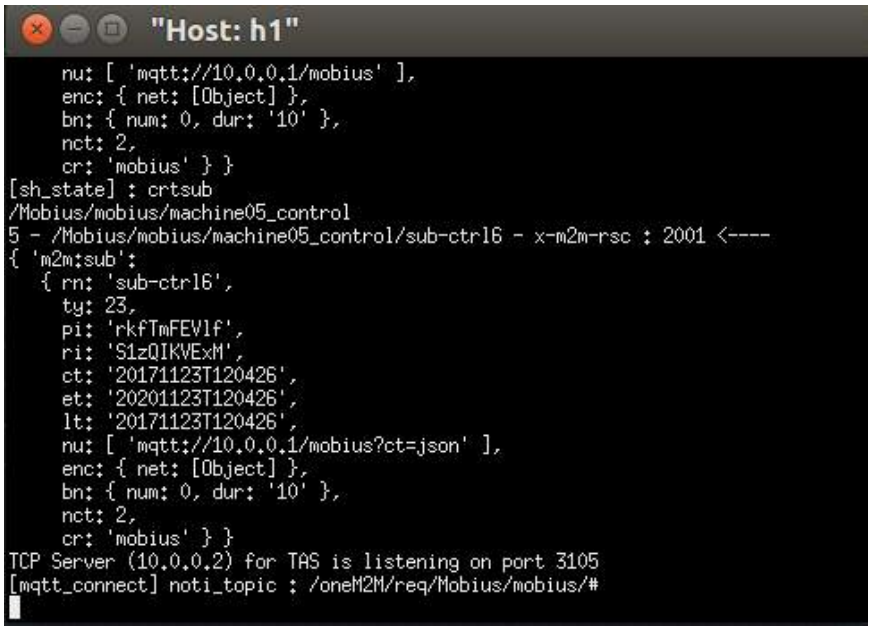
	7. container 내부에 SemanticDescriptor 정보를 확인 
판정	지원

6) Lifecycle 테스트케이스

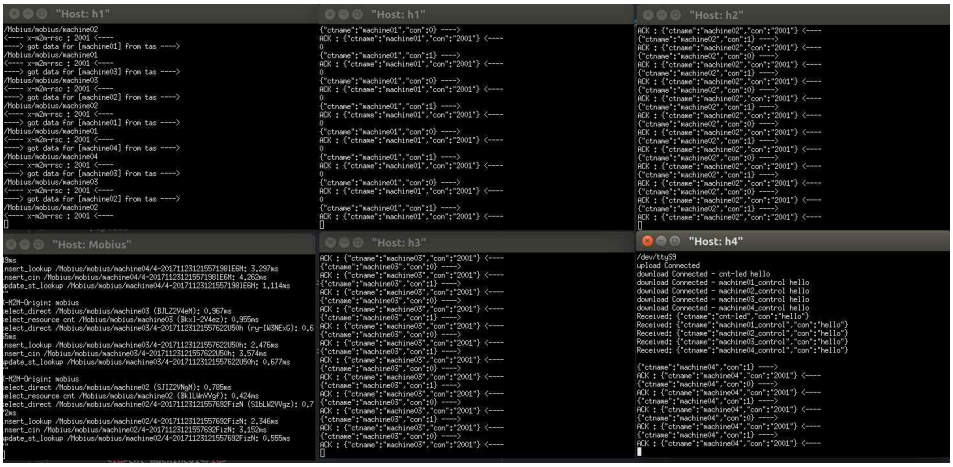
공정과정 관리에 대한 검증부분이다.

<표 6-31> Lifecycle 테스트케이스 : TC-LFC-01

테스트케이스		
TC-LFC-01		
시험항목	어플리케이션이 서버에 저장된 생산 제품에 대한 정보 및 현재 공정에 대한 정보의 수신 가능 여부	
관련요구사항	SF-Scal-Req-001	공장 기기 설비들의 연동 구조 실시간 모니터링
	SF-Inter-Req-013	구성 기기들의 서버로부터 생산 제품에 대한 정보 수신 가능 여부
	SF-Inter-Req-016	구성 기기들의 서버로부터 현재 공정에 대한 정보 수신 가능 여부
시험 방법	사용자는 Mobius서버에서 어플리케이션으로 제품의 정보 전달의 유무를 확인하기 위해 Mobius서버를 실행시키고 informationMonitor.js를 실행시켜 어플리케이션이 Mobius서버에 있는 제품의 정보를 잘 수신하는지 확인함	
시험 절차	1. Mininet simulator를 실행하여 가상 호스트 4개를 구성한다. 2. 가상 호스트 Mobius에 Mobius 서버를 실행 3. 가상 호스트 h1~h4에 각각 Thyme을 실행 4. machine01 ~ 04.js를 실행 5. machineMonitor.js를 실행 6. 제품의 정보를 터미널을 통하여 확인	
판정 기준	사용자가 Mobius서버와 어플리케이션을 실행시킨 후, 수신된 제품의 정보 및 공정의 정보가 터미널에 보여진다면 “지원”인 것으로 판단하고, 그렇지 않으면 “미지원”로 판단함	
시험 결과	1.Mininet simulator를 실행하여 가상 호스트 4개를 구성한다. 2. 가상 호스트 Mobius에 Mobius 서버를 실행 3. 가상 호스트 h1에 Thyme을 실행	

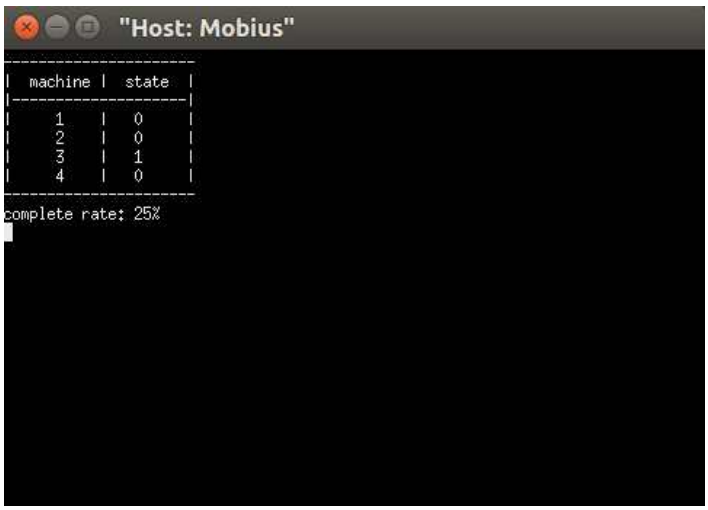


4. machine01 ~ 04.js를 실행



5. 가상 호스트 Mobius에서 machineMonitor.js를 실행

6. 제품의 정보를 터미널을 통하여 확인



판정	지원

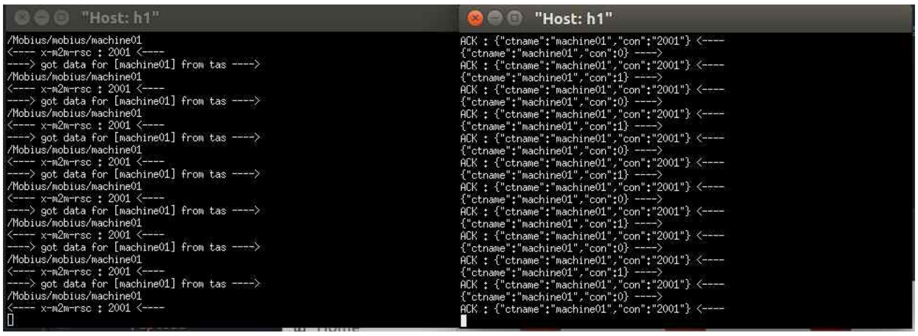
공정 설정변경에 대한 검증이다.

<표 6-32> Lifecycle 테스트케이스 : TC-LFC-02

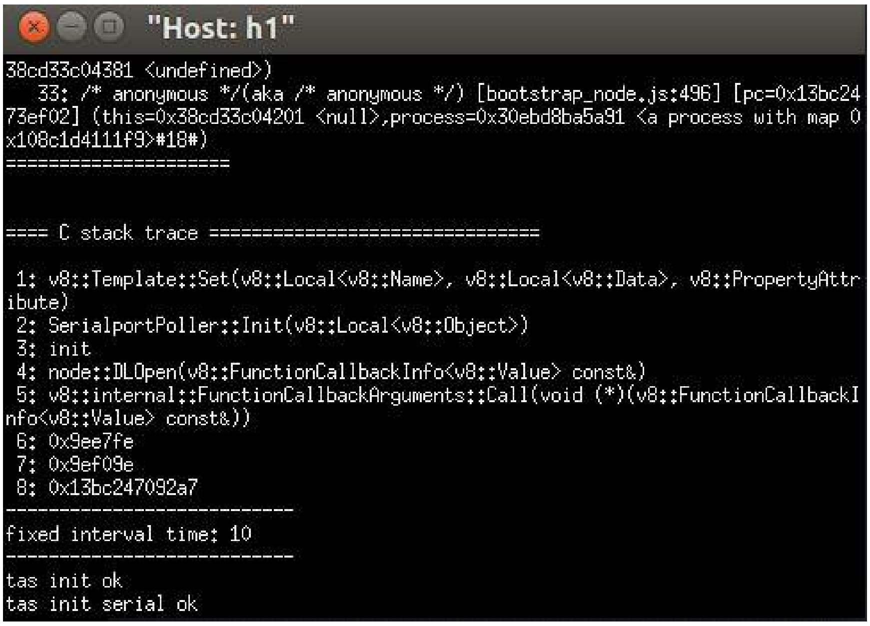
테스트케이스		
TC-LFC-02		
시험항목	Mobius 플랫폼 구성 어플리케이션의 공정 설정 변경 지원 확인	
관련요구사항	SF-Inter-Req-015	관리 서버의 공장 상태에 따른 공정 정보 설정 가능 여부
	SF-Inter-Req-017	구성 기기들의 공정 설정에 따른 작업 지원
	SF-Depend-Req-005	구성 기기들의 네트워크를 통한 원격 설정 가능 여부
	SF-Operate-Req-005	기기의 실제 위치를 고려하여 공정 과정을 설정하는 기능의 가능 여부
	SF-Operate-Req-007	센서를 이용하여 주변 환경을 측정하고 이에 따라 공정 상태를 제어하는 기능의 가능 여부
	SF-Operate-Req-008	제품의 주문량에 따라 유연하게 공정 속도를 제어하는 기능의 가능 여부
시험 방법	사용자는 Postman를 통해서 공정프로세스의 설정을 변경할 수 있는지 확인하기 위해 Postman에서 공정프로세스의 속도를 설정하는 메시지를 보내 그 속도가 변하는지 확인함	
시험 절차	1. Mininet simulator 실행 2. 가상 호스트 Mobius에 Mobius 서버를 실행 3. 가상 호스트 h1에 Thyme을 실행 4. 가상 호스트 h1에 machine01-change.js 5. 현재 machine01의 공정속도를 확인 6. Postman를 통해 machine01에 공정속도를 바꾸는 메시지를 보냄 7. 바뀐 공정속도를 확인	
판정 기준	사용자가 Postman을 이용하여 제어메세지를 보낸 후, cmd창을 통해서 공정속도가 달라지는 것이 확인되면 “지원”, 그렇지 않으면 “미지원”로 판단함	
시험 결과	1. Mininet simulator 실행 2. 가상 호스트 Mobius에 Mobius 서버를 실행	

3. 가상 호스트 h1에 Thyme을 실행

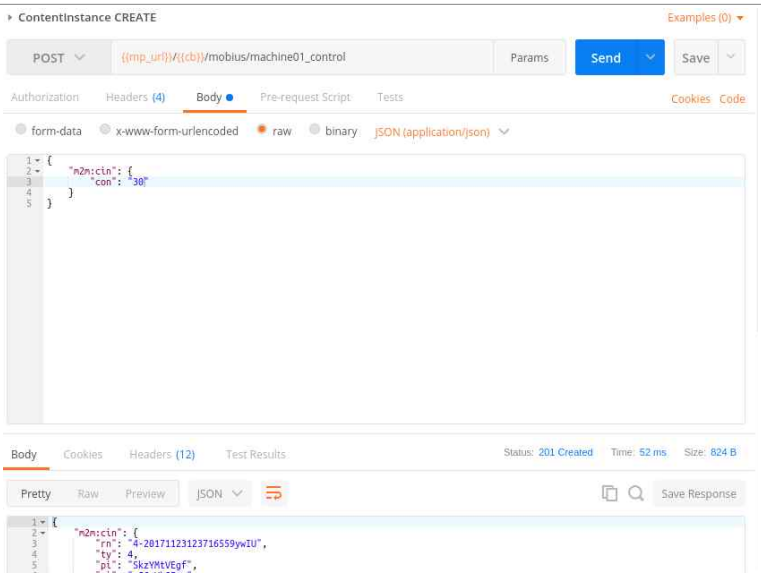
4. 가상 호스트 h1에 machine01-change.js



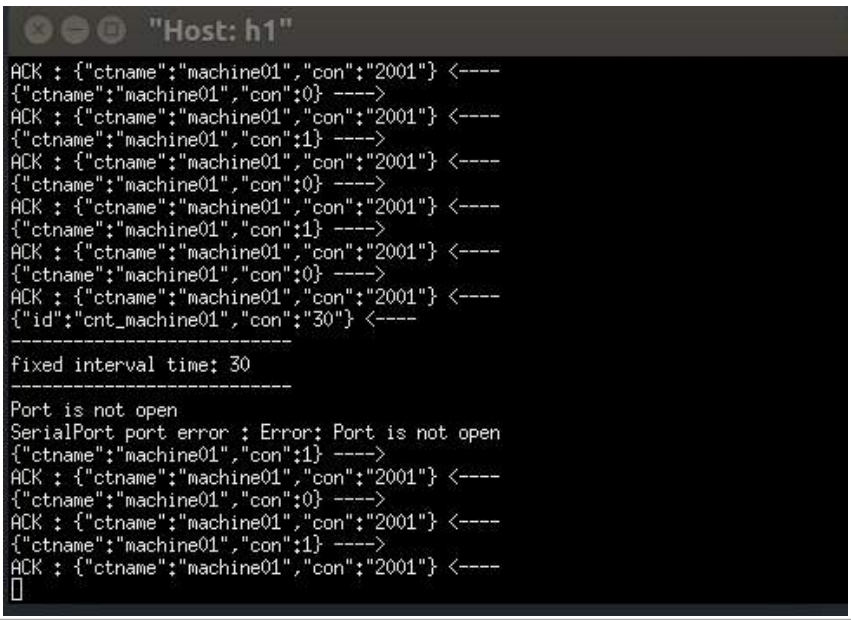
5. 현재 machine01의 공정속도를 확인



6. Postman를 통해 machine01에 공정속도를 바꾸는 메시지를 보냄



7. 바뀐 공정속도를 확인



판정	지원

제4절 대규모 CPS 지원 플랫폼 구성

지금까지 대규모 CPS 시스템구축을 위하여 CPS가 기본적으로 제공해야 하는 CPS 사용자 요구사항을 도출하고, CPS 소프트웨어 특성별로 시스템 요구사항을 분류하였다. 이 장 2절에서 CPS 요구사항을 중심으로 IoT 플랫폼이 지원하는 기능을 도출하고, 3절에서 그에 대한 검증을 시행하였다. 분석 결과 대규모 CPS 구축을 위해서는 IoT 플랫폼이 제공하지 못하는 기능을 CPS 플랫폼에서 추가해야할 필요성이 부각되었다. 이 절에서는 대규모 CPS 플랫폼 지원을 위해 필요한 추가기능을 정의하고 CPS 플랫폼 구조에 대해 제안하도록 하겠다.

1. CPS 특성별 요구사항 검증 결과 분석

1) CPS 확장성

대규모 CPS의 기능적, 성능적, 물리적 규모의 확장성과 관련된 시스템 확장성(Scalability)에 관련된 요구사항의 검증을 통하여 오픈 IoT 플랫폼 Mobius가 구성요소간의 데이터 연동과 이를 위한 다양한 통신 기술 지원 그리고 대규모 시스템에서 일관적으로 메시지가 해석될 수 있는 기능이 지원함을 확인하였다.

하지만 단순 데이터 연동 수준만을 지원하고 있어 네트워크나 도메인에 특화된 시스템에의 확장성 보장에는 한계점이 보이는 것 또한 확인하였다.

(1) 다양한 통신 기술 지원

Mobius 플랫폼이 다양한 통신 기술을 지원하여 스마트공장과 같은 다양한 시스템들이 존재하는 대규모 CPS의 시스템 확장성을 어느 정도 지원하는지 검증하였다.

Mobius 플랫폼에 연결된 설비들을 실시간으로 확인할 수 있는 기능을 명시한 SF-Scal-Req-001은 TC-LFC-01을 통하여 지원됨을 확인하였다. 또한 TC-TRP-01을 통하여 HTTP 프로토콜과 CoAP프로토콜의 설정 가능 여부를 확인하여 SF-SCAL-Req-003에서 요구하는 클라이언트/서버 통신 기술의 지원 여부를 확인할 수 있었고 TC-TRP-02를 통하여 SF-Scal-Req-04 ~ 06에서 요구하는 Publish/Subscribe 통신 프로토콜인 MQTT 지원 여부를 검증하였다.

(2) 온톨로지 기반의 데이터 일관성 보장 기능

연결되어 있는 모든 CPS 서브시스템들에서 동일한 의미를 가지는 메시지들이 동일하게 해석될 수 있도록 하는 기본적인 온톨로지 기능도 지원함을 보았다. TC-SEM-01을 통하여 SF-Scal-Req-010에 명시되어있는 모든 기기에서 메시지를 동일하게 해석할 수 있는 기능이 지원됨을 검증하였다.

(3) 플랫폼 구성요소 연동 기능

대규모 CPS를 구성하는 소프트웨어, 하드웨어와 같은 CPS 서브시스템이 생성되었을 때 이상 없이 Mobius 플랫폼에 연결되는 기능은 TC-MNG-03을 통하여 IoT 플랫폼에서 지원함을 실증하였다.

(4) 확장성 요구사항에 대한 IoT 플랫폼의 지원 한계

대규모 CPS를 구성하는 CPS 서브시스템들이 일부 변경되거나 변동이 생길 경우에 그에 대응하여 네트워크 연결구조나, 제어 시스템, 정보 시스템의 구조 등이 유연하게 변경되어 대처할 수 있는 기능은 일반적인 IoT 플랫폼을 통하여 제공하기 어렵다. 또한 대규모 CPS가 적용될 수 있는 도메인에 특화된 기능 요소는 SF-Scal-Req-011 ~ 012에 명시되어있는데 이러한 CPS 서브시스템의 설정 및 변경에 관련된 요구사항들 또한 기존 IoT 플랫폼에서는 전혀 고려되지 않음을 확인하였다.

일련의 과정을 통하여 오픈 IoT 플랫폼을 통해서는 대규모 CPS에 통신 기능을 제공하여 확장성을 일부 제공하지만 실질적으로 시스템 자체가 확장되거나 변동이 생길 경우에 그를 정밀하게 감지하고 진단하여 시스템에 반영하기 위한 환경 제공에는 한계점이 있음을 확인하였다.

2) CPS 융복합성

오픈 IoT 플랫폼인 Mobius가 대규모 CPS에서는 서브시스템들 간의 융복합성(Composability) 요구사항을 얼마나 만족시키는지 검증하였다.

대규모 CPS를 구성하는 소프트웨어 혹은 하드웨어 추가 제거나 버전 관리, 상태 모니터링

등 복잡하게 얽혀있는 서브시스템들을 관리에 얼마나 용이성을 제공하는지를 체크하고, 복잡하게 얽혀 있는 CPS 서브시스템들의 상태를 쉽게 추적할 수 있도록 각각 서버에 업로드하여 확인할 수 있는 환경이 오픈 IoT 플랫폼에서 제공되는지 확인하였다.

(1) CPS 서브시스템의 상태 정보 확인 기능

Mobius 플랫폼을 통하여 대규모 CPS의 S/W・H/W 서브시스템 상태 확인, 버전 관리 등의 기능을 제공할 수 있음을 TC-MNG-01, TC-MNG-03을 통하여 지원함을 검증하였다.

(2) CPS 서브시스템에 대한 접근 권한 관리 기능

CPS 서브시스템들에 접근하여 확인 및 관리를 위한 권한을 설정할 수 있음 또한 TC-SEC-01을 통하여 확인하였다.

(3) 융복합성 요구사항에 대한 IoT 플랫폼의 지원 한계

Mobius 플랫폼은 대규모 CPS의 변동에 따른 복잡성 변화를 확인하거나 최적화된 설계를 제안 및 변경하는 등의 유연하게 변화하는 시스템을 보조하기 위한 기능은 지원하지 않고 있는 상황이다.

또한, 대규모 CPS에서는 시스템 복잡도 계산, 최적화를 위하여 모델링&시뮬레이션을 거쳐 효과적으로 최적의 설계를 도출해내고 이를 사용자가 효과적으로 확인할 수 있는 환경이 필요하다. 하지만 현 IoT 플랫폼은 이에 대한 부분은 지원하지 않고 있다.

변화하는 시스템에 대한 정보를 사용자가 인지할 수 있는 인터페이스가 원활히 제공될 수 있는 환경이 필요하다는 것이 요구사항 SF-Com-Req-012에 명시되어있지만, Mobius 플랫폼은 이에 대하여 고려하고 있지 않다.

SF-Com-Req-017에서 명시되었듯이 스마트공장와 같은 산업용 대규모 CPS에서는 고객의 요구를 도출해 낼 수 있는 구성요소 구조 등의 도출을 위하여 M&S 등의 추가적인 기능이 필요하지만, Mobius 플랫폼은 IoT 플랫폼이기 때문에 이 역시 고려되지 않았다.

현재 지원되지 않는 CPS의 융복합성 요구사항을 만족시키기 위해서는 기존 IoT 플랫폼의 기능요소 이외에 복잡한 구조를 가지는 대규모 CPS를 사용자가 효과적으로 관리할 수 있는 환경과 최적의 설정을 도출해내기 위한 모델링&시뮬레이션 환경이 반드시 제공되어야 함을

확인하였다.

3) CPS 상호작용성 요구사항 검증 결과 분석

대규모 CPS에서는 서브시스템들 간의 상호작용성(interactivity)이 오픈 IoT 플랫폼인 Mobius 플랫폼을 통해 제공될 수 있는지 검증하였다.

(1) 제어 메시지 송·수신 기능

대규모 CPS에서는 서브시스템들 간의 제어 메시지 송·수신 여부를 Mobius 플랫폼에서 제공하는 통신 기능을 이용하여 확인할 수 있었다.

SF-Inter-Req-001과 CPS-Inter-Req-002에 명시되어 있듯이 Mobius 플랫폼이 현장 기술자로 하여금 관리 서버를 통해서 대규모 CPS를 구성하는 CPS 서브시스템들에게 제어메시지의 송·수신을 제공한다는 것을 TC-TRP-01와 TC-TRP-02를 통해서 확인하였다.

Mobius 플랫폼이 제공하는 통신 기능을 이용하여 SF-Inter-Req-002와 CPS-Inter-Req-001에 명시된 대규모 CPS의 서브시스템 간의 메시지 송·수신 기능의 지원 여부를 TC-TRP-03을 통해 검증하였다.

CPS-Inter-Req-002, CPS-Inter-Req-003, 그리고 CPS-Inter-Req-006에 명시되어 있듯이 IoT 플랫폼은 CPS내에 있는 서브시스템 간의 데이터 송·수신을 위한 적절한 통신기술을 제공한다는 것을 TC-TRP-02, TC-TRP-03, 그리고 TC-TRP-04를 통해 확인했다.

(2) 공정 설정 메시지 관리 기능

Mobius 플랫폼을 통해서 현장 기술자는 서버와 기기 간 공정과 관련된 설정을 변경하고 생산 중인 제품 및 현재 공정에 대한 정보를 수신 할 수 있다는 것을 시험해 보았다.

SF-Inter-Req-007, CPS-Inter-Req-004, CPS-Inter-Req-010, SF-Inter-Req-013와 SF-Inter-Req-016에 명시되어 있듯이 사용자가 생산 중인 제품 및 현재 공정에 대한 내용을 어플리케이션을 통해 사용자가 수신 받을 수 있다는 것을 TC-LFC-01을 통해 확인하였다.

그리고 TC-LFC-02를 통해 다른 공정들의 상황 또는 구성 기기들 간의 공장 상황에 따라 공정 정보를 설정할 수 있다는 것을 확인함으로써 SF-Inter-Req-015와 SF-Inter-Req-017을 검증하였다.

(3) 온톨로지 기반의 일관된 데이터 해석 기능

CPS 서브시스템 사이의 제어 메시지를 송·수신할 때 기기들이 메시지에 대한 적합한 해석 및 표시를 할 수 있는지 여부를 확인해 보았다.

기기들 간의 원활한 메시지 송·수신을 위해 특정 기기가 메시지를 수신했을 때, 보낸 기기에서 의도한 의미와 같은 의미를 해석하는지에 대한 SF-Inter-Req-003, CPS-Inter-Req-004, 그리고 CPS-Inter-Req-005 요구사항을 TC-SEM-01을 이용하여 검증할 수 있었다.

기기들 사이의 제어메시지를 사용자가 확인하기 위하여 특정 기기가 제어 메시지를 받았을 때 받은 메시지를 사용자가 확인할 수 있도록 표시하는 기능을 명시한 SF-Inter-Req-004와 CPS-Inter-Req-004를 제공하는지 TC-MNG-02를 통해 검증하였다.

(4) 상호작용성 요구사항에 대한 IoT 플랫폼의 지원 한계

기기들 간의 실시간 데이터 전송을 제공되지 않아 기기들 간의 상태 및 정보 메시지의 실시간 전송을 지원되지 않기 때문에 SF-Inter-Req-005, CPS-Inter-Req-001, CPS-Inter-Req-010, SF-Inter-Req-006, 그리고 CPS-Inter-Req-004 요구사항들을 만족시킬 수 없음을 확인하였다.

Mobius 플랫폼은 현장 기술자가 구성 기기로 데이터 전송을 지원하지만 실제적인 기기의 조작은 지원하지 않기 때문에 현장 기술자의 구성기기들 조작과 관련된 SF-Inter-Req-008, SF-Inter-Req-009, SF-Inter-Req-010, 그리고 CPS-Inter-Req-008 요구사항들을 고려하지 않고 있었다.

그리고 Mobius 플랫폼은 이상 상황을 감지할 수 없어 이를 서버나 현장 기술자에게 시각화하여 알릴 수 없기 때문에 SF-Inter-Req-011과 SF-Inter-Req-012 역시 검증 하는데 어려움이 있었다.

또한, Mobius 플랫폼의 Container 특성상 컨테이너의 변경이 그 컨테이너와 연관되어 있는 저장된 모든 데이터에게 영향을 주지 못하기 때문에 SF-Inter-Req-014에서 명시한 제품에 대한 정보 갱신은 일부분의 정보만 갱신한다는 점에서 Mobius 플랫폼을 통하여 검증할 수 없었다.

4) CPS 고신뢰성 요구사항 검증 결과 분석

오픈 IoT 플랫폼인 Mobius를 통해서 대규모 CPS에서는 서브시스템들 간의 고신뢰성(Dependability)을 충족시키는지 검증하기 위하여 CPS들 또는 CPS의 서브시스템들에 속해 있는 기기들 간 정보 접근에 따른 보안 체계를 지원하는지 확인하였다.

사용자 인증 여부에 따른 정보 접근 체계를 확인하기 위하여 ACP를 이용하여 검증받은 사용자만 정보 접근을 허용하고 인증 받지 않은 사용자가 정보에 접근할 때 ACCESS_DENIED되는 것을 TC-SEC-01을 통해서 확인하였다.

(1) 메시지 보안 기능

Mobius 플랫폼이 CPS에서 발생하는 메시지에 보안 체계를 제공하는 기능에 대한 유무를 검증하기 위하여 내부에 있는 SemanticDescriptor를 통해서 메시지를 구분할 수 있었다. 이를 통해서 인증되지 않은 사용자가 메시지를 취득하지 못하도록 하는 SF-Depend-Req-016 요구사항을 TC-SEM-01테스트 케이스를 이용하여 확인하였다.

(2) 정확한 원격 설정 기능

네트워크를 통한 기기들의 원격 설정을 할 때, 서로 정확한 설정을 할 수 있도록 지원하는지의 가능 여부를 확인하였다.

SF-Operate-Req-005에 명시되어 있듯이 네트워크를 이용하여 각 기기에게 공정 설정 관련과 같은 제어 메시지를 보냄으로서 기기 간의 정확한 설정을 할 수 있다는 것은 TC-LFC-02를 통해 확인할 수 있었다.

(3) 고신뢰성 요구사항에 대한 IoT 플랫폼의 지원 한계

Mobius 플랫폼은 메시지의 보안에 대한 기능만 제공하고 있어서 시스템에 대한 지능적인 이상 상황 판단, 최적화 및 재설정을 통한 항상성 유지 기능이 부족하고 그에 따른 데이터의 실시간 전송 또한 제공되지 않아 현재 시스템은 Fault-tolerance 등의 기능이 제공되지 않기 때문에 높은 신회성을 만족시킬 수 있는 환경을 제공하지 않는다.

IoT 플랫폼은 서브시스템, 네트워크 및 기기들을 실시간으로 모니터링하고 적합한 환경을 도출할 수 있어야 하고 이상 감지를 예측하여 예지보전을 할 수 있는 기능이 필수적인데

아직 이런 고신뢰성을 가진 기능을 지원하지 않는 것을 확인하였다.

SF-Depend-Req-013, SF-Depend-Req-007, 그리고 CPS-Depend-Req-002에 따르면 긴급한 상황에 따른 데이터 실시간 전송이나 중요성이 높은 작업에 대한 모니터링이 필요하지만 이를 지원하지 못하는 것을 확인하였다.

IoT 플랫폼은 송・수신되는 메시지에 대한 보안체계는 제공하지만 SF-Depend-Req-002에 명시되어 있듯이 네트워크 접근 보안체계는 IoT 플랫폼이 고려하는 보안 대상이 아니라 확인할 수 없었다.

이상 상황 감지 또는 현재 상황을 지능적으로 판단 후 그에 따른 제어로직을 설정하여 환경을 항상 적합하게 유지하는 기능이 필수적인데 IoT 플랫폼은 제어 로직을 구성할 수 없고 또한 제어 로직이 구성되어 실행되더라도 적절하게 실행되었는지 확인하는 기능이 제공되지 않기 때문에 추가적인 기능이 필요하다.

IoT 플랫폼은 적합한 환경을 도출하기 위해 서브시스템들 간의 최적의 연동을 통해 안정적인 상태를 도출해야하는데 이러한 기능 역시 제공하지 않았다.

5) CPS 실시간성 요구사항 검증 결과 분석

Mobius 플랫폼이 대규모 CPS의 실시간성(Timing) 관련 요구사항 중 SF-Timing-Req-006과 CPS-Timing-Req-003에서 드러난 서로 다른 노드 간 시간 동기화의 가능 여부에 대하여는 지원함을 확인했지만 대부분의 요구사항을 만족시키는 데에는 한계점을 보이는 것 또한 확인하였다.

(1) 실시간 시간동기화 기능

Mobius 플랫폼이 플랫폼 내 다른 노드 간 실시간 시간동기화를 확인할 수 있는지 판단하여 실시간성(Timing)을 어느 수준까지 지원하는지 검증하기 위하여 TC-NET-01을 통해 실시간으로 다른 노드 간 시간동기화 여부를 판단하여 SF-Timing-Req-006, CPS-Timing-Req-003의 요구사항을 만족함을 확인하였다.

(2) 실시간성 요구사항에 대한 IoT 플랫폼의 지원 한계

Mobius 플랫폼이 전송시간 및 각 노드의 주파수, 위상 수준의 동기화를 지원하지 못해 노

드 간 최적화된 동기화를 통한 정확한 실시간 서비스 제공이 불가능하다.

이러한 문제점을 해결하기 위하여 외부 스마트공장 및 이기종 시스템과의 연동된 Real-time task를 Mobius 플랫폼을 통해 실현하기 위해서는 실시간 네트워크 기술 및 이기종 시스템의 연동 표준의 적용이 필요하다.

대규모 CPS의 구성 시스템을 높은 수준으로 제어하기 위해 모델링&시뮬레이션 기능 및 시간 예측, 지능요소가 요구되나 Mobius 플랫폼은 단순한 데이터 전달 및 구성 노드 간 연결성만 제공하므로 한계가 존재한다.

6) CPS 상호운용성 요구사항 검증 결과 분석

대규모 CPS가 연동 네트워크 통신과 구성 요소들의 상호운용성(Interoperability) 요구사항을 Mobius 플랫폼이 얼마나 만족하고 있는지 검증하였다.

(1) 산업용 통신기술 지원

Mobius 플랫폼의 다양한 산업용 통신기술을 이용한 메시지 전달 지원 여부를 판단하여 통신 시스템의 이종성을 지원여부를 판단하였다.

TC-TRP-01, TC-TRP-02를 통해 Mobius 플랫폼이 MQTT, COAP 및 HTTP 등의 다양한 산업용 통신 환경에서 기기 설정 및 제어 메시지 전달 관련 요구사항인 SF-Operate-Req-001, SF-Operate-Req-002, SF-Operate-Req-010의 기능을 지원함을 확인하였다.

(2) 이종 CPS 서브시스템들의 상태 확인 기능

CPS 서브시스템 및 그 기기들의 위치와 센서 데이터를 이용하여 대규모 CPS 시스템인 스마트공장의 공정 설정, 공정 상태 및 공정 환경과 같은 서비스를 Mobius 플랫폼을 통해 확인, 제어가 가능할 수 있음을 TC-LFC-02로 검증하였다.

(3) 표준 준수 데이터 전달 기능

CPS 서브시스템에 있는 하드웨어와 소프트웨어의 상태 및 관리를 Mobius 플랫폼이 확인하고 관리할 수 있으며 표준에 준수한 데이터 전달을 지원함을 검증하였다.

Mobius 플랫폼이 스마트공장에 있는 하드웨어의 상태와 관리를 위한 메시지 전달 관련 기

능을 요구사항인 SF-Operate-Req-006을 TC-MNG-02를 통해 지원하고 있음을 보였다.

또한 TC-MNG-03로 CPS의 소프트웨어 요소들이 이종 노드의 적용을 위한 표준을 준수하여 추가적인 설정 변경 없이 Mobius 플랫폼이 다른 노드로의 이식을 지원함을 확인하였다.

(4) 상호운용성 요구사항에 대한 IoT 플랫폼의 지원 한계

CPS 서브시스템들의 센서를 이용하여 현재 상태 데이터에 따른 예기치 않은 동작을 예측하고 지능적인 대처를(요구사항 SF-Operate-Req-003)하는 기능은 Mobius 플랫폼의 지능성 부족으로 인해 제공하는데 어려움이 있다.

대규모 CPS 시스템의 유연한 동작과 사용자의 요구사항에 맞는 설계를 제공하기 위해 Mobius 플랫폼은 단순한 데이터 확인 및 전달 기능에 추가로 CPS 서브시스템 및 네트워크의 상태 판단과 재구성 기능을 지원해야 한다.

SF-Operate-Req-004, CPS-Operate-Req-001, CPS-Operate-Req-003은 다양한 CPS 모듈들과 통신하기 위해 네트워크의 요구사항 만족 여부 확인할 수 있어야 한다.

SF-Operate-Req-009, SF-Operate-Req-011을 위해 대규모 CPS 지원 플랫폼은 CPS 서브시스템들의 연동 구조 및 공정 설계에 관한 정보 확인과 설정을 지원할 수 있어야 한다.

7) CPS 지능성 요구사항 검증 결과 분석

일반적인 Mobius 플랫폼은 빅데이터 및 AI 기술 적용이 고려되지 않아 대규모 CPS의 지능성(Intelligence)관련 요구사항의 검증이 불가능하여 한계점에 대하여만 확인하였다.

(1) 지능성 요구사항에 대한 IoT 플랫폼의 지원 한계

Mobius 플랫폼은 지능적인 인지 및 변동성을 제공하지 않아 CPS의 지능성 관련 요구사항에 명시된 SF-Int-Req-001 ~ 003과 CPS-Int-Req-001 ~ 002의 지원에 어려움이 있다.

Mobius 플랫폼은 대규모 CPS를 구성하는 기기들의 공정 환경 및 상황을 인지하여 그에 따라 네트워크 연결구조, 제어 시스템, 정보 시스템의 구조 등이 유연하게 변경되어 대처되는 기능을 수행하지 못한다.

시스템 상태에 대한 지능적 인지 및 변동성 기능의 부재로 Mobius 플랫폼은 공장 내부의 현재 상황 인지하고, 최적의 상태를 조성 및 공정 또는 기기의 결함을 지능적으로 예측이

불가능하다.

대규모 CPS의 프로세스나 로직과 같은 동작 상황에 대한 정보 인지 및 지능적인 대처에 대해서는 현재의 Mobius 플랫폼이 시스템의 데이터 연동 및 전달만 주로 고려하고 있어 그를 활용하여 이상 상황 파악 및 회복하는 기능은 제공되지 않아 검증하는데 어려움이 있다.

또한 요구사항 SF-Int-Req-005에 명시되어 있듯이 시스템뿐만 아니라 네트워크 및 구성 기기들의 이상 상황을 파악하고 그에 대응하여 회복 가능 여부를 분석 기능을 제공해야 한다.

Mobius 플랫폼은 대규모 CPS를 구성하는 기기들이 생산중인 제품의 구성은 알 수 있지만 그 상태를 인지하는 기능은 제공할 수 없어 요구사항 SF-Int-Req-006을 만족시킬 수 없다.

2. 대규모 CPS 지원을 위한 추가 기능 정의

대규모 CPS 지원 플랫폼을 정의하고자 CPS 소프트웨어 특성별로 추가로 필요한 CPS의 기능에 정리하겠다.

1) CPS 소프트웨어 확장성

시험 검증 결과를 분석 과정에서 IoT 플랫폼이 CPS의 확장성 요구사항 중 데이터 교환 등 기본적인 통신 요구사항은 만족시키지만 시스템의 변동을 감지 및 반영할 수 있는 기능에는 한계점이 있음이 드러나 이를 보완하기 위한 소프트웨어 정의 네트워크, 디지털 트윈과 같은 CPS 플랫폼 추가 기능을 필요로 한다.

(1) 소프트웨어 정의 네트워크 기술을 통한 유연한 네트워크 구성 기능

대규모 CPS를 구성하는 CPS 서브시스템 간의 통신 과정에서 높은 확장성을 가지도록 소프트웨어 기반으로 유연한 네트워크를 설정해주는 기술로서 소프트웨어 정의 네트워크(SDN)가 있다. SDN은 소프트웨어로 네트워크를 유연하게 설정하는 기술로서 변화하는 응용, 트래픽, 네트워크 요구사항에 맞춰 네트워크 구성요소를 최적하여 QoS(Quality of Experience)를 만족시키는 데 사용된다.

본 테스트베드에서도 SDN 환경에서 IoT 플랫폼의 테스트케이스 검증을 진행하였지만 CPS를 고려하지 않은 네트워크 환경으로서 대규모 CPS에서 확장성 요구사항을 만족시키기에는 한계가 존재하는 것을 확인하였고, 대규모 CPS의 네트워크 확장성을 지원하기 위하여 CPS

서브시스템들의 연결 상태와 통신 요구사항들을 인지하여 네트워크 정책을 상황에 따라 유연하게 설정하고 반영하기 위한 기능이 플랫폼에 추가적으로 필요하다.

(2) 다양한 상황을 고려한 설정 및 제어 기능

CPS가 적용된 도메인에 최적화되어 시스템 구조에 변동이 생겼을 때 여러 상황을 고려하여 설정 및 제어할 수 있는 기능을 필요로 한다.

현재의 IoT 플랫폼을 활용할 경우 시스템에서 변동이 생겼을 때 데이터 연결이나, 알림 등의 기능을 통하여 시스템 구조의 변경이 있음을 인지할 수 있지만, 이러한 구조의 변경이 각각의 CPS요소에서 어떠한 영향을 끼치게 될 지는 예상하기가 어렵다.

(3) 디지털 트윈 기술을 이용한 CPS 서브시스템 관리

물리적 자산의 상태를 정밀하게 반영하는 사이버 모델인 디지털 트윈 기술을 통하여 CPS 서브시스템들을 효과적으로 관리할 수 있는 환경이 CPS 플랫폼에 필요하다.

(4) 디지털 트윈 모델을 기반으로 한 모델링&시뮬레이션 환경

CPS의 구조가 변동될 때 다양한 상황을 효과적으로 시험해 볼 수 있는 디지털 트윈 모델을 기반으로 한 모델링&시뮬레이션 환경을 제공하여 다양한 상황을 고려하여 설정 및 제어를 할 수 있어야 한다.

2) CPS 소프트웨어 융복합성

검증 과정을 통하여 IoT 플랫폼을 통해서는 대규모 CPS의 융복합성 요구사항으로 도출된 시스템 최적화에는 어려움이 있음을 확인하였고, 이를 보완하기 위해서는 시스템에 대한 정보를 인지하고 예측할 수 있는 모델링&시뮬레이션 환경의 지원이 추가적으로 필요하다.

(1) 시스템 상태 파악을 위한 환경 제공

대규모 CPS은 복잡한 구조를 가지기 때문에 시스템이 어떠한 상태인지, 각각의 CPS 서브시스템들이 어떠한 연결 구조를 가지는지 사용자가 분석하는 것은 매우 어렵기 때문에 이를 돕기 위한 환경이 CPS 플랫폼에 추가되어야 한다.

(2) 시각화 도구 기능 제공

CPS 서브시스템이 추가되거나 제거될 경우에 이를 사용자가 직접 확인하여 어떠한 영향을 끼치게 될지 분석하는 것은 매우 어렵다. 그러므로 대규모 CPS의 전체 상태를 시각적으로 확인할 수 있는 환경이 제공되어 비효율적인 혹은 의도하지 않은 구조로 작동 될 경우를 쉽게 진단해 낼 수 있어야 한다. 또한, 수집된 데이터들을 통하여 각각의 CPS 서브시스템들이 어떤 구조로 연동되어 어떠한 영향을 끼치는 지에 대한 정보 또한 추적 가능해야하고, 이를 시각화하여 사용자가 활용할 수 있는 환경이 필요하다.

(3) 모델링과 시뮬레이션 기술 제공

복잡한 구조를 가지는 대규모 CPS의 상태를 쉽게 확인하고 이것들이 변동되었을 때 영향을 시험하고 최적의 설정을 도출해 내기 위해서는 모델링과 시뮬레이션 기술을 반드시 필요로 한다. 복합한 구조를 가지는 대규모 CPS의 현재 상태를 효과적으로 인지하기 위해서는 각각의 CPS 서브시스템들의 정밀한 상태를 반영하는 가상 모델인 디지털 트윈 기술을 필요로 하며, 정밀한 디지털 트윈 모델들을 활용한 모델링&시뮬레이션을 통하여 시스템의 상태를 분석할 수 있는 환경이 제공되어야 할 뿐만 아니라 사용자가 최선의 선택을 할 수 있도록 분석 결과를 가시화 할 수 있는 기능 또한 제공되어야 한다.

3) CPS 소프트웨어 상호작용성

검증을 통해 대규모 CPS의 서브시스템과 현장기술자 사이의 통신은 가능했지만 데이터 전송의 실시간성에 어려움을 발견하였고 플랫폼을 통한 현장 기술사의 직접적인 기기 조작, 이상 상황 판단 및 알림, 생산 제품들의 정보 갱신, 그리고 다양한 네트워크와 연동할 수 있는 다른 기술이 추가적으로 필요하다는 것을 확인하였다.

(1) 가상 현실 및 증강현실을 활용한 HMI 기능 지원

현장기술자가 VR 및 AR을 통해 서브시스템들을 조작할 수 있게 인터페이스가 지원해야 하며 실시간 분산 시뮬레이션을 통해 정확하게 서브시스템이 조작되었는지 시각화하여 알려주는 기능이 추가적으로 필요하다.

(2) 다양한 환경에서 이상 상황을 판별하고 적절한 대처를 하는 기능 제공

서브시스템들의 다양한 환경 데이터를 이용하여 디지털 트윈 모델링을 하고 이를 이용하

여 이상 상황판별하고 그에 맞는 시스템 로직을 도출하여 이상 상황에 대처하여 그 결과를 현장기술자에게 알려주는 기능이 추가적으로 필요하다.

(3) 데이터 정보 변경 시 유관한 데이터의 일괄적인 변경 기능 제공

제품 공정 중에 일어날 수 있는 제품의 정보 변경이 있을 때, 이종 모델간 실시간 데이터 연동을 통해 이미 전송된 데이터의 정보까지 일괄적으로 변경이 가능한 기능이 제공되어야 한다.

4) CPS 소프트웨어 고신뢰성

오픈 IoT 플랫폼이 대규모 CPS의 고신뢰성(Dependability)을 지원하기 위해서는 추가적으로 실시간 이상 상황 감지 및 그에 따른 대처, 공정에 따른 제어 로직의 설정 및 실행 여부 확인, 적합한 작업 환경 도출 및 유지, 그리고 안정적인 작동을 위한 메모리 영역 보장의 기능이 제공되어야 한다.

(1) 지능형 시스템 예지보전

대규모 CPS의 각 CPS 서브시스템들을 모델링함으로써 IoT플랫폼이 적합한 기기의 상황을 학습하고 시뮬레이션을 통해 이상상황에 대한 적절한 대응 및 최적의 환경을 도출할 수 있는 기능이 추가되어야 한다.

(2) 실시간성 지원 태스크 스케줄링

시뮬레이션을 통해 메모리 할당 영역을 예측함으로써 최적의 태스크 스케줄링을 구현하여 메모리 부족으로 인한 이상 현상을 방지할 수 있는 RTOS 기능이 추가되어야 한다.

(3) 실시간 상황 판단을 위한 시뮬레이션 환경 구축

여러 IoT 플랫폼들이 가지고 있는 작업 환경 데이터를 실시간으로 공유하고 전반적인 상황을 판단하기 위한 적절한 시뮬레이션 환경의 구축이 필수적이다. 시뮬레이션을 통해 기기들마다 적합한 상황을 도출해야하고, 도출된 적합한 환경을 이용하여 기기별 제어 로직을 만들고 이를 기기별로 설정하여 제대로 실행되고 있는지 확인 할 수 있는 기능이 추가되어야한다.

5) CPS 소프트웨어 실시간성

오픈 IoT 플랫폼은 대규모 CPS의 실시간성(Timing) 요구사항을 만족하기 위해 연동 네트워크가 실시간성을 지원해야 하고 구성요소들이 실시간 네트워크를 활용할 수 있어야 한다.

(1) 실시간성 지원 네트워크 구성

대규모 CPS 시스템이 정확한 실시간 연동 서비스를 수행하기 위해 우선적으로 산업용 통신/네트워크 및 네트워크 인프라 디바이스들이 실시간 네트워크로서 구성되어야 한다. IEEE 802.1 Time-Sensitive Networking task group에서는 Ethernet 네트워크의 보장된 실시간성을 바탕으로 deterministic한 서비스를 제공하기 위해 Time Sensitive Networking이라는 이름하에 시간 동기화 및 패킷 전달 보장 메커니즘을 표준화하고 있다. 최근 산업용 기반 네트워크가 각 산업의 High bandwidth와 Low-latency communication 요구사항을 만족하기 위해 Ethernet기반의 real-time 네트워크로 변경되고 있다.

(2) 실시간 네트워크 연동 서비스

스마트공장 시스템의 공정 설정과 설계 시간을 예측하고 시간 내 메시지 전달 기능을 수행하려면 실시간 네트워크 인프라와 연동된 어플리케이션 설정 및 동작이 필요하다. 생산현장을 정보화 관점에서 볼 때 공정 데이터를 4M(Man, Machine, Material, Method)로 정의할 수 있다. 스마트공장의 공정이 정확하게 실시간으로 동작하기 위해 시스템 내 4M 데이터들이 실시간 네트워크 인프라와 연동되어 데이터가 전달될 수 있도록 시간 이벤트를 활용할 수 있는 판단 기술이 필요하다.

(3) 실시간 분산 시뮬레이션

복잡한 시스템들이 연동된 대규모 CPS는 분산 M&S를 통해 복잡한 구성요소들을 확인 및 제어할 수 있다. 그 연동은 실시간성을 만족해야 결과의 정확성과 신뢰성을 보장할 수 있다. 모델링 & 시뮬레이션의 이기종 시뮬레이터에 의해 생성된 모델을 공유하고 동시에 시뮬레이션하기 위하여 시뮬레이션 환경은 실시간성을 지원해야 한다.

6) CPS 소프트웨어 상호운용성

대규모 CPS의 상호운용성(Interoperability) 요구사항을 만족하기 위해 오픈 IoT 플랫폼은 이기종 서브시스템 및 노드 간 통신을 지원하고 구성 기기들의 원활한 재구성이 가능해야 한다.

(1) 이종 서브시스템간 데이터 공유

대규모 CPS를 구성하는 다양한 모듈들은 어떤 환경에서 작동하는지 여부에 관계없이 CPS 플랫폼을 통해 데이터 공유가 가능해야 한다. 서로 다른 서브시스템 간의 데이터 시멘틱 관리와 그를 기반으로 한 이종 데이터 처리 기능이 플랫폼을 통하여 제공되어야 한다. 또한 이종 시스템간의 효율적인 데이터 공유를 위하여 산업용 통신 기술인 OPC UA나 데이터 중심 미들웨어인 DDS와 같은 기술을 지원해야 한다.

추가로, 모델링&시뮬레이션 과정에서의 상호운용성을 보장하기 위하여 이기종 시뮬레이터 간의 분산 시뮬레이션을 지원하기 위하여 이종 모델간 실시간 데이터 연동 기능이 제공되어야 한다.

(2) 이종 시스템 간 실시간성 지원 분산 태스크 스케줄링

이종 시스템 간의 실시간성 지원 분산 태스크 스케줄링을 통하여 서브시스템들 간의 복잡한 연동 구조의 시스템을 효율적으로 연동 및 스케줄링하여 사용자의 요구사항을 만족시킬 수 있는 환경을 구축할 수 있어야 한다.

7) CPS 소프트웨어 지능성

오픈 IoT 플랫폼은 센싱 데이터 수집 및 관리에 기능이 집중되어 있어 대규모 CPS의 시스템 지능성(Intelligence) 요구사항에 명시된 것들과 같이 지능적인 인지 및 대처 기능에 대한 정의가 이루어져 있지 않다. 대규모 CPS 플랫폼을 위해서는 지능성을 제공하기 위한 추가 기능 정의가 필요하다.

(1) 지능형 네트워크 제어 기능

대규모 CPS를 구성하는 다양한 모듈들은 지능적으로 네트워크를 구축 및 수정할 수 있어야 한다. 이를 위하여 CPS 연동 네트워크가 스스로 장애를 감지해 지능적으로 동작하기 위해서는 SDN/NFV 기반 네트워크에 AI 기법을 적용한다면 지능화된 망 제어 및 관리를 제공할 수 있게 된다.

(2) 지능형 시스템 예지보전

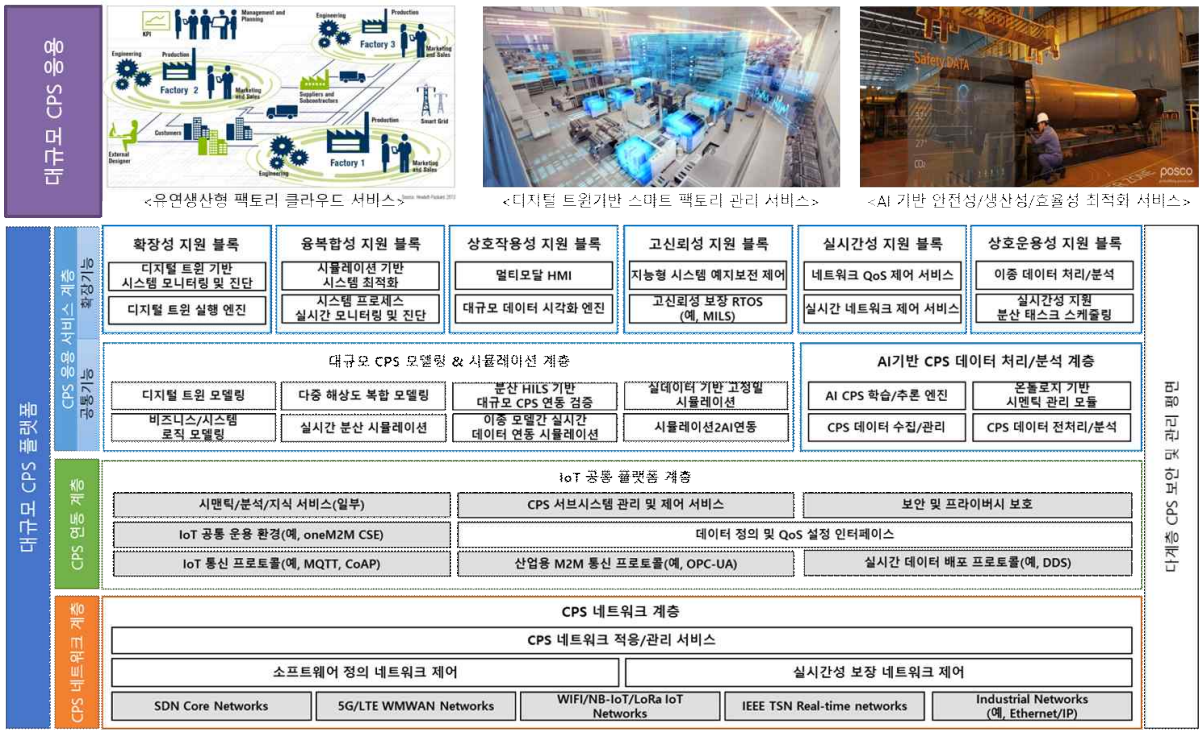
데이터 분석을 통하여 시스템 공정, 기기들의 상태를 지능적으로 예측해 상황에 따라 동작 상태를 변동하여 최적의 상태를 조성하고 제어 하는 기능을 구성하여 지능적으로 시스템 예지보전이 가능하도록 구성되어야 한다. 특히 CPS 시스템 구성요소들의 회복 가능 여부를 판단하고 동작 상태를 지능적으로 관리하기 위해 CPS 플랫폼에 프로세스를 실시간으로 모니터링하고 마이닝 할 수 있는 환경이 제공되어야 한다. 또한 CPS 서브시스템으로부터 수집되는되는 데이터들을 관리 및 분석함으로서 다양한 상황 변화에 유연하고 지능적으로 대처하여 최적의 환경을 제공할 수 있는 기술이 필요하다.

3. 대규모 CPS 지원 IoT 플랫폼 중심 CPS 플랫폼 구조

IoT 플랫폼 상에 CPS 기능 수행가능의 여부 분석 및 CPS 플랫폼 구축을 위하여 추가로 필요한 기능 정의 과정을 통하여 대규모 CPS 구성을 위하여 필요한 기능요소들을 지원할 수 있는 CPS 플랫폼 구조를 구성하였다. 대규모 CPS 지원 IoT 플랫폼 중심 아키텍처는 아래 드림에 나타난 것과 같이 크게 CPS 네트워크 계층, CPS 연결 계층, CPS 서비스 계층으로 나누어진다.

이러한 플랫폼을 활용하여 구성할 수 있는 CPS 응용의 예시들도 포함되어 있다. 유연생산형 팩토리 클라우드 서비스, 디지털 트윈기반 스마트공장 관리 서비스, AI 기반 안전성/생산성/효율성 최적화 서비스 등이 대규모 CPS를 활용하여 제공할 수 있는 대표적인 서비스들이다. 플랫폼을 통한 모델링과 시뮬레이션, 지능형 서비스 기반 제공 등을 통하여 다양한 서비스들을 제공할 수 있게 된다.

[그림 6-13] 대규모 CPS지원 IoT 플랫폼 기반 아키텍처



대규모 CPS 응용의 안정적이고 범용적인 작동 환경을 제공하기 위하여 정의된 플랫폼 계층에 보안성을 제공하기 위하여 네트워크, 서브시스템, 장비 등의 보안성 관리를 위한 다계층 CPS 보안 및 관리 기능이 전 계층을 아우르고 있다.

대규모 CPS를 위한 오픈 IoT 플랫폼 기반 CPS 플랫폼 아키텍처의 각 계층별 자세한 설명은 아래에 이어 설명한다. 그림에서 오픈 IoT플랫폼을 통해 이미 제공되고 있거나, CPS 플랫폼 이외 보편적으로 이미 사용되고 있는 네트워크 계층의 통신 기술(SDN, WIFI 등)은 회색 박스로 표시되어 구분할 수 있다.

1) CPS 서비스 계층

CPS 서비스 계층은 오픈 IoT 플랫폼에서 만족시킬 수 없는 CPS 요구사항들을 지원할 수 있는 기능들로 구성되어 있다. CPS 요구사항들이 공통적으로 필요로 하는 CPS 모델링&시뮬레이션 블록과 CPS 데이터 수집/처리 모듈을 기반으로 하여 요구사항 분류 항목별로 요구사항을 만족시킬 수 있는 기능들이 포함된 블록들이 정의된다.

(1) 대규모 CPS 모델링&시뮬레이션 블록

대규모 CPS 모델링&시뮬레이션(이하 M&S) 블록은 디지털 트윈 모델 기반의 분산 시뮬레이션을 통하여 CPS 시스템을 가상화 할 수 있는 기능을 제공한다. 디지털 트윈 모델링 기능과 시스템 로직 모델링 기능을 통하여 시스템의 구성과 작동 체계를 정의 하고 이를 실시간 분산 시뮬레이션을 통하여 검증 할 수 있는 환경을 제공한다. 또한 정의된 모델들은 목적에 맞추어 여러 해상도를 가지게 할 수 있도록 다중 해상도 모델링 기능도 제공한다.

다양한 환경과 상황에서의 시뮬레이션 검증을 위하여 다중 HILS 기반 대규모 CPS 연동 검증 기능과 이종 모델간 실시간 데이터 연동 기능이 포함된다. 이를 통하여 CPS 시스템에 대한 시뮬레이션을 좀 더 효율적으로 구성할 수 있게 된다.

CPS 시뮬레이션을 통하여 시스템 작동 상태 분석을 위하여 프로세스 마이닝을 통한 실데이터 기반 고정밀 시뮬레이션 기능이 추가적으로 제공되고, 대규모 CPS M&S 블록을 통하여 분석된 데이터와 시뮬레이션 데이터의 지능적인 활용을 위하여 시뮬레이션과 인공지능 엔진간의 연동성을 제공하는 기능이 포함된다.

(2) CPS 데이터 수집/처리 블록

CPS 연동 계층을 통하여 수집된 데이터와 CPS M&S 블록에서 발생된 데이터 등 대규모 CPS에서 발생하는 데이터들을 효과적으로 활용하기 위하여 CPS 데이터 수집/처리 블록이 구성된다.

디지털 트윈 데이터 수집/관리 모듈과 디지털 트윈 데이터 처리/분석 모듈을 통하여 디지털 트윈으로부터 발생된 대규모의 데이터들을 활용할 수 있는 환경이 제공된다.

CPS에 지능적인 서비스를 구성할 수 있도록 인공 지능 학습 엔진을 통한 AI 활용 환경을 CPS 데이터 수집/처리 블록에 구성하고 이종 환경에서 발생되는 데이터들의 의미를 관리하기 위한 온톨로지 기반 시멘틱 관리 모듈을 포함한다.

(3) 확장성 지원 블록

대규모 CPS의 확장성 요구사항을 위하여 정의된 블록으로 CPS M&S 블록에서 정의된 디지털 트윈을 해석하고 실행할 수 있는 엔진을 포함하여 이를 기반으로 하여 시스템 모니터링 및 진단 기능을 제공하여 시스템이 변동되어도 유연하게 대처할 수 있는 확장성을 제공한다.

(4) 융복합성 지원 블록

대규모 CPS의 융복합성 요구사항을 만족시키기 위하여 정의된 블록으로 하위 계층을 통하여 수집된 데이터를 바탕으로 시스템 프로세스의 실시간 모니터링 및 진단 기능을 제공하고, 이상 상황이나 비효율적인 상태가 진단되었을 경우에 CPS M&S 블록을 활용하여 시뮬레이션 기반의 시스템 최적화를 수행할 수 있다.

(5) 상호작용성 지원 블록

대규모 CPS 환경에서 사용자와 CPS 서브시스템 사이의 효율적인 상호작용을 위한 기능들이 포함된 블록으로 수집 및 분석된 대규모 데이터들을 시각화 할 수 있는 엔진을 통하여 데이터를 가시화하고 이를 VR/AR과 같은 다양한 방식의 인터페이스를 제공할 수 있는 멀티모달 HMI(Human-Machine Interface)를 통하여 사용자와 CPS 서브시스템간의 효과적인 상호작용성을 제공한다.

(6) 고신뢰성 지원 블록

대규모 CPS에서 발생되는 이상 상황의 예측과 그에 대한 지능적인 대처를 통하여 전체 시스템의 신뢰성을 제공하기 위한 블록으로 인공 지능 CPS 학습 엔진과 CPS M&S 환경을 모두 활용하여 지능형 시스템 예지보전 기능을 제공한다. 또한 시스템 태스크 스케줄링의 실시간성을 지원하여 태스크 처리의 고신뢰성을 제공하기 위하여 고신뢰성 보장 RTOS 환경이 포함된다.

(7) 실시간성 지원 블록

네트워크 수준의 실시간성을 제공하여 대규모 CPS에서 CPS 서브시스템간의 실시간 연결을 제공하기 위한 블록으로 네트워크 QoS 설정 기능과 TSN 설정 기능을 통하여 CPS 네트워크 계층을 제어하여 실시간성을 보장할 수 있는 네트워크를 구성할 수 있는 기능들을 포함한다.

(8) 상호운용성 지원 블록

대규모 CPS를 구성하는 이종 시스템간의 상호 운용성을 보장하기 위한 기능들이 제공되

는 블록으로 온톨로지 기반 시멘틱 관리 모듈로부터 정의된 이종 데이터들을 처리할 수 있는 기능을 제공한다. 또한 각 CPS 서브시스템간의 실시간성을 보장하기 위하여 태스크 스케줄링 기능을 제공한다.

2) CPS 연동 계층

CPS 연동 계층은 회색으로 표시된 오픈 IoT 플랫폼의 기능을 중심으로 CPS 서브시스템들의 데이터 수집 및 관리 분배가 이루어진다. CPS 서브시스템으로부터 발생하는 센싱 데이터들을 수집하기 위하여 MQTT, CoAP과 같은 오픈 IoT 플랫폼에서 지원하고 있는 IoT 통신 프로토콜을 사용한다. CPS 서브시스템 간의 효율적인 데이터 교환을 위하여 주고받을 데이터 정의 및 QoS를 설정하여 메시지를 주고 받을 수 있는 OPC UA와 같은 산업용 M2M 통신 프로토콜과 DDS(Data Distribution Service)와 같은 통신 미들웨어 기능이 구성된다.

이러한 IoT, IIoT 통신 기술을 상위 계층에서 활용할 수 있도록 수집된 데이터의 일부 시멘틱/분석 서비스가 제공되고, CPS 서브시스템의 관리 및 제어를 할 수 있도록 한다. 또한 수집된 CPS 서브시스템 데이터에 대한 보안 체계도 구성된다.

3) CPS 네트워크 계층

CPS 응용 서비스 계층의 실시간성 지원 블록으로부터 설정된 네트워크 요구사항을 반영하기 위하여 소프트웨어 정의 네트워크 제어 기능과 실시간성 지원 네트워크 제어 기능을 제공한다. 네트워크 QoS를 반영할 수 있도록 데이터 링크와 네트워크를 소프트웨어 정의 네트워크 제어 모듈을 통하여 설정하고, 실시간성을 제공하기 위하여 설정된 TSN 설정을 실시간성 지원 네트워크 인터페이스를 통하여 실제 Time Sensetive Network에 적용할수 있도록 한다. 또한 SDN과 TSN 이외에도 다양한 네트워크 IoT 센서 환경에서 다양하게 사용되는 5G/LTE WMWAN 네트워크와 WIFI/NB-IoT/LoRa IoT, 그리고 산업 현장에서 널리 사용되는 Ethernet/IP 등의 기능 또한 포함되어야 한다.

제7장 결론

제1절 연구의 요약

본 연구에서는 첫째, NIST PWG에서 정의한 CPS 분석 프레임워크를 기초로 자동차, 항공기 및 스마트공장 분야에서의 CPS의 역할과 사용자 요구사항을 정의하였다. CPS 분석 프레임워크에 따라 식별된 사용자 요구사항을 CPS 특성을 분석 틀로 사용하여 시스템 요구사항을 도출하였다. CPS 특성은 문헌 조사를 통하여 확장성, 융복합성, 상호작용성, 고신뢰성, 실시간성, 상호운용성과 지능성으로 규정하였다.

특히, 이종의 CPS 산업 간의 공통적인 기술적 요구사항을 도출하였고, 이는 기존 R&D에서 전혀 이루어지지 않은 시도였으며 유의미한 결과를 획득하였다. 시스템 확장성에 속하는 공통 시스템 요구사항은 통신 관련 기능이 대부분이었다. 스마트공장의 경우는 공장 도메인 기능에 속하는 공정 관련 사항과 공장 구성요소 변경에 관한 요구사항이 많아 공통 기능에는 포함하지 않았다. 융복합성, 고신뢰성, 상호운용성에 속하는 자동차 요구사항은 자동차 표준인 AUTOSAR에 관련된 것이 많았으며, 스마트공장의 경우도 도메인에 관련된 요구사항이 많았다. 지능성에 포함된 요구사항은 도메인에 속하는 요구사항이 많이 공통의 요구사항으로 도출하기는 어려운 면이 있다. 상호작용성의 경우는 통신에 관련된 일반적인 요구사항이 많아 공통요구사항으로 도출되었다.

둘째, 대규모 CPS 플랫폼 적용을 위한 기존 플랫폼 테스트 및 검증을 수행하였다. 검증대상 요구사항은 도출된 CPS 공통 시스템 요구사항과 스마트공장 시스템 요구사항으로 한정하였다. 검증대상 CPS 시스템 요구사항과 선정된 개방형 IoT 플랫폼이 제공하는 기능을 비교, 검토하여 최종적으로 검증이 가능한 CPS 시스템 요구사항을 선정했다. CPS 시스템 요구사항 중 CPS 특성별로 확장성 7종, 융복합성 6종, 상호작용성 12종, 고신뢰성 3종, 실시간성 1종 그리고 상호운용성 8종이 도출되어 총 37종의 요구사항이 검증 대상으로 식별되었고 지능적 요구사항은 AI와 같은 지능형 서비스와 관련된 요구사항들이 많아 IoT 플랫폼을 통한 검증에는 어려움이 있어 검증 대상에서 배제되었다.

CPS 시스템 요구사항을 IoT 플랫폼을 활용하여 기능 검증을 수행하였고, 검증결과 37종의 검증 대상 CPS 요구사항들은 IoT 플랫폼에서 구현이 가능함을 확인하였다. 결과적으로 대규모 CPS 플랫폼으로 확장 시 기존 개방형 IoT 플랫폼에 추가적으로 요구되는 기능을 도출하여 대규모 CPS 플랫폼 구조를 정의하였다. CPS 플랫폼에서 추가적으로 제공해야 하는 기

능들은 CPS 서브시스템의 유연한 변경을 위한 상황 파악 및 제어 기능, 모델링과 시뮬레이션을 통한 최적화 기능, CPS 서브시스템의 복잡성을 추적가능하게 하는 시각화 기능, 실시간성이 보장되는 오류 검출 및 이상상황 대응 기능, AI 등을 이용한 지능형 서비스 등이다.

CPS SW 플랫폼을 서비스, 연동, 네트워크 계층으로 구분하고, 서비스 계층은 시스템 확장성, 융복합성, 상호작용성, 고신뢰성, 실시간성, 상호운용성, 지능성의 CPS 특성을 지원하는 기능과 모델링과 시뮬레이션, 지능형 데이터 처리/분석 기능으로 구성하였다. 연동 계층은 검증 결과에서도 확인이 되었듯이 대부분 IoT 플랫폼의 기능을 사용하여 구성할 수 있다. 네트워크 계층의 요소는 IoT 플랫폼에서 제공하는 기능에 추가하여 실시간성을 보장하는 기능이 추가되어야 함을 발견했다.

이 연구에서는 CPS에서 필요한 요구사항을 CPS 플랫폼 형태로 구성하여 종합적인 CPS 시스템 요구사항을 정리하였으며, 각각의 요구사항 리스트를 제공함으로써 구체적인 요구사항을 정의하였다. 향후에 새로운 CPS의 체계적 연구개발의 수행과 응용 도메인에서 CPS를 개발하고자 하는 기업·연구소에서 핵심 참고자료로서 본 연구 결과물을 활용할 수 있다면 많은 시간과 노력을 절감할 수 있을 것이며, 보다 완성되고 신뢰성이 있는 CPS 개발이 이루어 질 것이라 사료된다. 더불어, 정부에서 미래 CPS 기술개발 로드맵을 설정하고 R&D 기획을 수행함에도 많은 기여를 할 것으로 기대한다. 이로써 본 연구가 국가의 미래 기술경쟁력 확보에 크게 일조할 것이라 믿는 바이다.

제2절 향후 연구

이 연구는 국내에서 시행된 CPS 공통 요구사항에 대한 최초의 연구로 자동차, 항공기, 스마트공장에 대한 요구사항을 도출한 후 CPS 공통의 요구사항 도출을 시도하였다. CPS의 응용도메인은 다양하고 특징이 달라, CPS 공통의 요구사항 도출이 쉽지 않았으나, CPS 라는 개념의 시스템이 요구하는 최소한의 기능을 정리하였다는 점에서 의미가 있다고 하겠다.

추출한 요구사항을 기반으로 CPS 플랫폼 구조를 정의하였으며, 향후 연구에서는 플랫폼을 구현하고, IoT 플랫폼과 그를 기반으로 하는 CPS 플랫폼의 연동이 유연하게 되어 요구하는 기능을 실현할 수 있을지 검증이 필요하리라 본다.

향 후 연구에서는 응용도메인별로 특징적인 요구사항을 구체화하는 연구가 필요하겠으며, 이 연구에서 정의한 수준별 CPS 단계에 따른 요구사항 도출 후 시스템에 적용하는 방안 마련이 필요하겠다.

제3절 연구의 한계

이 연구는 CPS 소프트웨어 요구사항만으로 제한하여, 소프트웨어 요구사항 도출에 중점을 두었으며, CPS 구현을 위해서는 CPS 하드웨어, 물리시스템과 CPS 소프트웨어 간의 상호작용도 고려하여 전체적인 요구사항 도출이 필요하다.

CPS 응용도메인의 다양성으로 인하여 본 연구는 자동차, 항공기, 스마트공장의 3가지 도메인으로 한정하였으며, 스마트도시, 스마트헬스 등의 도메인으로 확장하여 요구사항관련 연구를 수행하는 것이 요구된다.

CPS 응용 도메인의 한 분야인 자동차 도메인의 자율주행차나, 항공기 도메인의 무인 항공기의 경우도 발전 단계에 있어서 CPS 시스템을 위한 요구사항 도출 및 상세화에 대한 연구가 시스템 발전 정도에 따라 계속적으로 진행되어야 하겠다.

특히 CPS의 지능성에 대한 특징은 인공지능의 발전 정도에 따라 새롭고 다양한 요구사항이 도출 될 것으로 생각된다.

참 고 문 헌

국내 문헌

강남희 (2015), 사물인터넷 융합 서비스 보안 요구사항, 한국통신학회지(정보와통신), pp.45-50.

고영준, 김용진 (2015), IoT 상호운용성 및 서비스 개발, 한국통신학회지(정보와통신), pp.31-35.

김기형(2014), oneM2M 사물인터넷 서비스 플랫폼 표준화 현황

김다빈, 김경모, 조진성 (2015), IoT 보안 요구사항에 대한 고찰, 한국정보과학회 학술발표논문집, pp.1072-1074.

김동희, 윤석웅, 이용필 (2013), IoT 서비스를 위한 보안, 한국통신학회지(정보와통신), pp.53-59.

김상수, 인호, 최순황, 박수용 (2006), 국방 소프트웨어의 가치창출을 위한 요구공학 방법론, 정보과학회지, pp.5-13.

김원태 외(2010), CPS 기술동향, 정보통신산업진흥원 주간기술동향 통권 1455호, 2010.7.21.

김원태, 전인걸, 이수형, 박정민(2013), 대규모 자율제어 CPS 소프트웨어 플랫폼 기술, 정보과학회지 31(12), 2013.12., pp16-28

김원태(2015), 하이브리드 모델기반 고신뢰 CPS(Cyber-Physical Systems) SW 개발 환경, 2015.12

김재호 (2017), 『Mobius 2.0 IoT플랫폼 오픈소스 공개』, KETI.

김태열 외(2017), 소프트웨어 공학백서, 정보통신산업진흥원

문영식, 최성필, 이은규 외 2인 (2013), IoT 기반 컨테이너 보안 장치 및 시스템 성능 평가, 한국정보통신학회논문지, pp.2183-2190.

미래창조과학부(2017), 제품서비스 개발을 위한 IoT와 oneM2M의 이해

민경석(2013), 사물인터넷(Internet of Things), 한국인터넷진흥원

박수진, 박수용 (2011), 임베디드 시스템을 위한 그레이-박스 기반의 소프트웨어 요구사항 명세 기법, 정보과학회논문지, pp.485-490.

박수홍 (2016), 『OCF 사물인터넷 오픈소스 플랫폼 표준 동향』, TTA저널, vol. 166, pp. 41-45.

박종만(2015), 중소기업 스마트공장 기술 동향과 이슈, The Journal of Korean Institute of Communications and Information Sciences, Vol.40 No.12, 2015.12.

변정원, 류성열, 김진수 (2012), 사례 기반의 요구사항 정형화 및 선정 평가 기법, 한국컴퓨터정보학회논문지, pp.149-161.

손동환(2014), 항공기 시스템의 SW를 위한 표준, TTA 저널 Vol 154, 2014.7.8.

손상혁(2016), 융합의 또 다른 이름, 사이버물리시스템, 지식의 지평

손영성, 『IoT 플랫폼 기술 동향 [OCF]』, IoTF 사물인터넷포럼

스마트공장 사업소개(2015.12.), 스마트공장추진단, 2017년 산업기술 R&BD 전략

안미경, 박남춘 (2017), IoT 환경에서 상호운용성(Interusability) 개선을 위한 사물개성(Personality of Things) 모델링 방법에 대한 연구, 한국디자인학회 학술발표대회 논문집, pp.74-75.

원명규, 장덕우, 장정훈 외(2014), 사이버물리시스템 기반의 스마트 시티 기술, 한국통신학회지, 31(8), 2014.7.,pp45-53.

원명규 외(2013), 사이버물리시스템의 현재와 미래: 응용 어플리케이션 관점에서의 접근, 한국통신학회지(정보와 통신) 30(10), 2013.9. pp62-69.

윤주상, 이태진 (2016), IoT 환경에서 서비스 계층 상호운용성 연구, 한국컴퓨터정보학회 학술발표논문집, pp.77-78.

이강희, Web 2.0기반 유비쿼터스 소프트웨어 로봇 플랫폼의 구현, Journal of The Korea Society of Computer and Information July 2012
이상기, 이세윤, 김정철 (2016), A Study on Security Vulnerability Management in Electric Power Industry IoT, 한국디지털콘텐츠학회 논문지, pp.499-507.
이재기, 남궁한, 정연서 (2008), 요구사항의 도출과 우선 순위화, 대한전자공학회 학술대회, pp.115-120.
장승주(2016), 자율주행 자동차 관련 SW기술 동향, 한국통신학회지(정보와통신), 33(4),27-33
정보통신단체 표준(2015.12.), ICT 제조융합 스마트공장 참조 모델
정보통신산업진흥원(2017), SW공학백서 2017
정용식, 차재상 (2017), IoT 디바이스 보안 점검 기준, 한국통신학회지(정보와통신), pp.27-33.
조충기, 융희용(2015), 오픈소스 클라우드 컴퓨팅 플랫폼 분석 및 비교, 한국컴퓨터정보학회 동계학술대회 논문집 제23권 제1호,2015.1.
진회승(2017), 스마트공장 성공을 위한 소프트웨어 역할과 과제, SPRI 이슈리포트
최정은, 최순규, 이선아 (2002), 소프트웨어 요구사항 관리 사례 연구, 한국정보과학회 학술발표논문집, pp.445-447.
칼 위거스, 조이비티(2017), Software Requirements 3, 모든 프로젝트 이해관계자가 알아야 할 요구공학의 정석과 실천법, 2017.
한국전자통신연구원 (2014), WAVE 기반 차량 안전 및 C-ITS 기술 동향, 자동차IT융합연구실 융합 기술 연구부문.
한국전자통신연구원 (2016.), 가상-실공장 연동을 위한 CPS 기반 스마트팩토리 기술, 2016.11.
한국정보통신기술협회(2011), 임베디드 SW정의 및 분류 가이드라인, 정보통신단체표준(TTAS)
한국정보화진흥원(2016), 스마트시티 발전전망과 한국의 경쟁력, 2016.11.07.
허신욱, 김호원 (2017), 사물인터넷 보안 요구사항과 oneM2M 표준 보안 기술 분석, 정보과학회지, pp.16-22.
황재영, 안종관 외 3인 (2017), 시맨틱 기술을 활용한 글로벌 사물인터넷 상호연동 기술 개발 및 적용, 한국통신학회논문지, pp.2208-2216.
Ian Sommerville(2016), 소프트웨어 공학 제10판

해외 문헌

acatech(2015), Integrated research agenda Cyber physical systems, 2015.3.
Ahmad T. Al-Hammouri(2012), A comprehensive co-simulation platform for cyber-physical systems, Computer Communications 36 (2012) 8–19
Angelo Corsaro (2016), “DDS & OPC UA”, PRISMTECH.
AUTOSAR Standard (2016), “Foundation Release Overview”, AUTOSAR FO Release 1.0.0.
AUTOSAR Standard (2016), “Requirements on AUTOSAR Features”, AUTOSAR CP Release 4.3.0.
A. Al-Fuqaha, M. Guizani, M. Mohammadi et all.(2015), Internet of things: A survey on enabling technologies, protocols and applications, IEEE Communications Communications Surveys & Tutorials, 2015.6.
Carliss Baldwin, C. Jason Woodard(2008), The Architecture of Platforms: A Unified View, Harvard Business School white paper
Carlos Álvarez, Eduard Ayguadé, Jaume Bosch et al(2016), The AXIOM software layers, Microprocessors

and Microsystems 47 (2016), pp262-277
Chen Nan (2017), 『oneM2M Browser User Guide_v1.2_EN』, KETI.
Cristóbal Costa-Soria(2014), Software Engineering for Cyber-Physical Systems, EECA Cluster networking event, 2014.11.12.
C.S. Yu, R. Fiske, S.M.Park, and W.T.Kim (2012), "Many-to-One Communication Protocol for Wireless Sensor Networks", International Journal of Sensor Networks, Vol.12 Issue 3, pp. 160-170.
C.W.Min, H.K.Jum. W.T.Kim, and Y.I.Eom (2012), "Scalable Cache-Optimized Concurrent FIFO Queue for Multicore Architecture", IEICE Trans. on Information and Systems, Vol.E95-D, No.12, pp.2956-2957.
David Chappell(2011), What is an application platform?, 2011.12.
Di Paolo Emilio, Maurizio(2015), Embedded Systems Design for High-Speed Data Acquisition and Control, Springer
Dr. Karsten Schweichhart(2016), Reference Architectural Model Industrie 4.0 (RAMI 4.0) An introduction
Edward A. Lee(2006), Cyber-Physical Systems - Are Computing Foundations Adequate?, Position Paper in NSF Workshop On Cyber-Physical Systems: Research Motivation, Techniques and Roadmap
Edward A. Lee(2008), Cyber Physical Systems: Design Challenges, Object Oriented Real-Time Distributed Computing (ISORC), 2008 11th IEEE International Symposium on
Fieldbus Neutral Reference Architecture for Condition Monitoring in Factory Automation , 2014.4.
Frederick P. Brooks(1986), No Silver Bullet: Essence and Accidents of Software Engineering
G Karsai(2003), Model-Integrated Development of Embedded Software, Proceedings of the IEEE, 2003.01.
Helen Gill(2010), Cyber-Physical Systems: Beyond ES, SNs, and SCADA, Presentation in the Trusted Computing in Embedded Systems (TCES) Workshop, 2010.6.25.
H. Kim, S.H. Lee, et. al. (2011), "A Structure of Fast Discovery in DDS Middleware", Proc. of 6th International Symposium on Embedded Technology.
Ian Sommerville(2011), Software Engineering 9th edition
IDC(2016), Worldwide Embedded and Intelligent Systems Forecast, 2016-2020, 2016.05.
IEEE-CS Seminar(2008), ARINC 653, Courtesy of Wind River Inc., 2008.6.4.
IEEE(2015), Towards a definition of the Internet of Things, 2015.05.
ISO/IEC/IEEE 29148(2011), Systems and software engineering-Life cycle processes-Requirements engineering, 2011.
I. Chun, J. Park, W. Kim, W. Kang, H. Lee, S. Park (2010), "Autonomic Computing Technologies for Cyber-Physical Systems," Proc. of 12th Interational Conference on Advanced Communication Technology, Journal of Embedded Systems, Vol. 2, pp.1009-1014.
Jack Ganssle(2008), Mike Barr, Embedded Systems Dictionary
Jay Lee 외(2014), A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems, 2014.12.
Jesús Angel García Sánchez, Jorge J. Rodríguez Vázquez(2014), Open CPS Platforms, CPS20: CPS 20 years from now visions and challenges workshop, 2014.4.
John A. Stankovic(2014), Research directions for the internet of things, IEEE Internet of Things Journal, vol. 1, no. 1, 2014.2.

J. H. Shi, J. F. Wan, H. H Yan and H. Suo, A survey of cyber-physical systems, in Proc. of the Int. Conf. on Wireless Communications and Signal Processing, November 9-11, 2011.

J. Jeon, I.Chun, and W.Kim (2012), "Metamodel-Based CPS Modeling Tool," Lecture Notes in Electrical Engineering Volume 181, pp. 285-291.

J. M. Park, S.J. Kang, I.G. Chun, and W.T. Kim (2013), "An Approach to Generating Goal Model for Cyber-Physical Systems," In Proc. International Symposium on Embedded Technology, Daegu, Korea, pp. 23-24.

J. R. Kim, K.T. Kim, S.H. Lee, and J.W. Kim (2011), "Performance Evaluation of Reliable Real-Time Data Distribution for UAV-aided Tactical Networks", CA-CPS 2011.

Kevin Ashton(2009), That ' Internet of Things' thing, RFID Journal

Loucopoulos, P. & Potts, C. (Ed.)(1996), Requirements Engineering Journal. Springer Verlag, 1996.

Md. Masudur Rahman and Naushin Nower(2017), Requirements Model for Cyber-Physical System,

M. W. Alford(1977), A Requirements Engineering Methodology for Real-Time Processing Requirements, IEEE Transactions on Software Engineering, 1977.

NIST CPS PWS(2016), Framework for Cyber-Physical Systems Release 1.0, 2016.5.

NITRD(2012), CPS Vision Statement

OCF SPECIFICATION INTRODUCTION AND OVERVIEW, OCF, July 2017

Object Management Group (2007), "Data Distribution Service for Real-time System Ver. 1.2".

Object Management Group (2009), "The Real-time Publish-Subscribe Wire Protocol, DDS Interoperability Wire Protocol Specification Ver. 2.1".

Paolo Bellavista, et all.(2014), SOFTWARE ENGINEERING Key Enabler for Innovation, NESSI White Paper, 2014.07.

PCAST(2007), Leadership Under Challenge: Information Technology R&D in a Competitive World

PCAST(2010), Designing a Digital Future: Federally Funded Research and Development in Networking and Information Technology, 2010.12.26.

Peter Marwedel(2010), Embedded System Design, Springer, 2nd Edition

P. Bourque et all.(2014), Guide to the Software Engineering Body of Knowledge, Version 3.0, IEEE Computer Society, 2014.

Ragunathan(RAJ) Rajkumar, Insup Lee, Lui Sha, and et all.(2010), Cyber-physical systems: The next computing revolution, in Proc. of 47th IEEE/ACM Design Automation Conf., pp. 731-736.

Rajkumar et al (2016), Cyber-Physical Systems: The Next Computing Revolution, Design Automation Conference (DAC), 47th ACM/IEEE, 2010.6.

Robert Oshana & Mark Kraeling(2013), Software Engineering for embedded systems

Stefan Wiesner et all.(2016), Requirements Engineering for Cyber-Physical Systems Challenges in the Context of "Industrie 4.0", HAL archives, 2016.10.26.

Teade Punter(2002), Evaluating Evolutionary Systems, Springer, 2002.12.

Thomas J. Burke(2013), OPC Connectivity, OPC Foundation, 2013.10.23.

Volkan Gunes et all(2014), A Survey on Concepts, Applications, and Challenges in Cyber-Physical Systems, KSII TRANSACTIONS ON INTERNET AND INFORMATION SYSTEMS, 2014.12.

World Economic Forum(2016), The Future of Jobs, Employment, Skills and Workforce Strategy for the Fourth Industrial Revolution pp6~8, 2016.1.

Y.Y.Kim, S.H.Lee, W.T.Kim, and S.M.Park (2011), “Design of QoS Framework for Effectively Supporting DDS“, Proc. of 6th International Symposium on Embedded Technology.

기 타

스마트공장 추진단 홈페이지, https://www.smart-factory.kr/main.do.
미교통국 도로교통안전국 (NHTSA, National Highway Traffic Safety Administration), https://www.nhtsa.gov/.
위키피디아, https://ko.wikipedia.org/wiki/AUTOSAR.
artemis 홈페이지, https://artemis-ia.eu/.
oneM2M 홈페이지, http://www.onem2m.org/.
oneM2M, “Published Specifications”, http://onem2m.org/technical/published-documents.
oneM2M, “Draft Release 3 Technical Specifications”, http://onem2m.org/technical/latest-drafts.
OPC Foundation, https://opcfoundation.org/.
Open Connectivity Foundation (OCF), “OCF SPECIFICATION 1.0”, https://openconnectivity.org/developer/specifications.
Open Connectivity Foundation (OCF), “REFERENCE IMPLEMENTATION”, https://openconnectivity.org/developer/reference-implementation.
OPC Foundation, “Specifications”, https://opcfoundation.org/developer-tools/certification-specifications.
http://ir.interdigital.com/file/Index?KeyFile=390952300.
http://k-smartcity.kr/index.php.
http://mysnu.org/community/newtechnology.php?search_order=&search_part=&c_cate1=&mode=v&idx=11050&thisPageNum=.
http://news.inews24.com/php/news_view.php?g_serial=1034874&g_menu=022100&rrf=nv.
http://news.joins.com/article/17992244.
http://www.almanac-project.eu/news.php.
http://www.axiom-project.eu/.
http://www.edaily.co.kr/news/news_detail.asp?newsId=03781846615867912&mediaCodeNo=257&OutLnkChk=Y.
http://www.motie.go.kr/motiee/presse/press2/bbs/bbsView.do?bbs_seq_n=159206&bbs_cd_n=81, 2017.3.
http://www.yonhapnews.co.kr/bulletin/2017/02/16/0200000000AKR20170216065100009.HTML?input=1195m.
http://www.zdnet.co.kr/news/news_view.asp?artice_id=20171226180336&type=det&re=.
https://byline.network/2017/02/1-564/.
https://cps-vo.org/.
https://onetransport.io/.
https://worldindustrialreporter.com/harman-partners-with-interdigital-on-onem2m-compliant-end-to-end-iot-solutions/.
https://www.eurocps.org/eurocps-platforms/.
https://itea3.org/project/flex4apps.html.
https://www.nsf.gov/funding/pgm_summ.jsp?pims_id=503286.
https://www.prosysopc.com/blog/opc-day-europe-2016/.

주 의

1. 이 보고서는 소프트웨어정책연구소에서 수행한 연구보고서입니다.
2. 이 보고서의 내용을 발표할 때에는 반드시 소프트웨어정책연구소에서 수행한 연구결과임을 밝혀야 합니다.

93000
9 788961 084055
ISBN 978-89-6108-405-5