ISSN 2383-630X(Print) / ISSN 2383-6296(Online)
Journal of KIISE, Vol. 51, No. 5, pp. 414-425, 2024. 5
https://doi.org/10.5626/JOK.2024.51.5.414

멀티 인스턴스를 갖는 가상물리시스템의 통합 테스트를 위한 테스트 환경 개발

(The Development of a Test Environment for Integrated Test of Cyber-Physical Systems with Multiple Instances)

허 윤 아 [†] 정 세 진 [††] 유 준 범 [†††]
(Yoona Heo) (Sejin Jung) (Junbeom Yoo)

요 약 가상물리시스템은 다른 여러 시스템, 주변환경 또는 사용자와 밀접한 연관 관계를 가지며, 경우에 따라 특정 목적을 달성하기 위해 실행 시간 동안 멀티 인스턴스가 상호작용하는 다중 구성 환경에서 동작한다. 이 경우 시스템 구성의 변화에 따라 변경되는 여러 개발 산출물 및 위험 분석 결과와 그 연결 관계의 변화에 따른 테스트 케이스의 변경이 중요하게 분석되어야 한다. 멀티 인스턴스를 갖는 다중 구조의 가상물리시스템에 대한 통합 테스트를 위한 연구를 위해서는 실제 환경과 유사한 테스트 환경을 구축하는 것이 필요하다. 특히, 가상물리시스템은 가상 세계와 물리 세계가 통합하여 작동하는 시스템이므로 물리적 환경에 대한 문제도 포함시킬 필요가 있다. 본 논문에서는 이를 위해 개발한 두 종류의 시스템과 각 시스템의 멀티 인스턴스로 구성된 테스트 환경에 대해 소개한다. 개발한 테스트 환경은 통합 테스트 케이스의 생성 및 검증 등의 활동에 이용할 수 있다.

키워드: 가상물리시스템, 멀티 인스턴스, 다중 구성 시스템, 테스트 환경

Abstract Cyber-physical systems (CPSs) are closely related to other multiple systems, environments, or users. In some cases, they operate in multiple configuration environments where multiple instances interact during execution time to achieve certain objectives. In such cases, various development artifacts and hazard analysis results that change as the system configuration changes and the test cases due to changes in the connection relationship should be significantly analyzed. To study how to conduct integrated tests for multi-structured CPS with multiple instances, it is necessary to establish a test environment similar to the actual environment. Especially, since CPSs integrate the cyber and physical world, it is necessary to include problems in the physical environments. This paper introduces a test environment that consists of two types of systems and multiple instances of each system. A developed test environment can be utilized for tasks such as generating and verifying integrated test cases.

Keywords: cyber-physical system, multi-instances, multiple configuration system, test environment

· 이 논문은 2022학년도 건국대학교의 연구년교원 지원에 의하여 연구되었음

† 학생회원 : 건국대학교 컴퓨터공학과 학생
hyoona1202@naver.com
†† 정 회 원 : 진주교육대학교 컴퓨터교육과 교수
jsjj0728@gmail.com
††† 정 회 원 : 건국대학교 컴퓨터공학과 교수(Konkuk Univ.)
jbyoo@konkuk.ac.kr
(Corresponding author임)

논문접수 : 2023년 3월 31일
(Received 31 March 2023)
논문수정 : 2023년 12월 11일
(Revised 11 December 2023)
심사완료 : 2024년 2월 6일
(Accepted 6 February 2024)

정보과학회논문지 제51권 제5호(2024. 5)

1. 서 론

가상물리시스템은 가상 세계와 현실 세계를 연결하고 통합하여 동작하는 개념의 시스템으로 자동차, 교통, 의료 등의 다양한 분야에서 사용되고 있다[1,2]. 가상물리시스템은 다른 여러 시스템, 주변 환경, 또는 사용자와 밀접한 연관 관계를 갖고 동작하며, 경우에 따라서는 그 목적을 달성하기 위해 다른 시스템과 상호작용 또는 협력하여 동작하거나[3], 실행시간 동안 한 가상물리시스템의 여러 멀티 인스턴스 간에 서로 협력하기도 한다[4]. 시스템 간, 혹은 한 시스템의 여러 인스턴스 간의 협력 이외에도 시스템의 추가, 제거, 및 변경 등으로 인해 구성 시스템이 변화할 수 있어 여러 불확실성 (uncertainty)이 나타나는데, 이를 시스템의 설계 과정에서 모두 예상하는 것에는 어려움이 있다[5,6].

위와 같은 특성을 가지는 가상물리시스템의 테스팅 또는 위험/안전성 분석에는 개별 시스템은 물론 주변의 시스템 및 각 시스템의 멀티 인스턴스, 멀티 오브젝트 등을 고려할 필요가 있다[4,5]. 시스템 이론[7]에 따르면 시스템은 단순한 부분의 합으로 고려될 수 없으므로, System of Systems (SoS)와 같이 여러 시스템으로 구성되는 시스템의 테스팅 또는 위험 분석은 통합적으로 고려되고 적용될 필요가 있다. 가상물리시스템 또한 주변의 시스템 및 각 시스템의 멀티 인스턴스의 상호작용과 같은 특성을 고려하여 통합적으로 테스팅과 위험 분석을 수행해야 한다.

기존에 제안된 다양한 테스트베드들이 존재하지만[5, 8-13], 여러 서로 다른 시스템 및 각 산출물의 다양한 관계 분석을 포함하는 것은 다소 어려운 상황이다. 따라서, 멀티 인스턴스를 갖는 다중 구조의 가상물리시스템에 대한 통합 테스트 (integrated test)를 위하여 테스트 환경을 구축하는 것이 필요하다. 또한, 센서 또는 네트워크와 같은 물리적인 환경 또한 고려해야 한다는 가상물리시스템의 특성에 따라 이러한 문제에 대해서도 포함할 필요가 있다.

본 논문에서는 테스트 환경을 구성하기 위해 가상의 두 시스템과 각 시스템의 여러 인스턴스로 구성된 테스트 환경에 대해 소개한다. 개발한 테스트 환경은 오픈소스 하드웨어인 Arduino [14]와 여러 모듈, 각 모듈을 제어할 수 있도록 개발된 컨트롤러 SW, 그리고 각 시스템의 현재 상태를 확인할 수 있는 모니터링 SW를 포함한다. 추가로, 통합 테스트 케이스의 생성에 활용할 수 있도록 각 산출물의 관계 분석[15]을 완료해 각 관계들을 확인할 수 있도록 하였다. 이를 통해 테스트 환경에 대한 여러 종류의 관계 모델을 확인하고 추출할 수 있으며, 통합 테스트 케이스 생성, 통합 위험 분석 등의 연구에 이용할 수 있다.

본 논문의 구성은 다음과 같다. 2장에서는 본 논문에서 제안하는 테스트 환경에 대한 요구사항과 overview, 그리고 테스트 환경의 구현에 대해 소개한다. 3장에서는 테스트 환경에 대한 여러 개발 산출물의 관계 분석 결과와 테스트 환경을 이용한 두 가지 실험과 통합 테스트 케이스 생성, 그리고 향후 활용 방안에 대해 소개한다. 4장에서는 테스트베드 및 테스트 환경에 관한 관련 연구를 소개하고, 마지막으로 5장에서는 결론 및 향후 연구를 설명한다.

2. 테스트 환경 구축

이 장에서는 개발한 테스트 환경에 대한 overview 및 요구사항과 테스트 환경의 개발, 그리고 테스트 환경에 대한 관계 분석을 서술한다.

2.1 테스트 환경 Overview

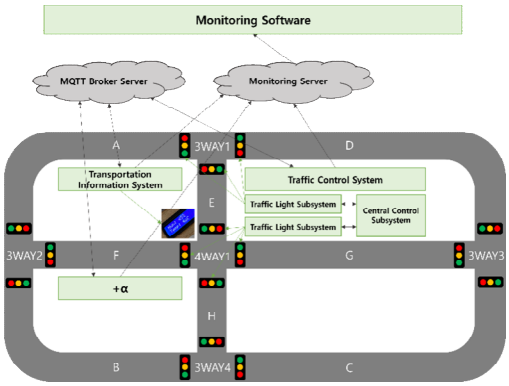


그림 1 테스트 환경 요약 화면
Fig. 1 Overview of the test environment

표 1 시스템 구성 요소
Table 1 Components of the systems

Communication component	
MQTT broker server	x 1
Monitoring SW server	x 1
Monitoring SW system	x 1
Wireless router	x 1
Transportation Information System	
Arduino D1	x 1
LCD module	x 1
Flame Sensor	x 1
DHT Sensor	x 1
Dust Sensor	x 1
Traffic Control System - Central Control Subsystem	
Arduino D1	x 1
RF module	x 1
Traffic Control System - Traffic Light Subsystem	
Arduino D1	x 1
Arduino Uno	x 2
RF module	x 1
3 color LED	x 4 (or x 3)

본 논문에서 제안하는 테스트 환경은 스마트시티[17]의 일부인, 축소된 형태의 지능형도로교통시스템 (Intelligent Transportation System, ITS) [18]으로 이해할 수 있으며, <그림 1>과 같은 구조를 가진다. 이 중 교통정보시스템 (Transportation Information System, TIS)은 총 5개의 인스턴스로 구성되어 가상 도로의 서로 다른 교차로에 배치된다. 신호체계관제시스템 (Traffic Control System, TCS)은 두 종류의 서브시스템으로 구성되며, 중앙통제 서브시스템 (Central Control Subsystem)을 통해 총 5개의 인스턴스를 가지는 신호등 서브시스템 (Traffic Light Subsystem)을 제어한다. 사거리 및 삼거리의 신호등 서브시스템이 각각 1개, 4개로 구성되어 각 교차로에 배치된다. <표 1>은 <그림 1>에 나타난 테스트 환경 중 각 시스템의 컨트롤러 SW를 제외한 구성 요소를 나타낸 표이다.

교통정보시스템과 신호체계관제시스템을 연결하기 위하여 IoT를 위한 통신 프로토콜인 Message Queuing Telemetry Transport (MQTT) 프로토콜[19]을 이용한 중앙의 정보 교환용 브로커 서버가 존재하고, 모니터링을 위한 서버와 모니터링 데이터를 출력하기 위한 SW를 포함하는 모니터링 시스템이 있다. 각 시스템이 서버와의 통신에 사용하는 WiFi 네트워크 통신은 무선 공유기 (Wireless router)를 통해 이루어진다. MQTT 프로토콜을 이용한 통신은 브로커 서버가 메시지의 발행 (publish)과 구독 (subscribe)을 관리하며 한 IoT 기기에서 발행한 메시지를 해당 토픽 (topic)을 구독하고 있는 다른 IoT기기에 전달하는 방식으로, 본 논문에서는 오픈소스 MQTT 브로커인 Eclipse Mosquitto™ [20]를 이용하여 브로커 서버를 구현한다.

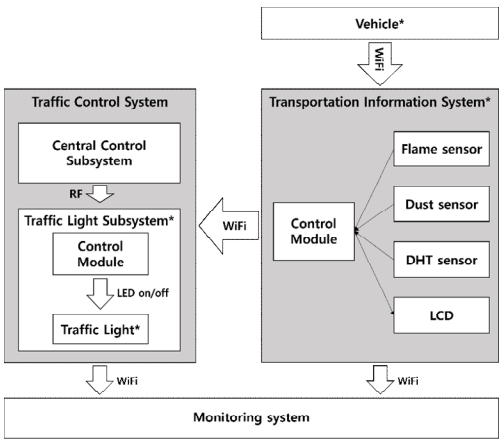


그림 2 테스트 환경의 전체 시스템 아키텍처

Fig. 2 The overall system architecture of the test environment

전체 테스트 환경의 시스템 아키텍처를 구성하면, 다음의 <그림 2>와 같다. 중앙통제 서브시스템은 RF 통신[21]을 통해 여러 개의 신호등 서브시스템을, 각 신호등 서브시스템은 여러 개의 신호등을 제어한다. 제어 모듈은 Arduino와 그 위에 올라가는 컨트롤러 SW를 포함한다.

교통정보시스템은 LCD로 도로 상황에 대한 데이터를 출력하기 위하여 환경 데이터를 수집한다. 이때 수집하는 환경 데이터는 센서 데이터와 WiFi를 통해 제공받는 차량 데이터로, 센서 데이터는 온습도 (Digital Humidity and Temperature, DHT) 센서 값, 불꽃감지 센서 값, 그리고 미세먼지 센서 값을 포함한다. 교통정보시스템은 수집한 데이터를 통해 긴급 상황이 도로 위에 발생했는지 아닌지 판단한다. 판단을 마치고 나면, 긴급 상황이 아닌 경우 LCD에 센서를 통해 수집한 날씨 데이터를 출력하고, 긴급 상황인 경우 긴급 상황을 알리는 메시지를 출력한다. 또한, 모든 데이터를 MQTT 서버와 모니터링 서버로 주기적으로 전송한다.

신호체계관제시스템 중 중앙통제 서브시스템은 교통상황에 따라 교통 흐름이 원활할 수 있도록 각 신호등 서브시스템을 수동으로 제어하는 것을 목표로 한다. 이때 입력 받는 값은 MQTT 서버를 통해 입력 받은 메시지와 관리자로부터의 수동 제어 입력이다. MQTT 서버 또는 관리자로부터 입력이 들어오면, 이를 처리하여 해당하는 신호등 서브시스템을 제어하기 위한 메시지를 RF 모듈을 통해 전송한다. 그리고 각 제어 메시지를 WiFi 네트워크를 통해 모니터링 서버로 전송한다. 신호체계관제시스템 중 신호등 서브시스템은 각 교차로에 배치된 신호등의 LED를 점/소등한다. 중앙통제 서브시스템으로부터 수동 입력을 받으면, 이를 처리하여 알맞게 동작하고, 아무런 입력도 들어오지 않으면 기본 규칙에 의해 동작한다. 신호등 서브시스템은 신호 정보를 주기적으로 모니터링 서버로 전송한다.

추가로, 본 테스트 환경의 전반적인 상황 및 입/출력에 대해 기록하고 확인하기 위한 방안으로 모니터링 소프트웨어 및 서버를 구현한다. 모니터링은 일종의 로그 추적 방식의 형태로 각 시스템의 현재 상태, 센서, LED의 상태 값, 통신 내용 등을 전달받아 출력할 수 있다. 각 시스템은 WiFi 네트워크 통신을 통해 모니터링 서버와 연결되고, 모니터링 소프트웨어는 모니터링 서버로 들어온 데이터를 출력한다.

2.2 테스트 환경 구성 시스템 요구사항

제안하는 테스트 환경의 구성 시스템인 교통정보시스템과 신호체계관제시스템의 시스템 수준에서의 요구사항들을 정리하기 위해 작성한 PFR (Preliminary Functional Requirements) 문서에서 나타나는 시스템의 주요 기능은 다음과 같다.

▪ 교통정보시스템

- 빙결, 화재, 침수 등의 긴급 상황 또는 교통 혼잡이 발생하는 경우 이 정보를 습득하고 처리하여 LCD에 출력하고 외부 시스템에 이 데이터를 제공한다.
- 긴급 상황 또는 교통 혼잡이 발생하지 않으면 센서 데이터를 종합하여 날씨 정보를 제공한다.

▪ 신호체계관제시스템

- 관리자로부터 수동 제어 명령을 입력 받으면 이에 따라 수동으로 신호등을 제어한다.
- 외부 시스템으로 긴급 상황 정보 또는 교통 혼잡 정보를 제공받으면 이에 따라 수동으로 신호등을 제어한다.
- 외부 시스템으로부터의 입력이 없으면 기본 규칙에 의해 신호등을 제어한다.

표 2 각 시스템의 소프트웨어 수준의 기능적 요구사항
Table 2 SW-level functional requirements of each system

Transportation Information System	
TIS-REQ-1	Connect to WiFi network
TIS-REQ-2	Connect to monitoring server
TIS-REQ-3	Connect to MQTT server
TIS-REQ-4	Receive data from MQTT server
TIS-REQ-5	Transmit data to MQTT server
TIS-REQ-6	Transmit data to monitoring server
TIS-REQ-7	Collect sensor data
TIS-REQ-8	Process collected sensor data
TIS-REQ-9	Decide traffic congestion level
TIS-REQ-10	Determine detour path
TIS-REQ-11	Print weather data through LCD
TIS-REQ-12	Print emergency data or congestion data through LCD
Traffic Control System - Central Control Subsystem	
TCS-C-REQ-1	Connect to WiFi network
TCS-C-REQ-2	Connect to MQTT server
TCS-C-REQ-3	Receive external data from MQTT server
TCS-C-REQ-4	Receive manual control from operator through serial
TCS-C-REQ-5	Connect to Traffic Light Subsystems through RF
TCS-C-REQ-6	Transmit control data through RF
TCS-C-REQ-7	Process external data or manual control
TCS-C-REQ-8	Control Traffic Light Subsystem manually
Traffic Control System - Traffic Light Subsystem	
TCS-T-REQ-1	Connect to WiFi network
TCS-T-REQ-2	Connect to monitoring server through WiFi
TCS-T-REQ-3	Transmit current signal data to monitoring server
TCS-T-REQ-4	Connect to Central Control Subsystem through RF
TCS-T-REQ-5	Receive control data through RF
TCS-T-REQ-6	Check current signal status periodically
TCS-T-REQ-7	Control signal (Normal mode)
TCS-T-REQ-8	Control signal (Manual mode)

PFR 문서에서 정의한 시스템 수준의 요구사항을 만족하기 위한 컨트롤러 SW의 요구사항은 SRS (Software Requirements Specification) 문서로 작성되었다. 각 시스템의 컨트롤러 SW의 기능적 요구사항의 요약은 <표 2>와 같다. 각 요구사항은 1:1로 SRS 문서의 기능적 요구사항과 연결되며, 이 외에 통신을 위한 인터페이스 등 추가적인 정보와 멀티 인스턴스 개념을 포함하는 것으로 인한 특징은 SRS 문서에 상세하게 기술하였다.

2.3 테스트 환경 개발 (Implementations)

2.3.1 HW 구현

<그림 3>은 본 논문에서 제안하는 테스트 환경의 시스템 구성에 대한 그림으로 각 시스템은 <표 1>의 구성 요소에 따라 오픈소스 HW인 Arduino와 각종 센서, LCD 모듈, 3색 신호등 LED 모듈, RF 통신 모듈 등으로 구성된 구조를 가진다. 현재 각 교차로에 교통정보시스템과 신호등 서브시스템이 하나씩 설치되어 있다. RF 통신에는 nRF24L01 모듈을, WiFi 통신에는 Arduino D1 R1 보드에 내장된 ESP8266 WiFi 모듈을 사용한다. 신호등 서브시스템의 경우 Arduino D1 보드에 두 개의 Arduino Uno 보드가 SoftwareSerial 통신 [22]으로 연결되어 동작한다.

<그림 4>는 교통정보시스템과 신호체계관제시스템의 서브시스템인 신호등 서브시스템의 구성에 대한 사진으로 <표 1>에서 나타난 바와 같이 여러 보드와 센서, 3색 신호등 LED 모듈, LCD 모듈, 그리고 RF 통신 모듈로 구성된 것을 확인할 수 있다. 이 중 빨간 테두리로 표시된 신호등 LED 모듈과 RF 모듈이 신호등 서브시스템에 해당하고, 노란 테두리로 표시된 세 가지 센서와 LCD 모듈이 교통정보시스템의 구성을 나타낸다.

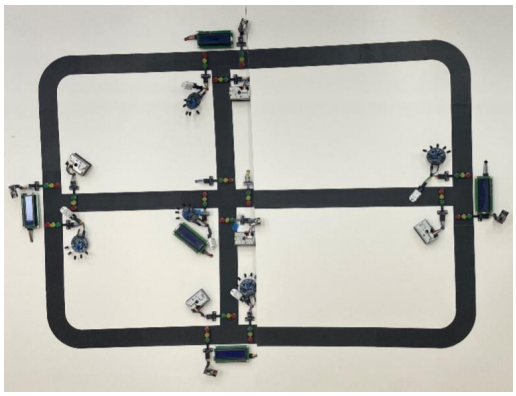


그림 3 시스템 구성 HW 전체 화면
Fig. 3 A photograph of the overall compositions of HW for the whole system

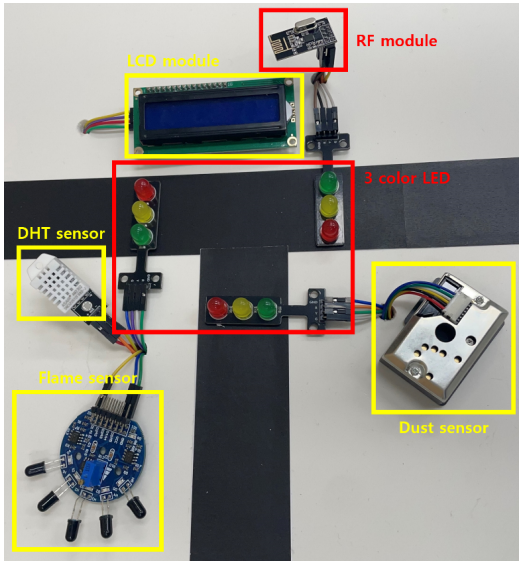


그림 4 교통정보시스템 및 신호등 서브시스템 구성
Fig. 4 A configuration of transportation information system and traffic light subsystem

2.3.2 SW 구현

본 논문에서 테스트 환경 개발을 위해 개발한 SW는 총 4종류로 교통정보시스템의 컨트롤러 SW, 신호체계 관제시스템의 서브시스템인 중앙통제 서브시스템과 신호등 서브시스템의 컨트롤러 SW, 그리고 모니터링 서버가 수집한 데이터를 화면에 나타내기 위한 모니터링 SW가 있다. 모니터링 시스템을 제외한 나머지 구성 시스템의 컨트롤러 SW는 SW 기본 생명 주기 (SW Development Life Cycle, SDLC) [23]에 따라 개발되었다.

개발 산출물 중 SW 요구사항 문서인 SRS 문서는 작성 가이드 표준인 ISO/IEC/IEEE 29148:2018 [24]을 바탕으로 작성되었으며, 구조적 분석 방법론을 통해 top-down 방식으로 요구사항을 분석하였다. 각 시스템의 컨트롤러 SW는 일종의 임베디드 시스템 SW로 입력-제어-출력의 기본 형태로 구성되며 본 논문에서는 이에 대한 요구사항을 두 수준으로 분석하였다. High-level 요구사항은 시스템 수준의 요구사항으로부터 도출한 컨트롤러 SW의 기본 요구사항이며, low-level 요구사항은 high-level 요구사항을 data flow diagram (DFD) [25]과 state transition diagram (STD) [26]을 이용해 분석한 요구사항이다. SW 설계 문서인 SDS (Software Design Specification)는 작성 가이드 표준인 IEEE 1016:2009 [27]을 바탕으로 하여 작성되었다. 해당 문서에서는 low-level 요구사항에서 분석한 DFD에 따라서 structured chart를 그리고 인터페이스와 데이터 등을 분석하였다. 구현에는 Arduino를 위한 C 언어 [28]를 활용하였으며, 분석한 요구사항 및 설계에 맞게 작성하였다. 모니터링 SW는 Java 언어를 사용하여 별도로 개발되었으며, 테스트 환경에 모니터링의 필요성이 생기면서 편입되었다.

<알고리즘 1>을 통해서는 교통정보시스템의 컨트롤러 SW의 동작을, <알고리즘 2>와 <알고리즘 3>을 통해서는 각각 중앙통제 서브시스템과 신호등 서브시스템의 컨트롤러 SW의 동작을 확인할 수 있다. 각 알고리즘은 앞서 <표 2>에 나타낸 각 시스템의 컨트롤러 SW의 기능적 요구사항을 만족한다.

<알고리즘 1>은 교통정보시스템의 컨트롤러 SW 알고리즘으로, 그 동작을 설명하면 다음과 같다. 우선, WiFi 네트워크와 각 서버에 연결한다. 이후 연결에 성공하면 무한 반복문을 계속 돌며 1초마다 센서 값과 차량으로부터 들어오는 데이터를 분석하여 emergency E의 값을 확인한다. 만일 E의 값이 참이면 우회 도로 D 값을 분석하고 R 값과 M 값을 업데이트한다. E의 값이 거짓인 경우에는 도로의 날씨 정보를 분석하여 해당 값을 LCD에 출력할 텍스트로 변경하고 R 값과 M 값을 업데이트한다. 업데이트한 R 값과 M 값은 각 서버로 전송된다. E 값이 참인 경우 별도의 루프 안에서 매 10초마다 flag F 값을 토대로 긴급 데이터를 출력할 지 우회 경로를 출력할 지 판단하여 동작한다.

<알고리즘 2>를 통해서는 중앙통제 서브시스템의 컨트롤러 SW 알고리즘을 살펴볼 수 있으며, 동작을 설명하면 다음과 같다. 우선, WiFi 네트워크를 통해 MQTT 서버에, RF 통신을 통해 각 신호등 서브시스템과 연결한다. 이후 무한 반복문 안을 계속 돌며 관리자로부터의 I를 입력 받거나 MQTT 서버를 통해 R 값을 입력 받으면 각 교차로의 신호등 서브시스템을 제어하기 위한 C 값을 생성한다. 또한, 1초마다 생성된 C 값을 RF 통신을 통해 전송한다.

마지막으로 <알고리즘 3>을 통해서는 신호등 서브시스템의 컨트롤러 SW 알고리즘을 확인할 수 있고, 동작은 다음과 같다. 우선, WiFi 네트워크를 통해 모니터링 서버에, RF 통신을 통해 중앙통제 서브시스템에 연결한다. 이후 무한 반복문 내부를 돌며 중앙통제 서브시스템으로부터 C를 입력 받으면 C에 맞게, 만일 입력 받지 않으면 기본 신호 제어 규칙에 따라 신호등을 제어하기 위해 S_C를 신호등 제어 보드로 SoftwareSerial 통신을 통해 전송한다. 또한, 신호등을 직접 제어하는 것이 아니므로 신호등이 잘 제어되고 있는지 확인하기 위해 1초마다 신호등 제어 보드에 S_S를 요청한다. 이후 S_S를 모니터링 서버로 전송한다.

알고리즘 1. 교통정보시스템의컨트롤러소프트웨어알고리즘
Algorithm 1. Algorithm for controller SW of Transportation Information System

Input: vehicle data V, Dust/Flame/DHT sensor data S_D,S_F,S_{DHT}
Output: road condition R, monitoring data M, text data T_{LCD}

```
Connect to WiFi network
Connect to MQTT server & monitoring server
while True do
   Collect S_D, S_F, S_DHT, V, and T
   for every 1 second
      for each S_D, S_F, S_DHT, and V
         Analyze S_D, S_F, S_DHT, and V
         Check value of emergency E
      end for
      if E == True then
         Analyze detour path D
         R = emergency data
         M = "Restriction"
      else
         Analyze weather data W with S_D, S_F, S_DHT
         Change T_LCD into W
         R = normal data
         M = "Normal"
      end if
      Send R to MQTT server
      Send M to monitoring server
   end for
   if E == True then
      for every 10 seconds
         if flag F == True then
            Change T_LCD into emergency data
            F = False
         else
            Change T_LCD into D
            F = True
         end if
      end for
   end if
end while
```

알고리즘 2. 중앙통제서브시스템의컨트롤러소프트웨어알고리즘
Algorithm 2. Algorithm for controller SW of Central Control Subsystem

Input: road condition R, operator input I
Output: Signal control msg C_A, C_B, C_C, C_D, C_E

```
Connect to WiFi network & RF
Connect to MQTT server
while True do
   if I != "" then
      Generate C_A, C_B, C_C, C_D, C_E according to I
   else if R != "" then
      Parse R and generate C_A, C_B, C_C, C_D, C_E
   end if
   for every 1 sec
      if C_A, C_B, C_C, C_D, C_E!= "" then
         Send C_A, C_B, C_C, C_D, C_E through RF
      end if
   end for
end while
```

알고리즘 3. 신호등서브시스템의컨트롤러소프트웨어알고리즘
Algorithm 3. Algorithm for controller SW of Traffic Light Subsystem

Input: Control msg C, Current signal status S_S
Output: Signal control S_C

```
Connect to WiFi network & RF
Connect to monitoring server
while True do
   if C != "" then
      Generate S_C to control each Traffic light
      according to C
   else
      Generate S_C to control each Traffic light
      according to a basic rule
   end if
   for every 1 sec
      Request S_S through SoftwareSerial communication
      Send S_S to monitoring server
   end for
end while
```

2.4 관계 분석

본 논문에서는 개발한 테스트 환경을 이용해 다중 시스템에서 나타날 수 있는 여러 상황을 분석하고 검증하여 테스트 케이스 생성 시에 활용할 수 있도록 가상물리시스템의 추적성을 포함한 관계 분석 방법 및 모델[15,16]을 사용하여 관계 분석을 수행하였다. 이는 여러 상호작용이 존재하는 다중 시스템의 산출물 간의 계층 관계, 상호작용 관계 등 다양한 산출물 간의 관계들을 한 번에 확인하고 관리하기 위해 제안된 방법으로 본 논문에서 개발한 시스템의 산출물의 계층 관계는 <그림 5>와 같다. 각 시스템은 시스템의 PFR 문서와 컨트롤러 SW의 high-level SRS, low-level SRS, SDS 문서를 개발산출물로 가지며, 시스템의 컴포넌트로 HW 모듈을 포함한다.

관계 분석을 통해 SW의 요구사항 추적성 분석을 포함하여 다중 구성 시스템에서 나타날 수 있는 각 산출물의 여러 계층과 연결 관계를 link type으로 지정해 분

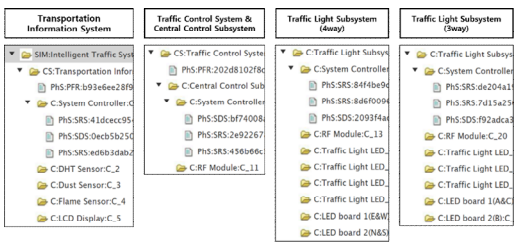
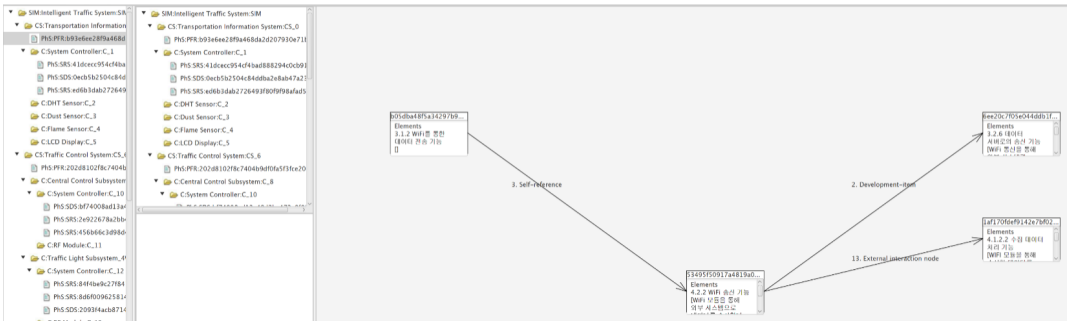


그림 5 시스템 구조 및 개발 산출물의 계층 관계
Fig. 5 A hierarchical structure of development artifacts

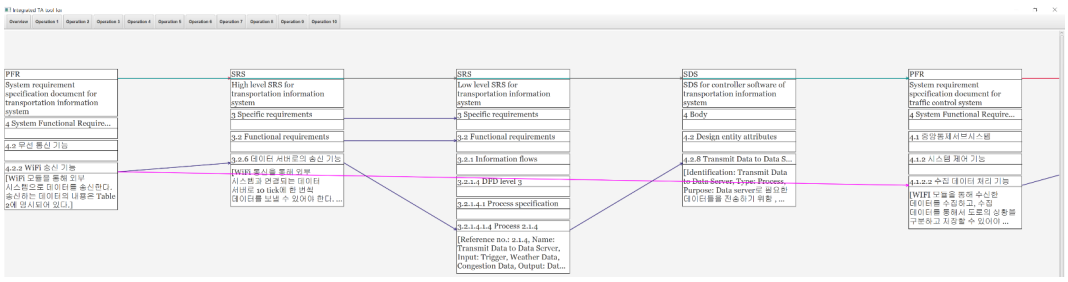
석하였으며, 결과에 따라 시스템의 변화와 같은 생명주기의 활동에 사용할 수 있는 관계 모델을 추출할 수 있다. <그림 6>은 테스트 환경의 두 시스템의 개발 산출물 사이의 관계 분석 결과와 도출한 관계 모델에 대한 그림으로, 이와 같이 본 논문에서 제안하는 테스트 환경을 구성하는 시스템 중 하나의 시스템이 특정 동작을 할 때 서로 다른 두 시스템의 상호작용 과 상호 연결 관계를 산출물들 사이의 관계로 확인할 수 있다. <그림 6>에 나타난 관계 모델과 연결 모델은 [15]와 [16]에서 그 정의를 확인할 수 있다.

<그림 6>의 (a)에서는 교통정보시스템의 PFR에서 하나의 element를 골라 해당 element와 교통정보시스템의 다른 개발 산출물 및 신호체계관제시스템의 개발 산출물과의 연결 관계를 확인할 수 있다. (a)에 나타난 세 가지 관계, development-item과 self-reference, 그리고 external interaction node는 각각 '개발 산출물에서 특정 항목의 선후 관계를 나타내는 연결', '하나의 개발 산출물 모델 내부에서의 개발 산출물의 연결', 그리고 '서로 다른 시스템에서 기능적으로 연결되어 있거나 상호작용하는 개발 산출물의 연결'을 의미한다. 여기서 선후관계란 개발 시에 작성하는 문서 간의 계층구조에서의 선후 관계를 의미한다. 개발 산출물 모델은 [16]에서의 DEM (Development Element Model)을 의미하며, 각 개발 산출물인 요구사항 문서와 설계 문서는 <그림 5>에서 PhS (Phase Structure)로 나타나 있으며, 그 내부에 각 장을 나타내는 AIS (Artifact Item Structure)가 있고, 그 내부에 각 요구사항에 대한 명세, 설계에 대한 명세 등이 E (Element)로 표현된다. 사용한 도구에서의 self-reference는 논문에서는 interior-reference로 표현되었다

<그림 6>의 (b)는 (a)에서 분석한 연결관계를 포함하여 연결 모델로 나타낸 것으로, 이처럼 구성 시스템의 여러 개발 산출물 간의 연결 관계를 확인할 수 있다. (b)에 나타난 연결 관계를 표현하기 위한 선의 색상은 [16]에 설명되어 있다. 동일한 시스템의 멀티 인스턴스가 존재하는 경우 서로 다른 인스턴스 사이의 상호작용에 대한 분석을 포함해야 하며, 관계 분석을 위해 개발한 도구의 한계로 (b)에서는 드러나지 않았으나 동일한 시스템에 대한 개발 산출물 간의 상호작용에 대한 연결 관계를 포함할 수 있다.



(a) An example of analyzing relationships among development artifacts



(b) An example of relationship models related to a development artifact in (a)

그림 6 관계 분석 화면

Fig. 6 A screen of relationships analysis

3. 테스트 환경의 활용

3.1 발생 가능한 긴급 상황에 대한 실험 결과

본 논문에서 제안하는 테스트 환경을 통해서는 테스팅 하고자 하는 상황에 대한 특정 값을 입력하여 해당 환경에서의 시스템의 동작을 확인할 수 있다. 예를 들어, 삼거리 교차로 2번의 교통정보시스템이 해당 교차로에 화재가 발생했음을 감지하면, 해당 정보는 MQTT 브로커 서버로 전송되고, 중앙통제 서브시스템에서 이를 수집 및 분석하여 삼거리 교차로 2번에 위치한 신호등 서브시스템을 통제한다. 이를 위해 교통정보시스템의 불꽃감지 센서에 불을 켠 라이터를 가까이 가져가면, 교통정보시스템은 이 센서 값을 수집하여 화재가 발생했음을 감지한다. 이때 교통정보시스템은 해당 교차로가 통제되고 있다는 메시지를 LCD를 통해 출력해야 한다. 또한 테스트 환경의 현재 버전에서는 교차로에서 긴급 상황이 발생한 경우 해당 교차로의 신호를 전부 통제하므로, 삼거리 교차로 2 번의 신호등 서브시스템은 해당 교차로의 신호를 모두 정지 신호로 변경해야 한다. 각 시스템 및 서브시스템에서 발생한 변화는 모니터링 서버로 전송되어 모니터링 시스템의 동작 화면에 동일하게 출력되므로, 해당 예시에 대한 실험 결과를 모니터링 시스템으로 확인한 결과는 <그림 7>과 같다.

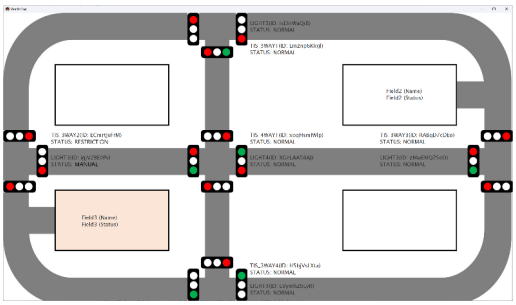


그림 7 삼거리 교차로 2 번에 화재가 발생한 상황에서의 모니터링 시스템의 동작 화면

Fig. 7 An operating screen of the monitoring system in fire situations at three-way intersection no.2

또 다른 예시로는, 도로 위의 특정 구간에 갑작스럽게 사고가 발생한 경우 사고 발생 정보를 도로 위의 차량으로부터 제공받는 경우가 있을 수 있다. 현재 버전에서는 아직 차량과 관련된 시스템을 개발하지 않았기 때문에, 차량과 교통정보 시스템은 WiFi 통신을 통해 연결될 수 있다고 가정하고, MQTT 브로커 서버를 통해 교통정보시스템으로 임의의 차량 데이터를 제공하는 방식으로 진행할 수 있다. 위 상황을 만들어주기 위해 MQTT 브로커 서버에 임의로 삼거리 교차로 1번과 사거리 교차로 사이의 E 구간에 사고가 발생했다는 내용의 차량 데이터를 입력하면, 교통정보시스템은 해당 구간에 접근이 불가능하다는 정보와 우회 도로 정보를 LCD를 통해 출력한다. 신호체계관제시스템의 중앙통제 서브시스템에서는 MQTT 브로커 서버를 통해 교통정보시스템으로부터 해당 데이터를 전달받고, 이 데이터를 기반으로 신호등 서브시스템을 통제한다. 삼거리 교차로 1번과 사거리 교차로에 위치한 신호등 서브시스템은 해당 도로로 향하는 방향의 신호가 직진 또는 좌회전 신호가 되지 않도록 하고, 나머지 도로로 향하는 방향의 신호는 정상적으로 동작할 수 있도록 한다. 이를 모니터링 시스템으로 확인한 결과는 <그림 8>과 같다.

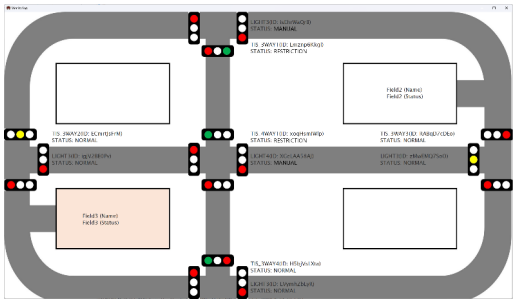


그림 8 E 구간에 사고가 발생한 상황에서의 모니터링 시스템의 동작 화면

Fig. 8 An operating screen of the monitoring system when an accident occurred in section E

3.2 통합 테스트 케이스 생성

본 테스트 환경은 멀티 인스턴스가 존재하는 가상물리시스템의 통합 테스트를 위하여 개발된 테스트 환경이므로, 이에 대하여 통합 테스트 케이스의 생성 및 검증 등의 활동을 수행할 수 있다. 우선적으로 안전성(safety) 등의 품질 특성에 대해서는 고려하지 않고 기능 요구사항에 대한 테스트만 고려하여 연결 관계를 고려한 통합 테스트 케이스를 생성할 수 있다.

전체 시스템의 연결 관계를 고려하여 기능에 관련된 테스트 케이스를 생성하는 절차는 다음과 같다: 1) [15]와 [16]에서 제안한 내용에 따라 각 시스템과 그 내부의 컴포넌트 사이의 연결 관계를 분석한다. 2) 분석한 연결 관계를 토대로 기능 요구사항에 대한 테스트 케이스를 도출한다. 이때, 추적성을 제외한 연결 관계를 발견했다면 이를 고려하여 테스트 케이스를 시나리오의 형태로 도출하고, 연결 관계가 없는 경우 개별적인 테스트 케이스를 도출한다. 이때, PFR, SRS, SDS 등 시스템 개발

시에 작성하는 개발과 관련된 문서는 상세하게 작성되어 있어야 한다.

이러한 테스트 케이스 생성 절차를 따르면, 3.1절에서 예시로 든 것과 같이 특정 구간 또는 교차로에서 화재가 발생한 상황에 관련하여 도출한 개별적인 시스템 및 관리자 관점에서의 시스템 테스트 케이스는 예시는 <표 3>과 같다. 개별 시스템 테스트 케이스에 대하여 모든 테스트 케이스가 통과하더라도, 이를 시나리오의 형태로 연결 지어 테스트를 수행하면 다른 결과가 나올 수 있기 때문에 연결 관계가 있는 시스템 컴포넌트 간의 테스트 케이스들을 시나리오의 형태로 연결 지어 테스트를 수행하는 것이 필요하다. 따라서 생성한 테스트 케이스들 중 연결 관계가 있는 테스트 케이스들을 연결 지어 시나리오의 형태로 변환하면 다음과 같이 정리할 수 있다. S-1의 경우 중앙통제 서브시스템과 신호등 서브시스템의 멀티 인스턴스와의 통신을 고려하여 작성되었음을 확인할 수 있다.

- **S-1:** 교통정보시스템에서 화재 정보를 입력 받으면 이후 서버를 통해 중앙통제 서브시스템으로 통제 정보를 전송한다. 중앙통제 서브시스템에서는 입력 받은 통제 정보를 토대로 신호등 서브시스템(들)의 위치를 특정하고 제어 명령을 생성하여 RF 통신을 통해 지속적이고 순차적으로 전송한다. 해당하는 신호등 서브시스템은 제어 명령을 입력 받으면 이에 따라 동작하고, 신호등 제어 보드로부터 현재 신호를 받아와 1초마다 모니터링 서버로 전송한다.
- **S-2:** 교통정보시스템에서 화재를 감지하였으나 모니터링 서버에만 통제 정보를 전송하고 MQTT

표 3 화재 발생 상황에서의 테스트 케이스 예시
Table 3 Examples of test cases under fire conditions

Transportation Information System	
TIS-1	Fire data is inputted into the system. Then the system has to transmit information on which section or intersection has to be restricted to the monitoring server within 1 second.
TIS-2	Fire data is inputted into the system. Then the system transmits information on which section or intersection has to be restricted to the MQTT server within 1 second.
TIS-3	Fire data is inputted into the system. Then the system prints out information on which section or intersection has to be restricted on LCD.
TIS-4	Fire data is inputted into the system. Then the system decides a detour path within 1 second.
TIS-5	Fire data is inputted into the system. Then the LCD prints out restriction information and detour path information alternately every 10 seconds.
Central Control Subsystem	
C-1	Restriction information about a certain section or intersection restriction from the MQTT server is inputted. Then the system specifies the traffic light subsystem at the corresponding location.
C-2	Corresponding traffic light subsystem is specified. Then the system generates control commands based on the inputted restriction information.
C-3	Restriction information is being provided. Then the system transmits generated control commands to the corresponding traffic light subsystem over RF communication until restriction information is no longer provided.
C-4	No restriction information is provided. Then the system follows basic operation rules to control traffic light subsystems.
Traffic Light Subsystem	
T-1	Central control subsystem provides manual control commands. Then the system operates accordingly.
T-2	Manual control commands are being provided by the central control subsystem. Then the system requests and receives current signal information from the traffic light board every second and transmits to the MQTT server.
T-3	Manual control commands are being provided by the central control subsystem. Then the system requests and receives current signal information from the traffic light board every second and transmit to the monitoring server.
Operator	
O-1	Operation information between the TIS and the TCS does not match on the monitoring system. Then the operator controls the TCS in person.

서버에는 해당 정보를 전송하지 않는 경우, 관리자는 이를 확인하고 중앙통제 서브시스템에 접근하여 신호등 서브시스템을 직접 조작한다.

마지막으로 이렇게 도출된 테스트 케이스의 예시를 테스트 환경에 적용하면, 테스트의 결과를 확인할 수 있다. 현재까지 도출된 화재 발생 상황에서의 테스트 케이스 예시에 대하여는 모든 테스트 케이스가 통과함을 확인하였다. 단, 생성한 테스트 케이스 예시의 커버리지 (coverage) [29]에 대한 분석은 진행하지 않았으므로 추후 연구를 통해 논의되어야 한다.

3.3 향후 활용 방안

본 논문에서는 이처럼 두 종류의 시스템이 멀티 인스턴스를 가지고 동작하는 테스트 환경을 개발하였고, 이를 통해 특정 입력을 제어하고 이에 따른 시스템의 변화를 확인할 수 있다. 또한, 관계 분석을 통해 해당 테스트 환경에 대해 여러 관계 모델을 추출할 수 있는 것을 확인하였다. 따라서 향후 다중 구성 시스템의 통합 환경에 대한 더 구체적인 테스트 케이스의 생성 및 검증과 위험 분석의 수행에 개발한 테스트 환경을 활용할 수 있고, 다중 구성 시스템의 시스템 추가, 변경, 및 삭제 시에 전체적인 관리가 가능하다. 또한, 다중 구성 시스템 환경에 대한 계층적 위험 분석, 위험 분석 요소와 테스트 케이스 사이의 관계 확인 및 생성 방법 등 여러 연구를 수행할 수 있다.

4. 관련 연구

가상물리시스템은 가상 세계와 물리 세계를 연결하는 실시간 시스템으로 해당 부분들을 고려하여 시스템에 대한 검증이 이루어져야 한다. 기존에 이를 지원하기 위한 다양한 시뮬레이션 도구와 테스트베드들이 존재한다.

4.1 테스트베드 개발 연구

[8]에서는 CPSoS (Cyber Physical System of Systems)의 예시 중 하나인 platooning system에 대해 LEGO를 활용하여 물리적인 실험 환경을 조성하여 가상의 시뮬레이션 뿐 아니라 물리적인 실험까지 가능하도록 exemplar인 Platooning LEGOs를 개발하였다. 해당 논문에서는 최종적으로 장애물 발생에 대한 두 가지 샘플 시나리오를 통해 실험 결과를 보여줌으로써 물리적으로 실험이 가능한 환경의 중요성에 대해 강조한다.

[9]과 [10]에서는 연구실 환경에서 사용 가능한 smart power system을 위한 하드웨어 기반의 테스트베드 플랫폼을 소개한다. [9]에서는 해당 테스트베드의 구성 요소 및 아키텍처와 동작을 소개하고, [10]에서는 해당 시스템에 대한 모니터링·제어·보호 시스템에 대해 소개하면서, 해당 테스트베드의 활용성에 대해 강조한다.

[11]에서는 대표적인 가상물리시스템 중 하나인 무인 비행체 (Unmanned Aerial Vehicle, UAV)를 위한 테스트베드인 OpenUAV를 제안한다. OpenUAV는 클라우드 서비스를 지원하며, 여러 사용자로부터의 동시다발적인 접근을 지원하는 시뮬레이션 도구로, 멀티 인스턴스에 대한 직접적인 언급은 없으나 도구 내에서 동시에 여러 개의 UAV 인스턴스를 다룰 수 있다.

4.2 테스트베드 개발 및 활용 연구

1장에서도 언급되었듯이, 시스템 이론에 따르면 시스템은 단순한 부분의 합으로 고려될 수 없으므로, 여러 시스템으로 구성되는 시스템은 단순히 기능적인 부분 이외에도 여러 품질 특성 (Quality Attribute)이 중요하게 다뤄져야 한다. 가상물리시스템에서는 이러한 특성 때문에 안전성이나 보안성 등의 품질 특성에 관해서도 중요하게 다뤄진다.

이와 관련해서 [12]에서는 Western System Coordinate Council (WSCC) 3 machine, 9 bus 테스트 케이스 기반의 전력 시스템 [30]을 실시간 가상물리시스템 테스트베드를 개발한다. 또한, 개발한 테스트베드를 활용하여 사이버 보안 (cyber security)과 안정성 제어 (stability control)의 두 가지 측면에 대하여 실험적 연구를 진행한다. 본 실험을 통해 사이버 공격이 물리적 전력망에 미치는 영향을 보이고자 하였으며, 시스템의 안정성을 최대한 유지하기 위한 전략 또한 제시한다. 또한 이러한 실시간 적응형 제어 검증은 보다 복잡한 시스템에 적용될 수 있다.

[13]에서도 [12]에서와 유사하게 실시간 가상물리시스템, 그 중 전력 시스템을 위한 테스트베드를 개발하고 사이버 보안 측면에서의 공격이 발생했을 때에 대한 사례 연구를 진행한다. 테스트베드는 RTDS (Real Time Digital Simulator)와 OPNET이라고 하는 네트워크 상에서 동작하고, 사례 연구의 대상이 되는 시스템은 11 bus로 구성된 시스템이다. 이 테스트베드는 또한 모델링 기능을 제공함으로써 충격 분석 (impact analysis)을 수행할 수 있다.

이 외에도 가상물리시스템에 관련된 다양한 시뮬레이션 도구와 물리적 환경을 조성할 수 있는 테스트베드에 관한 기존의 연구들이 존재한다. 본 연구에서는 멀티 인스턴스를 가지는 가상물리시스템에 대해 가상의 시뮬레이션이 아닌 하드웨어 기반의 테스트 환경을 구성하고자 하였으며, 테스트 환경을 모니터링할 수 있는 시스템까지도 포함하였다. 또한, 이 테스트 환경을 활용하여 안전성 등의 품질 특성과 여러 시스템 컴포넌트 간의 연결 관계까지 고려할 수 있는 통합 테스트 케이스 또한 생성할 수 있도록 한다.

5. 결론 및 향후 연구

본 논문에서는 멀티 인스턴스를 갖는 다중 구조의 가상물리시스템의 통합 테스트 케이스 생성, 위험 분석 연구 등의 활동에 활용할 수 있는 두 종류의 시스템으로 구성된 테스트 환경을 제안하고 개발하였다. 제안한 테스트 환경은 멀티 인스턴스로 구성되며, SW 개발 프로세스에 따라 요구사항 분석, 설계, 구현 등의 활동들을 수행하고 산출물을 작성하였다. 또한 여러 시스템의 상호작용 등의 관계에 대해 분석하기 위해 추적성을 포함한 다양한 개발 산출물 간의 연결 관계를 분석하고, 해당 결과들을 이용해 통합된 테스트 케이스의 생성 및 검증 등의 활동에 활용할 수 있음을 확인하였다.

현재 버전의 테스트 환경은 서로 다른 인스턴스 간에 구분이 되지 않는다는 한계점과 한 번에 하나 이상의 긴급 상황이 발생할 경우 우선순위가 존재하지 않기 때문에 시스템에 충돌이 발생할 수 있는 등의 한계점이 있으므로, 향후 동일 시스템의 멀티 인스턴스 간의 상호작용을 더욱 활성화하기 위해 요구사항을 보완하여 테스트 환경을 개선할 것이다. 또한, 앞서 밝힌 바와 같이 본 테스트 환경을 이용해 다중 시스템의 테스팅을 위한 통합 테스트 케이스 생성, 테스트 케이스의 연결성 분석, 계층적 위험 분석 등의 연구를 진행할 계획이다. 추가로, 활용한 두 시스템, 교통정보시스템과 신호체계관제시스템 뿐 아니라 다른 가상물리시스템까지도 테스트 환경에 추가할 것이다. 또한, 본 논문에서 언급한 활용 방안을 더 고도화하기 위해 커버리지 측정과 같은 실험 결과의 평가 척도들을 측정하는 방법을 추가로 연구하여 활용 방안을 늘릴 계획이다.

References

[1] W. Wolf, "Cyber-physical Systems," *Computer,* vol. 42, no. 3, pp. 88-89, Mar. 2009.

[2] L. Bass, P. Clements, R. Kazman, "Software Architecture in Practice," *Addison-Wesley Professional*, 2003.

[3] Ali, N., Hussain, M., Hong, J.-E., "SafeSoCPS: A Composite Safety Analysis Approach for System of Cyber-Physical Systems," *Sensors*, vol. 22, no. 4474, 2022.

[4] M. Daun, J. Brings, T. Bandyszak, P. Bohn and T. Weyer, "Collaborating Multiple System Instances of Smart Cyber-physical Systems: A Problem Situation, Solution Idea, and Remaining Research Challenges," *2015 IEEE/ACM 1st International Workshop on Software Engineering for Smart Cyber-Physical Systems*, pp. 48-51, 2015.

[5] X. Zhou, X. Gou, T. Huang and S. Yang, "Review on Testing of Cyber Physical Systems: Methods and Testbeds," *IEEE Access*, vol. 6, pp. 52179-52194, 2018.

[6] S. Kabir and Y. Papadopoulos, "Computational Intelligence for Safety Assurance of Cooperative Systems of Systems," *Computer*, vol. 53, no. 12, pp. 24-34, Dec. 2020.

[7] P. Checkland, "*Systems Thinking, Systems Practice*," John Wiley & Sons, 1981.

[8] Y. -J. Shin, L. Liu, S. Hyun and D. -H. Bae, "Platooning LEGOs: An Open Physical Exemplar for Engineering Self-Adaptive Cyber-Physical Systems-of-Systems," *2021 International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, pp. 231-237, 2021.

[9] V. Salehi, A. Mohamed, A. Mazloomzadeh and O. A. Mohammed, "Laboratory-Based Smart Power System, Part I: Design and System Development," *IEEE Transactions on Smart Grid*, vol. 3, no. 3, pp. 1394-1404, Sept. 2012.

[10] V. Salehi et al., "Laboratory-Based Smart Power System, Part II: Control, Monitoring, and Protection," *IEEE Transactions on Smart Grid*, vol. 3, no. 3, pp. 1405-1417, Sept. 2012.

[11] M. Schmittle et al., "OpenUAV: A UAV Testbed for the CPS and Robotics Community," *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*, pp. 130-139, 2018.

[12] S. Poudel, Z. Ni, N. Malla, "Real-Time Cyber Physical System Testbed for Power System Security and Control," *International Journal of Electrical Power & Energy Systems*, vol. 90, pp. 124-133, 2017.

[13] B. Chen et al., "Implementing a real-time cyber-physical system test bed in RTDS and OPNET," *2014 North American Power Symposium (NAPS)*, pp.1-6, 2014.

[14] A. G. Smith, "Introduction to Arduino," 2011. [Online]. Available: http://hdl.handle.net/1/5499 (accessed 2023, Sep 14)

[15] S. Jung, E.-S. Kim, J. Yoo, "A Traceability Analysis for Integrated Relationship Analysis of Development/Safety Artifacts of Cyber Physical Systems," *Journal of KIISE*, vol.48, no.1, pp.107-118, 2021. (in Korean)

[16] S. Jung, "A Comprehensive Relationship Analysis for Heterogeneous Artifacts in Multiple-Collaborative Safety-Critical Systems." Ph.D. dissertation, Konkuk Univ., 2022. [Online]. Available: http://www.dcollection.net/handler/konkuk/000000588860

[17] K. Su, J. Li, H. Fu, "Smart City and the Applications," *2011 International Conference on Electronics, Communications and Control (ICECC)*, pp. 1028-1031, 2011.

[18] G. Dimitrakopoulos, P. Demestichas, "Intelligent Transportation Systems," *IEEE Vehicular Technology*

Magazine, vol. 5, no. 1, pp. 77-84, 2010.
[19] R. A. Light, "Mosquitto: Server and Client Implementation of the MQTT Protocol," *Journal of Open Source Software*, vol. 2, no. 13, pp. 265, 2017.
[20] Cedalo, Eclipse Foundation (2018, Jan 8). Eclipse Mosquitto™(ver. 2.0.15) [Online]. Available: https://mosquitto.org/ (accessed 2023, Mar 9)
[21] D. K. Misra, "*Radio-Frequency and Microwave Communication Circuits: Analysis and Design*," John Wiley & Sons, 2012.
[22] Arduino, "SoftwareSerial Library | Arduino Documentation" [Online]. Available: https://docs.arduino.cc/learn/built-in-libraries/software-serial (last modified 2023, Sep 11, accessed 2023, Sep 14)
[23] S. Shylesh, "A Study of Software Development Life Cycle Process Models," *National Conference on Reinventing Opportunities in Management, IT, and Social Sciences*, pp. 534-541, 2017.
[24] International Organization of Standardization (ISO), International Electrotechnical Commission (IEC), Institute of Electrical and Electronics Engineers (IEEE), "*ISO/IEC/IEEE 29148-2018: Systems and software engineering – Life cycle processes – Requirements engineering*," Nov 2018.
[25] Q. Li, Y-L. Chen, "Data Flow Diagram," *Modeling and Analysis of Enterprise and Information Systems*, pp. 85-97, 2009.
[26] R. J. K. Jacob, "A State Transition Diagram Language for Visual Programming," *Computer,* vol. 18, no. 8, pp. 51-59, 1985.
[27] Institute of Electrical and Electronics Engineers (IEEE), "*IEEE Std 1016-2009: IEEE Standard for Information Technology–Systems Design-Software Design Descriptions*," Jul 2009.
[28] J. Bayle, "*C Programming for Arduino*," Packt Publishing Ltd, 2013.
[29] P. Ammann, J. Offutt, "*Introduction to Software Testing*," Cambridge University Press, 2016.
[30] P. M. Anderson, A. A. Fouad, "*Power System Control and Stability*," John Wiley & Sons, 2008.

허 윤 아

2021년 건국대학교 물리학과·컴퓨터공학과 졸업 (학사). 2023년 건국대학교 컴퓨터공학과 졸업 (석사). 2023년~현재 건국대학교 컴퓨터공학과 박사 과정. 관심분야는 소프트웨어 공학, 위험/안전성 분석

정 세 진

2015년 건국대학교 컴퓨터공학과 졸업 (학사). 2016년 건국대학교 컴퓨터 정보통신공학과 졸업(석사). 2022년 건국대학교 컴퓨터 정보통신공학과 졸업 (박사) 2023년~현재 진주교육대학교 컴퓨터교육과 조교수. 관심분야는 소프트웨어 공학, 위험/안전성 분석

유 준 범

2005년 KAIST 전자전산학과 전산학 전공 졸업 (박사). 2008년 삼성전자주식회사 통신연구소 책임연구원. 2008년~현재 건국대학교 컴퓨터공학부 교수. 관심분야는 소프트웨어 공학, 안전성 분석, 정형기법