JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 26 No. 9, pp. 105-114, September 2021
https://doi.org/10.9708/jksci.2021.26.09.105

An Effective ESICD Verification Strategy: A case study of Military Satellite Communications System II

Kee-Sung Lee*, Jun-Ho Choi*, Jeong-Jin Shin*, Hye-Jin Yoon*, Seung-Ho Kim**

*Chief Research Engineer, C4I R&D Lab, LIG Nex1 Company, Gyeonggi-do, Korea
*Senior Engineer, C4I R&D Lab, LIG Nex1 Company, Gyeonggi-do, Korea
*Senior Engineer, C4I R&D Lab, LIG Nex1 Company, Gyeonggi-do, Korea
*Senior Engineer, C4I R&D Lab, LIG Nex1 Company, Gyeonggi-do, Korea
**Senior Researcher, Defense Space Technology Center, ADD, DaeJeon, Korea

[Abstract]

ESICD(Electrical Signal Interface Control Document) refers to a document that describes protocols and data for communication between components consist of a system. Each component developer gathers at a specific place to conduct an integrated test for ESICD verification. In this case, it often happens that the integration test is delayed due to a simple mistake of software developers. There are two reasons for this situation: First, software developers do not perform sufficient verification because it is difficult to configure the system environment in a Lab, and second, they do not immediately find the cause of errors occurred during integration tests. Therefore, in this paper, we propose a strategy to effectively perform ESICD verification, which takes a lot of time between the production and implementation stage of the weapon system development stage and the system integration test stage.

▸ **Key words**: ESICD Verification, Weapon System, Simulator, System Integration Strategy

[요 약]

소프트웨어 연동통제문서는 시스템을 구성하는 구성품 간 연동을 위해 프로토콜 및 데이터 등을 기술하는 문서를 의미한다. 무기체계 개발 특성상 소프트웨어 연동통제문서 검증을 위해서는 각 구성품 개발자들이 특정 장소에 모여서 통합시험을 진행하게 되는데, 이때 소프트웨어 개발자들의 단순한 실수로 인하여 통합시험이 지연되는 경우가 종종 발생하게 된다. 이러한 상황이 발생하는 가장 큰 이유는 소프트웨어 개발자들이 연구실 환경에서는 시스템 환경을 구성하기 어려운 문제로 인해 충분한 검증을 수행하지 못한 경우와 통합시험 간 발생한 오류에 대해서 즉각적으로 원인을 발견하지 못한 경우 때문이다. 따라서 본 논문에서는 무기체계 개발 단계의 제작 및 구현단계부터 시스템 통합시험 단계 사이에 많은 시간이 소요되는 소프트웨어 연동통제문서 검증을 효과적으로 수행하기 위한 전략을 제안한다.

▸ **주제어**: 소프트웨어 연동통제문서 검증, 무기체계 소프트웨어 개발, 시뮬레이션 환경, 시스템 통합 전략

• First Author: Kee-Sung Lee, Corresponding Author: Kee-Sung Lee
*Kee-Sung Lee (keesung.lee@lignex1.com), C4I R&D Lab, LIG Nex1 Company
*Jun-Ho Choi (junho.choi2@lignex1.com), C4I R&D Lab, LIG Nex1 Company
*Jeong-Jin Shin (jungjin.shin@lignex1.com), C4I R&D Lab, LIG Nex1 Company
*Hye-Jin Yoon (hyejin.yoon@lignex1.com), C4I R&D Lab, LIG Nex1 Company
**Seung-Ho Kim (seungho_kim@add.re.kr), Defense Space Technology Center, ADD
• Received: 2021. 08. 13, Revised: 2021. 09. 08, Accepted: 2021. 09. 09.

http://www.ksci.re.kr pISSN:1598-849X | eISSN:2383-9945

I. Introduction

무기체계 개발은 Fig. 1과 같이 요구사항 분석, 기본설계(PDR), 상세설계(CDR), 제작 및 구현, 구성품 단위 시험, 시스템 성능 시험(연동시험), 개발 시험평가(DT/OT)의 7단계로 진행된다[8].

Fig. 1. Development Step of Weapon System

소프트웨어 측면에서 무기체계 개발 단계별로 수행하는 업무를 살펴보면, 1단계부터 3단계는 요구사항을 기반으로 요구사항에 적합한 무기체계의 소프트웨어에 대해서 기본설계와 상세설계를 수행하고, 상세설계 단계에서 시스템을 구성하는 구성품 간 소프트웨어 연동을 위해 프로토콜 및 데이터 등을 기술하는 문서인 소프트웨어 연동통제문서(ESICD: Electrical Signal Interface Control Document)를 시스템의 요구사항을 만족할 수 있도록 각 구성품의 담당자들과 협의하여 작성한다. 4단계인 제작 및 구현단계에서는 각 구성품 본연의 기능을 구현한다. 5단계인 구성품 단위 시험단계에서는 이전 단계에서 구현한 구성품 본연의 기능 시험을 수행하면서 소프트웨어 연동통제문서를 기반으로 연동 시험에 필요한 기능을 구현한다. 6단계인 시스템 성능 시험에서는 개발된 구성품들을 하나의 시스템으로 통합하여 시스템 연동 시험을 진행하고, 7단계인 개발 시험평가에서는 시스템과 시스템을 연동하여 종합적인 개발 시험평가를 진행하게 된다. 총 7단계로 진행되는 무기체계 개발을 보다 효과적으로 수행하기 위해서는 각 단계를 진행하면서 해당 단계에서 발생할 수 있는 위험들을 사전에 식별하고, 이전 단계에서 위험을 제거하는 전략이 필요하다[1].

본 논문에서는 무기체계 개발 단계 중 6단계인 시스템 성능 시험과 7단계인 개발 시험평가에서 발생할 수 있는 위험 요소로 '소프트웨어 연동통제문서 연동 오류'를 식별하였으며, 무기체계 개발 특성상 6단계와 7단계 각 구성품 개발자들이 특정 장소에 모여서 소프트웨어 연동통제문서를 기반으로 통합시험을 진행하게 되는데, 이때 개발자들의 단순한 실수(바이트 오더링, 비트 해석 오류 등)로 인해 발생한 상황에 대해서 즉각적으로 원인 분석이 어려워 통합시험이 지연되는 경우가 종종 발생한다[1].

이러한 상황이 발생하는 가장 큰 원인은 개발자들이 연구실에서 시스템 환경을 구축하기 어렵고 그로 인해 충분한 검증을 하지 못한 상황에서 통합시험을 진행하면서 발생한 것이다. 이러한 오류를 줄이기 위해서는 개발자들이 연구실에서 전체 시스템을 구성하여 통합 시험을 사전에 준비할 수 있어야한다. 따라서 본 논문에서는 식별된 위험을 4단계부터 제거할 수 있는 방법으로 각 구성품의 소프트웨어 개발 담당자가 연구실에서 전체 시스템을 구성하여 소프트웨어 연동통제문서를 검증할 수 있는 E2V(Efficient ESICD Verification) 시스템을 제안한다.

본 논문의 구성은 2장에서 제안하는 시스템과 유사한 기능을 가지고 있는 선행 연구들에 대해서 조사하고, 3장에서는 제안하는 시스템에 대해서 소개를 하고 4장에서는 제안하는 시스템을 이용하여 실제 무기체계 개발 단계에서 활용 가능한 사례를 제시하고, 5장에서 결론을 맺는다.

II. Preliminaries

1. Related works

일반적으로 무기체계 소프트웨어를 검증하기 위한 전략은 통합시험환경(System Integration Laboratory)과 모의환경을 구성하여 검증하는 전략으로 나뉜다[2-3, 5-6]. 통합시험환경은 실 장비와 연동하기 전 개별 구성품의 기능 및 연동 기능에 대해서 회기시험을 통한 검증이 가능한 환경을 의미한다. 모의환경은 통합시험환경을 구성하지 못하는 경우에는 최소한의 구성품들을 소프트웨어 형태로 구성하여 소프트웨어 연동통제문서에 대한 검증을 개발자 레벨에서 패킷 단위로 수행하는 환경을 말한다.

1.1 Verification based on System Integration Lab

통합시험환경을 구성하는 무기체계로는 유도무기, 항공전자시스템 등이 있다. 특히 무인항공기의 경우에는 개발과정에서 개별 구성품의 기능 및 연동 기능 검증을 위한 통합시험환경을 구성하는 것이 일반적인 전략이다[3, 5-6]. 통합시험환경에서는 비행체, 지상통제체계에 탑재되는 실 장비를 실제 환경과 같이 연결하여 다양한 기능 및 성능 시험을 수행하여 실제 환경에서 발생 가능한 오류 식별을 조기에 발견하고 수정함으로써 통합시험기간 단축 및 체계연동의 효율성을 높이는 역할을 수행한다[5].

그러나 이러한 통합시험환경에서도 시간적인 이유로 시스템 전체의 시험항목을 매번 시험하기 어렵기 때문에 소프트웨어 연동통제문서 변경이 발생한 시점에 시스템의 영향성을 판단하여 선택적 회기시험을 수행한다[3, 6-7].

선택적인 회기시험 역시 소프트웨어 변경에 따른 검증시험에 많은 시간이 소비되기 때문에 자동코드 생성기를 개발하여 소프트웨어 연동통제문서 검증 및 구성품의 기능에 대한 검증을 수행하는 방법들도 함께 사용되고 있다[3, 5-6]. 자동코드 생성을 통한 소프트웨어 검증의 장점으로는 개발자의 개입을 최소화하기 때문에 소프트웨어 신뢰성이 높아지고 빠른 시간 내에 코드에 대한 유효성 검증이 가능하지만, 무기체계의 특성상 구성품의 수가 방대하고 개발 언어도 상이하여 각각의 구성품에 대해 자동코드 생성기를 개발하여 검증하는데 어려움이 존재한다.

1.2 Verification based on Simulation Software

무기체계 개발과정에서 통합실험환경을 구성하기 위해서는 통합실험환경을 관리하고 운용할 인원들과 실제 환경과 동일한 구성품 등을 구비해야하는 어려움으로 인해서 통합시험환경을 구성하지 못하는 경우도 다수 존재한다. 이러한 경우에는 최소한의 모의환경을 구성하여 소프트웨어 연동통제문서에 대한 검증을 개발자 레벨에서 패킷 단위로 진행한다. 이러한 모의환경에서 패킷 검증을 위해서는 네트워크상에서 패킷 수집과 분석이 가능해야한다[2].

네트워크에서 패킷을 수집하기 위해서는 WinPcap과 같은 네트워크 인터페이스로부터 수신된 패킷들을 수집할 수 있는 라이브러리를 사용한다. 이렇게 수집된 패킷은 핵사코드 형태로 수집되기 때문에 가독성이 낮고 해당 패킷의 오류 여부를 검증하기 위해서는 반드시 소프트웨어 연동통제문서를 참고해야하는 번거로움이 존재한다. 따라서 수집된 패킷을 분석하여 개발자가 이해할 수 있는 소프트웨어 연동통제문서의 텍스트 형태로 변환하는 과정이 필요하다.

모의환경은 개발자가 손쉽게 구축할 수 있으나, 구축된 모의환경에 대한 검증 과정이 별도로 이루어지지 않기 때문에 모의환경 자체에 대한 오류가 존재하는 한계가 있다[2]. 따라서 모의환경을 시스템 통합하는 담당자가 개발 후 충분히 검증된 모의환경을 각 구성품 개발자에게 배포하여 시스템 통합 환경과 유사한 환경을 구축할 수 있도록 지원함으로써 개발자들의 단순한 실수로 인해 통합시험이 지연되는 경우를 사전에 대비할 수 있게 된다[1].

III. The Proposed Framework

이 장에서는 본 논문에서 제안하는 무기체계 개발 4단계부터 소프트웨어 연동통제문서를 효과적으로 검증할 수 있는 아키텍처에 대해서 설명한다.

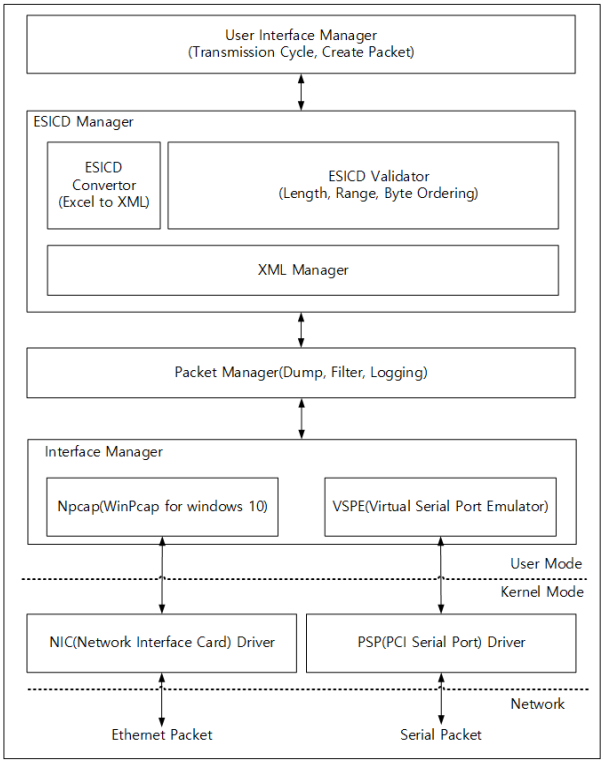

Fig. 2. E2V(Efficient ESICD Verification) System Architecture

본 논문에서 제안하는 프레임워크는 Fig. 2와 같이 Interface Manager, Packet Manager, ESICD Manager, User Interface Manager로 구성되어있다.

Interface Manager는 네트워크 인터페이스 카드와 시리얼 포트로부터 패킷을 수신하는 역할을 수행한다. Packet Manager는 수신된 패킷 전체 또는 사용자에 의해 설정된 특정한 패턴의 패킷을 ESICD Manager로 전달하고, 패킷들을 저장하는 역할을 수행한다. ESICD Manager는 엑셀로 작성된 소프트웨어 연동통제문서를 XML 형태로 변환 후, Packet Manager로부터 수신한 패킷들에 대해서 구문 오류를 분석하는 역할을 수행한다. User Interface Manager는 소프트웨어 연동통제문서 테스트를 위해 사용자가 손쉽게 특정한 패턴의 패킷들을 생성하여 주기적으로 전송하거나 수신된 패킷을 가공하여 특정 구성품으로 전송하는 역할을 수행한다.

1. Interface Manager

Interface Manager는 네트워크 인터페이스 카드부터 수신되는 패킷을 처리하는 Npcap(WinPcap for windows 10)과 시리얼 포트로부터 수신되는 패킷을 처리하는 VSPE(Virtual Serial Port Emulator)모듈로 구성되어 있다. 무기체계 구성품들은 시스템에 장착되는 위치나 구성품의 역할에 따라서 다양한 연동 인터페이스로 설계된다. 최근에는 ALL IP 기반으로 구성품들에 대한 연동 인터페이스를 설계하지만, 성능개량과 같이 기존 구성품들이 시리얼 통신 인터페이스로 연동을 해야 하는 경우도 존재하기 때문에 이더넷과 시리얼 통신이 모두 지원 가능한 형태의 시스템을 설계하였다.

Npcap은 리눅스/유닉스에서 사용하는 Libpcap을 윈도우 개발 환경에 맞도록 포팅한 것으로 Windows 10 이하 버전에서는 WinPcap을 사용하지만, Windows 10에서는 Npcap을 사용한다[4]. Npcap을 사용하는 대표적인 프로그램으로는 네트워크 패킷을 분석하는 도구인 와이어샤크(WireShark)가 있다. 와이어샤크의 주요 기능으로는 네트워크 인터페이스 카드로 수신되는 패킷을 캡처하여 사용자가 정의한 규칙에 맞는 패킷들만 응용 프로그램으로 전달하는 역할을 수행한다.

VSPE(Virtual Serial Port Emulator)는 가상의 COM 포트를 만들고, 실제 포트와 서로 연결시키는 기능을 제공한다. 본 논문은 기존 구성품의 연동 인터페이스가 시리얼 통신에서 IP 통신으로 변경이 필요한 상황과 시리얼 인터페이스로 연동이 되는 구성품들의 통신과정을 모니터링하고 분석하기 위해서 VSPE 모듈을 적용하였다.

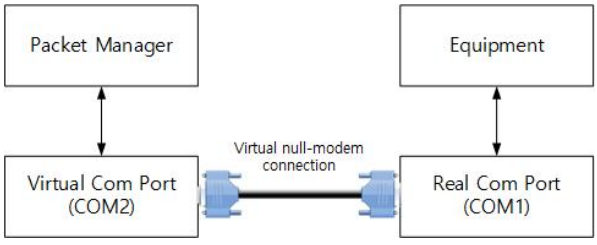

Fig. 3. VSPE Connection Diagram

예를 들어 Fig. 3과 같이 COM1과 COM2 포트는 논리적인 형태로 연결 설정을 하고, 구성품을 실제 COM1에 연결한 후 구성품에서 자신의 상태정보를 송신하게 되면, COM1 포트와 연결된 COM2에서 상태정보 데이터를 수신하여 Packet Manager에서 데이터를 수신할 수 있는 환경이 된다. 이러한 구성은 이더넷 패킷과 시리얼 데이터를 공통 모듈에서 처리할 수 있다는 장점을 가지게 된다.

2. Packet Manager

Packet Manager는 Interface Manager로부터 전달받은 UDP 패킷을 ESICD Manager로 전달하는 역할을 수행한다. 디버깅 관점에서 살펴보면, 네트워크를 통해서 송수신되는 모든 패킷을 개발자가 모니터링 할 필요가 없기 때문에 현 시점에서 모니터링 하고자하는 계획된 타깃 패킷들을 제외하고 다른 패킷들은 필터링하는 것이 매우 효율적인 디버깅 방법이다.

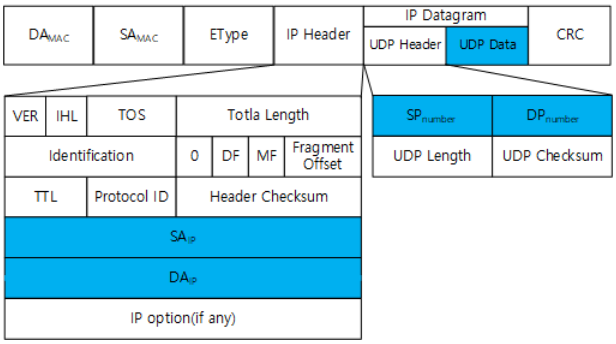

Fig. 4. UDP IP Structure

이러한 관점으로 본 논문에서는 Fig. 4와 같은 구조에서 필터링을 위한 파라미터로 SP_{ip}, DP_{ip}, SP_{number}, DP_{number}, UDP Data를 선정하였으며, 개발자가 선정된 파라미터를 설정할 수 있는 사용자 인터페이스를 Fig. 5와 같이 제공한다. 사용자 인터페이스에서 패킷 필터링 적용 체크박스를 선택하면 수신된 패킷들중에서 SP_{ip}, DP_{ip}, SP_{number}, DP_{number} 값을 모두 만족하는 패킷만 ESICD Manager로 전달하고 나머지 패킷들은 폐기시킨다. 이와 반대로 UDP Data는 패킷 Discard 패턴을 등록하고, 적용을 선택하면 이후 수신되는 패킷에서 등록된 패턴을 비교하여 패킷에 등록된 패턴이 존재하면 해당 패킷을 폐기하고, 그렇지 않은 패킷은 ESICD Manager로 전달한다.

Fig. 5. User Interface for Packet Filtering

3. ESICD Manager

ESICD Manager는 Packet Manager로부터 전달받은 패킷들을 가공하여 사용자에게 직관적인 형태로 제공하기 위해 소프트웨어 연동통제문서를 XML로 변환하는 ESICD Convertor, 수신된 패킷들의 문법적인 오류를 체크하는 ESICD Validator로 구성되어 있다.

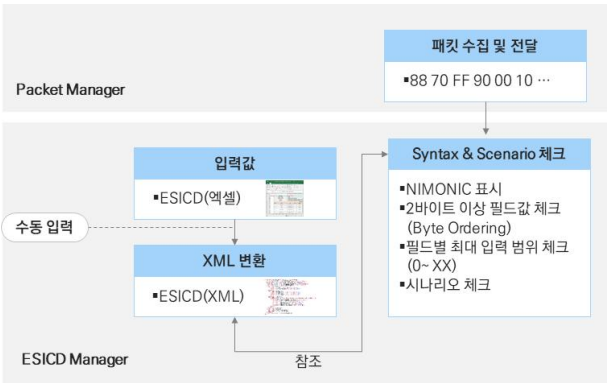

Fig. 6. Data Processing Flow in ESICD Manager

Packet Manager로부터 전달받은 패킷들은 Fig. 6과 같은 핵사코드 형태로 수신되기 때문에 개발자가 직관적으로 패킷의 명칭, 필드 오류, 시퀀스 오류 여부를 확인하기 어렵다. 따라서 패킷을 직관적으로 분석하기 위해서는 소프트웨어 연동통제문서를 참고 해야 한다.

본 논문에서는 이러한 문제를 해결하기 위해서 소프트웨어 연동통제문서를 XML로 변환하는 과정을 수행하여 수신된 패킷들을 처리하는 방법을 제안한다. 그러나 소프트웨어 연동통제문서는 다양한 문서 형태로 작성이 가능하기 때문에 본 논문에서는 소프트웨어 연동통제문서가 엑셀 형태로 작성되어 있는 상황을 가정하였고, 한글 문서나 MS 워드로 작성되어 있는 부분들에 대해서는 수동으로 XML을 추가하였다.

Fig. 7은 엑셀로 작성된 소프트웨어 연동통제문서를 XML로 변환한 예시이다. 소프트웨어 연동통제문서는 시스템을 구성하는 구성품별로 작성이 되었기 때문에 XML의 구성 요소도 구성품을 나타내는 Unit 단위로 기술하였다. 구성품은 제어와 감시의 항목들이 존재하는데, 본 예시에서는 감시 항목을 기준으로 설명을 하고자 한다. 감시 항목으로는 구성품의 상태정보와 SW 버전정보, C-BIT(Continuous Built In Test) 결과정보로 구성되며 각각의 항목은 구성품의 정보를 제공하는 하나의 단위가 되기 때문에 Item을 하나의 요소로 식별하였으며, 각각의 Item들은 sItem(sub Item)을 갖는 구조로 설계하였다. sItem은 바이트(Byte) 단위로 패킷들을 처리하지만, sItem의 하위에 있는 sItem_property를 정의하여 비트(Bit) 단위로 패킷을 처리 할 수 있도록 기술되었다.

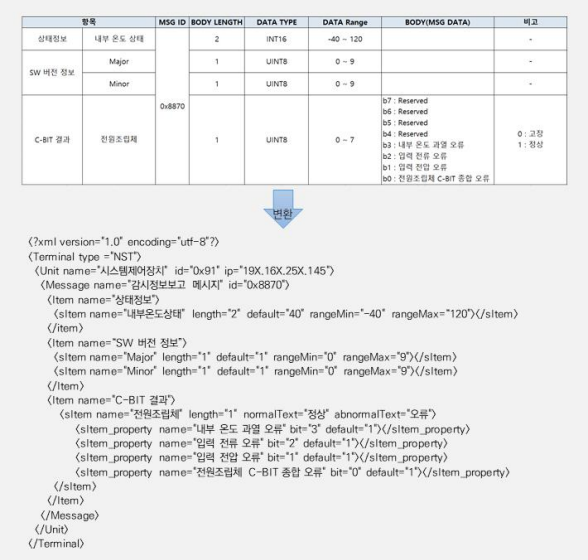

Fig. 7. XML Sample Converted in Excel

변환된 XML을 기초로 하여 Packet Manager로부터 수신된 패킷들의 헤더를 분석하여 Fig. 8처럼 직관적인 니모닉(Mnemonic) 형태로 표현이 가능하다.

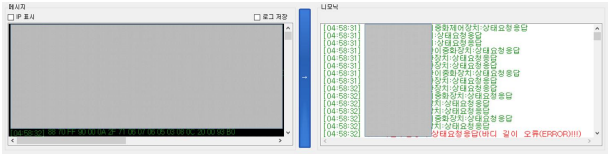

Fig. 8. Sample Captured Packet with Nimonic

XML의 sItem의 속성 중 length 값이 1보다 큰 경우에는 해당 필드에 대해서 바이트 오더링 검사를 수행하여 Fig. 8의 니모닉 화면에 정상(초록색) 또는 비정상(빨간색) 여부를 표시할 수 있다. 메모리에 데이터가 저장되는 순서를 의미하는 바이트 오더링은 빅 엔디안(Big Endian)과 리틀 엔디안(Little Endian)방식이 있다.

AMD, PPC 계열의 CPU는 빅 엔디안 방식, Intel 계열의 CPU는 Little Endian 방식을 사용한다. 예를 들면 2Byte를 표현하는데 실제 값이 “1”인 경우에 빅 엔디안의 경우는 핵사코드 “00 01”로 저장되지만, 리틀 엔디안의 경우에는 “01 00”으로 저장되어 실제 값과 큰 차이를 보이게 된다. 이렇게 서로 다른 CPU에서 저장된 데이터를 네트워크로 유통시키게 되면 바이트 오더링의 해석 문제로 인해서 구성품들이 오동작을 하게 된다. 따라서 시스템 관점에서는 유통되는 바이트 오더링 방식을 정해야하며,

개발자들은 패킷들을 네트워크로 유통시킬 때 바이트 오더링을 고려하여 패킷들을 스왑하는 과정을 거쳐야한다. 바이트 오더링과 더불어 sItem의 속성인 rangeMin값과 rangeMax값을 활용하여 수신된 패킷의 정상 범위 값이 유통되는지 확인이 가능하다.

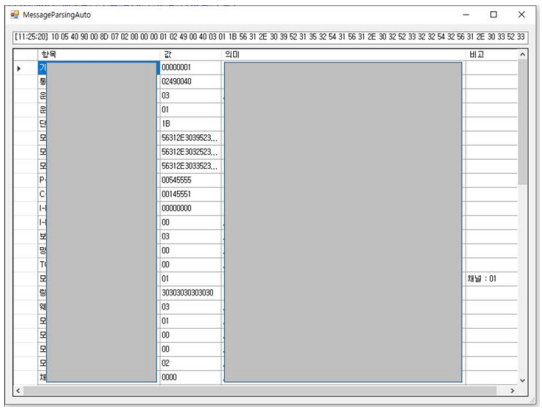

Fig. 9. Screen of Packet Analysis

또한, Fig. 8과 같이 수신된 패킷의 니모닉을 선택하면, Fig. 9와 같이 XML로 정의된 ESICD의 sItem의 속성 중 name 속성 값을 이용하여 필드들의 항목을 표현하고, length 값을 이용하여 각각의 필드 길이에 맞도록 패킷을 분리하여 직관적으로 패킷의 정상 유무에 대한 확인이 가능하다.

4. User Interface Manager

User Interface Manager에서는 구성품을 모의하기 위해서 XML 기반으로 테스트 패킷을 생성하여 특정한 IP로 전달하는 기능과 시스템 제어장치를 모의하기 위해서 수신한 패킷에서 메시지 헤더의 송신 필드 값을 변경하여 특정한 IP로 전달하는 기능을 제공한다. 본 논문에서는 구성품은 무기체계에서 특정한 임무를 수행하는 장치, 시스템 제어장치는 각 구성품으로부터 수집된 정보를 사용자 GUI가 탑재된 컴퓨터로 전달하는 장치를 의미한다.

Fig. 10은 소프트웨어 연동통제문서를 기반으로 생성된 XML을 이용하여 구성품 제어를 위한 모의 화면이다. 구성품과 GUI를 각각 다른 개발자들이 구현하기 때문에 해당 모의기를 통해서 구성품 개발자들은 자신들이 구현한 구성품의 입력범위와 그에 따른 구성품의 동작을 확인하는 용도로 사용할 수 있다. 제안하는 시스템을 이용하여 패킷을 생성하는 절차는 다음과 같다.

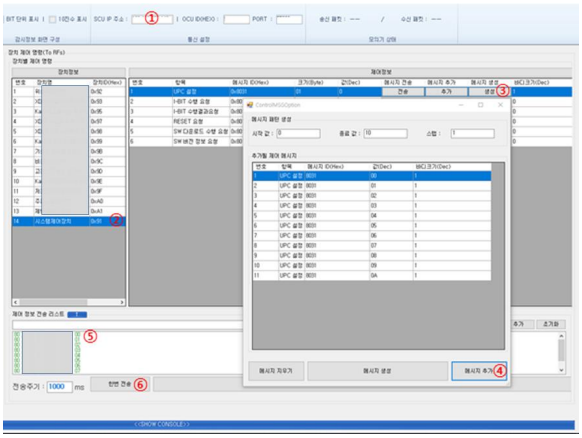

Fig. 10. Screen of Create Packet

① 해당 패킷을 수신하는 구성품의 IP주소를 입력한다.
② 테스트 하고자 하는 구성품을 선택한다.
③ 선택한 구성품의 제어 메시지를 선택한다.
④ 메시지의 시작 값, 종료 값, 구간에 대한 증가하는 값을 입력하고, 메시지 추가 버튼을 누른다.
⑤ 생성된 메시지를 확인한다.
⑥ 전송주기를 설정하고 전송버튼을 눌러 구성품으로 생성한 패킷을 전송한다.

Fig. 11은 시스템 제어장치를 모의하는 화면으로 수신된 패킷에서 메시지 헤더의 송신 필드 값을 변경하여 특정한 IP로 전달하는 기능을 제공한다.

그러나 본 논문에서는 Npcap을 사용하여 네트워크 인터페이스에서 들어오는 모든 패킷에 대한 수신을 가정하고 있기 때문에 시스템 제어장치를 모의할 때는 반드시 로컬에서 생성된 패킷에 대해서는 수신을 할 수 없도록 필터링을 해야 한다. 이러한 필터링을 하지 않을 경우에는 특정한 IP로 전달하고자 하는 패킷도 Interface Manager에서 재수신되기 때문에 무한 루프가 발생하게 된다.

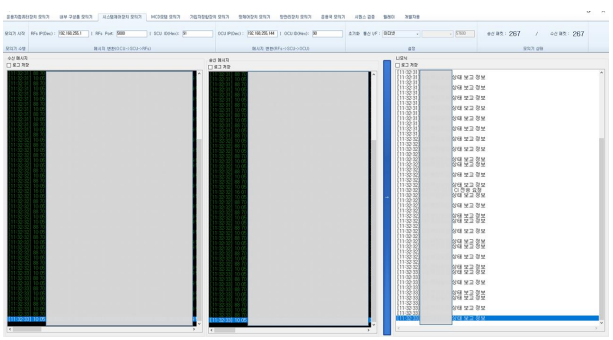

Fig. 11. Screen of Simulation for System Control Unit

IV. Scenario and Evaluation

본 장에서는 군 위성통신체계-II 체계 개발에 적용한 E2V 시스템을 활용한 사례 중심으로 논의하고자 한다. 군 위성통신체계-II 시스템의 경우 구성품의 종류를 RF 구성품, 시스템 제어장치, 운용자컴퓨터장치(GUI)로 구분할 수 있다. 무기체계 소프트웨어 개발은 소프트웨어 연동통제문서를 정의한 이후에 각 구성품들이 동시에 개발을 진행하기 때문에 무기체계 개발 4단계부터 제안하는 시스템을 적용하였다.

1. Weapon System Development Stage 4-5 Support Simulation Scenario

무기체계 개발4단계와 5단계에서는 RF구성품 개발을 지원하는 모의환경, 시스템 제어장치 개발을 지원하는 모의환경, GUI 개발을 지원하는 모의환경에 적용이 가능하다.

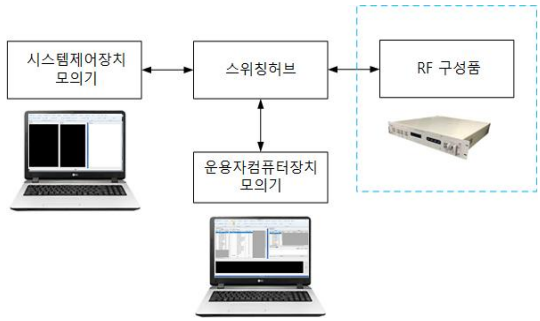

Fig. 12. Environment for RF Equipment Developer

RF 구성품은 실제 통신 시스템에서 각각의 임무를 수행하는 장치들을 의미한다. RF 구성품들의 소프트웨어적인 요소들은 많지 않으나, 시스템 측면에서는 RF 구성품의 수량이 많고, 하나의 RF 구성품이 정상동작하지 않는 경우 전체 시스템이 영향을 받을 수 있기 때문에 충분한 검증이 필요한 구성품들이다.

RF 구성품 개발자를 위한 모의환경은 Fig. 12와 같이 노트북 각각에 설치된 시스템 제어장치 모의기, 운용자컴퓨터장치 모의기, 실제 개발 중인 RF 구성품을 하나의 스위칭허브에 연결하여 구성한다. RF 구성품의 정상 동작 여부를 확인하기 위해여 운용자컴퓨터장치 모의기에서 RF 구성품 제어 메시지를 시스템 제어장치 모의기로 전송하고, 시스템 제어장치 모의기는 해당 메시지를 변환하여 RF 구성품으로 전달한다. RF 구성품 개발자는 수신된 제어 명령을 받은 RF 구성품이 정상 동작하는지 확인하고, 그 결과를 시스템 제어장치 모의기로 전송한 후 운용자컴퓨터장치 모의기에서 확인한다. 이러한 과정을 제어 명령 전체에 대해서 수행을 하게 되면 RF 구성품 개발자의 입장에서는 수신한 데이터와 송신한 데이터에 대해서 소프트웨어 연동통제문서를 검증할 수 있게 된다.

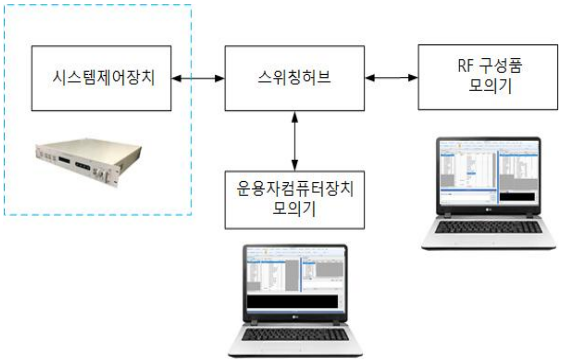

Fig. 13. Environment for System Control Equipment Developer

시스템 제어장치는 하나의 네트워크에서 수신된 데이터를 분석하여 해당 데이터가 어떤 구성품으로 전달되어야 하는지 판단하여 전달하는 역할을 한다. 수신되는 데이터의 양이 많으면 처리하는 속도에 영향을 받기 때문에 부하시험 등의 검증 시간이 필요한 장치이다.

시스템 제어장치 개발자를 위한 모의환경은 Fig. 13과 같이 노트북 각각에 설치된 RF 구성품 모의기, 운용자컴퓨터장치 모의기, 실제 개발 중인 시스템 제어장치를 하나의 스위칭허브에 연결하여 구성한다.

시스템 제어장치의 정상 동작 여부를 확인하기 위해서 운용자컴퓨터장치 모의기에서 RF 구성품 제어 메시지를 시스템 제어장치로 전송하고, 시스템 제어장치에서 해당 메시지를 변환하여 RF 구성품으로 전달하는지 확인한다. 또한 RF 구성품 모의기에서 응답 메시지를 시스템 제어장치로 전달하고, 그 결과를 운용자컴퓨터장치 모의기에서 확인하여 시스템 제어장치의 정상 동작여부를 확인한다. 이러한 과정을 제어 명령 전체에 대해서 수행을 하게 되면 시스템 제어장치 개발자의 입장에서는 수신한 데이터와 송신한 데이터에 대해서 소프트웨어 연동통제문서를 검증할 수 있게 된다.

운용자컴퓨터장치는 사용자의 명령을 생성하고 전달하며, 구성품의 상태를 실시간으로 모니터링 역할을 수행하기 때문에 다른 구성품에 비해 개발 시간과 검증에 많은 시간이 필요한 장치이다.

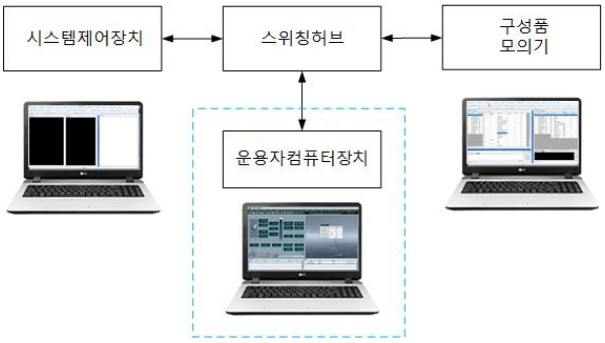

Fig. 14. Environment for GUI Developer

운용자컴퓨터장치 개발자를 위한 모의환경은 Fig. 14와 같이 노트북 각각에 설치된 시스템 제어장치 모의기, RF 구성품 모의기, 실제 개발 중인 운용자컴퓨터장치를 하나의 스위칭허브에 연결하여 구성한다.

운용자컴퓨터장치의 정상 동작 여부를 확인하기 위하여 운용자컴퓨터장치에서 RF 구성품 제어 메시지를 시스템 제어장치 모의기로 전송하고, 시스템 제어장치 모의기는 해당 메시지를 변환하여 RF 구성품 모의기로 전달한다. 운용자컴퓨터장치 개발자는 RF 구성품 모의기에서 수신된 제어 명령을 확인하고, RF 구성품 모의기에서 전송하는 결과는 운용자컴퓨터장치에서 정상 동작 여부를 확인한다. 이러한 과정을 제어 명령 전체에 대해서 수행을 하게 되면 운용자컴퓨터장치 개발자 입장에서는 수신한 데이터와 송신한 데이터에 대해서 소프트웨어 연동통제문서를 검증할 수 있게 된다.

2. Weapon System Development stage 6-7 Support Simulation Scenario

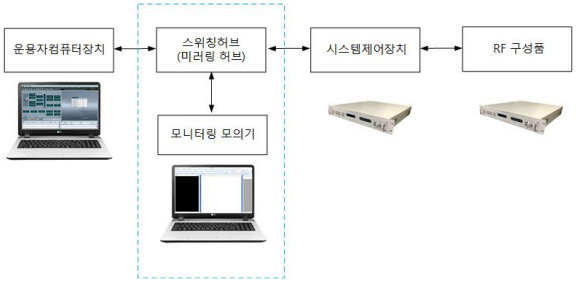

Fig. 15. Environment for System Integration

무기체계 개발 4~5단계에서 구성품들 단독으로 소프트웨어 연동통제문서를 검증하여 시스템 통합을 위한 무기체계 6단계에 진입하면 Fig. 15와 같이 실제 구성품들과 모니터링 모의기를 연결하여 구성한다.

미러링 허브는 지정된 포트로 들어오는 이더넷 패킷을 복사하여 또 다른 지정된 포트로 복사된 이더넷 패킷을 전송하는 기능을 수행한다. 모니터링 모의기는 ESICD에 정의된 패킷들이 올바르게 송/수신 되고 있는지 확인하는 역할을 수행한다. 이러한 미러링 허브와 모니터링 모의기의 기능을 활용하여 위와 같은 시스템을 구성하고, 통합시험 전에 사전 시험을 수행하면 구성품, 시스템 제어장치, 운용자컴퓨터장치 간 오류를 조기에 발견할 수 있고, 발생한 오류에 대해서 명확하게 원인을 규명을 하고 문제를 해결할 수 있다.

3. Evaluation

본 논문에서 제안한 E2V 시스템은 군 위성통신체계-II 체계 개발 4단계~7단계까지 적용하여 ESICD 검증 효용성을 확인하였다. Table 1과 같이 계발 4단계~6단계에서는 기능 추가 및 변경으로 인한 ESICD 개정이 이루어지기 때문에 문법오류와 바이트 오류에 대한 검증이 70%의 빈도로 사용되었으며, 바이트 오더링과 비트 오류의 경우에는 개발단계 4~5단계에서 개발자들이 빈번하게 발생하는 오류였고, 개발단계 7단계에서는 시퀀스 오류에 대한 검증이 주로 이루어 졌다.

각 항목에 대한 평가는 해당 항목의 오류를 사용자가 확인할 수 있는 단계를 제안하는 시스템과 기존방식(수동)으로 비교하였다. 예를 들어 바이트 오더링의 오류 여부를 검증하기 위해서 기존 방식에서는 패킷 캡쳐→ ESICD확인 → 필드위치 확인의 3단계를 거치는 반면 제안하는 시스템은 Fig .8과 같이 패킷 캡쳐 1단계만으로도 해당 항목을 검증할 수 있다.

제안하는 시스템은 평균적으로 기존 방식에 비해 약 2배정도 검증 단계를 줄였다. 그러나 시간적인 측면에서는 기존 방식은 ESICD 문서를 찾는 시간과 패킷을 분석하는 시간이 소요되기 때문에 제안하는 시스템을 활용하여 개발단계에서 많은 시간을 절약할 수 있었다.

Table 1. System Evaluation Result

검증항목	E2V시스템 (Depth)	수동 (Depth)	사용빈도 (%)
문법오류	2	3	50
바이트 오류	2	3	20
바이트 오더링	1	3	15
비트오류	2	4	5
시퀀스 오류	1	2	10
평균	1.60	3.00	

V. Conclusions

본 논문에서는 무기체계 개발 간 시스템 통합 시험이 개발자들의 단순한 실수(바이트 오더링, 비트 해석 오류 등)나 오류에 대해서 즉각적으로 원인을 발견하지 못한 경우 때문에 지연되는 상황을 사전에 방지하기 위하여 소프트웨어 연동통제문서 검증 시험을 무기체계 개발 4단계부터 소프트웨어 개발자들이 연구실 환경에서 시스템 환경을 구축하고 충분한 검증 시험이 가능한 전략을 제시하였다.

또한 제안한 E2V 시스템은 군 위성통신체계-II 체계 개발 4단계~7단계까지 적용하여 ESICD 검증 효용성을 확인하였으며, 제안하는 시스템은 평균적으로 기존 방식(수동)에 비해 약 2배정도 검증 단계를 줄였지만, 시간적인 측면에서는 기존 방식은 ESICD 문서를 찾는 시간과 패킷을 분석하는 시간이 소요되기 때문에 제안하는 시스템은 개발 단계에서 오류 발생 시 오류를 즉각적으로 발견하여 문제점을 빠르게 해결할 수 있었다.

본 논문에서 제안하는 E2V 시스템을 향후 다른 무기체계 개발 간에 적용하기 위해서는 다양한 형태로 정의되는 ESICD를 XML로 변환하는 과정에 대한 고려를 통해서 재사용이 가능할 것으로 예측된다.

REFERENCES

[1] Eunchul Park, and Song-Hwa Park, “The Improvement of Development Process for Communication Software”, The Korean Institute of Information Scientists and Engineers, Vol.36(2(B)), pp. 82-86, Nov. 2009

[2] Junyong Shim, Jiyeon Song and Sounghyouk Wi, “A Method for Processing Packet Based on Interface Control Document for Analyzing Interoperable Data of Weapon System Software”, The Korean Institute of Information Scientists and Engineers, pp. 181-183, Jun. 2013.

[3] Sungho Park, Woohyeok Chang, Hungbeen Ham, Jaesung Park and Yoonsuk Park, “DB-based Automatic Code Generation for improving the reliability of the UAV Avionics System Integration Laboratory”, The Korean Society for Aeronautical & Space Sciences, pp. 1200-1203, Nov. 2014.

[4] Jaeheon Yoo, Keunhyoung Lee and Jinmo Kim “Design of GUI-based Packet Analyzer Using WinPcap”, The Korean Institute of Information Scientists and Engineers, pp. 2042-2044, Jun. 2015.

[5] Gukbo Yang, Wanggguk Lee and Manjo Kim, "Development of System Integration Laboratory for the Verification of UAV System Interface Design", The Korean Society for Aeronautical & Space Sciences, pp. 950-953, Nov. 2013

[6] Seung-Gu Yang, Yeon-Je Cho, Kyoung-Yong Jo and Chang Myung Ryu, "Design of Automatic Model Verification for System Integration Laboratory", Advanced Navigation Technology, Vol. 23(5), pp. 361-366, Oct. 2019

[7] Byung Hoon park and Yeong Geon Seo, “Process Improvem- ent for Quality Increase of Weapon System Software Based on ISO/IEC/IEEE 29119 Test Method”, Journal of the Korea Society Computer and Information, Vol. 23(12), pp. 115-122, Dec. 2018.

[8] Sung Il, Gwangtae Kim and Bongki Lee, “A Study on the Documentation Case for Weapon System Development”, Journal of the KIMST, Vol.11(2), pp.73-79, Nov, 2008

Authors

Kee-Sung Lee received the Ph.D. degrees in Computer Science and Engineering from Inha University, Korea, in 2014. Form 2014 to 2015, he worked as a postdoctoral researcher in Inha University.

Dr. Lee joined the C4I R&D Lab of LIG Nex1, Korea, in 2015. He is currently a Chief Research Engineer in the Satellite Communication Team R&D Site, LIG Nex1 Co. He is interested in Satellite Communication, Intelligent Assistant System, Information Visualization and User Interface Design.

Jun-Ho Choi received the BS degree and MS degree in Information Communication Engineering from Inha university, Korea, in 2012 and 2014, respectively. He joined the C4I R&D Lab of LIG Nex1, Korea, in 2014.

He is currently a Senior Researcher in the Satellite Communication Team R&D Site, LIG Nex1 Co. He is interested in Satellite Communication, Cognitve Radio, Wireless Network and User Interface Design.

Jeong-Jin Shin received the BS degrees in Computer Science and Engineering from Hongik University, Korea, in 2015. He joined the C4I R&D Lab of LIG Nex1, Korea, in 2015. He is currently a Senior Researcher in the Satellite Communication Team R&D Site, LIG Nex1 Co. He is interested in Satellite Communication, Software Reliability and User Interface Design.

Hye-Jin Yoon received the BS degree in Communication Engineering from Daejin university, Korea, in 2015. She joined the C4I R&D Lab of LIG Nex1, Korea, in 2015. She is currently a Senior Researcher in the Satellite Communication & Microwave Team R&D Site, LIG Nex1 Co. She is interested in Satellite Communication, Wireless Network and User Interface Design.

Seung-Ho Kim received the M.S. degree in Information Communication Engineering from Gwangju Institute of Science and Technology, Korea in 2015. Mr. Kim joined the Agency for Defense Development(ADD), Korea in 2015. He is currently a Senior Researcher in Defense Space Technology Center, ADD. He is interested in Satellite Communication and Laser Communication.