



저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

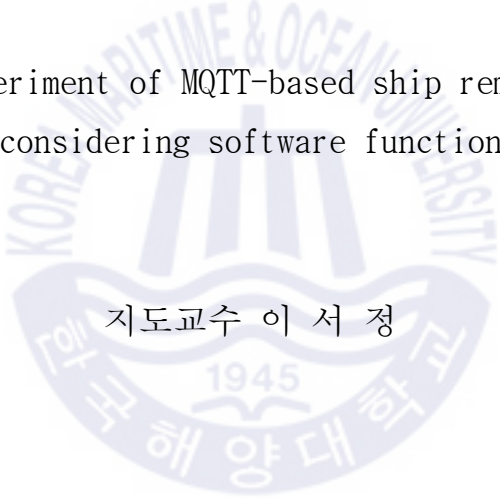
이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

공학석사 학위논문

소프트웨어 기능안전성을 고려한
MQTT 기반 선박시스템 원격 모니터링
플랫폼 설계 및 실험

Design and experiment of MQTT-based ship remote monitoring
platform considering software functional safety



지도교수 이 서 정

2020년 2월

한국해양대학교 대학원

컴퓨터공학과
이 창 의

본 논문을 이창의의 공학석사 학위논문으로 인준함

위원장 : 박 휴 찬



위원 : 이 서 정



위원 : 이 장 세



2019년 12월 26일

한국해양대학교 대학원

목 차

List of Tables	iii
List of Figures	iv
Abstract	v

1. 서 론

1.1 연구배경 및 필요성	1
1.2 연구목적 및 범위	4

2. 관련연구

2.1 선박 IT융합기술 국제표준화	5
2.2 조선/해양분야 소프트웨어 품질관리 표준화	7

3. 원격 모니터링 플랫폼 요구사항 분석

3.1 ISO 19847/19848 요구사항 분석	9
3.2 IEC 61508 요구사항 분석	12
3.3 위험 및 운용 연구(HAZOP Study) 절차 분석	14

4. 선박시스템 원격 모니터링 플랫폼 설계

4.1 ISO 19847에 대한 기능 안전 평가	18
4.2 반정형기법을 이용한 위험성 감소방안 제안	20
4.3 원격 모니터링 소프트웨어 아키텍처 설계	38

5. 시스템 구현 및 검증	
5.1 개발 환경	48
5.2 실험 환경	49
5.3 시스템 구현	52
5.4 실험 및 검증	55
6. 결론 및 향후 연구	62
참고문헌	65



List of Tables

Table 1	Configuration of IEC 61508	12
Table 2	Hazard Severity (referenced IEC 61126)	16
Table 3	Hazard probability(referenced MIL-STD-882C)	16
Table 4	Matrix for risk level determination	17
Table 5	Definition of process parameters	19
Table 6	Definition of guide words	19
Table 7	Declaration of deviation	20
Table 8	Identify cause of deviation	16
Table 9	Identify consequence of deviation	18
Table 10	Determine risk level	19
Table 11	Identify safeguards of deviation	20
Table 12	Semi-formal methods	21
Table 13	Definition of message type	39
Table 14	Additional definition of message type	40
Table 15	Additional definition of configuration code	40
Table 16	Software development environment	48
Table 17	Experimental variable of data input/output control	56
Table 18	Experimental variable of error checking mechanism	57
Table 19	Experimental variable of reconnection mechanism	60

List of Figures

Fig. 1 International Shipbuilding · IT standardization organization	5
Fig. 2 Industry standards based on IEC 61508	7
Fig. 3 ISO 19847/19848 concept model	10
Fig. 4 MQTT communication method	11
Fig. 5 HTTP communication method	11
Fig. 6 IEC 61508-3 Software development life cycle	13
Fig. 7 HAZOP study process	14
Fig. 8 High risk deviations, causes and safeguards	20
Fig. 9 Frequently publish of MQTT protocol	22
Fig. 10 Time-Petri diagram of high publish situation	23
Fig. 11 Sequence of minimum publish interval setting	24
Fig. 12 Communication failure between ship and shore	25
Fig. 13 Sequence of QoS0 in MQTT protocol	26
Fig. 14 Sequence of QoS1 in MQTT protocol	27
Fig. 15 Sequence of QoS2 in MQTT protocol	28
Fig. 16 Suggest of error checking mechanism between ship and shore ...	29
Fig. 17 TCP connection process	30
Fig. 18 TCP disconnect process	31
Fig. 19 Diagram of TCP state transition	32
Fig. 20 Sequence of acceptance of TCP server	33
Fig. 21 Petri-Net diagram of frequently reconnect situation	34
Fig. 22 Improve of reconnection sequence	35
Fig. 23 Termination of connection due to an error of the ship server ...	36

Fig. 24	Countermeasure in Server error situation through system redundancy ..	37
Fig. 25	Structure of MQTT protocol	38
Fig. 26	Structure of fixed header	39
Fig. 27	Structure of variable header	39
Fig. 28	Sequence of applying minimum publish interval and reconnection interval ..	41
Fig. 29	Structure of fixed header of PUBACK message	42
Fig. 30	Structure of variable header of PUBACK message	42
Fig. 31	Suggest of QoS1 sequence in MQTT protocol	43
Fig. 32	Suggest of QoS2 sequence in MQTT protocol	44
Fig. 33	Concept of Load Balancer	45
Fig. 34	Architecture of system redundancy	46
Fig. 35	Error handling procedure by system redundancy	47
Fig. 36	Diagram of experimental environment	49
Fig. 37	Facility of experiment	50
Fig. 38	Device placement inside the control room	51
Fig. 39	MQTT subscriber devices in an office environment	51
Fig. 40	Screenshot of MQTT Publisher	52
Fig. 41	Screenshot of MQTT Broker	53
Fig. 42	Screenshot of MQTT Subscriber	54
Fig. 43	Experimental result of data input/output control	56
Fig. 44	Results of applying error checking mechanism in QoS1	58
Fig. 45	Results of applying error checking mechanism in QoS2	58
Fig. 46	Result of data usage applying error checking mechanism in QoS1 ...	59
Fig. 47	Result of data usage applying error checking mechanism in QoS2 ...	59
Fig. 48	Result of data usage applying reconnection mechanism	61
Fig. 49	Number of ports allocated of applying reconnection mechanism ..	61

소프트웨어 기능안전성을 고려한 MQTT 기반 선박시스템 원격 모니터링 플랫폼 설계 및 실험

이 장 의

한국해양대학교 대학원

컴퓨터공학과

초 록

최근 자율운항 선박 및 원격 유지보수에 대한 요구가 높아지면서 선박용 장비 제조사들은 문제가 발생하면 수리를 하거나 정기점검을 통해 관리하던 기존 방식에서 벗어나 자사 제품을 원격에서 모니터링하고 자사 제품과 연관된 타사제품들을 공통 데이터 통신 프로토콜을 기반으로 통합하여 관리하려는 움직임을 보이고 있다. 이에 따라 국제표준화기구(ISO, International Organization for Standardization)에서는 선박시스템들의 데이터를 수집하고 원격지에서 관리하기 위한 해상데이터 공유 표준인 ISO 19847/19848을 2018년 10월에 제정하였다.

본 논문에서는 ISO 19847/19848이 산업현장에 적용되기에 앞서 소프트웨어 기능안전성 측면에서 HAZOP(Hazard and Operability Study) 방식의 위험성 분석을 수행하여 위험 및 안전대책을 도출하였다. 도출된 안전대책에 대해 IEC 61508-3의 반정형기법(Semi-formal methods)을 이용하여 위험을 구체적으로 분석하고 위험성 감소방안을 제안하였다. 위험성 감소방안에 따라

시스템을 설계하고 MQTT(Message Queue Telemetry Transport) 프로토콜 기반 선박시스템 원격 모니터링 플랫폼을 개발하여 실험을 통해 이를 검증하였다.

이를 위하여 관련된 선행 연구 자료를 수집하여 분석하고 원격 모니터링 플랫폼의 데이터 교환 표준인 ISO 19847/19848의 요구사항을 분석하였다. 그리고 소프트웨어 기능안정성 평가를 위해 관련 표준인 IEC 61508의 요구사항과 HAZOP 절차를 분석하였다.

본 논문에서 제안하고자 하는 내용의 효용성에 대해 더욱 신뢰할 수 있는 자료를 확보할 수 있도록 실제 선박을 이용하여 실험 환경을 구축하고 황산화물 저장시스템의 운용 데이터를 수집하여 이를 원격으로 모니터링하는 실험을 수행하였다.



키워드 : ISO 19847, IEC 61508, 기능안정성, HAZOP, MQTT

Design and experiment of MQTT-based ship remote monitoring platform considering software functional safety

Lee, Chang Ui

Department of Computer Engineering,
Graduate School of Korea Maritime and Ocean University

Abstract

With the recent increase in demand for autonomous ships and remote maintenance, ship equipment manufacturers try to remotely monitor their products and integrate related third-party products based on common data communication protocol, instead of repairing them or managing them through regular inspections when problems arise. Accordingly, the International Organization for Standardization(ISO) established ISO 19847/19848 in October 2018, a maritime data sharing standard for collecting and remotely managing ship systems data.

In this paper, before applying ISO 19847/19848 to industrial sites, risk and safety measures were derived by performing HAZOP(Hazard and Operability Study) risk analysis in terms of software functional safety. The derived safety measures were analyzed in detail using the semi-formal methods of IEC 61508-3 and suggested ways to reduce the

risk. The system was designed according to the risk reduction method, and the MQTT(Message Queue Telemetry Transport) protocol based ship system remote monitoring platform was developed and verified through experiments.

To do this, we collected and analyzed relevant prior research data and analyzed the requirements of ISO 19847/19848, the data exchange standard for remote monitoring platforms. In addition, the requirements of IEC 61508 and HAZOP procedure were analyzed to evaluate software functional safety.

In order to obtain more reliable data on the utility of the contents proposed in this paper, we conducted an experiment to establish a test environment using real vessels and collect the operating data of De-SOx Scrubber system and monitor it remotely.

KEY WORDS : ISO 19847, IEC 61508, Functional Safety, HAZOP, MQTT

제 1 장 서 론

1.1 연구배경 및 필요성

최근 ICT 융합기술이 선박과 해양분야에 빠르게 적용되고 있지만, 선박 사고로 인한 대형 인명, 경제적 및 환경적 피해에 대한 염려에 따라 등장한 자동화선박과 무인 항해 관련 기술이 상용화되기까지는 아직 어려움이 남아있었다. 하지만, 2014년 5월 영국의 롤스로이스에서는 10년 이내에 무인화물선을 상용화한다고 발표한 이후 노르웨이, 핀란드를 비롯한 유럽 국가들과 중국 및 일본을 비롯한 많은 나라가 차례로 자율운항선박 계획을 발표하고 있으며, 각 나라와 회사들이 미래의 핵심 사업으로 중점적으로 개발하고 있다[1]. 자율운항선박에 대한 국제적 차원의 논의는 2017년 6월에 개최된 제98차 국제해사기구(IMO, International Maritime Organization) 해사안전위원회(MSC, Maritime Safety Committee)에서 자율운항선박을 MASS(Maritime Autonomous Surface Ship)로 규정함으로써 본격화되었다[2].

MASS는 자율적으로 운항하는 선박을 만드는 것이 아니라, 자율운항에서 안전성(Safety), 신뢰성(Reliability), 효율성(Efficiency)을 위한 단계적 고도화 범위 내에서 선박의 무인화, 자율화 및 운송의 효율화를 위한 종합적인 제품 및 서비스라고 할 수 있다. 궁극적으로 MASS는 초지능화, 초고속화, 디지털화를 바탕으로 선박의 모든 데이터가 실시간으로 상호 연계되는 초연결성을 바탕으로 무인화를 통한 해운/조선산업의 새로운 패러다임의 전환이 될 것이다[3].

육상과 선박의 연결은 MASS 운용 발전에 있어서 가장 중요한 문제라고 파악되고 있으며, 이를 위하여 국제 표준화 기구에서는 다양한 형태의 표준을 제정하거나 개정하고 있다.

선박에서 항해통신장비와 관련된 부분은 e-navigation 패러다임의 진행에 따라 메이커와 선사를 중심으로 필요성이 인식되어 표준 제정 및 관련 연구가 활발하게 진행되고 있다. 하지만, 선박은 물 위를 떠다니는 건축물이라 할 만큼 항해와 관련된 시스템을 제외하고도 전력관리시스템, 수화물 관리시스템 등의 다양한 시스템들이 존재하고 있다.

최근 들어 자율운항 선박 및 원격 유지보수에 대한 요구가 높아지면서, 선박용 장비 제조사들은 문제가 발생하면 수리를 하거나 정기점검을 통해 관리하던 기존 방식에서 벗어나 자사 제품을 원격에서 모니터링하고 자사 제품과 연관된 타사제품들을 공통 데이터 통신 프로토콜을 기반으로 통합하여 관리하려는 움직임을 보이고 있다. 하지만, 이 플랫폼들은 표준화되어 있지 않고 고유의 인터페이스를 가지고 있는 경우가 대부분으로 회사간 또는 시스템간 플랫폼 호환성이 없으므로 플랫폼이라 칭하기 어려운 실정이다. 또한, 소프트웨어 개발에 있어 확장성을 고려하지 않은 경우가 많아 연동되는 시스템을 추가할 경우 시스템의 수정이 많이 필요하며 유지보수가 어려운 측면이 있다.

이에 따라 최근 국제표준화기구(ISO, International Organization for Standardization)에서는 선박시스템들의 데이터를 수집하고 원격지에서 관리하기 위한 해상데이터 공유 표준인 ISO 19847 선박 및 해양기술-해상에서 필드 데이터 공유를 위한 선박 데이터 서버(Ships and marine technology-Shipboard data servers to share field data at sea)와 ISO 19848 선박 및 해양기술-선박용 기기 및 장비에 대한 표준 데이터(Ships and marine technology-Standard data for shipboard machinery and equipment)를 2018년 10월에 제정하였고, IMO에서는 지속 가능한 소프트웨어를 개발하기 위해 IMO MSC/Circ. 1512 e-navigation을 위한 소프트웨어 품질인증 및 인간 중심 설계에 관한 가이드라인(Guideline on software quality assurance and human-centered design for e-navigation)을 도입하는 등 선박시스템의 데이터를 육상으로 송신하기 위한 표준화 활동이 진행 중이다. 특히 IMO MSC/Circ. 1512 가이드라인에서는 소프트웨어 설계 시 고려되어야 하는 품질 특성으로 제품 및 데이터 품질, 사용자 요구 충족, 보안, 기능안전성에 대해 강조하고 있다.

자동차, 철도, 항공, 원자력, 국방 분야와 마찬가지로 해양분야에도 소프트웨어 안전성은 중요한 요소이며 소프트웨어에 의한 사고를 예방하기 위한 소프트웨어 안전성에 관한 연구는 소프트웨어 개발단계에서 매우 중요한 부분이다. 소프트웨어에 의한 사고는 조작 미숙 혹은 외적인 요인에 의해서 발생할 수도 있지만, 가장 근본적인 원인은 소프트웨어 개발단계에서의 안전성에 대한 검증이 제대로 이루어지지 않기 때문에 발생하게 된다. 소프트웨어 분석 및 설계 단계에서 안전성에 관한 연구는 대부분 안전성 분석 및 평가, 또는 소프트웨어 아키텍처 설계기법을 통한 안전성의 확보를 목표로 수행하는 것이 일반적인 추세이다.

가장 많이 알려진 소프트웨어 안전성 분석 기법은 시스템에서 발생할 가능성이 있는 고장의 형태와 영향을 분석하는 고장유형 및 영향분석(Failure-Mode & Effect Analysis, FMEA)과 서브시스템이 전체 시스템에 어떠한 결과를 미치는지 분석하는 결함 위험분석(Fault Hazard Analysis, FHA), 특정한 사고에 대하여 그 사고의 원인이 되는 기기의 결함이나 운전자의 실수 등을 하향식(Top-Down)으로 찾아가는 결함 트리 분석(Fault Tree Analysis, FTA), 대상 시스템에 대한 전문가들이 모여서 자료를 토대로 위험요소들과 운전상의 문제점을 찾아내어 그 위험성을 제거하는 위험성 평가방법인 위험 및 운용 분석 (Hazard and Operability analysis, HAZOP) 등이 있다.

자율운항선박은 하드웨어와 소프트웨어 간 상호 연계성을 통한 통합의 집합체로 볼 수 있으며 특히 소프트웨어의 역할이 점점 더 증대되고 있다. 자율운항선박을 위해서는 무엇보다 시스템이 안전해야 하며, 만약 문제가 발생하더라도 큰 사고로 이어지지 않고 적절하게 조치가 되어야 한다. 즉, 소프트웨어의 기능안전성이 확보되어야만 자율운항선박의 안전한 항해가 보장된다는 것이다. 최근 시스템 간의 데이터교환과 통합을 위한 다양한 표준들이 제정되고, 자율운항선박을 위한 테스트베드가 구축되고 있는 시점에서 시스템 안정성 증대를 위해 국제표준과 시스템에 대해 소프트웨어 기능안정성을 고려해야 할 필요성이 강조되고 있다.

1.2 연구목적 및 범위

자율운항선박을 실현하기 위한 기술은 크게 육상에서의 원격 모니터링 기술과 선박에서의 자율운항시스템기술과 선박과 육상을 연결해 주는 해상 연결성 기술로 구분할 수 있다.

육상 원격 모니터링 기술에서 대상이 되는 시스템은 선박의 운항과 관련된 항해통신시스템과 그 외 선박시스템으로 나누어 볼 수 있다. 항해통신시스템의 원격 모니터링은 e-navigation으로 불리는 새로운 패러다임을 통해 관련 연구와 실체화가 진행되고 있으므로 본 논문에서는 항해통신시스템을 제외한 선박시스템의 원격 모니터링에 대해 논하고자 한다.

본 논문에서는 해상데이터 공유표준인 ISO 19847/19848을 중심으로 하여 선박시스템들의 데이터를 수집/저장하고 원격지에서 모니터링하기 위해 고려해야 할 국제표준들에 대해 분석하고, 이를 바탕으로 소프트웨어 기능안정성 측면에서 위험성 분석을 수행하고 이를 극복하기 위한 방법을 제시하고자 한다. 이를 바탕으로 선박시스템 원격 모니터링 플랫폼의 기능을 수정 설계하고 황산화물 저감시스템을 사례로 제안한 방법이 타당함을 검증하고자 한다.

본 논문의 구성은 다음과 같다. 2장에서는 본 논문과 관련된 연구와 국제표준 동향을 파악하여 현황을 확인한다. 3장에서는 선박 원격 모니터링과 관련한 국제표준과 소프트웨어의 품질과 관련한 국제표준을 분석하여 요구사항을 도출한다. 4장에서는 2~3장까지 분석한 내용을 토대로 요구사항의 기술적 문제점을 정의하고 극복방안을 제시한 후 플랫폼 아키텍처를 설계한다. 5장에서는 설계된 아키텍처를 실체화하고 검증하기 위하여 황산화물 저감시스템을 사례로 플랫폼 아키텍처를 구현하고 검증한다. 6장에서는 본 연구 논문의 결론과 앞으로의 남은 연구를 살펴보도록 한다.

제 2 장 관련 연구

조선해양기자재산업은 선박이 세계의 여러 나라를 항해하고 다니는 특성에 따라 전통적으로 국제법의 규제를 받는다. 이는 선박의 장비나 시스템들이 국제 표준의 요구사항을 준수하고 이에 맞는 장비의 개발이 필요하다는 것을 의미한다. 따라서 선박시스템의 개발 및 연구를 위해서는 표준에 대한 정확한 이해가 필요하다.

본 장에서는 주요 국제 표준화 기구들의 표준화 제정 동향을 선박 IT 융합기술과 소프트웨어 품질관리로 나누어 살펴보고, 산업계 동향을 알아보려고 한다.

2.1 선박 IT융합기술 국제표준화

최근 정보통신기술의 발전으로 인해 선박에 디지털 및 전자장비들이 도입되었고, 폐쇄적인 구조에서 IT기반의 통합형 형태로 바뀌어 가고 있다. 이에 따라서 국제 표준화 기구들에서는 다양한 표준들을 제정 또는 개정 중이다.

선박과 관련된 주요 국제 표준화 기구들은 Fig. 1에서와 같이 선박 및 해양과 관련된 국제 규약을 담당하고 있는 IMO와 이를 지원하기 위한 국제 표준화 기구들, 그리고 국제 민간 표준 단체들과 이에 대응하기 위한 국내 표준 단체들로 구분된다[4].

International
Regulations



International
Standards



International
Telecommunication
Union



International
Hydrographic
Organization



International
Organization for
Standardization



International
Electrotechnical
Commission

Industrial
Standard



National Marine
Electronics Association

Fig. 1 International Shipbuilding · IT standardization organization

IMO는 해운과 조선에 관한 국제적인 문제들을 다루기 위해 설립된 국제기구로 유엔(UN)의 산하기관으로 각국의 정부만이 회원 자격이 있는 정부간 기구으로써 해양 및 선박관련 국제 규약을 제정하는 업무를 수행한다[1].

2008년 12월 제 85차 MSC에서 e-Navigation 이행전략이 승인되어 단계별 이행전략에 따라 선박의 안전운항에 관한 규약이 채택되었고 이를 지원하기 위해 표준화 기구들에서는 이더넷 기반 항해통신장비 데이터교환 표준인 ISO/IEC 61162-450과 네트워크 안정성 및 보안표준인 ISO/IEC 61162-460 표준을 비롯한 항해통신장비와 관련된 표준문서를 제정하였다.

최근 자율운항선박이 대두됨에 따라 IMO에서는 2018년 5월에 제 99차 MSC에서 자율운항선박에 대한 정의와 자율화 정도에 대하여 잠정적으로 결정하고, 자율운항선박에 대한 기술개발을 고려하여 도입에 따른 법적근거와 기술표준을 만들기 위한 전 단계로 현행 국제협약의 적용과 이행을 위하여 필요한 사항과 조치에 대하여 작업반(WG)과 통신작업반(CG)을 통해 작업을 수행하고 있다.

국제전기연합(IEC, International Electrotechnical Commission)은 주로 해상 무선통신 및 항해 장비와 시스템 기술위원회(TC80)에서 선박에 탑재되는 장비들과 장비들 간의인터페이스 표준에 대한 기능과 테스트 방안에 대한 표준을 다루고 있다. 그리고, 국제표준화기구인 ISO에서는 주로 선박의 설계 및 항해 통신장비와 무선 통신 장치를 제외한 선박의 장치들에 대한 표준을 정의하고 있다. 특히, ISO 선박 기술 전문위원회(TC8)에서 주로 관련 표준을 다루고 있다[1].

IMO의 자율운항선박 이행이 결정됨에 따라 e-navigation 이행전략으로 인해 표준화가 잘 이루어진 항해통신장비에 비해 선박시스템들은 상대적으로 표준화가 미비하여 데이터 교환 및 관리에 한계가 존재하였고 이를 해결하기 위하여 ISO에서는 선박의 모든 장치로부터 데이터들을 모으기 위한 선상 및 육상서버와 이들 간의 통신을 위한 구조 및 데이터 인터페이스에 대한 표준으로써 ISO 19847과 ISO 19848 표준을 2018년 10월에 정식 표준으로 공표하였다. 그리고, 선박의 시스템들이 통합되고 네트워크로 연결됨에 따라 엔진과 항해 네트워크를 제외한 선박시스템들을 설치하고 통합하기 위한 가이드라인인 ISO 16425가 2013년 2월에 제정되어 발표되었다.

2.2 조선/해양분야 소프트웨어 품질관리 표준화

조선/해양분야의 사용자들에게 지속 가능하고 품질이 좋은 소프트웨어와 사용하기 편한 시스템을 제공하기 위해 e-navigation에서는 소프트웨어 품질 보증을 강조하고 있다. 2015년 6월 MSC 95차 회의에서 MSC/Circ. 1512를 표준 문서로 최종 승인하였다. 이 문서에는 시스템품질 요소 중의 하나로 기능안전성을 언급하고 있으며 일반적으로 기능안전성을 확보하기 위해서는 IEC 61508 전기/전자/프로그램 가능한 전자 안전 관리 시스템의 기능 안전(Functional Safety of electrical/electronic/programmable electronic safety-related systems)과 산업분야별 정의된 특정 표준을 따르는 것을 권장하고 있다[5].

IEC 61508에서는 기능안전성을 소프트웨어의 기능에 위험이 생기거나 사고가 날 염려로부터의 자유로 정의하고 있다. 2000년 초반부터 원전, 항공, 의료, 철도, 장치산업 등의 다양한 분야에서 IEC 61508을 기반으로 하는 안전 표준들을 작성하여 기능안전, 신뢰성, 품질 및성능에 대한 검증을 요구하고 있으며 점차 모든 분야로 확산하고 있다. Fig. 2는 IEC 61508을 기반으로 하는 각 분야의 표준들을 나타낸다[6].

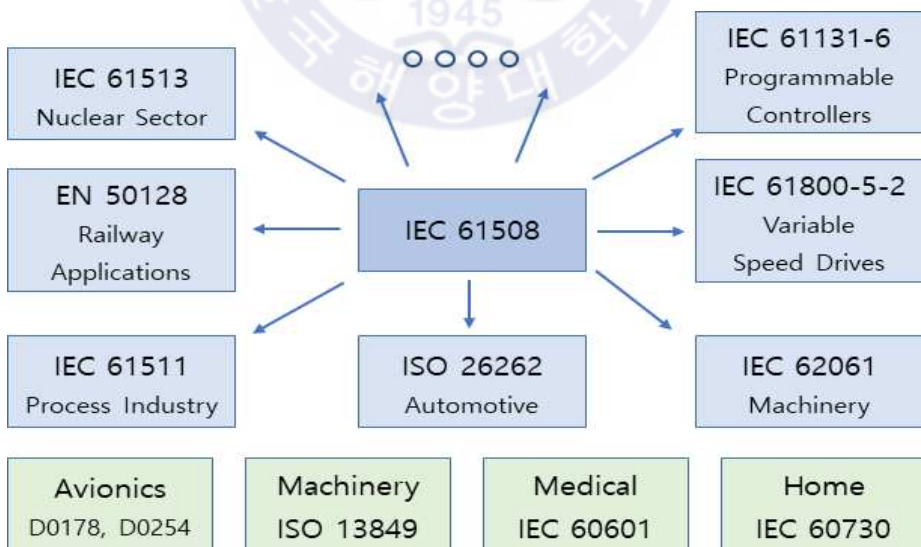


Fig. 2 Industry standards based on IEC 61508

IEC 61508에서는 기능안전성 확보를 위한 방법을 다루었지만 표준문서의 특성상 필요한 산출물만이 제시되어 있을 뿐 실행방법에 대한 명세는 작성되어 있지 않다. 미래창조과학부 산하 정보산업진흥원(NIPA, National IT Industry Promotion Agency)에서는 기능안전성을 국내의 모든 분야의 소프트웨어에 도입하고자 소프트웨어 안전성 공통 개발 가이드라인을 개발하였다. 소프트웨어 안전성 공통 개발 가이드라인은 현장에서 활용하기 위해 개념 제시 수준을 넘어 실제 실무에 활용할 수 있도록 구체적인 방안을 제시하였다.

하지만, 조선해양 분야에서는 IEC 61508을 기반으로 하는 안전표준이 없을 뿐만 아니라 NIPA의 소프트웨어 안전성 공통 개발 가이드라인에서도 조선/해양 분야에 대해서는 언급을 하고 있지 않으므로, 조선/해양분야 기술 표준화는 ISO에서 관리하는 표준규격과 IMO에서 결의되는 강제적 규정만을 포함하여 이루어지고 있다.



제 3 장 원격 모니터링 플랫폼 요구사항 분석

원격 모니터링 플랫폼 개발을 위해서는 기본적으로 기술적 요구사항을 파악하고 있어야 하며 지속 가능한 플랫폼 소프트웨어 개발을 위해서는 소프트웨어 품질을 고려한 플랫폼 설계가 바탕이 되어야 한다.

본 장에서는 원격모니터링 플랫폼의 기술적 요구사항인 ISO 19847/19848과 소프트웨어의 기능안정성 확보를 위한 IEC 61508에서의 요구사항을 살펴보도록 한다.

3.1 ISO 19847/19848 표준 분석

ISO 19847은 해상 필드 데이터 공유를 위한 선박 데이터 서버에 관한 국제표준으로 선박 내 시스템에서 생성된 데이터를 저장하고 육상으로 전달하는 데이터 서버로서 기능, 성능, 서비스 및 안전요건에 대해 제시하고 있다. 또한, ISO 19848은 선박시스템과 데이터 서버간 데이터 교환에 관한 표준으로 선박에서 데이터를 교환하고 처리하는 것을 용이하도록 하기 위해 시스템을 분류하여 구분할 수 있는 데이터 규칙 및 식별자를 정의하고 있다[7].

Fig. 3은 두 표준의 개념모델로써 (A) 부분은 ISO 19847의 개념모델로 입력기능, 출력기능, 데이터 저장소로 구성된다. 수신된 데이터는 기본적으로 저장소에 저장되도록 정의하고 있으며 스트리밍으로 받은 데이터를 중계하여 스트리밍 서비스를 수행하고 요청에 따라 저장된 데이터를 검색하여 응답하는 서비스를 수행한다. (B) 부분은 ISO 19848의 개념모델로 입력데이터와 출력데이터의 형식에 대해서 정의하고 있다. 계층적 구조를 통해 데이터를 추상화하고 데이터의 형식과 종류, 범위 등을 정의할 수 있도록 데이터 구조를 정의하고 있다[7][8].

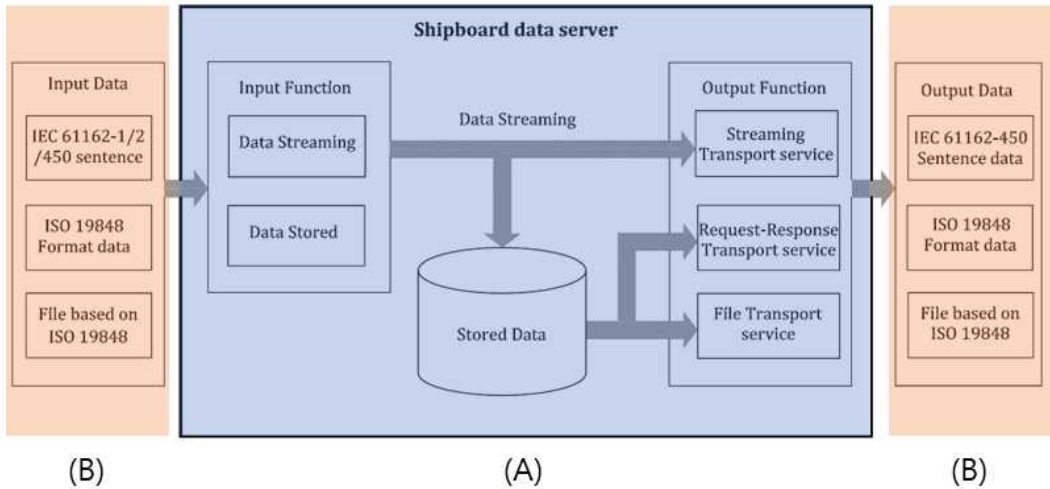


Fig. 3 ISO 19847/19848 concept model

ISO 19847에서는 선박에서 육상으로 데이터를 전송하는 출력기능을 3가지 방식으로 제안하고 있다.

(1) 스트리밍(Streaming) 방식으로 수신된 데이터를 지연 없이 중계하는 방법으로 MQTT 프로토콜을 사용하도록 제안하고 있다.

MQTT는 2013년 OASIS(Organization for the Advancement of Structured Information Standards)에서 IoT를 위한 표준 프로토콜로 발표되었으며 M2M(Machine to Machine), IoT(Internet of Things) 등과 같이 대역폭이 낮은 장치들에 적합한 프로토콜로써 최소한의 전력과 패킷량으로 통신하는 프로토콜이다. 프로토콜이 비교적 가벼워서 IoT와 같은 사양이 높지 않은 장비의 메시지 교환을 위해서 많이 사용되고 있다.

MQTT는 전통적인 클라이언트-서버 구조가 아닌 중계자(Broker), 출판자(Publisher), 구독자(Subscriber) 구조로 이루어져 있다. Fig. 4에서와 같이 구독자가 중계자가 중계 서버로 존재하는 상태에서 구독자가 서버에 주제(Topic)를 구독하고자 신청하고, 출판자가 서버에 주제를 발행하면 이를 구독하는 구독자에게 메시지를 전달해 주는 방식이다.

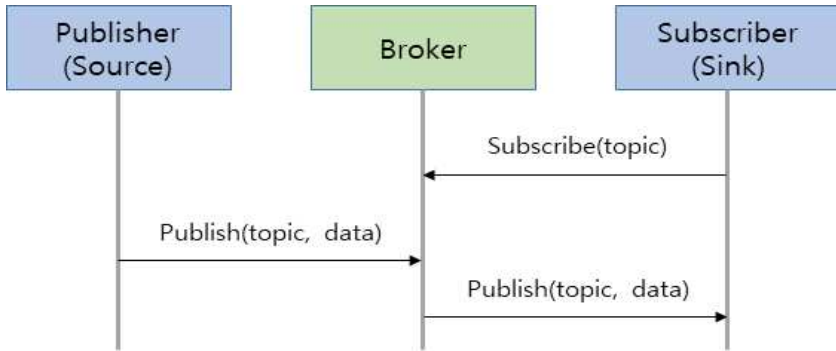


Fig. 4 MQTT communication method

(2) 요청-응답 방식으로 사용자의 요청에 따라 데이터 저장소에서 조회하여 응답하는 방식으로 REST API를 사용한 HTTP 프로토콜을 제안하고 있다.

HTTP(Hyper Text Transfer Protocol)는 전형적인 서버-클라이언트 구조로 Fig. 5에서와 같이 클라이언트가 정보를 요청하면 서버는 요청에 따른 응답을 클라이언트에 전송한다. 연결을 계속 유지하는 것이 아니라 클라이언트의 요청에 응답한 후 바로 연결을 끊어버리는 비 연결성(Connectionless)이 특징이다.

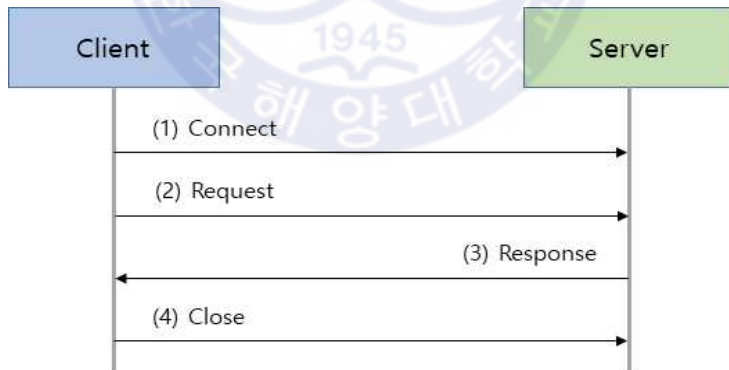


Fig. 5 HTTP communication method

(3) 파일 전송방식으로 사용자의 요청에 따라 데이터 저장소에서 조회하여 파일을 생성하여 HTTP 프로토콜이나 FTP 프로토콜을 이용하여 파일 형태로 전송하는 방식이다.

3.2 IEC 61508 요구사항 분석

IEC 61508은 전체 산업 분야의 기능 안전 개념을 소개하는 표준으로 전기, 전자 및 프로그램 가능한 전자 기기(Electrical/Electronic/Programmable Electronic Safety-related Systems)를 개발할 때 필요한 기능 안전을 정의하고 안전 요구사항을 도출하며, 안전에 관련된 기능과 안전무결성의 관계를 기술한다. 이 표준은 전 산업을 대상으로 하며, 특히 인간의 안전과 관련된 분야를 중심으로 한다. 즉, 어떤 시스템의 고장이나 결함으로 인해 인간에게 위해(harm)를 미칠 수 있는 경우는 모두 대상이 된다. 총 7개의 파트로 구성되어 있으며, 각 파트별 내용은 Table 1과 같다[6].

Table 1 Configuration of IEC 61508

Part	Contents
IEC 61508-1	일반적인 요구 사항
IEC 61508-2	E/E/PE 안전 관련 시스템을 위한 요구 사항
IEC 61508-3	소프트웨어 요구 사항
IEC 61508-4	정의 및 약어(Definitions and Abbreviations)
IEC 61508-5	SIL의 결정을 위한 방법에 대한 예시
IEC 61508-6	IEC 61508-2, 3의 적용 가이드 라인
IEC 61508-7	IEC 61508에서 요구하는 기술 및 수단의 개요

IEC 61508에서는 안전수명주기를 통하여 시스템의 개념단계에서 구현, 양산, 관리, 폐기에 이르기까지의 전사적 과정을 명시하고 있다. 안전제어시스템은 개발부터 운영환경 및 다양한 상황에서 잠재된 위험원을 도출하고 위험원을 허용할 수 있는 수준까지 감소시키고자 한다[6]. 이처럼 위험원을 도출하고 감소시키기 위한 다양한 기법들이 개발 및 발전되고 있으며, 도출된 요구사항을 바탕으로 이에 적합한 안전제어시스템이 설계되었는가를 하드웨어 및 소프트웨어 나아가 시스템 수준에서의 평가가 이뤄지게 된다.

이러한 안전무결성의 도출 및 평가의 기준으로 IEC 61508은 안전 무결성 수준(SIL, Safety Integrity Level)을 명시하고 있다. 이는 안전제어시스템의 안전무결성으로서, 위험원에 따라 요구되는 수준이 결정되고 이는 다시 정성적 및 정량적인 기법들을 통하여 평가된다. 이때의 SIL은 4등급으로 구분되고 높은 등급을 가질수록 엄격한 수준의 안전무결성이 요구된다[9].

IEC 61508-3에서는 소프트웨어를 설계하는데 요구되는 활동 및 설계기술의 요구사항들이 정의되어 있으며 소프트웨어 개발 수명 주기 참조모델은 Fig. 6에서 보는 바와 같이 전통적인 V-모델에 안전과 관련된 활동을 추가하여 안전이 검증된 소프트웨어를 개발하도록 하고 있다. 품질과 안전 보증절차는 안전 수명주기의 각 활동과 통합되어야 하며, 소프트웨어 수명주기의 각 단계는 입력과 그 결과인 산출물이 명확히 구분되어야 한다. 하지만, 표준에서 제시하는 수명주기 단계들은 대체로 규모가 큰 시스템에 적합하므로 프로젝트의 복잡성과 안전무결성 정도에 따라서 V모델 단계들의 정도와 수, 작업 규모를 조정할 수 있도록 하고 있다[10][11].

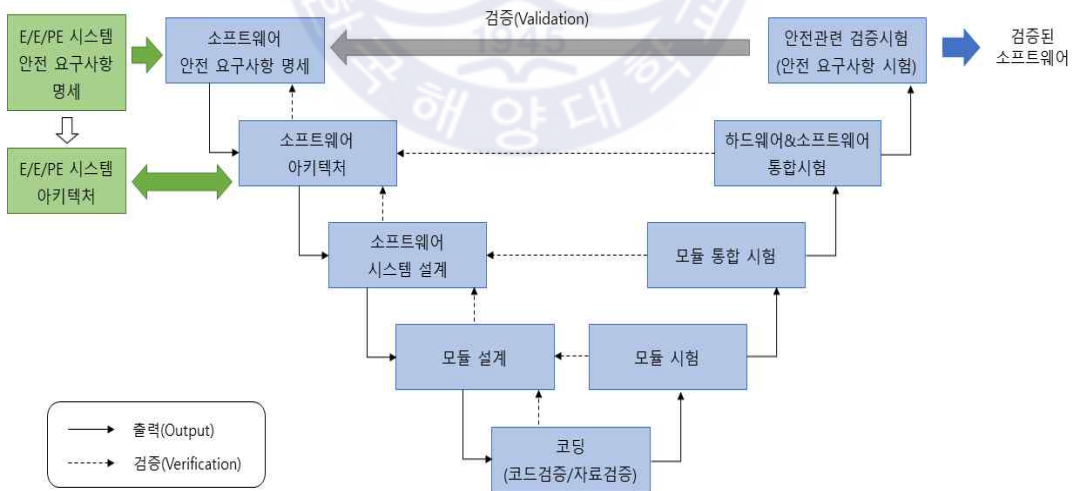


Fig. 6 IEC 61508-3 Software development life cycle

3.3 위험 및 운용 연구(HAZOP Study) 절차 분석

IEC 61508은 소프트웨어 개발 프로세스에서 안전과 관련된 위험성 평가절차를 추가로 수행하는 것으로 소프트웨어 개발 초기 단계에 위험성 분석을 수행해야 한다. IEC 61508에서는 위험성 분석 방법을 결정/진리표(Decision tables), 위험 및 운용 연구(Hazard and Operability Study(HAZOP), 공통 원인 고장 분석(Common cause failure analysis), 마코브 모델(Makov models), 신뢰성 블록 다이어그램(Reliability block diagrams), 몬테카를로 시뮬레이션(Monte-Carlo Simulation)의 6가지를 제시하고 있다. 본 논문에서는 NIPA의 소프트웨어 안전성 공통 개발 가이드라인에서 제시하고 있는 HAZOP 연구를 기반으로 한 한국해양대학교 소프트웨어공학 연구실(연구책임자 이서정 교수)에서 고안한 방안을 따른다. HAZOP 연구 절차는 Fig. 7과 같다.

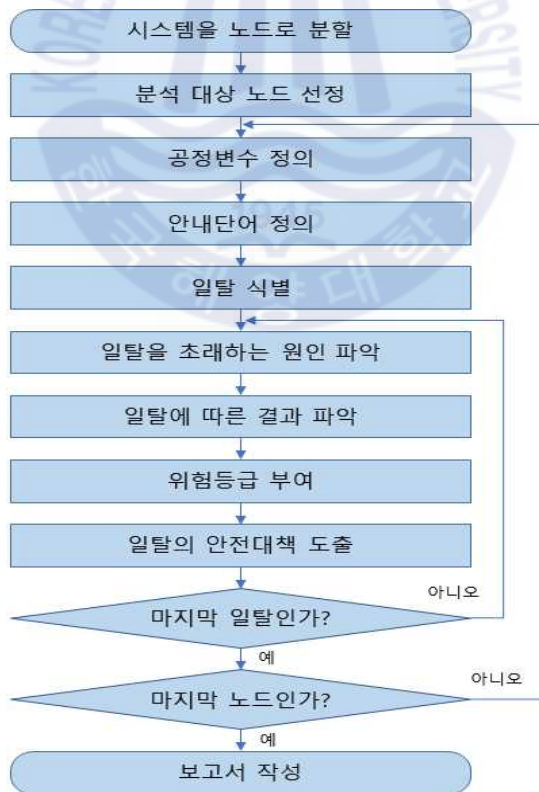


Fig. 7 HAZOP study process

(1) 분석 대상 노드(node) 선정 : 노드란 시스템이나 프로세스상의 주요 변화가 발생하는 한 부분으로 분석을 위한 범위를 한정한다. HAZOP 분석을 위해서는 해당 노드의 설계자료와 운용 매뉴얼 등이 사전에 준비되어 있어야 한다.

(2) 공정변수(process parameter) 정의 : 시스템의 기능이나 운용상태를 나타내기 위한 물리적 특성을 도출한다. 분석 대상 시스템이나 프로세스에 따라서 공정변수는 달라질 수 있으며 하드웨어 시스템의 경우에는 흐름, 온도, 압력, 수위, 점도 등이 있을 수 있으며 소프트웨어 시스템의 경우에는 입력, 저장, 출력, 통신 등이 될 수 있다.

(3) 안내단어(guide word) 정의 : 관계자들이 모여 난상토론(Brain Storming)을 할 때 문제들을 상상해내기 쉽게 하도록 도움이 되는 짧은 단어나 문구를 정의한다. 일련의 표준 질문을 공정변수에 반복적으로 적용하여 정상 조건에서 벗어나는 일탈(deviation)을 체계적이고 일관성 있게 도출하도록 해줄 수 있도록 하며 일반적으로 없음, 과다, 과소, 역전, 부분적, 대등한 등의 안내단어를 사용한다.

(4) 일탈(deviation) 식별 : 안내단어를 선정된 공정변수에 적용하여 의도한 정상 설계에서 벗어나는 일탈을 찾는다. 일탈은 안내단어와 공정변수의 조합으로 이루어지므로 매트릭스를 활용하여 쉽게 정의할 수 있다.

(5) 일탈을 초래하는 원인(causes) 파악 : 일탈 중 하나를 선정하여 그 일탈이 발생하게 된 원인을 분석한다. 일탈을 초래하는 원인은 한 가지 이상일 수도 있으며, 발생 가능성이 지나치게 낮은 원인은 관계자들의 판단에 따라 리스트에서 제외될 수도 있다. 원인의 유형으로는 오퍼레이터, 설계가, 제조자, 기타 관련 작업자의 행위로 인해 심각한 결과를 초래하는 위험이 발생하는 사람 오류(Human error)와 기계적, 구조적, 운용상의 결함으로 인한 장비 오류(Equipment failure), 정전이나 날씨 등에 의한 외부요인이 노드 운용에 영향을 주는 외부요인(External events)이 있다.

(6) 일탈에 따른 결과(consequence) 파악 : 일탈 발생에 따른 피해나 문제점을 안전성(Safety), 환경(Environmental), 경제성(Economic) 측면에서 분석한다.

(7) 위험등급 부여 : 일탈이 발생하였을 때 미치는 영향이 얼마나 심각한지를 나타내는 심각도와 일탈이 발생할 가능성을 고려하여 위험등급을 결정한다. NIPA의 소프트웨어 안전가이드에서는 Table 2와 Table 3에서 보는 바와 같이 IEC 61126과 MIL-STD-882를 참조하여 심각도와 발생 가능성을 산정하였다.

Table 2 Hazard Severity (referenced IEC 61126)

내용	범주	정 의
파국적	A	사망, 시스템 상실, 심각한 환경 파손
치명적	B	심각한 재해, 심각한 직업병, 상당한 시스템 또는 환경 파손
한계적	C	사소한 재해, 사소한 직업병, 사소한 시스템 또는 환경 파손
무시가능	-	사소한 재해나 직업병보다 더 낮음, 사소한 시스템이나 환경파손보다 더 낮음

Table 3 Hazard probability(referenced MIL-STD-882C)

내용	수준	확률 추정치
자주 발생	A	자주 발생할 수 있음
빈번히 발생	B	수명 기간에 몇 차례 발생함
가끔 발생	C	수명 기간에 한, 두 차례 발생할 수 있음
거의 발생하지 않음	D	수명 기간에 발생하지는 않으나, 배제할 수는 없음
발생 가능성 없음	E	확실히 발생하지는 않으나, 일어나지 않으리 라고 가정함

또한, 일탈에 대한 위험의 심각도와 발생확률을 조합하여 Table 4와 같이 위험성 수준 결정을 위한 매트릭스를 제시하고 있다. 이 표에서는 치명적인 심각도와 비교적 빈번한 발생확률을 갖는 일탈은 높은 위험등급을 가지도록 하고 있다.

Table 4 Matrix for risk level determination

구 분	위험원 범주			
	파국적	치명적	한계정	무시가능
자주 발생	높음	높음	높음	중간
빈번히 발생	높음	높음	중간	낮음
가끔 발생	높음	높음	중간	낮음
거의 발생하지 않음	높음	중간	낮음	낮음
발생 가능성 없음	중간	낮음	낮음	낮음

(8) 일탈의 안전대책(Safeguards) 도출 : 일탈 발생을 막거나 막는 것이 불가능하다면 발생 빈도를 줄여 변이가 주는 피해를 완화하는 데 도움이 되는 안전 대책을 도출한다. 안전대책은 문제를 해결하기 위한 개략적인 권고안의 목적으로 도출되어야 하며 시스템을 재설계하는 수준으로 도출되어서는 안 된다.

(9) HAZOP 분석절차 반복 및 결과 보고서 작성 : 선정된 노드와 도출된 모든 일탈에 대해서 원인분석, 결과분석, 위험등급 부여, 안전대책 도출의 과정을 반복 수행한다. HAZOP 분석이 완료되면 식별된 위험과 권고사항을 기록한 분석 보고서를 작성한다.

제 4 장 선박시스템 원격 모니터링 플랫폼 설계

선박시스템 원격 모니터링 플랫폼 소프트웨어의 기능안정성을 확보하기 위해서는 시스템의 설계 단계에서부터 위험성평가를 수행하여 플랫폼의 잠재된 문제를 발견하고 이를 해결할 수 있도록 설계를 수정해야 한다.

본 장에서는 선박 시스템 원격 모니터링 플랫폼의 기능안전성을 향상시키기 위하여 데이터 교환 표준인 ISO 19847에 대해 IEC 61508에서 정의하고 있는 위험성 평가절차에 따라 위험성을 평가하고, 준 정형 기법을 이용하여 위험성 감소방안을 제안한다. 그리고 위험성 감소방안에 따라 플랫폼을 수정 설계한다.

4.1 ISO 19847에 대한 기능 안전 평가

IEC 61508에서는 6가지의 기능안전 평가 방법을 제시하고 있으나 본 논문에서는 NIPA의 소프트웨어 안전성 공통 개발 가이드라인에 따라 시스템의 기능으로부터 오동작을 도출하고 안내단어를 이용한 난상토론을 통해 위험을 식별하는 체계적인 방식인 HAZOP 연구를 통해 ISO 19847에 대한 기능안전 평가를 수행하였다.

ISO 19847에 대한 HAZOP 연구를 수행하기 위해 NIPA의 소프트웨어 안전성 공통 개발 가이드라인과 항해장비의 위험성 분석 사례(임상우, 2017) 및 자동차 분야의 사례를 참고하여 안내단어의 후보들을 설정하고 본 연구에 맞게 내용을 바꾸고 종류를 결정하였다.

(1) 분석대상 노드 선정

ISO 19847의 데이터 입출력을 위한 개념모델에서 선박에 설치되어 데이터를 수집, 저장, 전송하는 선박 데이터 서버(Shipboard data server)를 분석 대상 노드로 선정하였다.

(2) 공정변수(process parameter) 정의

ISO 19847과 ISO 19848에서 정의한 개념모델의 요구사항을 분석하여 Table 5와 같이 서버시스템 기능에 대한 부분과 선박과 육상간 통신에 대한 부분으로 구분하여 공정변수를 도출하였다.

Table 5 Definition of process parameters

분류	공정변수
서버시스템 기능	데이터 입력 기능
	데이터 출력 기능
	데이터 저장 기능
	데이터 가공 기능
선박-육상간 통신	전송 주기
	전송 속도
	통신 신뢰성
	통신 안정성

(3) 안내단어(guide word) 정의

IEC 61508-7에서 제안하고 있는 안내단어 중에서 선행 연구사례들을 참고하여 소프트웨어에서 어떠한 동작도 할 수 없거나 일부 기능만 수행이 되는 경우 또는 그 기능의 성능이 현저하게 떨어지는 경우, 너무 많은 동작에 따른 과부하인 경우는 쉽게 발생하지만, 그 영향력이 소프트웨어의 충돌(Crash)이나 경제적 손실로 이어질 때가 많으므로 Table 6과 같이 4개의 안내단어를 선정하였다.

Table 6 Definition of guide words

항목	의미
없음(None)	행동이 없는 경우
과다(More)	적정 수준보다 더 많은 경우
과소(Less)	적정 수준보다 더 적은 경우
부분적(Part of)	의도한 설계의 일부분만 정상적으로 수행

(4) 일탈(deviation) 식별

앞서 선정된 공정변수가 안내단어에 따라 변화될 때에 생겨나는 일탈을 Table 7과 같이 식별하였다.

Table 7 Declaration of deviation

구분	없음	과다	과소	부분적
데이터 입력	입력이 없음	입력이 너무 많음	입력이 적음	일부 시스템만 입력이 들어옴
데이터 출력	출력이 안됨	출력이 너무 많이 발생	출력이 적게 발생	-
데이터 저장	저장이 안됨	데이터 저장이 너무 발생	저장이 적게 발생	-
데이터 가공	데이터 가공이 안됨	데이터 가공이 너무 자주 발생	시스템 가공이 너무 적게 발생	-
전송 주기	전송이 없음	전송 주기가 너무 빠름	전송 주기가 너무 느림	-
전송 속도	전송이 없음	전송 속도가 빠름	전송 속도가 너무 느림	-
통신 신뢰성	통신 신뢰성 없음	통신 신뢰성 높음	통신 신뢰성 낮음	-
통신 안정성	통신 안정성 없음	통신 안정성 높음	통신 안정성 낮음	-

(5) 일탈을 초래하는 원인(causes) 파악

식별된 일탈에 대해서 일탈에 이르게 한 원인을 분석하였다. HAZOP 절차에 따라서 하나의 일탈에 대해서 원인, 결과, 위험등급 부여, 안전대책 도출이 이루어진 후 다른 일탈에 대해서 과정이 반복되어 수행되었으나 그 결과를 정리하여 Table 8과 같이 모든 일탈에 대한 원인을 나타내었다.

Table 8 Identify cause of deviation

구분	없음	과다	과소	부분적
데이터 입력	하위 시스템 연결 안됨	하위 시스템의 데이터 변화가 많음	하위 시스템의 데이터 변화가 적음	하위 시스템 중 일부만 정상 작동함
데이터 출력	시스템 작동 불가능	서버는 MQTT 중계자 역할만을 수행하므로 하위 시스템의 변화가 많아서 발생함 서버는 HTTP 요청에 응답을 해야 하므로 육상의 잦은 요청이 있음	서버는 MQTT 중계자 역할만을 수행하므로 하위 시스템의 변화가 적음 서버는 HTTP 요청에 응답을 해야 하므로 육상의 요청이 별로 없음	-
데이터 저장	시스템 작동 불가능 저장 매체 오류	하위 시스템의 데이터 변화가 많음	하위 시스템의 데이터 변화가 적음	-
데이터 가공	시스템 작동 불가능	하위 시스템의 데이터 변화가 많음	하위 시스템의 데이터 변화가 적음	-

전송 주기	시스템 작동 불가능	서버는 MQTT 중계자 역할만을 수행하므로 하위 시스템의 변화가 많아서 발생함 서버는 HTTP 요청에 응답을 해야 하므로 육상의 잦은 요청이 있음	서버는 MQTT 중계자 역할만을 수행하므로 하위 시스템의 변화가 적음 서버는 HTTP 요청에 응답을 해야 하므로 육상의 요청이 별로 없음	-
전송 속도	시스템 작동 불가능	문제 발생 안함	통신망이 지원하는 전송 속도가 느림	-
통신 오류	문제 발생 안함	통신망 품질이 떨어져서 목적지까지 패킷이 전송되지 못함	문제 발생 안함	-
통신 신뢰성	통신망 품질이 현저히 떨어져서 목적지까지 패킷이 전송되지 못함	문제 발생 안함	통신망 품질이 다소 떨어져서 패킷이 일부 유실됨	
통신 안정성	음영지역이라서 접속이 매우 자주 끊어지거나 통신이 단절됨	문제 발생 안함	음영지역의 경계지점에 있어 통신 품질이 떨어져서 접속이 자주 끊어짐	

(6) 일탈에 따른 결과(consequence) 파악

일탈이 발생한 경우 시스템상 미칠 수 있는 영향을 시스템의 안정성과 시스템 운영을 위한 경제성 측면에서 분석하여 Table 9와 같이 도출하였다.

Table 9 Identify consequence of deviation

구분	없음	과다	과소	부분적
데이터 입력	선박 시스템 정보 최신화 안됨 오류상황 대처 불가	서버시스템 과부하 시스템 작동불능으로 이어질수 있음	선박 시스템 정보 최신화가 느림	작동하지 않는 시스템에 대해서는 정보 최신화 안 됨
데이터 출력		서버시스템 과부하 통신비용 과다	선박 시스템 정보 최신화가 느림	-
데이터 저장		서버시스템 과부하 시스템 작동불능으로 이어질수 있음	선박 시스템 정보 최신화가 느림	-
데이터 가공		서버시스템 과부하 시스템 작동불능으로 이어질수 있음	선박 시스템 정보 최신화가 느림	-
전송 주기		서버시스템 과부하 통신비용 과다	선박 시스템 정보 최신화가 느림	-
전송 속도		-	선박 시스템 정보 최신화가 느림 정보 발생량 대비 속도가 느린 경우 버퍼가 모자를 수 있음	-
통신 신뢰성		빈번한 재전송으로 인한 트래픽 과다 통신비용 과다	-	빈번한 재전송으로 인한 트래픽 과다 통신비용 과다
통신 안정성	빈번한 재접속으로 정보 업데이트가 늦음 트래픽과다로 통신비용 증가	-	빈번한 재접속으로 정보 업데이트가 늦음 트래픽과다로 통신비용 증가	-

(7) 위험등급 부여

도출된 일탈에 대해 NIPA의 소프트웨어 안전가이드에서 제시하고 있는 위험성 수준 결정 매트릭스를 이용하여 위험등급을 부여하였다. Table 10의 음영되어 있는 것과 같이 위험등급이 높은 일탈을 11개를 식별하였다.

Table 10 Determine risk level

구분	없음	과다	과소	부분적
데이터 입력	낮음 심각도-C 확률-D	높음 심각도-B 확률-B	중간 심각도-C 확률-C	중간 심각도-C 확률-C
데이터 출력	높음 심각도-B 확률-C	높음 심각도-B 확률-B	중간 심각도-C 확률-C	-
데이터 저장	중간 심각도-C 확률-C	중간 심각도-C 확률-B	중간 심각도-C 확률-C	-
데이터 가공	중간 심각도-C 확률-C	중간 심각도-C 확률-B	중간 심각도-C 확률-C	-
전송 주기	높음 심각도-B 확률-C	높음 심각도-B 확률-B	중간 심각도-C 확률-B	-
전송 속도	높음 심각도-B 확률-C	낮음 심각도-무시가능 확률-C	높음 심각도-C 확률-A	-
통신 신뢰성	높음 심각도-B 확률-A	낮음 심각도-무시가능 확률-C	높음 심각도-B 확률-A	-
통신 안정성	높음 심각도-B 확률-A	낮음 심각도-무시가능 확률-C	높음 심각도-B 확률-A	-

(8) 일탈의 안전대책(Safeguards) 도출

일탈의 원인과 결과의 상관관계를 분석하여 일탈의 위험성을 감소시키기 위한 권고안을 Table 11과 같이 도출하였다.

Table 11 Identify safeguards of deviation

구분	없음	과다	과소	부분적	
데이터 입력	서버 이중화 지속적인 서버의 상태 확인	서버에 의한 능동적 입력량 조절 기능 추가(MQTT)	서버에 의한 능동적 입력량 조절 기능 추가(MQTT)	작동하지 않는 시스템에 대해서는 정보최신화가 됨	
데이터 출력		서버에 의한 능동적 입력량 조절 기능 추가(MQTT) - 너무 잦은 요청에 대한 서비스 불가 또는 무시기능 추가(HTTP)	서버에 의한 능동적 입력량 조절 기능 추가(MQTT) - 서버는 중계자일 뿐이므로 입력이 조절되면 출력은 자동적으로 조절됨	-	
데이터 저장		서버에 의한 능동적 입력량 조절 기능 추가(MQTT)	서버에 의한 능동적 입력량 조절 기능 추가(MQTT)	-	
데이터 가공		서버에 의한 능동적 입력량 조절 기능 추가(MQTT)	서버에 의한 능동적 입력량 조절 기능 추가(MQTT)	-	
전송 주기		서버에 의한 능동적 입력량 조절 기능 추가(MQTT) - 너무 잦은 요청에 대한 서비스 불가 또는 무시기능 추가(HTTP)	서버에 의한 능동적 입력량 조절 기능 추가(MQTT) - 서버는 중계자일 뿐이므로 입력이 조절되면 출력은 자동적으로 조절됨	-	
전송 속도		-	데이터 출력량 조절, 버퍼 오버플로우 대비	-	
통신 신뢰성		오류확인 메커니즘 적용	-	오류확인 메커니즘 적용	-
통신 안정성		재접속 메커니즘 적용	-	재접속 메커니즘 적용	-

4.2 반정형기법을 이용한 위험성 감소방안 제안

HAZOP 분석을 통해 도출된 8종의 공정변수와 4종의 안내단어를 이용하여 25종의 일탈을 도출하고 Table 10에서와 같이 위험등급을 부여하였다. 이 중에서 위험등급이 높은 11가지의 일탈과 원인 및 안전대책을 정리하여 보면 Fig. 8과 같다.

도출된 25종의 일탈 중에서 “데이터 저장” 과 “데이터 가공” 의 공정변수에 대응하는 일탈과 “부분적” 의 안내단어에 대응하는 일탈은 모두 위험등급이 상대적으로 낮아 Fig. 8의 도식에서 제외하였다. 위험등급이 높은 11종의 일탈들을 HAZOP 절차에 따라 도출된 내용을 살펴보면 6종의 공정변수와 3종의 안내 단어를 통해 도출된 11종의 일탈들이 4종의 안전대책으로 요약됨을 알 수 있다.

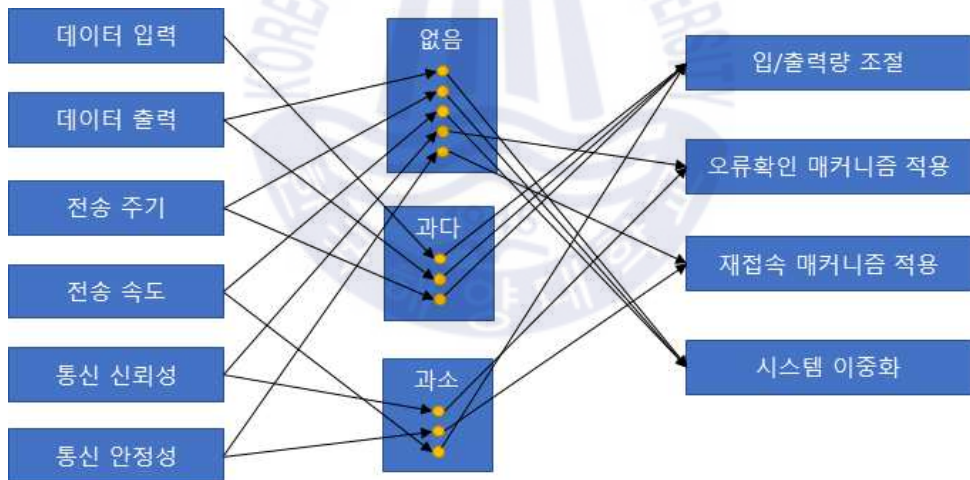


Fig. 8 High risk deviations, causes and safeguards

HAZOP 분석 과정에서 도출된 안전대책은 문제 해결을 위한 구체적인 해결책은 아니므로 IEC 61508-3의 소프트웨어 안전 요구사항에서 제시하고 있는 반정형기법(Semi-formal methods)을 이용하여 일탈을 구체적으로 분석하고 위험성 감소방안을 제안하고자 한다.

HAZOP을 통한 기능안전 평가에서는 스트리밍 방식을 사용하는 MQTT 프로토콜과 요청-응답 방식을 사용하는 HTTP방식을 모두 고려하였으나 본 논문에서는 MQTT 프로토콜을 사용하는 스트리밍 방식의 위험성 감소방안에 대해서만 고려하기로 한다.

IEC 61508에서는 높은 수준의 안전성이 요구되는 시스템에 대해 반정형기법의 적용을 요구하고 있다. 반정형 기법은 수학적 기반으로 엄격하게 정의된 언어를 통해 시스템의 명세와 설계를 수행하고 수학적 증명을 통해 검증함으로써 모호한 자연어명세와 일반적 검증절차에서 발생할 수 있는 인적 오류의 개연성을 억제하고자 하는 것이다. 이것은 하드웨어나 소프트웨어 시스템은 불가피하게 점점 더 규모, 기능면에서 커지고 복잡해지고, 이러한 복잡성의 증가 때문에 작은 결함이 존재할 가능성이 커지고 있기 때문이다.

Table 12는 IEC 61508-3의 표준에서는 제안하고 있는 반정형기법의 세부 기법으로 본 논문에서는 HAZOP 분석의 결과에 따른 세부적인 문제점 분석에 적절한 기법을 선택해서 적용하였다.

Table 12 Semi-formal methods

기법/수단		참조	SIL1	SIL2	SIL3	SIL4
1	논리/함수 블록 다이어그램	비고 참조	R	R	HR	HR
2	순서도	비고 참조	R	R	HR	HR
3	데이터 흐름 다이어그램	C.2.2	R	R	R	R
4	유한상태 기계/상태전이 다이어그램	B.2.3.2	R	R	HR	HR
5	시간 페트리 넷	B.2.3.2	R	R	HR	HR
6	결정/진리표	C.6.1	R	R	HR	HR
비고 논리/함수 블록 다이어그램과 순서도는 KS C IEC 61508-3에서 기술						
* 안전무결성 수준에 따라서 적절한 기법/수단이 선택되어야 한다.						

4.2.1 데이터 입/출력량 조절

선박 데이터 서버가 스트리밍(Streaming) 방식으로 데이터 전송기능을 수행할 때 선박 내 시스템의 변화가 많은 경우 잦은 출판(Publish) 메시지를 전송하게 된다. Fig. 9에서 보는 바와 같이 선박의 압력계측시스템인 MQTT 출판자가 압력의 변화가 많아 MQTT 메시지를 자주 발행하게 되면 중계자는 출판자로부터 받은 메시지를 육상의 데이터 서버인 구독자에게 계속해서 메시지를 전달하게 된다.

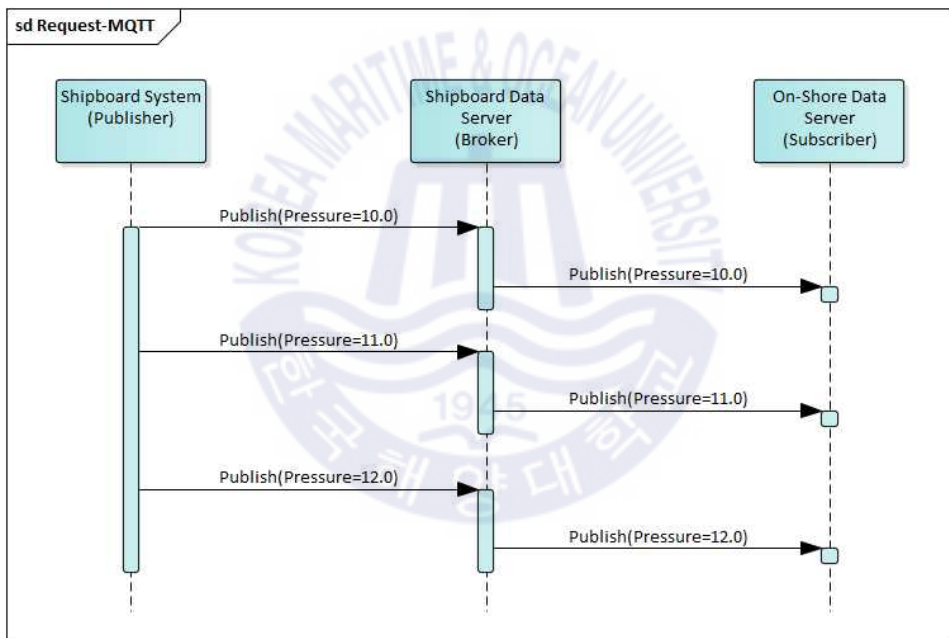


Fig. 9 Frequently publish of MQTT protocol

특히 MQTT 프로토콜은 중계자(Broker)인 서버가 메시지를 단순히 중계하는 역할만을 수행하므로 출판자가 메시지를 발행하면 구독자에게 충실히 전달하기만 할 뿐이다.

선박 내 시스템으로부터 출판 메시지가 많아지게 되면 이를 처리하기 위한 대기큐에 데이터가 쌓이게 된다. Fig. 10의 시간-페트리넷 다이어그램에서 보는 바와 같이 시간이 지남에 따라 대기상태에 자원(토큰)이 많이 쌓이게 되어 서비스의 지연이 발생하게 된다.

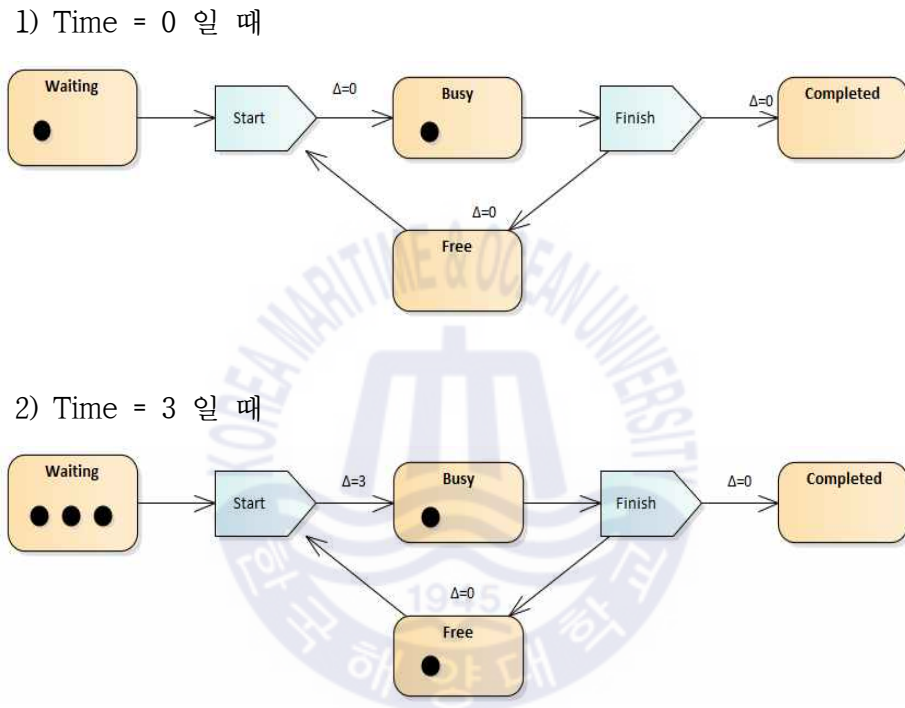


Fig. 10 Time-Petri diagram of high publish situation

초기 시스템이 가동할 때는 메시지가 많이 수신되지 않아 1개의 자원만이 대기 상태에 있게 되지만, 시스템이 처리할 수 있는 속도보다 더 많은 출판 메시지가 수신되기 시작하면서 대기 상태에 있는 메시지가 늘어나게 된다. 이렇게 되면 메시지를 저장하는 버퍼(Buffer)의 한계에 도달하게 되고 시스템의 비정상적인 종료로 이어질 수 있다.

그러므로 출판 메시지가 많이 발생하지 않도록 MQTT 프로토콜에 최소 전송 주기를 설정하는 기능을 추가하고자 한다. Fig. 11에서 보는 바와 같이 선박 데이터 서버는 선박 내 시스템으로 최소 전송주기를 10초로 설정하게 되면 10초가 지나기 전에는 시스템의 변화가 발생하더라도 출판 메시지를 보내지 않는다.

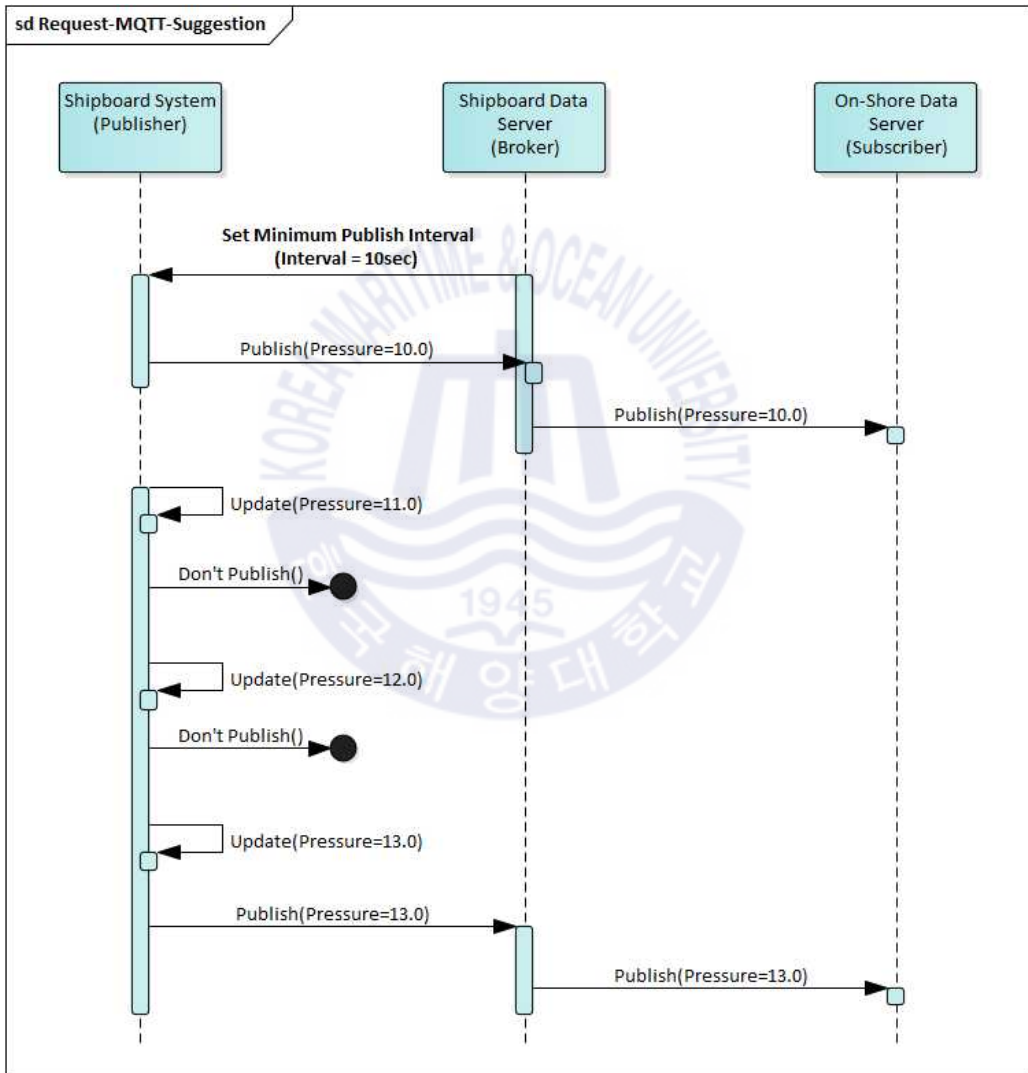


Fig. 11 Sequence of minimum publish interval setting

4.2.2 오류확인 메커니즘 적용

선박과 육상간의 통신은 무선통신을 사용하는 경우가 대부분이며 무선통신의 특성상 기상 상황이나 주변 상황에 의해 잡음과 간섭이 커져 신호가 왜곡되거나 도달하지 못하여 통신품질이 떨어지는 경우가 자주 발생한다. 선박에서는 데이터 통신을 위해 위성통신을 주로 사용하는데, 긴 전송지연시간과 상대적으로 높은 데이터 에러율로 인해 Fig. 12에서 보는 바와 같이 출판자가 발행한 메시지를 구독자에게 전달되지 못하는 상황이 자주 발생하게 된다.

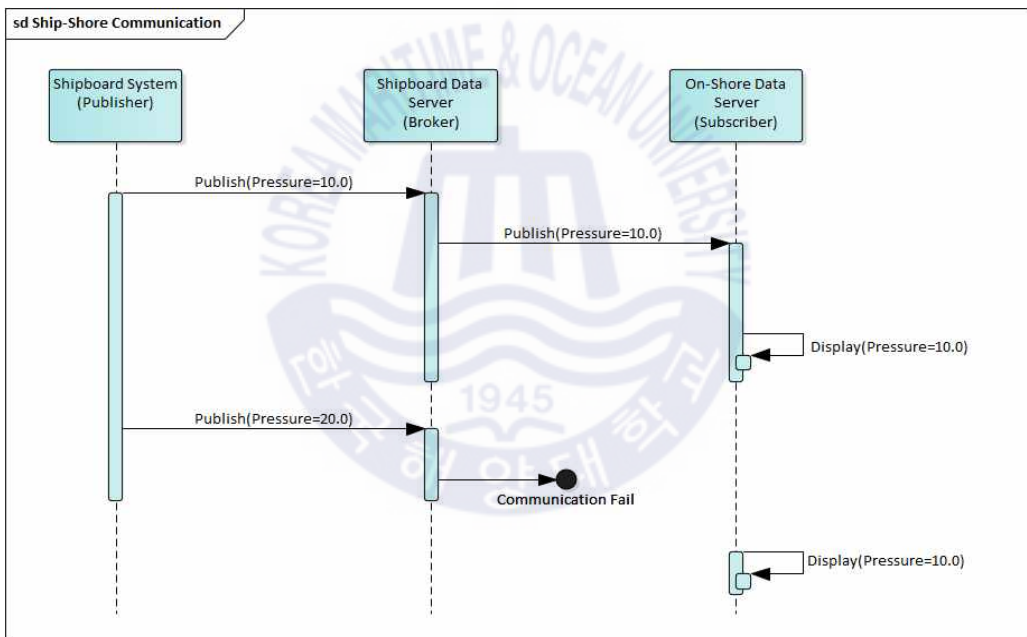


Fig. 12 Communication failure between ship and shore

MQTT 프로토콜에서는 구독자에게 신뢰성 있는 메시지 전달을 위하여 “많아야 한 번“, “적어도 한 번“, “정확히 한 번“이라는 3가지 QoS(Quality of Service) 레벨을 정의하고 있다.

(1) QoS0은 한 번 전송하고 성공 여부는 확인하지 않으며, 만약 전송이 실패 하더라도 재전송하지 않는다. Fig. 13에서와 같이 출판자는 데이터를 출판한 뒤 전송 성공 여부는 고려하지 않고 메시지를 삭제한다. 그러므로 메시지는 구독자와의 연결이 끊어지거나 서버에 장애가 발생하는 경우 메시지가 손실될 수 있으므로 구독자에게 한 번 전달되거나 전혀 전달되지 않는다. 즉, “많아야 한 번” 전송이 된다.

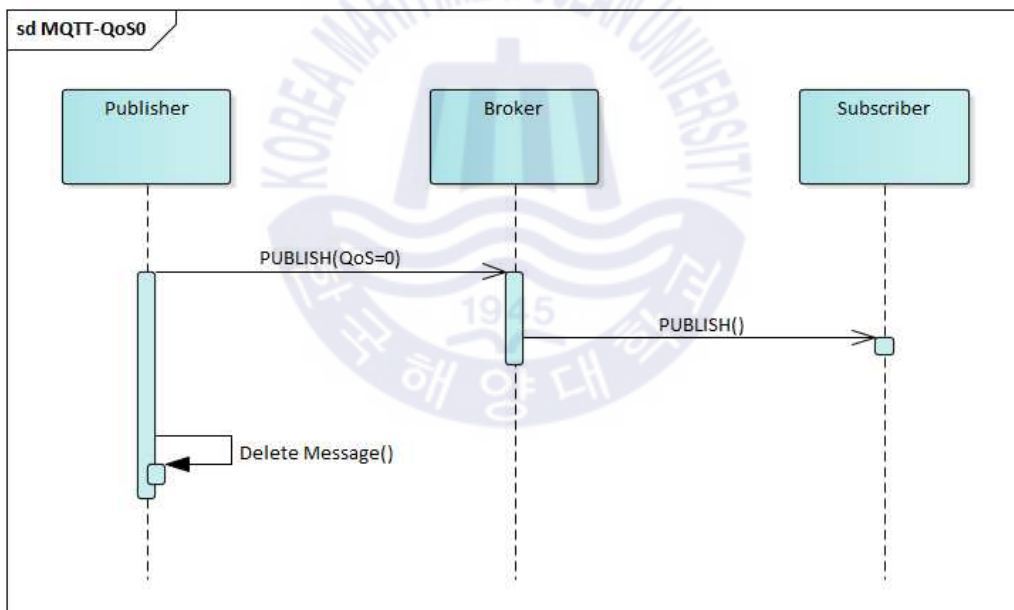


Fig. 13 Sequence of QoS0 in MQTT protocol

(2) QoS1은 최소 한번은 전송을 보장하지만, 중복패킷을 받을 수 있다. Fig. 14에서와 같이 출판자가 중계자에게 메시지를 보내고, 중계자는 구독자에게 메시지를 보낸다. 중계자는 출판자에게 PUBACK 메시지를 통해 메시지를 전달했음을 알리게 되고 출판자는 해당 메시지를 삭제한다. 하지만, 만약 이 PUBACK 메시지가 분실되었다면 출판자는 다시 메시지를 출판하게 될 것이므로 구독자에게 메시지가 중복되어 전달될 수 있지만 “적어도 한 번”은 전달된다.

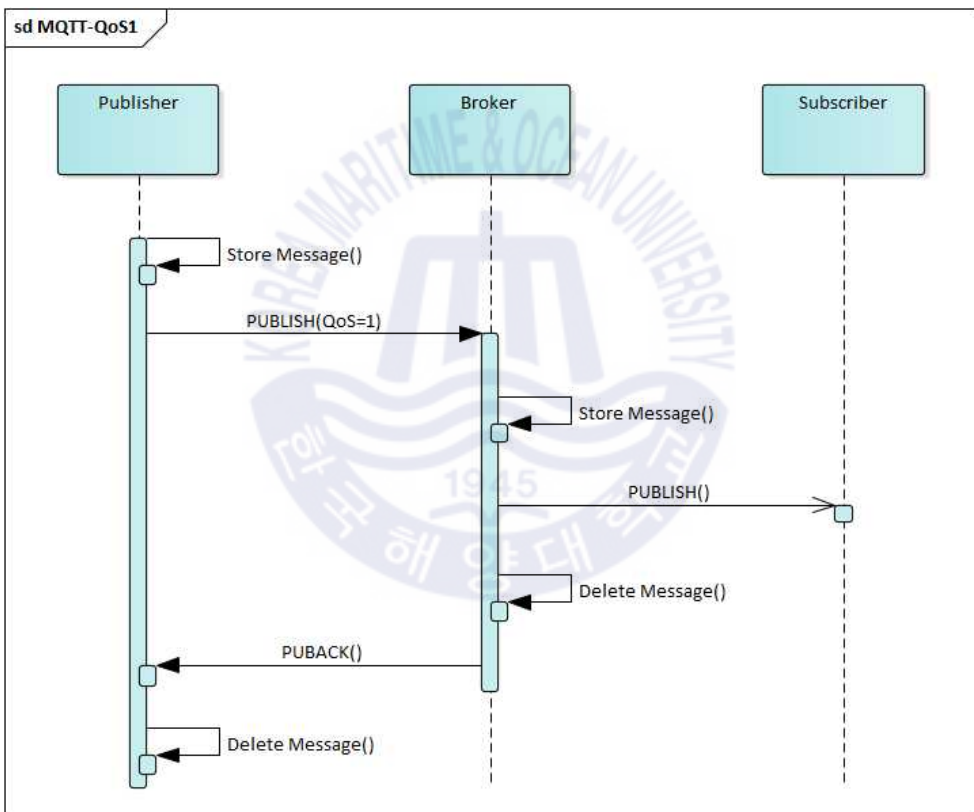


Fig. 14 Sequence of QoS1 in MQTT protocol

(3) QoS2는 앞서 언급한 QoS1의 중복 전달의 문제점을 해결하기 위해서 PUBACK 과정을 핸드셰이킹(Hand-Shaking) 형태로 변경하였다. 중계자가 메시지를 수신하면 PUBREC 메시지를 출판자에게 전달하고 이를 다시 응답으로 받으면 메시지를 삭제한다. 만약 위의 과정에서 PUBREC 메시지가 손실되었다 하더라도 중계자는 이미 보냈다는 사실을 알고 있으므로 구독자에게 새로 보내지 않으므로 “정확히 한 번” 전달이 된다.

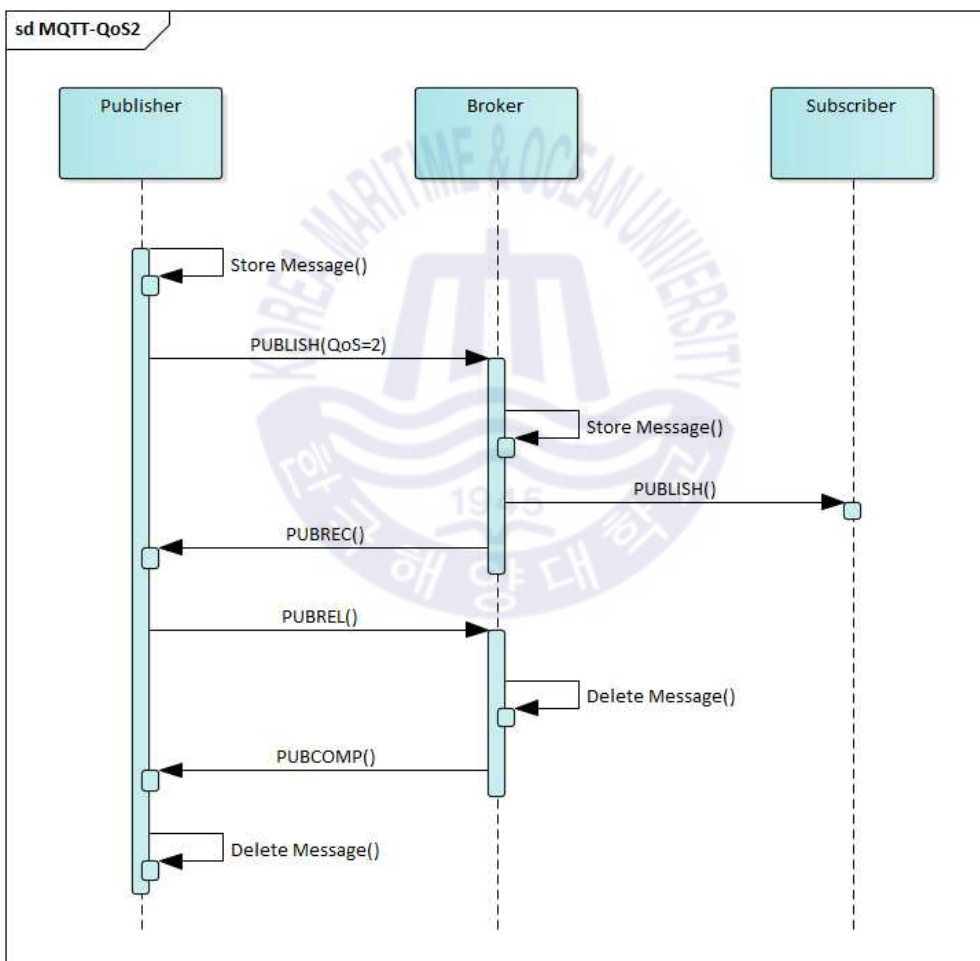


Fig. 15 Sequence of QoS2 in MQTT protocol

MQTT 프로토콜은 출판자와 중계자 간의 통신품질이 떨어질 것을 대비하여 출판자와 중계자 사이의 통신을 보강하고 있으나 일반적으로 선박에서 이 구간은 유선통신 구간인 경우가 많다.

선박에서 통신의 신뢰성이 떨어지는 구간은 위성을 이용한 무선통신을 하는 중계자와 구독자 사이의 통신이지만 MQTT 프로토콜에서 정의하고 있는 QoS에서는 이 구간에 대한 통신 신뢰성은 확인하지 않고 있다. 그러므로 본 논문에서는 Fig. 16에서와 같이 구독자가 출판 메시지를 수신하면 PUBACK 메시지를 중계자에게 보내서 출판 메시지를 정상적으로 수신하였음을 알려 중계자와 구독자 사이의 신뢰성 있는 메시지 전달을 할 수 있도록 하고자 한다.

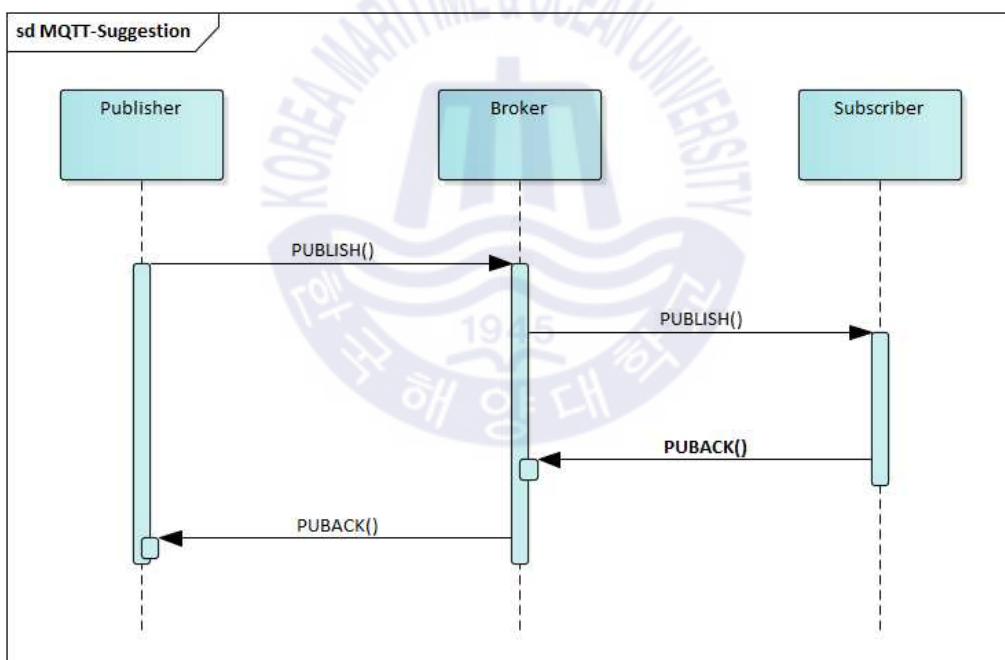


Fig. 16 Suggest of error checking mechanism between ship and shore

4.2.3 재접속 메커니즘 적용

선박이 육상과 무선통신을 하던 중 음영지역으로 진입하게 되면 통신 품질이 떨어지게 된다. MQTT 프로토콜은 TCP통신을 기반으로 하고 있으므로 Fig. 17 과 Fig. 18에서와 같이 TCP 연결 및 종료를 위한 절차를 거친다.

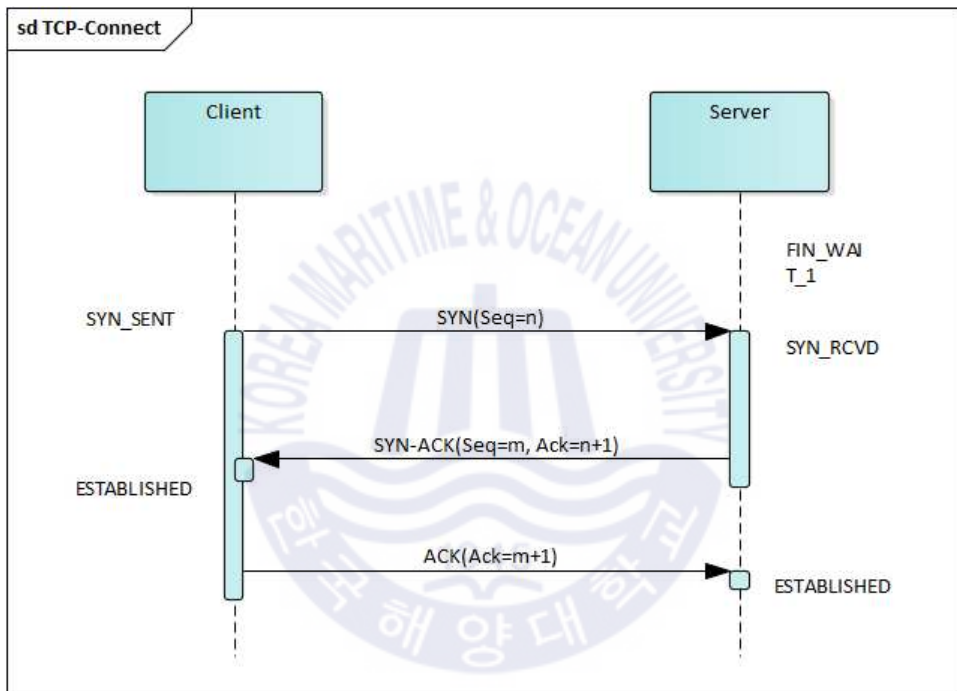


Fig. 17 TCP connection process

클라이언트가 서버에 접속하기 위해서는 클라이언트는 서버에게 접속을 위한 SYN 패킷을 보내고 응답을 기다리는 SYN_SENT 상태가 된다. 서버는 요청을 수락한다는 의미로 SYN-ACK 패킷을 전송하고 SYN_RCVD 상태로 변경된다. 클라이언트가 서버의 응답을 수신하면 ESTABLISHED 상태로 변경되고 서버에게 ACK 패킷을 전송하게 되고 그 이후로부터는 연결이 맺어지고 실제 데이터가 오가게 된다.

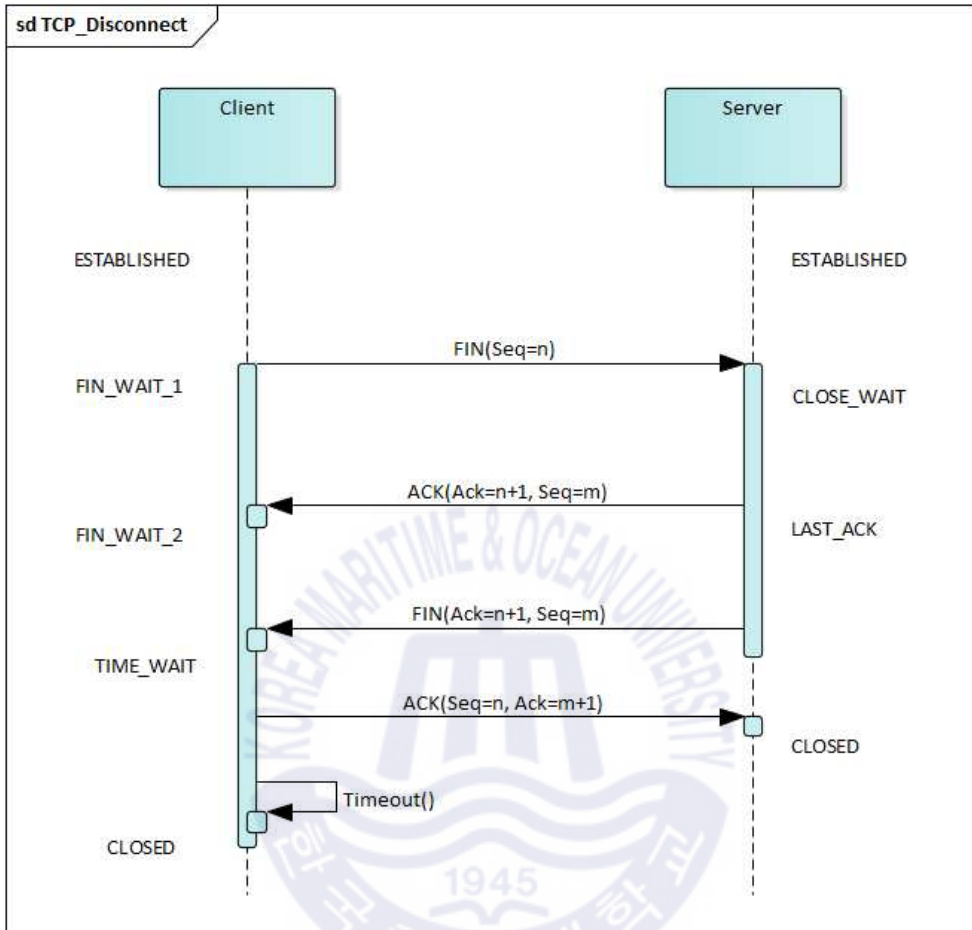


Fig. 18 TCP disconnect process

클라이언트가 접속을 종료하기 위해서는 클라이언트가 서버에게 접속을 종료하겠다는 FIN 패킷을 전송한다. 서버는 우선 ACK 패킷을 이용해 확인 메시지를 보내고 자신의 통신이 끝날 때까지 기다리는 CLOSE_WAIT 상태로 변경된다. 서버에서 보내야 할 데이터를 모두 보내고 나면 서버가 클라이언트에게 FIN 패킷을 보내게 되고 클라이언트는 ACK 패킷으로 응답한 후 TIME_WAIT 상태로 변경된다. 클라이언트는 일정 시간 이후에 완전히 접속을 종료한다.

TCP 접속 및 종료 절차를 상태 다이어그램으로 나타내면 Fig. 19와 같다[13].

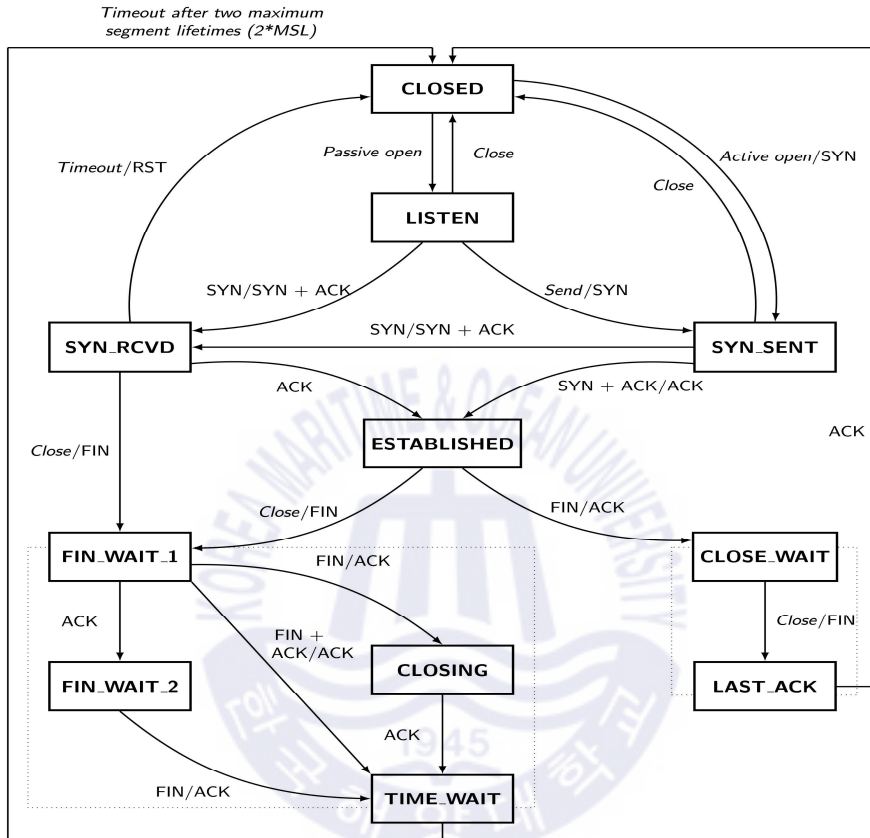


Fig. 19 Diagram of TCP state transition

클라이언트는 서버의 FIN 패킷 이전에 보내진 데이터가 라우팅 지연이나 패킷의 유실로 인해 재전송되어 FIN 패킷을 받은 이후에 수신될 수 있으므로 일정 시간 동안 세션을 남겨놓고 잉여 패킷을 기다린다. 다시 말해 클라이언트는 FIN_WAIT2 상태에서 서버로부터 FIN 패킷을 수신하고 난 뒤 FIN_WAIT 상태를 일정 시간 동안 유지함으로써 지연된 패킷을 수신할 수 있게 된다.

서버는 TCP 연결을 기다리는 포트는 하나지만, Fig. 20에서와 같이 클라이언트가 연결을 시도하면 로컬 포트를 임의로(ephemeral port) 바인딩하여 자식 소켓을 생성하면서 서버의 소켓과 연결된다. 실제로 서버와 클라이언트간 데이터 전송은 서버가 생성해준 자식 소켓과 클라이언트간 이루어지게 된다.

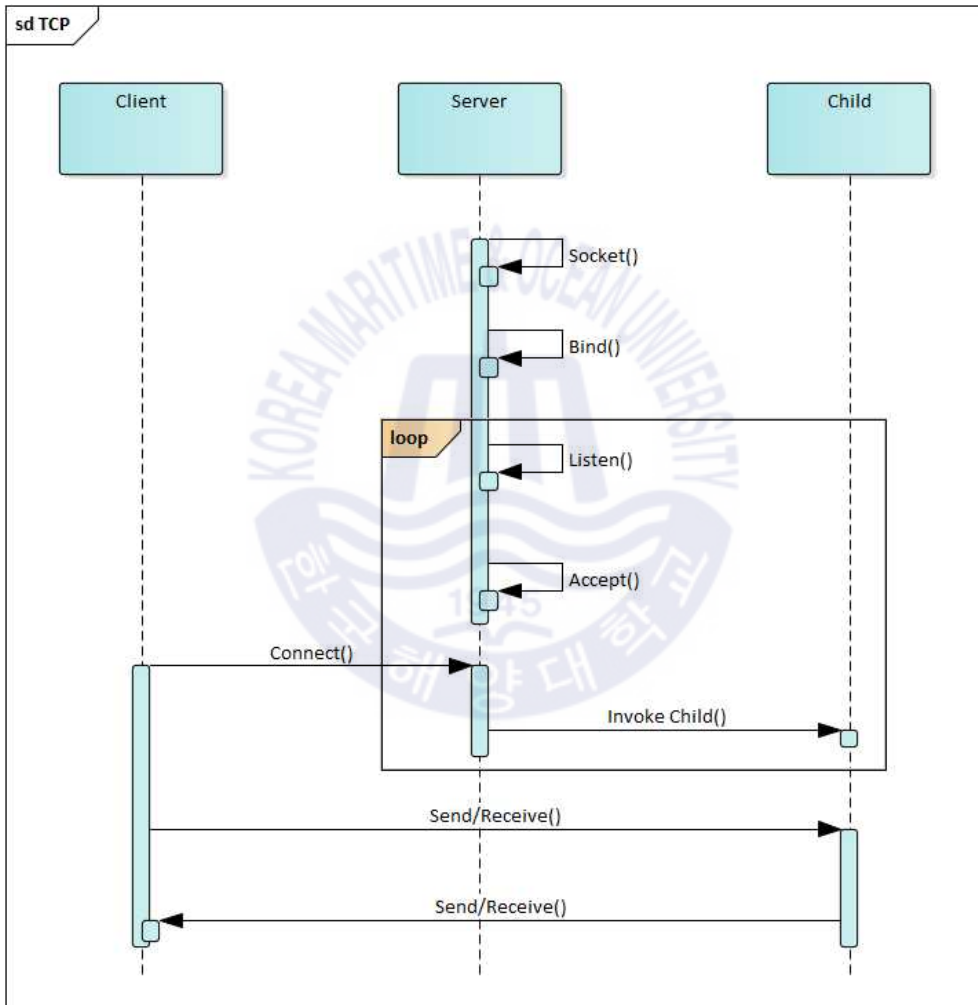


Fig. 20 Sequence of acceptance of TCP server

앞서 Fig. 19에서 언급한 바와 같이 TIME_WAIT 상태는 연결 종료시 필수적으로 생길 수밖에 없으며, 특히 서버쪽에 TIME_WAIT가 남게 되면 일정 시간 동안 상태를 지속하게 되는데 모든 클라이언트들의 세션 종료마다 서버측에 TIME_WAIT가 발생한다면, 서버측에 부하가 될 뿐만 아니라 최악의 경우 클라이언트와 연결을 맺기 위한 새로운 TCP 포트 자원이 고갈되어 더 이상 새로운 연결을 받아들일 수 없는 상황이 발생할 수 있다.

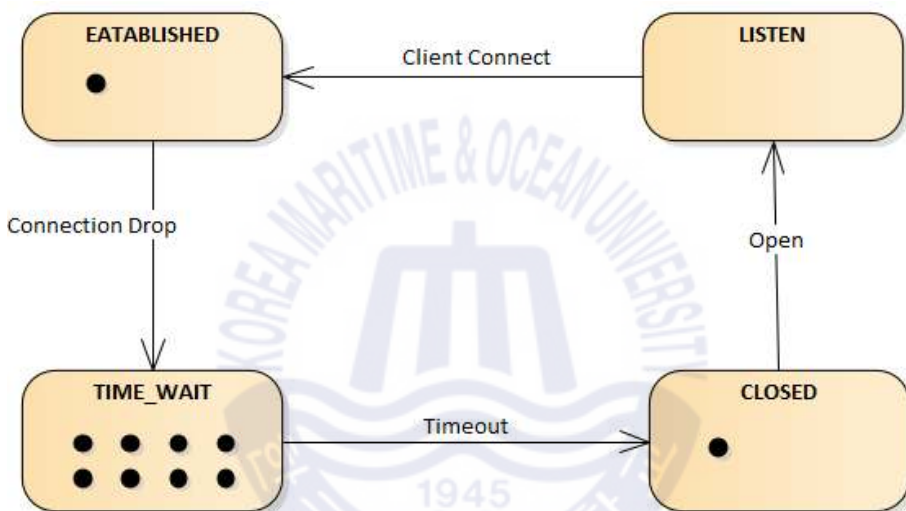


Fig. 21 Petri-Net diagram of frequently reconnect situation

Fig. 21의 페트리넷 다이어그램에서와 같이 가용할 수 있는 자원인 TCP 포트는 10개인 상황에서 잦은 재접속으로 인해 TIME_WAIT에 있는 자원이 늘어나게 되면 새로운 연결이 시도될 때 가용할 수 있는 자원이 부족하여 더 이상 연결이 허용되지 않는다.

재접속을 시도하는 시간은 클라이언트의 설정에 의존하므로 서버는 자신이 가지고 있는 TCP 포트 자원의 고갈을 방지하는 방법이 필요하다. 이에 본 논문에서는 Fig. 22에서와 같이 TCP 포트 자원의 고갈을 방지하고 빈번한 재접속으로 인한 서버의 과중한 부하를 예방하기 위하여 서버의 재접속 시간 설정에 따라 클라이언트에게 서버와 연결이 종료되더라도 바로 재접속하지 않고 일정 시간 이후에 재접속을 할 수 있도록 하는 메커니즘을 제안한다.

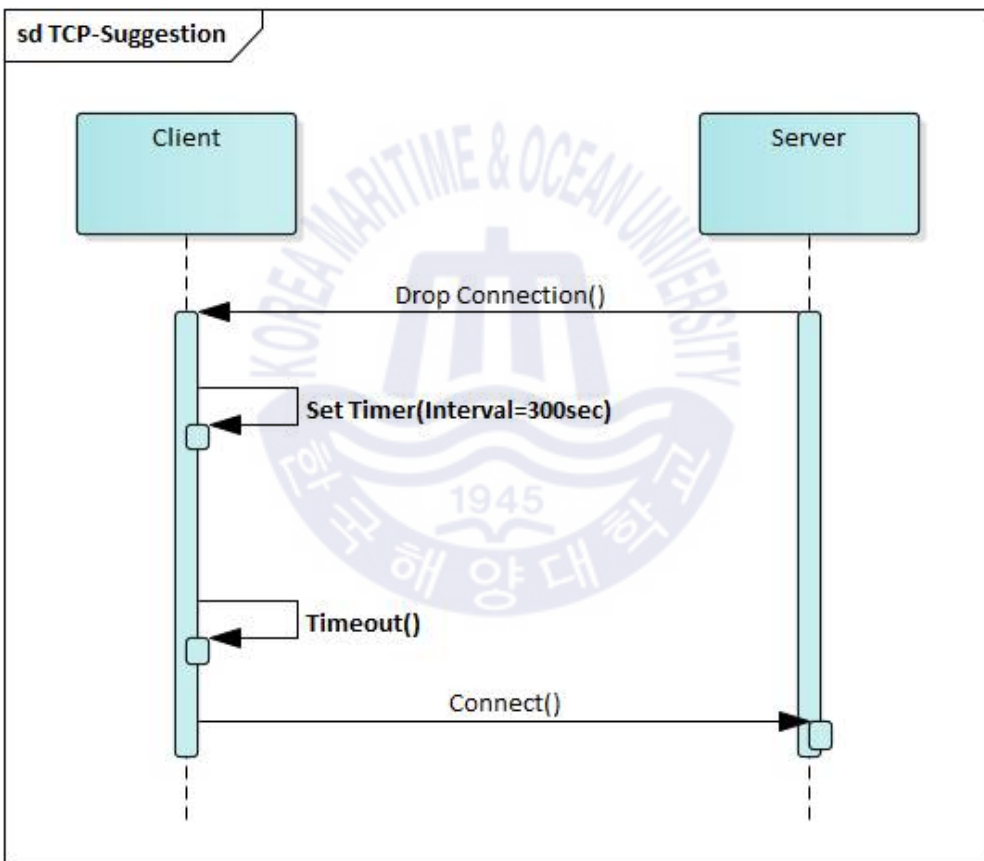


Fig. 22 Improve of reconnection sequence

4.2.4 시스템 이중화

ISO 19847에서는 HTTP 및 MQTT 서비스를 위해서 선박에 서버를 구성하도록 요구하고 있다. Fig. 23에서와 같이 선박 및 육상의 시스템은 선박에 있는 서버에 접속하고, 선박 내부의 각 시스템은 정보변경이 발생하면 서버로 데이터를 전송하거나 육상의 데이터 요청이 있으면 데이터를 조회하여 응답한다. 만약 선박 내 서버에 문제가 발생할 경우에 선박 내부 시스템이나 육상의 시스템은 정상적으로 작동 중이지만, 메시지가 전달되지 못하고 연결이 종료되어 버리는 문제가 발생한다.

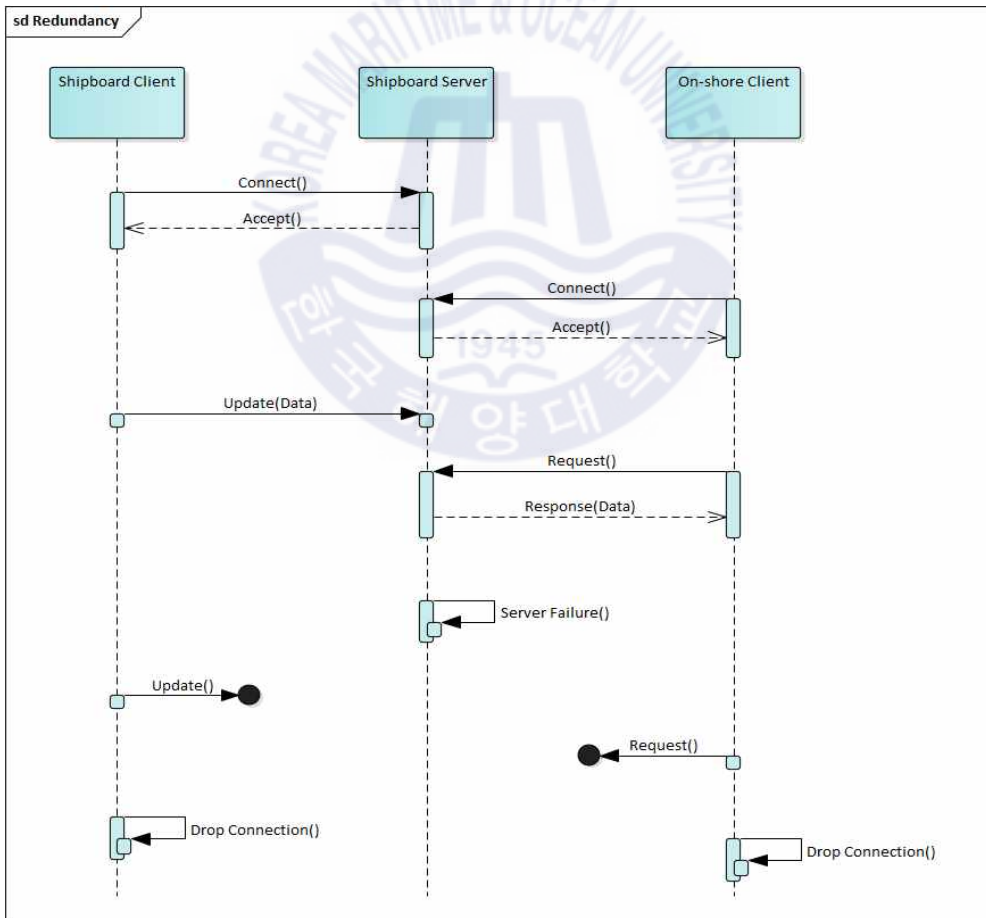


Fig. 23 Termination of connection due to an error of the ship server

이를 해결하기 위해 선박에 이중화된 서버를 추가로 설치하고, 선박 서버의 상태를 주기적으로 감시한다. 만약 서버에 문제가 발생하게 되면 이를 감지하여 이중화된 보조 서버가 서버 역할을 대신하게 되어 지속적인 서비스를 가능하게 한다.

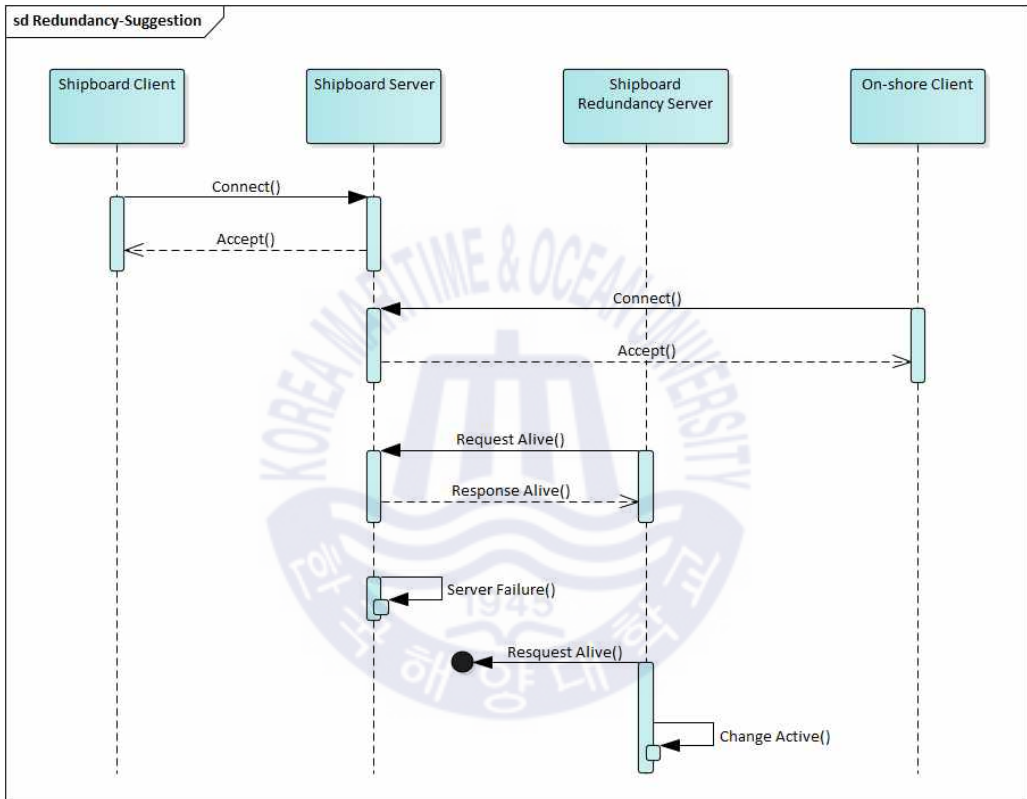


Fig. 24 Countermeasure in Server error situation through system redundancy

4.3 원격 모니터링 소프트웨어 아키텍처 설계

앞서 4.1절에서는 원격 모니터링 플랫폼의 데이터 교환 표준인 ISO 19847에 대해 HAZOP 과정을 통해 도출된 일탈과 그에 따른 안전대책에 대해 살펴보았고, 4.2절에는 반정형기법을 이용하여 안전대책을 구체적으로 분석하여 4가지 위험성 감소방안을 제시하였다.

이번 4.3절에서는 앞서 제시한 4가지 위험성 감소방안에 따라 기존의 소프트웨어 아키텍처를 수정하여 설계하는 것을 다룬다. 위험성 감소방안 중 데이터 입/출력량 조절과 재접속 메커니즘 적용은 아키텍처 설계시 함께 고려되어야 하므로 하나로 묶어서 4.3.1절에서 설명하고, 오류확인 메커니즘과 시스템 이중화는 그 뒤에 별도의 절로 나누어 설명하였다.

4.3.1 데이터 입/출력량 조절 및 재접속 메커니즘 설계

MQTT 패킷 구조는 Fig. 25와 같이 2바이트의 Fixed header와 메시지 종류(Message Type)에 따라 포함될 수 있는 Variable Header 및 Payload로 이루어져 있다.

Bit	7	6	5	4	3	2	1	0
Byte 1	Message Type				DUP Flag	QoS Level		RETAIN
Byte 2	Remaining Length							
Byte 3 ... Byte n	Optional: Variable Length Header							
Byte n+1 ... Byte m	Optional: Variable Length Message Payload							

Fig. 25 Structure of MQTT protocol

2바이트의 Fixed Header는 다음 Fig. 26과 같이 정의되어 있으며 실제로 사용되는 필드는 Message Type과 Remaining Length 필드이며, DUP Flag, QoS Level, RETAIN 필드는 사용하지 않는다.

Bit	7	6	5	4	3	2	1	0
Byte 1	Message Type (15)				DUP Flag	QoS Level		RETAIN
	1	1	1	1	x	x	x	x
Byte 2	Remaining Length (3)							
	0	0	0	0	0	0	1	1

Fig. 26 Structure of fixed header

Variable Header는 다음 Fig. 27과 같이 정의되어 있으며, 설정 코드 1바이트와 설정될 값 2바이트로 구성되어 있다.

Bit	7	6	5	4	3	2	1	0
Byte 1	Configuration Code							
Byte 2, 3	Configuration Value							

Fig. 27 Structure of variable header

Fig. 26에서 표시된 Fixed Header의 Message Type은 Table 13과 같이 정의되어 있다[14].

Table 13 Definition of message type

Mnemonic	Enumeration	Description
Reserved	0	Reserved
CONNECT	1	Client request to connect to Server
CONNACK	2	Connect Acknowledgment
PUBLISH	3	Publish message
PUBACK	4	Publish Acknowledgment
PUBREC	5	Publish Received
PUBREL	6	Publish Release
PUBCOMP	7	Publish Complete
SUBSCRIBE	8	Client Subscribe request
SUBACK	9	Subscribe Acknowledgment

UNSUBSCRIBE	10	Client Unsubscribe request
UNSUBACK	11	Unsubscribe Acknowledgment
PINGREQ	12	PING Request
PINGRESP	13	PING Response
DISCONNECT	14	Client is Disconnecting
Reserved	15	Reserved

본 논문에서는 데이터 입/출력량 조절 및 재접속 메커니즘을 적용하기 위하여 예약으로 남겨져 있는 Fixed Header의 Message Type을 Table 14와 같이 정의하고 Variable Header의 Configuration Code를 Table 15와 같이 정의한다.

Table 14 Additional definition of message type

Mnemonic	Enumeration	Description
MQTTCFG	15	MQTT Service Configuration

Table 15 Additional definition of configuration code

Mnemonic	Enumeration	Description
Reserved	0	Reserved
PUBMIN	1	Minimum publish interval(seconds)
RECONNMIN	2	Minimum reconnection interval(seconds) : When TCP connection lost

입출력량 조절을 위한 최소 출판(Publish) 주기를 설정하기 위해서는 Fig. 28과 같이 클라이언트가 서버로 접속을 시도하고 서버가 연결을 수락한 이후 PUBMIN을 설정코드(Configuration Code)로 하여 MQTTCFG 메시지를 클라이언트로 전송한다. 마찬가지로 서버는 클라이언트의 최소 재접속 시도시간을 설정

하기 위하여 RECONNMIN을 설정코드로 하여 MQTTCFG 메시지를 클라이언트로 전송한다. 서버는 클라이언트의 불필요한 동작을 사전에 방지하기 위해 접속 성공 후 바로 제어 메시지를 송신하도록 하며, MQTTCFG 메시지는 상황에 따라 언제든 송신하여 서버가 클라이언트를 제어할 수 있도록 한다.

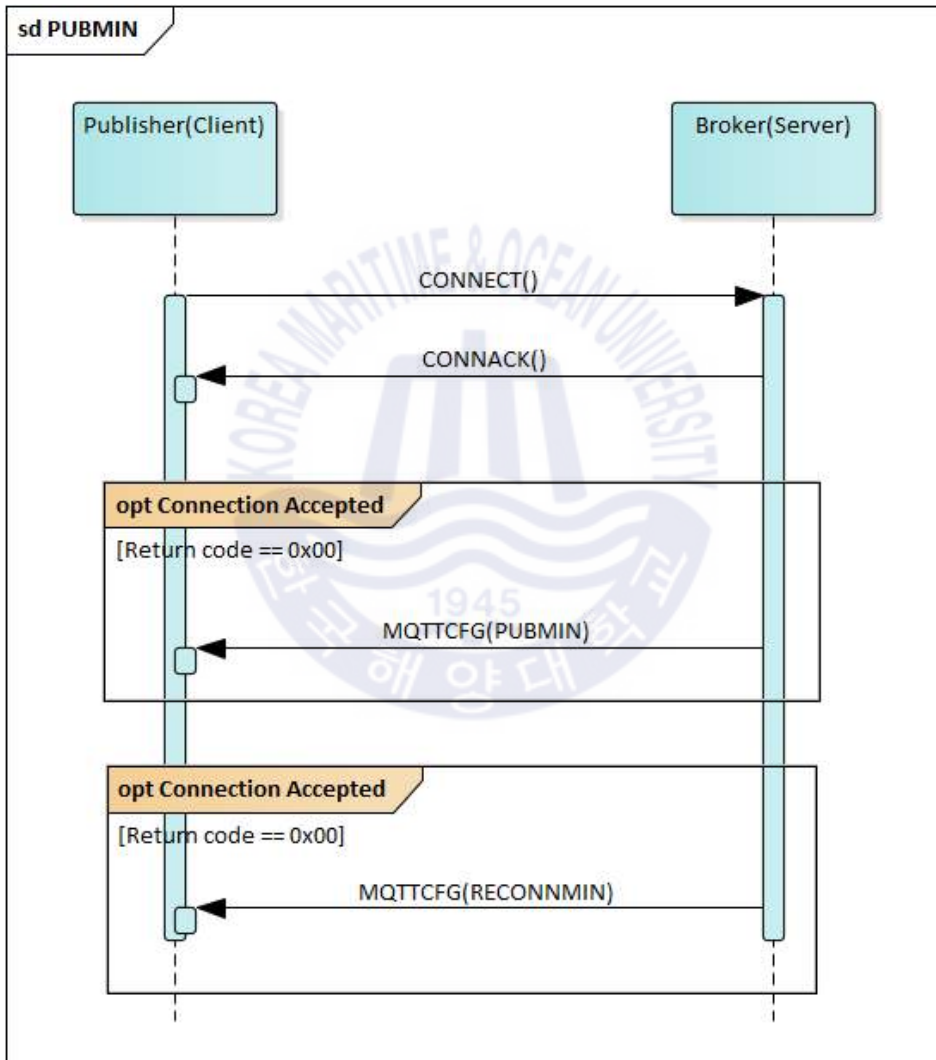


Fig. 28 Sequence of applying minimum publish interval and reconnection interval

4.3.2 오류확인 메커니즘 설계

MQTT 프로토콜은 데이터의 안정적 송수신을 위하여 QoS를 위한 메시지를 정의하고 있다. 본 논문에서는 기존의 QoS를 위한 MQTT 프로토콜의 구조적인 부분은 그대로 사용하고, 절차적인 부분의 개선을 제안한다.

QoS를 위한 PUBACK 메시지의 구조는 Fig. 29와 Fig. 30에서 보는 바와 같다. Fixed Header의 Message Type은 PUBACK를 의미하는 4로 설정되고, Remaining Length는 뒤에 나올 Variable Header의 크기인 2로 설정되며 나머지 필드는 사용되지 않는다.

Bit	7	6	5	4	3	2	1	0
Byte 1	Message Type (4)				DUP Flag	QoS Level		RETAIN
	0	1	0	0	x	x	x	x
Byte 2	Remaining Length (2)							
	0	0	0	0	0	0	1	0

Fig. 29 Structure of fixed header of PUBACK message

또한, Variable Header는 PUBLISH 메시지를 보낼 때 사용하였던 Message ID를 설정하여 해당 메시지가 잘 수신되었음을 응답한다.

Bit	7	6	5	4	3	2	1	0
Byte 1	Message ID MSB							
Byte 2	Message ID LSB							

Fig. 30 Structure of variable header of PUBACK message

본 논문에서 제안하고자 하는 QoS의 절차적인 개선의 기본 아이디어는 구독자가 PUBLISH 메시지를 받은 후 잘 수신하였다는 확인 메시지(PUBACK)를 중계자에 송신하도록 추가하도록 하는 것이다.

기존의 QoS1은 중계자가 구독자에게 출판 메시지를 송신한 후에 바로 메시지를 삭제하였지만, 개선된 방식은 구독자가 출판 메시지를 받으면 PUBACK 메시지를 중계자에게 송신하고 중계자는 구독자로부터 PUBACK를 수신할 때까지 메시지를 삭제하지 않고 보관하도록 한다.

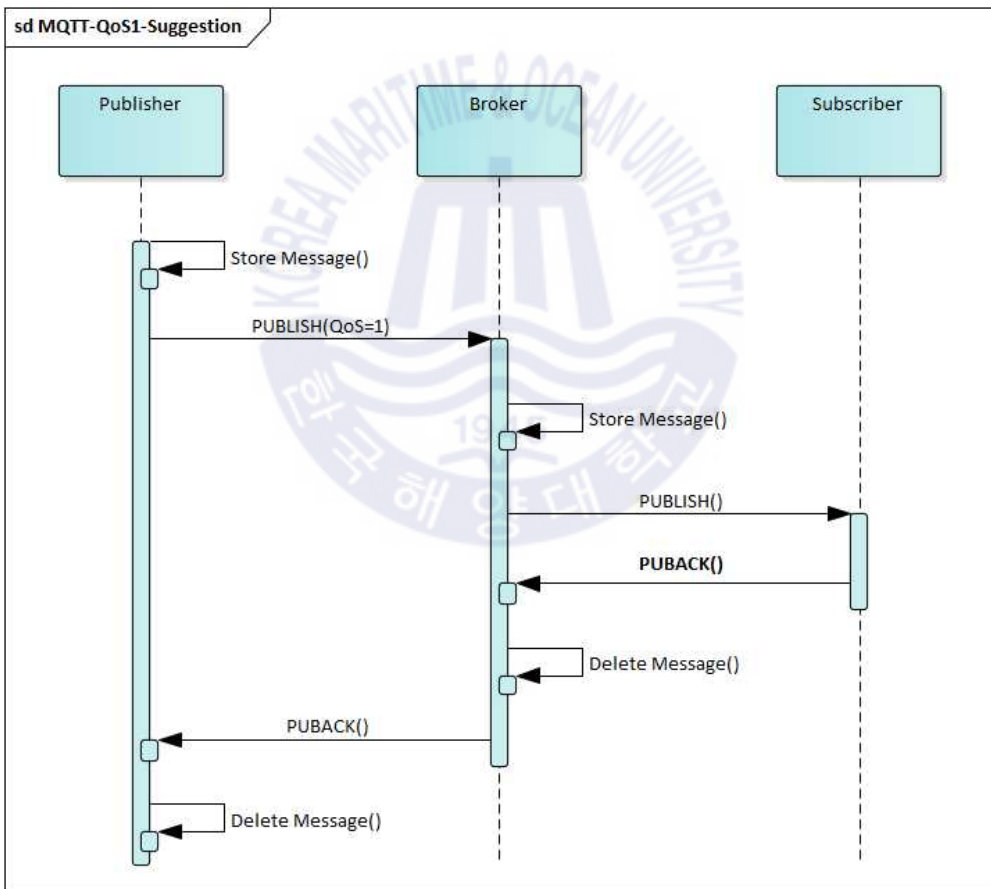


Fig. 31 Suggest of QoS1 sequence in MQTT protocol

기존의 QoS2는 중계자가 구독자에게 출판 메시지를 송신한 후에 PUBREC 메시지를 통해 중계자가 출판자로부터 메시지를 잘 수신하였다는 메시지를 전송하고 그에 따른 응답으로 PUBREL 메시지를 수신 후 중계자는 메시지를 삭제하였다. 하지만 앞선 QoS1에서와 마찬가지로 구독자로부터 PUBACK 메시지를 이용해 구독자가 메시지를 잘 수신하였다는 응답을 받은 후 메시지를 삭제하도록 절차를 변경하였다.

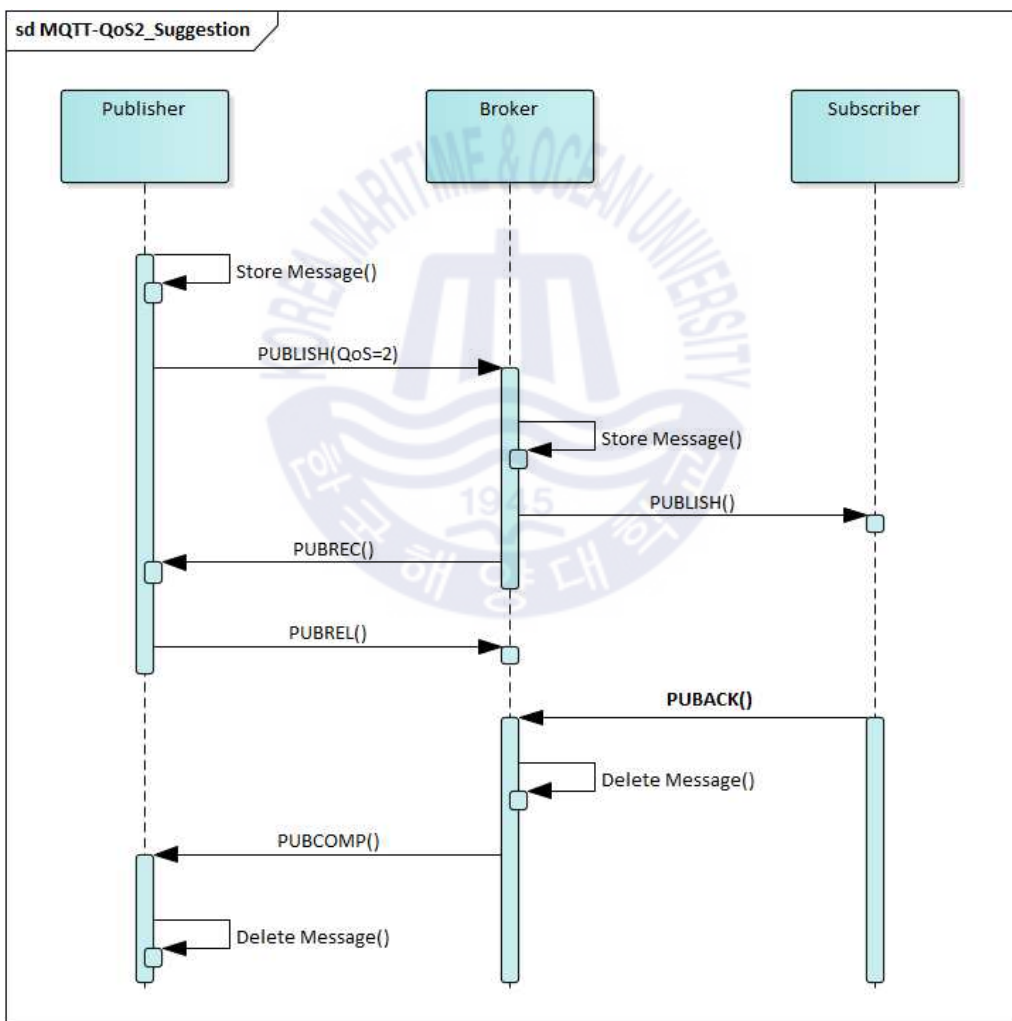


Fig. 32 Suggest of QoS2 sequence in MQTT protocol

4.3.3 시스템 이중화

서버를 이중화하여 하나의 서버에 문제가 발생하면 다른 하나의 서버가 가동하여 서비스를 지속시켜 주는 것은 좋은 방법이다. 하지만 프로그램적으로 깊이 있게 살펴보면 서버에 TCP 연결이 되어 있고 이를 종료하고 다시 재접속하는 메커니즘이 필요하게 된다. 이는 클라이언트 프로그램의 구성을 복잡하게 만드는 원인이 된다. 그러므로 로드 밸런서를 이용하여 이를 극복하고자 한다. 로드 밸런서는 트래픽이 많을 때 여러 대의 서버의 상태(부하율 증가, 부하량, 속도 저하, 작동불능 등)를 고려하여 적절히 트래픽을 분산처리하여 해결해 주는 장비를 말한다.

많은 양의 사용자가 서버에 접속하는 경우 대량의 트래픽이 발생하고, 이로 인한 네트워크 부하, 서버 시스템상의 부하 등으로 서비스 장애가 발생할 수 있다. 로드 밸런서는 Fig. 33에서와 같이 대량의 트래픽을 여러 대의 서버로 분산하여 안정적인 네트워크 환경을 유지하고, 서버의 시스템 오류 또는 하드웨어 오류 등으로 시스템 장애가 발생하더라도 정상적으로 구동되는 서버를 통해 시스템 장애 상황에서도 안정적인 서비스가 가능하도록 해준다.

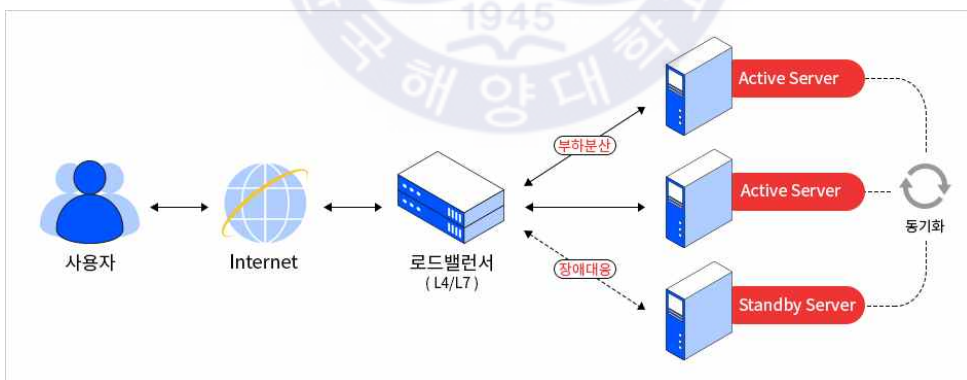


Fig. 33 Concept of Load Balancer

로드 밸런서가 추가되면 기술적으로는 패킷 검사와 패킷 라우팅에 추가적인 시간이 더 소요될 수 있으며, 경제적으로는 장비의 설치 비용과 공간이 더 필요

요하게 된다. 하지만, 안정적인 서비스의 제공과 선박시스템의 보안을 위하여 발트국제해사협의회(BIMCO, Baltic and International Maritime Council) 및 영국 선급(LR, Lloyd's Register), 미국선급(ABS, American Bureau of Shipping)은 사이버 보안에 관한 다양한 지침들을 마련하고 있으며, 이러한 지침들의 사례로써 본 논문에서는 Fig. 34와 같이 로드 밸런서를 이용한 시스템 이중화 아키텍처를 제안한다.

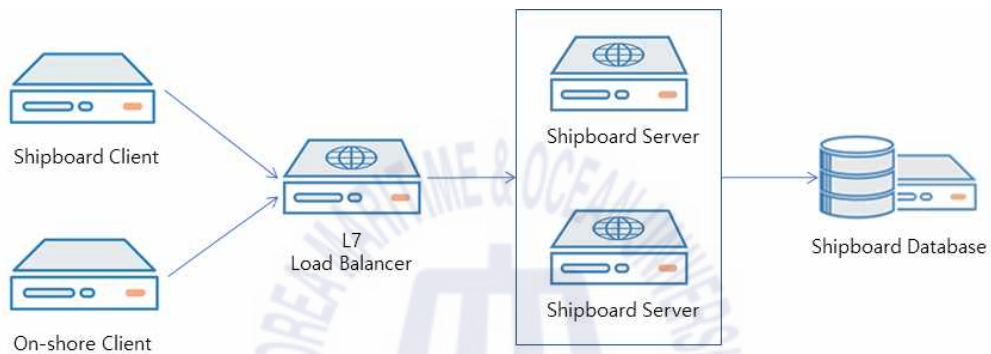


Fig. 34 Architecture of system redundancy

로드 밸런서를 전면에 배치하고 후단에 서버를 이중화하여 트래픽을 분산하고 서버가 갑작스러운 작동 불능이 되었을 때 로드 밸런서에 의해 나머지 서버에 적절하게 패킷을 전달하여 지속적인 서비스를 가능하게 해준다. 일반적으로 로드 밸런서는 OSI 모델의 4계층 로드 밸런서와 7계층 로드 밸런서가 있는데, 애플리케이션 수준의 세션 유지 및 자연스러운 서비스를 위해서 L7 로드 밸런서 사용을 하고자 한다.

클라이언트가 서버에 접속을 시도하거나 데이터를 송신할 때 로드 밸런서가 패킷을 먼저 수신하고 서버 상태를 확인한 후 어느 서버에서 패킷을 처리할지를 결정한다. 만약 서버1이 작동 불능 상태가 되더라도 로드 밸런서는 서버 상태를 확인하여 서버2로 패킷을 전달한다. 클라이언트와의 연결은 서버1과 이루어졌지만, L7 로드 밸런서에 의해 애플리케이션 수준의 세션 유지가 된 상태로 서버2로 서비스가 이어질 수 있다.

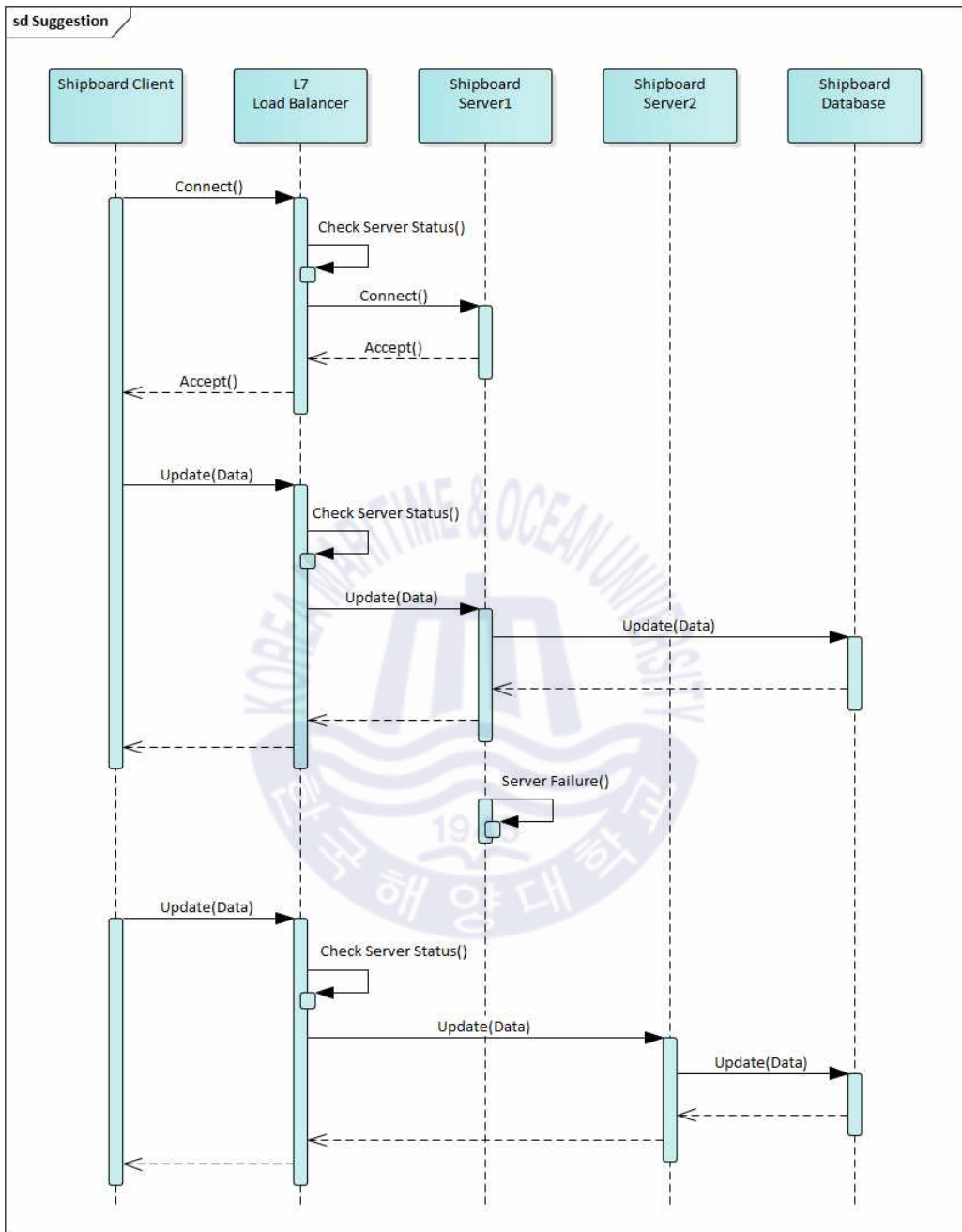


Fig. 35 Error handling procedure by system redundancy

제 5 장 시스템 구현 및 검증

본 장에서는 4장에서 ISO 19847의 기능안정성 평가를 통해 도출된 문제점을 보완하기 위해 설계한 내용을 토대로 시스템을 구현하고 실험을 통해 선박시스템 원격 모니터링 플랫폼 설계의 타당성을 검증한다.

5.1 개발 환경

선박시스템 원격 모니터링 플랫폼의 개발 환경은 Table 16과 같으며, MQTT 공개 라이브러리인 Mosquitto와 오픈소스를 본 논문에서 설계한 형태로 변경하여 재컴파일해야 하므로 다양한 외부 라이브러리를 사용하였다.

Mosquitto 라이브러리의 소스 코드는 리눅스 기반의 C언어로 작성되어 있어 윈도우즈 환경에서 컴파일할 수 있도록 개발PC 및 개발도구의 환경설정을 변경하여 Visual Studio 2015 환경에서 라이브러리를 수정 및 생성하였다. 실험을 위한 MQTT 중계자와 출판자 및 구독자의 응용 프로그램은 Microsoft Visual Studio 2015 환경에서 .NET Framework 4.5.2 기반의 C#언어를 이용하여 Windows Forms 응용 프로그램으로 개발하였다.

Table 16 Software development environment

Item	Environment
Operating System	Microsoft Windows 10 Professional x64
Development Tool	Microsoft Visual Studio 2015
Development Language	Microsoft .NET Framework 4.5.2 Based C, C#
External Library	Mosquitto v.1.6.7
	pthread-win32 v.2.9.1
	OpenSSL Win32 v1.1.1d
	libwebsocket v2.4.1

5.2 실험 환경

시스템을 실험하는 데 필요한 테스트 바지(Barge)선과 황산화물 저감시스템, 인말새트(Inmarsat) FB-250 위성통신 시스템들은 기존에 구축된 설비들을 활용하였으며, 본 논문에서 개발한 소프트웨어를 기존의 임베디드 장비에 탑재하여 실험을 진행하였다. 실험을 위해서는 통신환경이 좋지 않은 상황이 필요하므로 정지궤도 위성인 인말새트 FB-250이 설치된 테스트 바지선에서 실제 선박에서의 통신환경과 유사하게 구성하였다.

Fig. 36의 구성도에서 보는 바와 같이 테스트 바지선에서 설치된 황산화물 저감 시스템(SOx Scrubber)에서 발생한 데이터는 MQTT 출판자를 거쳐 MQTT 중계자에게 전달된다. MQTT 중계자는 인말새트 FB-250 통신망을 이용하여 육상의 구독자에게 메시지를 전달한다.

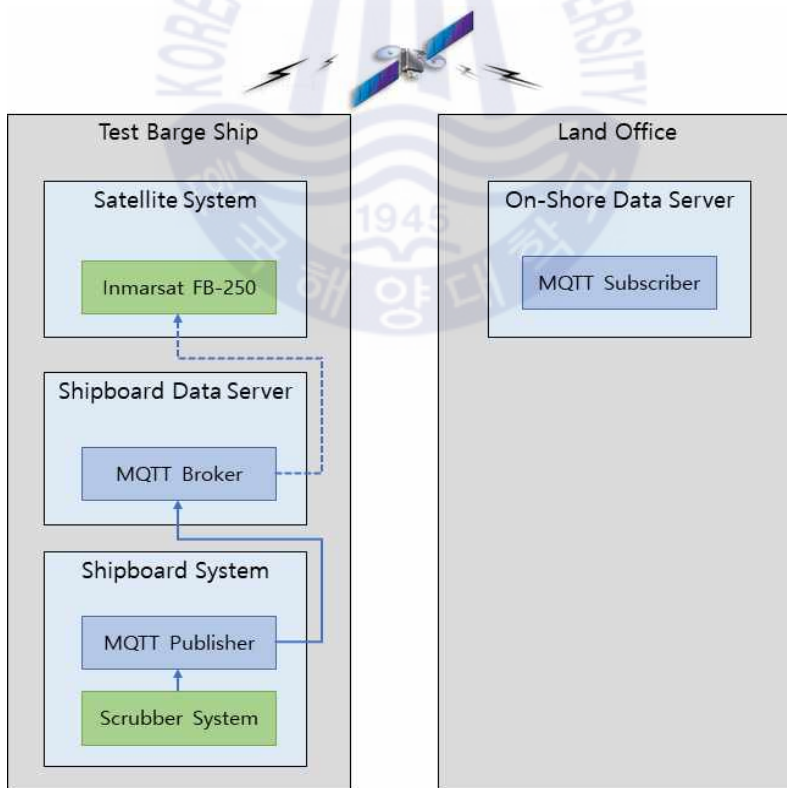


Fig. 36 Diagram of experimental environment

테스트 바지선에는 Fig. 37의 (A)에서 보는 바와 같이 인말세트 FB-250 안테나가 설치되어 있으며, (B)는 실험에서 실제 데이터를 생성하는 역할을 할 선박 시스템인 황산화물 저감시스템의 제어기 판넬의 모습이다.

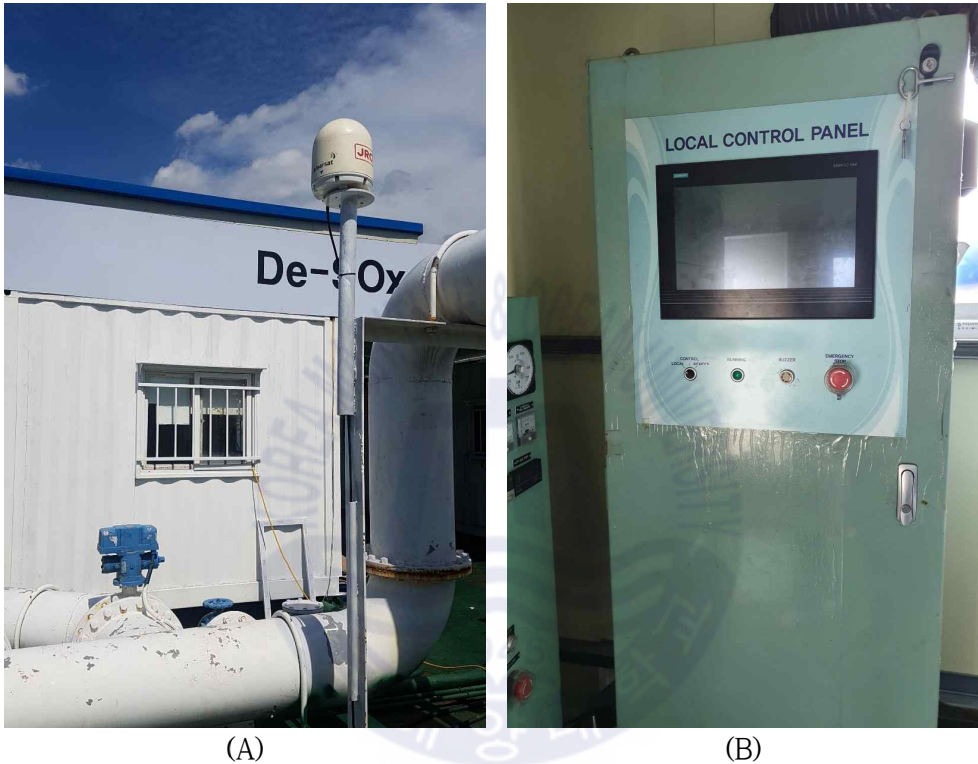


Fig. 37 Facility of experiment

제어실 내부에는 Fig. 38에서와 같이 수신된 위성 전파를 통신 신호로 바꾸어 주는 모뎀인 (A)부분과 방화벽인 (B)부분, 황산화물 저감시스템과 연결되어 데이터를 출판하는 MQTT 출판자 장치인 (C)부분, MQTT 중계자가 탑재될 (D)부분으로 구성되어 있다.

육상의 MQTT 구독자는 노트북을 이용한 사무실 환경으로 Fig. 39와 같다.

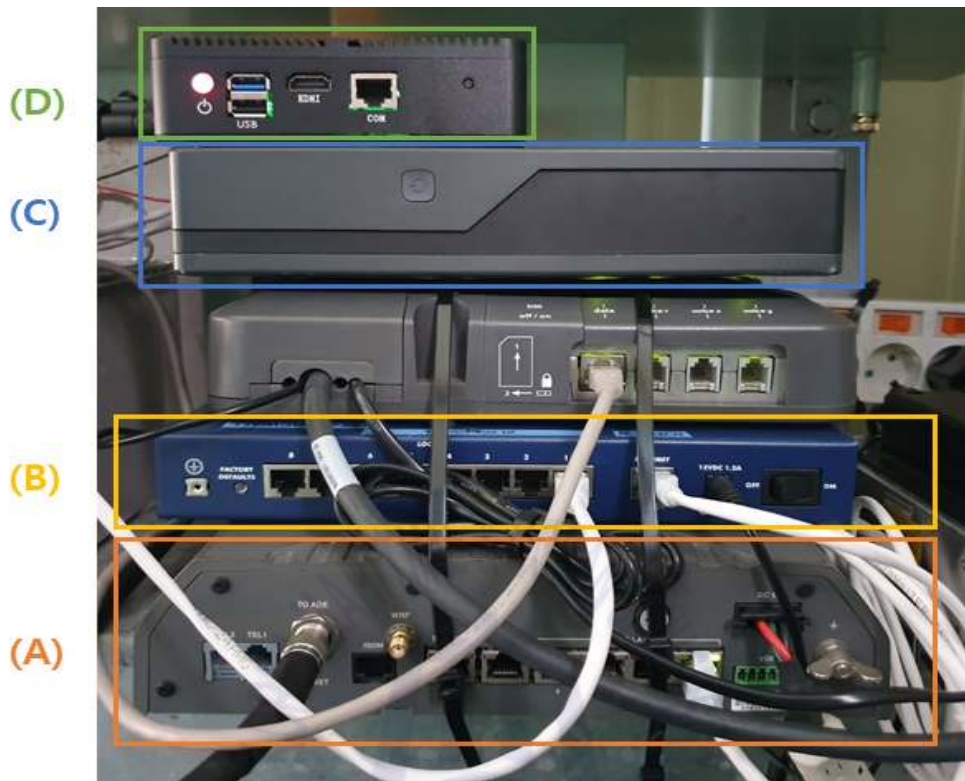


Fig. 38 Device placement inside the control room



Fig. 39 MQTT subscriber devices in an office environment

5.3 시스템 구현

본 논문에서 제시한 소프트웨어 아키텍처에 따라 Mosquitto 오픈소스 라이브러리를 수정하여 MQTT 중계자와 MQTT 출판자, MQTT 구독자 라이브러리를 생성하였다. 원래 Mosquitto 오픈소스는 콘솔 응용프로그램 형태로 개발되어 있으나 실험의 원활한 진행과 분석을 위하여 GUI(Graphic User Interface)를 가지는 응용 프로그램을 추가로 개발하였다.

Fig. 40은 MQTT 출판자 화면으로 접속을 위한 기본정보와 출판을 위한 정보들을 설정할 수 있다. 황산화물 저감시스템은 온도, 압력, 염도, 탁도 등의 다양한 센서 정보를 전달하고 있으나 실험을 위해서 하나의 센서값을 선정하여 출판할 수 있도록 구성하였다. 그리고 신뢰성 있는 전달을 위한 QoS 레벨을 설정하고 기존의 QoS 메커니즘과 본 논문에서 제안한 QoS 메커니즘 중 하나를 선택할 수 있도록 하였다. 또, 최소 출판 주기와 최소 재접속 주기는 설정할 수 없으며 중계자가 설정한 값을 표시하도록 하였다.

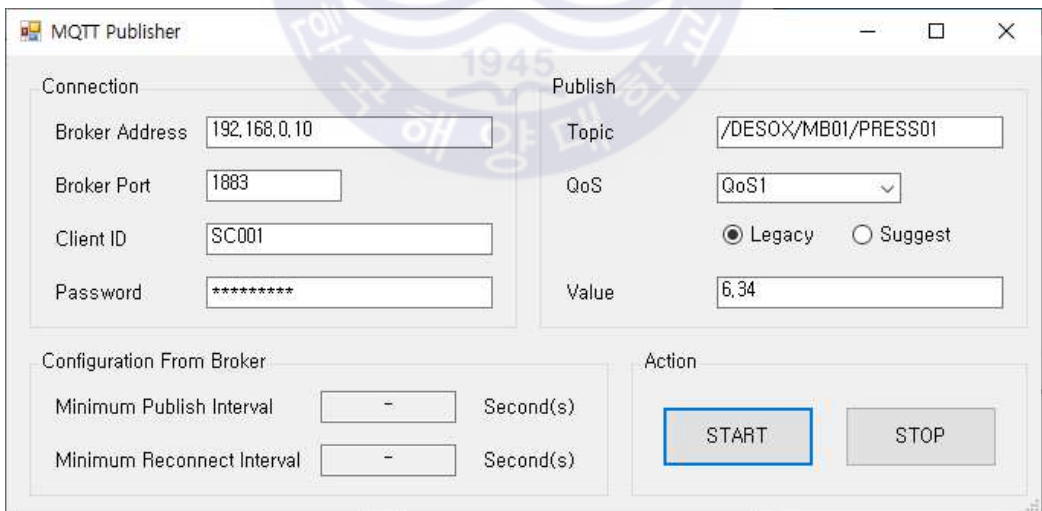


Fig. 40 Screenshot of MQTT Publisher

MQTT 중계자는 화면 좌측에 주제를 트리(Tree)구조로 구성하여 나타내고 주제의 말단 노드를 선택하면 최근 수신 일자와 전송된 값을 표시한다. 화면의 우측에는 연결된 출판자와 구독자 목록을 보여주고 있으며, 목록에 있는 출판자 또는 구독자 중 하나를 선택하고 화면 하단의 최소 출판 주기와 최소 재접속 주기를 설정하면 그 내용이 클라이언트에 전송된다.

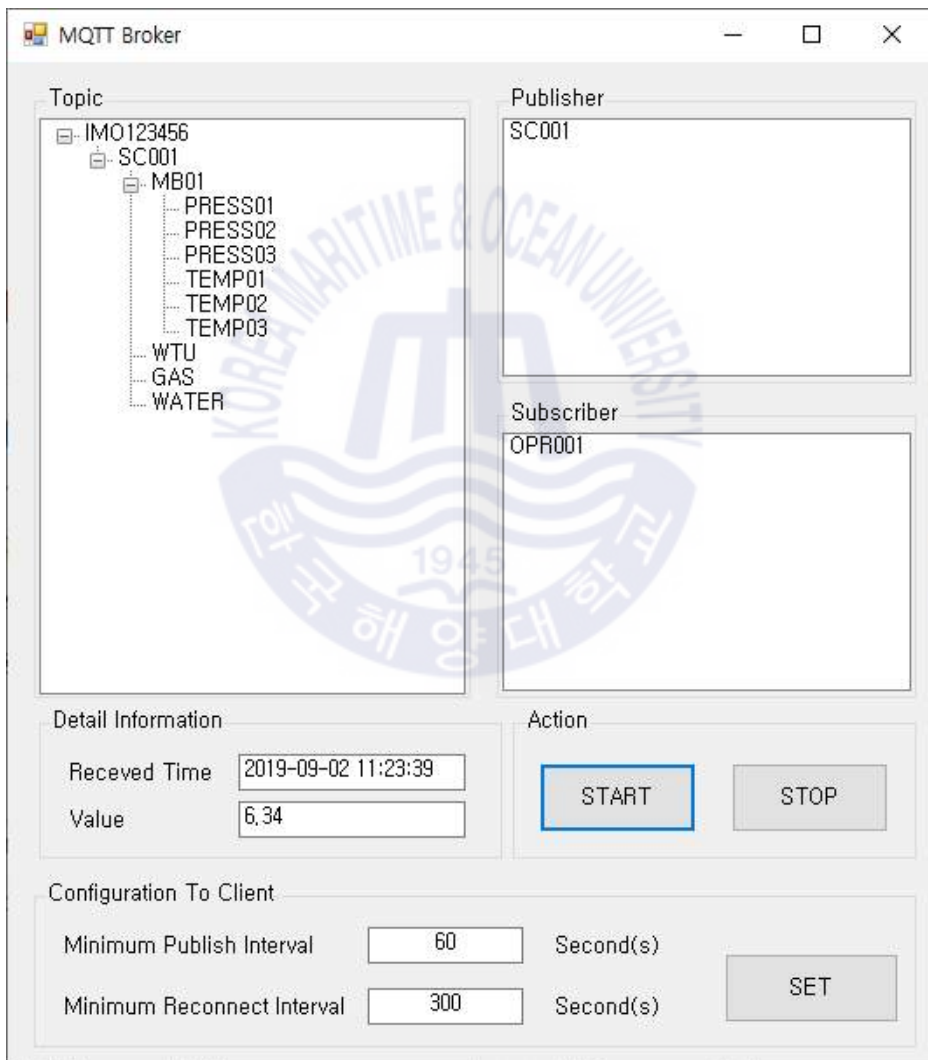


Fig. 41 Screenshot of MQTT Broker

MQTT 구독자는 출판자와 유사한 화면 구성을 하고 있다. 중계자에 접속하기 위한 접속정보와 구독을 위한 주제를 설정할 수 있도록 하였으며 마찬가지로 여러 주제를 구독할 수 있지만, 실험을 위하여 하나의 주제만을 구독할 수 있도록 구성하였다. 중계자로부터 연결이 끊어졌을 때 재접속 시도를 위한 최소 재접속 주기는 임의로 설정할 수 없으며 중계자가 설정한 값에 따라 동작하도록 하였다.

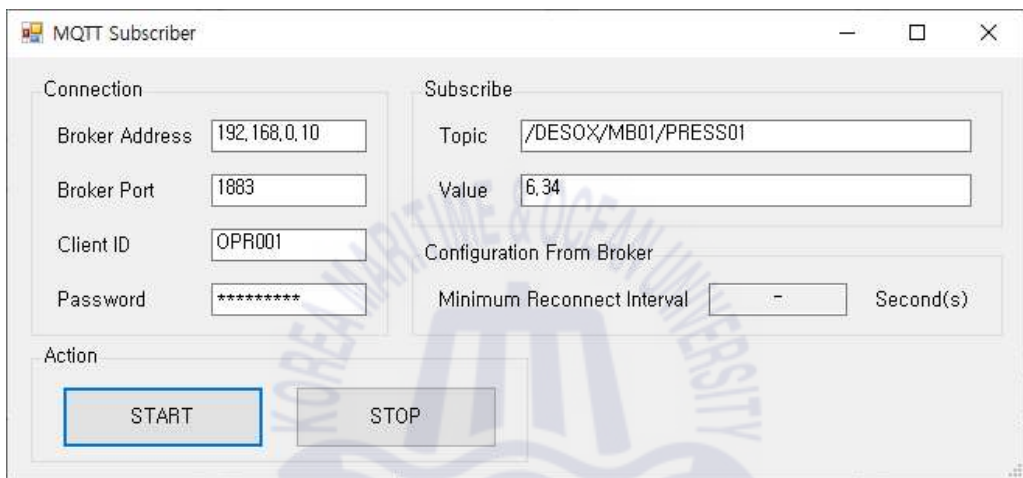


Fig. 42 Screenshot of MQTT Subscriber

MQTT 출판자, 중계자, 구독자 프로그램에는 화면으로 구성되어 있지는 않으나 실험의 진행 과정을 저장하기 위한 데이터 기록 기능을 내장하여 실험 데이터의 분석 및 디버깅을 쉽게 하였다.

5.4 실험 및 검증

테스트 바지(Barge)선에 설치된 실제 운영 중인 황산화물 저감시스템에 MQTT 출판자와 중계자 장치를 연결하고 인말새트(Inmarsat) FB-250 위성통신을 이용하여 본 논문에서 제안하고 있는 데이터 입/출력량 조절과 오류확인 메커니즘, 재접속 메커니즘을 검증하였다. 하지만, 시스템 이중화에 대한 시스템 구현 및 검증은 로드 밸런서 장비를 구매하여야 하므로 여건의 제약으로 인하여 제외하기로 하였다.

실험의 신뢰성을 확보하기 위하여 실험하는 동안 일정하게 유지해야 하는 통제변수를 도출하고, 실험의 목적을 검증하기 위해 의도적으로 조작하는 독립변수와 실험에 따른 결과인 종속변수를 정의하였다.

데이터 사용량을 정확하게 측정하기 위하여 Fig. 38의 (B)에서와 같이 방화벽 장비를 설치하고 실험에 사용된 통신 포트(Port) 외에는 모두 차단하도록 설정하였다. 또한, 패킷 분석도구를 사용하여 패킷 사용량을 측정할 경우에는 실제 사용량과 달라질 수 있으므로 매 실험마다 위성통신 모뎀의 데이터 사용량 카운터를 초기화하여 실제로 위성통신망을 통해 사용한 데이터량을 측정하였다.

5.4.1 데이터 입/출력량 조절

대상 시스템인 황산화물 저감시스템의 컨트롤러는 수십 밀리-세컨드(ms) 이내의 아주 빠른 주기로 아날로그 센서 정보를 수집하고 제어를 수행하며, 변화된 데이터를 일반적으로 10초에 1번의 주기로 외부 시스템에 전송한다. 이대로 MQTT를 적용할 경우 빈번한 출판으로 데이터 사용량이 많아질 수 있으므로 이를 조절하기 위해 본 논문에서 제안한 방식으로 시스템을 변경하고 Table 17과 같이 실험의 변수를 정의하여 실험하였다.

Table 17 Experimental variable of data input/output control

독립변수	전송주기 : 제어 없음(10초), 제어(1분)
종속변수	일간 데이터 사용량
통제변수	주제 및 수량 : 압력(Pressure), 1개
	QoS 레벨 : QoS0

실험의 결과 1분에 1회로 전송주기를 변경하였더니 Fig. 43에서와 같이 하루 동안 약 700KByte를 사용하던 것이 약 280KByte 정도로 사용량이 줄어들었다. 이상적인 상황에서는 데이터 사용량이 1/6로 감소 해야 하지만 위성 접속을 위한 오버헤드, 접속 유지를 위한 오버헤드, 재접속을 위한 오버헤드 등이 추가되어 감소 폭이 다소 적었다. 하지만, 출판 주기를 중계자가 조절할 수 있으므로 위성통신의 상황에 따라 능동적으로 조절할 수 있음을 알 수 있다.



Fig. 43 Experimental result of data input/output control

5.4.2 오류확인 메커니즘 적용

기존 MQTT 프로토콜의 QoS1 및 QoS2 메커니즘에서 중계자와 구독자 사이의 오류확인 메커니즘을 보강하여 설계된 시스템의 성능을 확인하기 위하여 Table 18에서와 같이 실험변수를 설정하여 인말새트 FB-250 위성통신망에서의 기존 QoS1 및 QoS2 메커니즘과 개선된 메커니즘의 일간 데이터 수신율을 비교하였다.

Table 18 Experimental variable of error checking mechanism

독립변수	QoS 메커니즘 : 기존과 개선된 QoS1 및 QoS2 메커니즘
종속변수	일간 데이터 수신율, 일간 데이터 사용량
통제변수	주제 및 수량 : 압력(Pressure), 1개
	전송주기 : 1분

실험의 결과 1분의 전송주기로 하루 동안 출판하는 정보는 8,640개이며, Fig. 44와 Fig. 45에서와 같이 수신율은 기존의 QoS1 및 QoS2 메커니즘을 사용하는 경우에는 평균 71.1%의 수신율을 보였지만 본 논문에서 제안하고 있는 메커니즘을 사용하였을 때는 평균 98.7%의 상당한 성능향상을 보였다. 위성통신 특성상 실험 환경에 따라 정도의 차이는 있겠지만 위성통신 구간에서 상호간 송수신 확인 메커니즘의 추가가 수신율에 영향을 미쳤음을 알 수 있다.

반면에 기존의 메커니즘이나 개선된 메커니즘 모두 QoS1에서 QoS2로 변경한 경우에는 수신율에 큰 차이가 생기지는 않았다. QoS1과 QoS2의 차이는 구독자의 데이터 중복수신 여부이므로 수신율에는 큰 영향을 미치지 않을 뿐만 아니라 중복수신을 회피하는 방식이 출판자와 중계자간의 확인절차에 의해서 이루어지므로 중계자와 구독자 사이의 데이터 수신율에는 별다른 영향을 미치지 않은 것으로 파악된다.

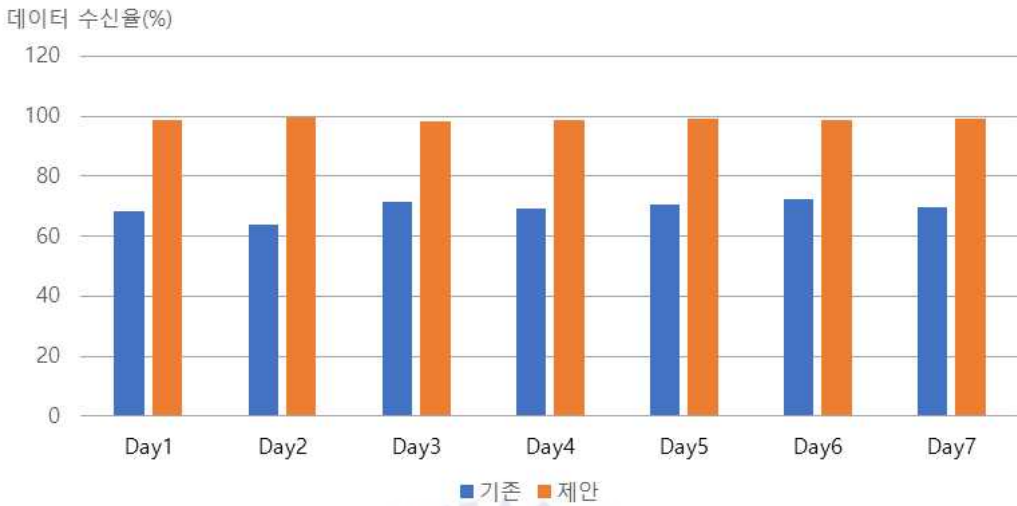


Fig. 44 Results of applying error checking mechanism in QoS1

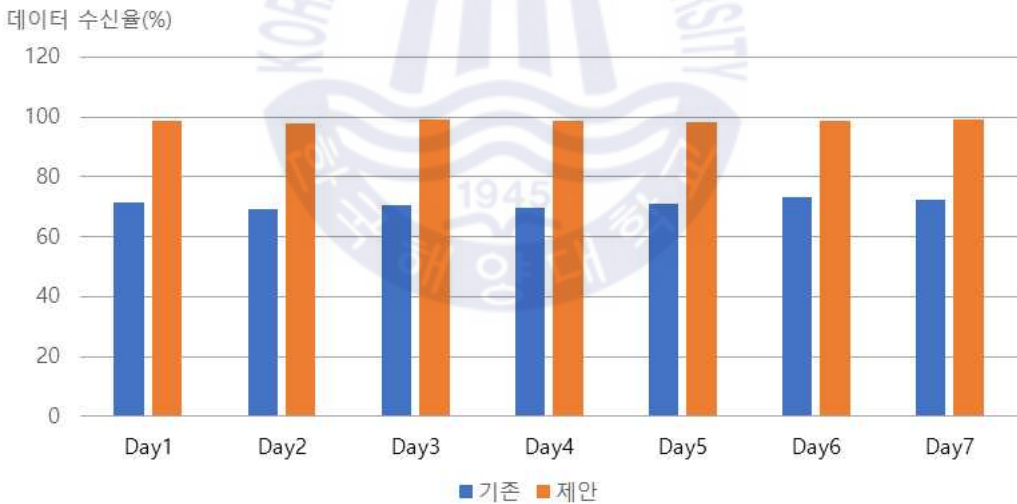


Fig. 45 Results of applying error checking mechanism in QoS2

개선된 오류확인 메커니즘은 선박과 육상 간 통신 구간에 대해 오류 확인을 위한 절차를 추가하였으므로, 위성통신 데이터 사용량이 증가할 것으로 예상된다. 위 실험에서 기존의 방식과 논문에서 제안하는 방식의 데이터 사용량을 비

교환 결과 그림46과 그림47에서 보는 바와 같이 약 34KBytes의 추가적인 데이터가 사용되는 것을 확인하였다. 또한 QoS1과 QoS2의 매커니즘 차이가 위성통신 부분에 영향을 주지는 않으므로 데이터 사용량에는 별다른 영향을 주지 않았다.

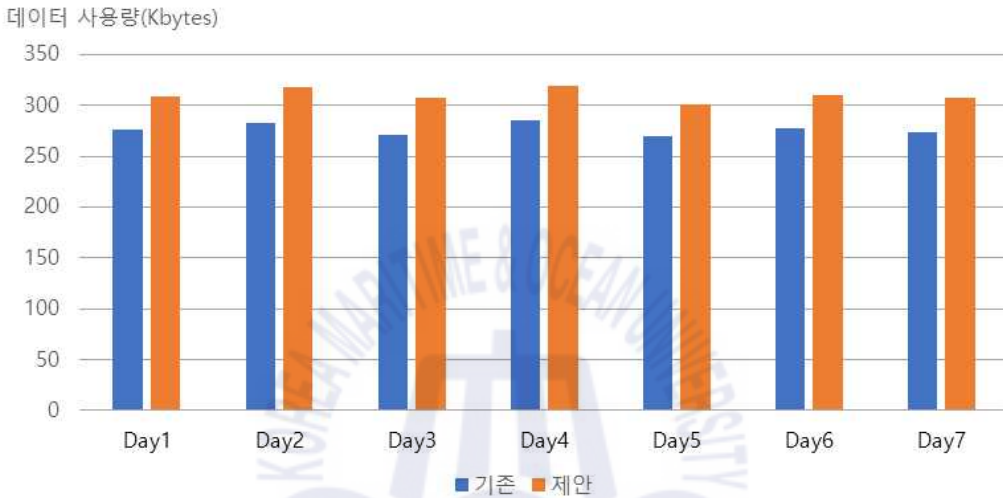


Fig. 46 Result of data usage applying error checking mechanism in QoS1

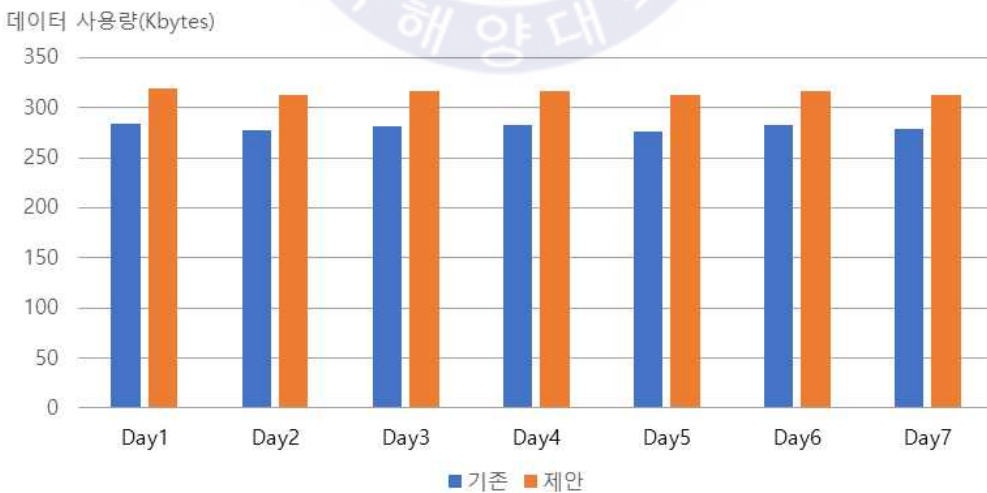


Fig. 47 Result of data usage applying error checking mechanism in QoS2

5.4.3 재접속 메커니즘 적용

일반적으로 MQTT 구독자는 연결이 끊어짐을 감지하면 연결을 유지하기 위해 바로 재접속하도록 소프트웨어를 설계한다. 본 논문에서는 재접속을 시도하는 시간을 중계자(서버)가 설정할 수 있도록 제안하였고, 그로 인한 데이터 사용량 절감효과와 TCP 포트 사용량을 확인하기 위해 Table 19와 같이 실험변수를 설정하여 실제 데이터는 보내지 않고 접속을 유지하는데 자원을 얼마나 소모하는지 실험하였다.

이 실험에서 MQTT 구독자는 연결이 끊어짐을 감지한 뒤 30초 주기로 재연결을 시도하도록 설계되어 있으며, 재접속 주기를 중계자(서버)가 5분으로 설정한 경우와 비교하여 데이터 사용량 및 할당된 TCP 포트 수의 변화를 살펴보기로 한다.

Table 19 Experimental variable of reconnection mechanism

독립변수	재접속 주기 : 30초, 5분
종속변수	데이터 사용량, 할당된 TCP 포트 수
통제변수	주제(Topic) 출판 : 없음
	QoS 레벨 : QoS0

실험의 결과 재접속 주기를 30초로 설정한 경우 연결을 유지하기 위해 하루 평균 131KByte를 사용하는 것을 확인할 수 있었다. 반면에 재접속 주기를 5분으로 변경할 경우 일일 평균 데이터 소모량이 32Kbyte로 감소하였다. 중요한 점은 재접속을 위한 주기는 구독자의 설정에 의해 결정되므로 데이터의 소모량을 중계자가 제어할 수 없었으나 본 논문에서 제안하는 방식을 사용함으로써 중계자가 데이터 소모량을 제어할 수 있다는 것이다.

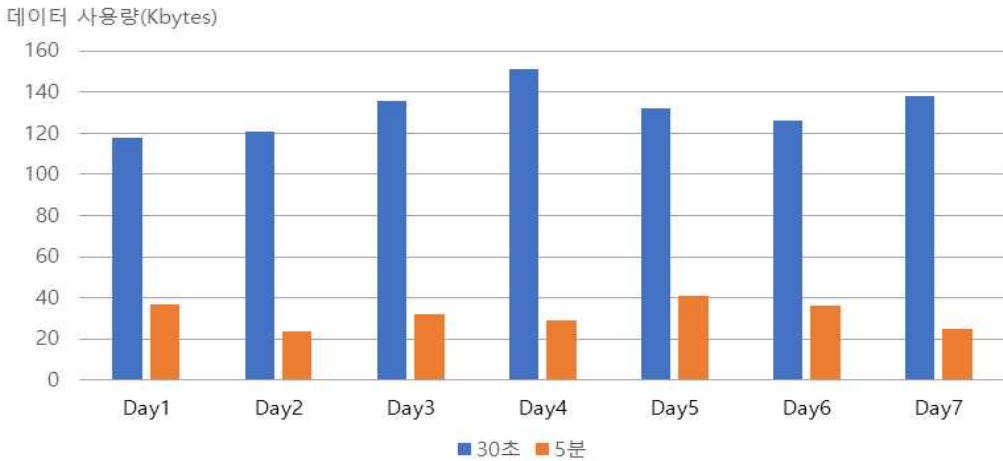


Fig. 48 Result of data usage applying reconnection mechanism

이 실험에서 중계자는 출판자와 구독자가 각 하나씩만 존재하므로 이상적인 경우 TCP 포트는 2개만 할당되어야 한다. 하지만 Fig. 47에서와 같이 통신 안정성이 부족한 경우에는 비정상적 종료와 재접속 등의 이유로 인하여 할당된 TCP 포트가 해제되지 못하고 남아있게 된다는 것을 알게 되었다. 또, 중계자에 의한 재접속 주기 변경을 통해 할당된 TCP 포트 수가 감소함을 확인하였다.

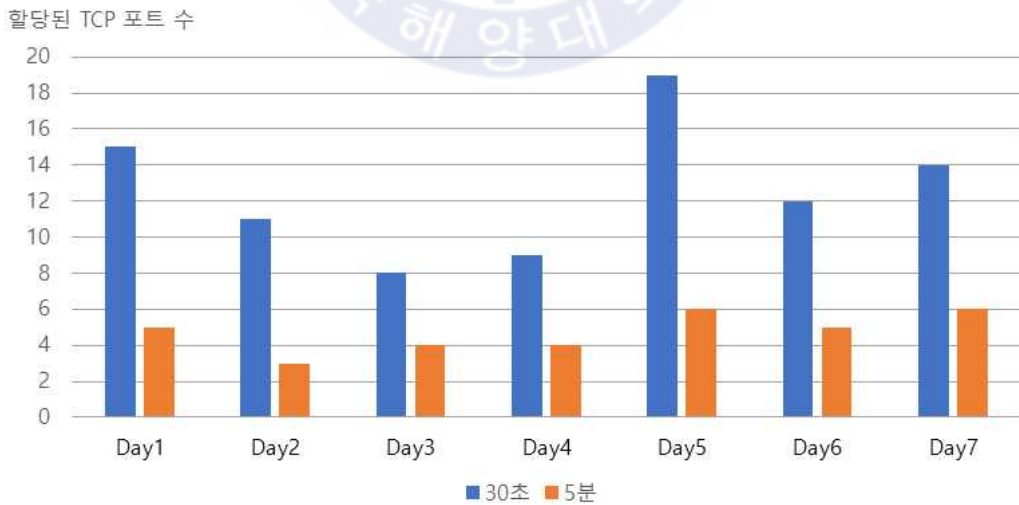


Fig. 49 Number of ports allocated of applying reconnection mechanism

제 6 장 결론 및 향후 연구

자율운항 선박 및 원격 유지보수에 대한 요구가 높아지면서 선박용 장비 제조사들은 문제가 발생하면 수리를 하거나 정기점검을 통해 관리하던 기존 방식에서 벗어나 자사 제품을 원격에서 모니터링하고 자사 제품과 연관된 타사제품들을 공통 데이터 통신 프로토콜을 기반으로 통합하여 관리하려는 움직임을 보이고 있다. 이에 따라 ISO에서는 선박시스템들의 데이터를 수집하고 원격지에서 관리하기 위한 해상데이터 공유 표준인 ISO 19847/19848을 2018년 10월에 제정하였다.

본 논문에서는 ISO 19847/19848이 산업현장에 적용되기에 앞서 소프트웨어 기능안정성 측면에서 HAZOP 방식의 위험성 분석을 수행하여 위험 및 안전대책을 도출하였다. 도출된 안전대책에 대해 IEC 61508-3의 반정형기법을 이용하여 위험을 구체적으로 분석하고 4가지 위험성 감소방안을 제안하였다. 위험성 감소방안에 따라 시스템을 설계하고 MQTT 오픈소스 라이브러리를 수정하여 플랫폼을 개발하였고, 실험을 통해 이를 검증하였다.

IEC 61508의 기능안정성 평가를 통해 본 논문에서 제안하는 ISO 19847/19848 표준의 MQTT 기반 선박 시스템 원격모니터링 시스템의 변경사항은 4가지이다. 첫 번째 중계자에 의한 재전송 주기 설정을 통한 통신 사용량 감소, 두 번째 중계자와 구독자 사이의 QoS 메커니즘 변경을 통한 데이터 수신율 향상, 세 번째 중계자에 의한 재접속 주기 설정을 통한 통신 사용량 감소 및 TCP 포트 자원 확보, 마지막 네 번째 서버 이중화를 통한 서비스의 안정적 공급이다.

실험용 바지(Barge)선에 실제 운영 중인 인말세트 FB-250 위성통신망과 황산화물 저감시스템을 이용한 실험을 통해 첫 번째에서 세 번째까지의 제안은 효용성이 있음을 검증하였으며, 네 번째 제안은 여건의 문제로 검증하지 못하였다.

본 논문에서는 IEC 61508의 기능안정성 평가를 통해 시스템의 기능이 향상될 수 있음을 확인하였다. IEC 61508 표준에서는 기능안전성 확보를 위한 방법을 다루었지만, 구체적인 실행방법에 대한 명세는 존재하지 않았다. 본 논문을 통해 조선/해양 분야에서 IEC 61508 기반 소프트웨어 기능안전성 확보를 위한 사례 연구(Case Study)가 되었을 것으로 생각된다. 또한, 본 논문에서 제안하고 개발한 시스템을 통하여 ISO 19847/19848 기반의 원격 모니터링 플랫폼을 구현하는 데 있어 도움이 될 것으로 기대된다.



감사의 글

평생을 해야 하는 것이 공부라고 하지만, 모든 일에는 시기가 있고 공부 또한 그 시기가 중요하다고 생각합니다. 사정상 그 시기를 지나버리고 꽤 많은 시간이 흘렀습니다. 결혼하고 아내가 셋째 아이를 임신하고 있던 어느 날, 더는 늦어져서는 안 되겠다는 생각에 다시 시작한 대학원 공부였습니다. 힘들고 어려운 시간 견뎌준 아내에게 고맙다는 말을 전하고 싶습니다.

대학을 졸업하고 10년 동안 소프트웨어를 개발하면서 개발자의 현실에 대해 고민이 많았던 저에게 소프트웨어공학이라는 학문을 알려주시고 앞으로 나아가야 할 길을 보여주신 이서정 지도 교수님께 무한한 감사의 말씀을 올립니다. 항상 한발 앞서 생각하시고 활발한 연구 활동과 새로운 일에 도전하시는 모습이 저에게는 큰 감명이었습니다. 교수님의 가르침을 기억하고 한 단계 성장한 개발자이자 설계자로서의 모습을 보여드릴 수 있도록 노력하겠습니다.

논문 작성에 있어 방향을 제시하여 주시고 세심하게 지도해주신 이서정 지도 교수님께 다시 한번 감사의 말씀을 드립니다. 그리고 바쁜 시간 내어 논문심사에 신경 써 주신 박휴찬 교수님, 정보보안에 대해 많은 가르침 주시고 논문심사에도 좋은 말씀 해주신 이장세 교수님께도 감사의 말씀을 드립니다.

제 인생에 멘토이신 유강주 박사님, 학업과 일을 병행할 수 있도록 도와주시고 조언을 아끼지 않으신 천상규 박사님, 네트워크 및 표준화 동향에 대해 많은 가르침을 주신 이광일 교수님, 같이 수학했던 김효승님, 심호용님, 정지은님, 김건홍님, 신현철님, 김반석님께도 감사의 말을 전합니다.

마지막으로 저를 낳아주시고 어려운 환경에도 이만큼 키워주신 아버지, 어머니, 늘 묵묵히 응원해 주신 장인어른, 장모님께 고개 숙여 감사에 말씀드립니다. 그리고 영원한 동반자이자 사랑하는 아내 김무정, 세상 그 무엇보다도 예쁜 세 딸 지연, 지민, 지은에게 감사의 마음을 전합니다.

참고문헌

- [1] 이광일, 황승욱, 자율운항선박 국제표준화 동향, 한국정보통신기술협회, 175호, pp.115-122
- [2] 박혜리, 박한선, 김보람, 2018. 8, 자율운항선박 도입 관련 대응정책 방향 연구, 한국해양수산개발원 현안연구 2018-07
- [3] 이윤석, 2018, 자율운항선박(MASS)개발 동향 분석, 한국정보통신기술협회, 178호, pp.20-26
- [4] 이광일, 조선·IT 융합기술 표준화 동향, 2013. 8, 전자통신동향분석, 제28권 4호, pp.10-22
- [5] 임상우, 이서정, 양희석, 2017. 4, 항해장비 소프트웨어 기능안전성 확보를 위한 위험분석 단계 연구, 디지털콘텐츠학회논문지, Vol. 18, No. 2, pp.393-401
- [6] 임상우, 2017, 소프트웨어 기능안전성 확보를 위한 위험분석과 소프트웨어 요구분석 및 항해장비 사례 연구, 한국해양대학교 대학원 석사학위논문
- [7] 우윤태, 2019, ISO 19847/19848 기반 선박 기관부 데이터 관리 시스템 및 모니터링 어플리케이션 개발, 한국해양대학교 대학원 석사학위논문
- [8] ISO, 2013, ISO Std. 19847, Ship and marine technology-Shipboard data servers to share field data at sea
- [9] 김성규, 김용수, 2014, 기능안전을 위한 IEC 61508의 안전수명주기에 관한 연구, 신뢰성응용연구 제14권 제1호, pp.81-91
- [10] 정보통신산업진흥원, 2016. 11, SW신뢰·안전성 확보를 위한 공통 가이드
- [11] IEC 61508, 2010, Functional safety of electrical/electronic/programmable electronic safety-related systems. IEC Publication
- [12] 이용희, 2002, “제품안전을 위한 리스크 평가의 원칙과 체계”, 대한인간공학학회 춘계학술대회
- [13] IETF, 2013. 03, RFC 793 - Transmission Control Protocol
- [14] ISO/IEC, 2016, Information technology - Message Queuing Telemetry Transport(MQTT)