2014년 2월 전자공학회 논문지 제 51권 제 2 호
Journal of The Institute of Electronics and Information Engineers Vol. 51, NO. 2, February 2014 http://dx.doi.org/10.5573/ieie.2014.51.2.197

논문 2014-51-2-26

함정 전투 시스템을 위한 메시지 지향 모델링 도구 설계

(A Design of Message Oriented Management and Analysis Tool for Naval Combat Systems)

송 경 섭*, 김 동 성**, 최 윤 석***

(Kyoung-Sub Song, Dong-Seong Kim© and Yoon-Suk Choi)

요 약

본 논문에서는 함정 전투 시스템을 위한 메시지 지향 모델링 도구의 구조 설계에 대하여 연구한다. 함정 전투 시스템은 다양하고 대규모의 장비와 통신 서비스 그리고 데이터 분산 서비스 등으로 구성되어 있다. 각각의 장비들은 컴포넌트로서 대규모의 메시지를 발생시킨다. 이러한 메시지를 관리하기 위해 메시지 지향 모델링 도구가 개발되었다. 기존 모델링 도구는 중복되는 데이터베이스 테이블로 인해 어플리케이션 성능이 낮은 문제점이 있다. 본 논문에서는 이러한 문제점을 해결하기 위해 보다 효율적인 데이터베이스 설계 방안을 제안한다. 중복되는 테이블 수를 줄이고 어플리케이션의 응답 속도 및 처리시간을 향상시킨다. 실험 결과들은 제안하는 방법이 메시지 지향 모델링 도구 어플리케이션에 적용 가능함을 보이고, 클라이언트 노드로부터 서버로 전송되는 총 데이터양과 서버부하의 감소에 대하여 보여준다.

Abstract

This paper investigates a design of optimization database structure layout of Message Oriented Analysis and Management Tool (MOMAT) for naval combat systems (NCS). The NCS is composed of heterogeneous and large-scale component such as communication service and data distribution servcies (DDS). Each components are massively made the data as components. To manage the messages, MOMAT is developed. Typical modeling tool have problems that low performance due to duplicate database tables. An efficient design that one of the database optimization is proposed to solve the problems in this paper. It reduces the number of tables and improves application response and processing time. Experiment results shows that an availability of proposed method in MOMAT and decrease of both amount of data from client node to server and sever load.

Keywords : Naval Combat Systems, Message Oriented Modeling and Analysis Tool, Application Design

Ⅰ. 서 론

현대 무기 시스템 기술 발전으로 함정 전투 시스템의 설계 및 구조에 변화가 생겼다. 함정 전투 시스템은 예측하지 못한 곳에서 공격하는 적군에 대항하기 위해 실시간성을 보장하고 신뢰성 있는 정보를 제공해야 한다. 다각도로 변화하는 전장의 상황을 연속적으로 분석하기 위해 수많은 센서와 장비들이 실시간으로 데이터를 주고받는다[1~3].

함정 전투 시스템에서 미들웨어의 사용은 다양한 종류의 데이터를 관리할 수 있는 방법 중 하나이다. 특히, Data Distribution Service (DDS)는 메시지 지향 시스템 환경에서 널리 사용되는 미들웨어이다. DDS에서 통신의 수립은 기본적으로 데이터 제공자와 사용자 (publisher / subscriber)의 연결 형태로 이루어진다. 이러한 DDS를 바탕으로 Naval Combat Systems (NCS)와 같은 대규모 시스템에서는 수많은 데이터와 메시지들을 관리할 수 있다, 하지만 이와 같은 대규모 시스템

* 학생회원, ** 정회원, 금오공과대학교
(Kumoh National Institute of Technology)
*** 정회원, 삼성탈레스
(Samsung Thlaes)
© Corresponding Author(E-mail: dskim@kumoh.ac.kr)
※ 본 연구는 교육부와 한국연구재단의 지역혁신인력양성사업으로 수행된 연구결과임
(NRF-2012H1B8A2026109).
접수일자: 2013년9월25일, 수정완료일:2014년2월3일

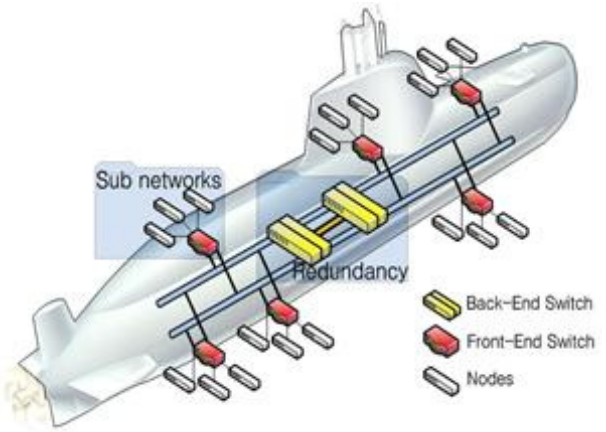

그림 1. 함정용 제어 통신망 배치도의 예
Fig. 1. The example of control network layout for naval combat system.

을 개발하기 위해서는 수많은 개발자들이 참여하고, 협력하여야 한다.

대규모 시스템에서 컴포넌트는 다양한 개발자들에 의해서 구현되어진다. 이 구현과정에서 메시지의 생성, 수정, 삭제 등의 과정이 정확하게 컴포넌트에 적용되지 않는다면 개발 효율이 감소하고, 메시지의 유지보수가 힘들게 된다. 이러한 문제점을 해결하기 위해 message definition & management system (MDMS), message oriented software modeling and anaylysis tool (MOSMAT), real-time data distribution service (RDDS)와 같은 다양한 모델링 도구들이 개발되어져 왔다[4][5]. 그러나 기존 모델링 도구들은 웹 기반 어플리케이션으로서 낮은 확장성, 여러 컴포넌트 간의 인터페이스 관계 파악의 어려움 그리고 유연한 데이터베이스 구조 설계에 관한 문제점들을 가지고 있다.

개발자가 생성한 메시지에 필드를 생성할 경우, 메시지 테이블 및 이 메시지의 위치를 표시한 컴포넌트 테이블에 여러 번 접근해야한다. 수많은 개발자가 동시에 서버로 접근하여 이와 같은 다중 명령들을 수행하게 되면 서버에 집중적인 부하가 가중되고, 이로 인해 MDMS 기반의 개발 도구의 전체 성능이 감소한다.

이를 개선하기 위해서, 본 논문에서는 대규모 시스템 기반의 함정 전투 시스템을 위한 최적화된 데이터베이스 설계 방안을 적용한 MOMAT을 제안한다. 최적화된 데이터베이스 구조는 메시지 테이블과 컴포넌트 테이블의 접근 횟수를 줄임으로서 서버 부하 감소와 동시에 이에 따른 개발 툴의 성능향상에 도움을 준다. OPNET modeler 16.0과 MONyog를 이용하여 기존 개발 도구의 데이터베이스 구조와의 성능 비교를 통하여 MOMAT 데이터베이스 설계 방안의 가능성을 보였다[11~12].

본 논문의 구성은 다음과 같다. Ⅱ장에서는 함정 전투 시스템과 MOMAT에 대하여 다룰 것이다. Ⅲ장에서는 MOMAT에서 발생되는 문제점에 대하여 언급한다. Ⅳ장에서는 MOMAT의 문제점을 해결하기 위한 최적화된 MOMAT 데이터베이스 설계 방안을 제안한다. 마지막으로 Ⅴ장에서는 성능 분석 및 결론을 다룰 것이다.

Ⅱ. 함정 전투 시스템과 MOMAT

1. 함정 전투용 분산 제어 시스템

NCS는 기본적으로 대규모 시스템으로 다양한 무기 시스템과 감시 장비들로 구성되어 있다. 다양한 탐지 및 추적 센서들과 전술 데이터 링크를 통해 수집된 다량의 표적 정보와 전술정보들을 실시간으로 처리, 융합하여 시스템 표적을 생성하고 종합된 전술상황을 전시한다[2]. 이러한 과정에서 대량의 메시지 트래픽이 발생하며, 시스템 크기에 따라 발생되는 정도가 다르다. 그림 1 과 같이 함정 전투 시스템 정보망은 이중화로 구성된 백본망을 기준으로 무기제어, 레이더, 감시, 물체추적 등 각 영역별 서브넷이 연결되어있다.

표 1은 함정 전투 시스템의 노드 수에 따른 분류로서 노드 규모별 메시지관련 특성을 보여주고 있다. 함정 전투 시스템은 DDS를 기반으로 메시지 통신이 이루어지고, 각 시스템 규모별 생성되는 데이터의 양과 크기는 각각 다르다. 총 종간노드는 함정 전투 시스템의 전체 제공자와 사용자를 나타낸다. 총 노드 수는 레이더, 무기, 함정 제어 시스템 등을 나타낸다. 최대 토픽 사이즈는 각 전투 시스템 규모별로 8 KB 와 1 MB 로

표 1. 함정 전투 시스템의 노드수에 따른 분류
Table 1. Classification by number of nodes in NCS.

	Medium-scale systems	Large-scale systems
Total endpoints	13019	24911
Publisher / Subscriber	5413 / 7606	10284 / 14627
Total Nodes	48	140
The largest topic size	8 KB	1 MB
High frequent data	20 Hz	200 Hz
Bursty traffic	3000 Packets / sec	3500 packets / sec

제한된다.

다양한 양과 크기로 생성되는 데이터를 생성하는 각각의 장비들은 컴포넌트(component)로 간주 할 수 있다. 이러한 컴포넌트의 데이터들은 대규모 시스템에서 중규모 시스템보다 더 많은 트래픽을 발생 시키므로 메시지 관리를 통해 수많은 데이터들을 제어해야 한다[3].

2. Data Distribution Service

함정 전투 시스템에서 각 노드별 생성되는 메시지는 DDS와 같은 통신 미들웨어를 이용하여 제어 및 관리된다. DDS는 객체 지향 미들웨어 (Object-Oriented Middleware)와 메시지 지향 미들웨어 (Message Oriented Middleware)로 양분된 통신 미들웨어 시장에서 가장 많이 사용되는 미들웨어중 하나이다. 객체 지향 미들웨어로 대표되는 미들웨어로는 common object request broker architecture (CORBA)와 distributed component object model (DCOM)이 있다. 메시지 지향 미들웨어로는 object management group data distribution service (OMG DDS), RTI사의 DDS, 삼성탈레스의 STC DDS 등이 있다. 대표 메시지 지향 미들웨어인 DDS는 함정 전투 시스템에 널리 사용되며, 통신이 이루어지는 과정은 그림 2와 같다[3-7].

데이터 제공자와 사용자간의 통신이 수립되면 함정 전투 시스템의 다양한 장치들의 데이터들은 제공자로서 역할을 하며 글로벌 데이터 공간에 제공한다. 이곳에 모여진 데이터들은 사용자에게 제공되고, 사용자들은 자신이 필요로 하는 데이터만 가져간다[8~10].

데이터 송수신이 이루어 질 때 DDS는 QoS (Quality of Service)를 이용하여 네트워크 트래픽 및 데이터 전송을 관리한다. QoS는 DDS에서 가장 중요한 기능이며 통신 서비스의 동작을 제어하는 특성을 보여준다. DDS에 관련된 다양한 QoS에 대한 자세한 내용은 [9]를 참고하면 된다.

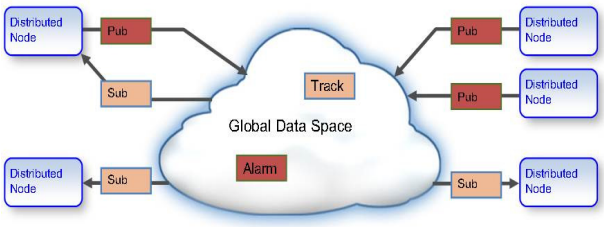

그림 2. 함정 전투 시스템의 DDS 통신 구조
Fig. 2. DDS communication structure of naval combat system.

3. MOMAT의 전체구조

MOMAT은 DDS 표준 기반의 데이터 중심 함정 전투 시스템 제어 통신망에서 각각의 컴포넌트에서 발생되는 수많은 데이터를 관리하기 위하여 개발되었다. MOMAT의 장점 중 하나는 다양한 데이터 메시지들의 버전을 형상관리 기능과 같이 버전별로 관리 할 수 있다. 또한, OMG 그룹에서 정의한 인터페이스 정의 언어 (Interface Definition Language)를 이용하여 코드생성 기능을 제공 할 수 있다. 이러한 장점을 바탕으로 MOMAT은 대규모 시스템의 데이터들을 쉽고 간단하게 관리 할 수 있으며, 새로운 기능을 추가 할 수 있는 확장성도 가지고 있다.

MOMAT은 어플리케이션 운영을 위해 객체 지향 언어이며, 다른 운영체제 및 플랫폼과의 호환성이 높은 C#, MYSQL 기반의 데이터베이스, 컴포넌트, 메시지, 그리고 제공자 / 사용자 이렇게 4 개의 영역으로 구성되어 있다. 그림 3은 MOMAT의 구조를 보여준다. MOMAT에서 중요한 영역은 컴포넌트와 메시지, 그리고 제공자 / 사용자 영역이다. 각각의 영역에 대한 자세한 설명은 다음과 같다.

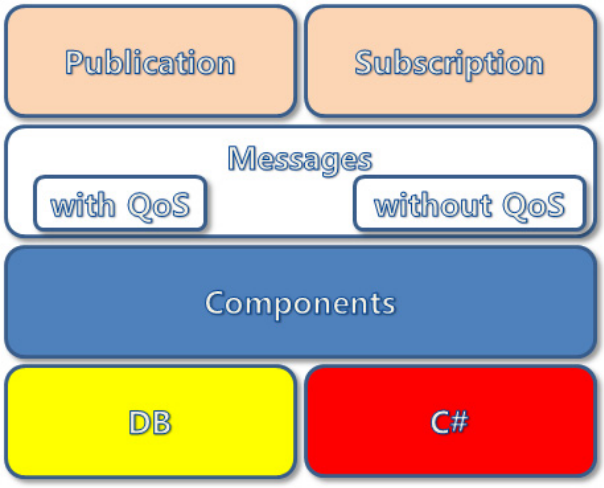

그림 3. MOMAT 설계화의 계층적 구조
Fig. 3. The hierarchical architecture of MOMAT design.

1) 컴포넌트 : MOMAT은 컴포넌트를 기초로 모든 메시지를 관리 할 수 있다. 컴포넌트는 computer software configuration item (CSCI), computer software component (CSC), 그리고 computer software item (CSU)로서 3 가지로 구분할 수 있다. 즉, 각각의 컴포넌트는 다양한 센서 노드들로부터 생성되는 데이터들을 공유하기 위해 글로벌 데이터 공간에 참여한 컴포넌트들과 통신을 할 수 있다.

2) 메시지 : 메시지는 제공자 / 사용자로 등록 될 수 있다. 메시지는 프로젝트에서 유저가 정의하는 데이터 구조가 될 수 있고, DDS에서 토픽과 같은 역할을 한다. 메시지를 통한 데이터 전송은 컴포넌트를 기반으로 동작한다. 각각의 메시지는 QoS를 포함하고 있으며, 이를 이용하여 메시지의 전송방법에 영향을 줄 수 있다.

3) 제공자 / 사용자 : 제공자 / 사용자는 MOMAT에서 중요한 부분 중 하나이다. 글로벌 데이터 공간에서 통신이 이루어지는 최소 단위가 이 제공자 / 사용자 단위이기 때문이다. 앞서 언급한 메시지가 이 둘을 서로 연결하는 매개체가 되고 제공자 / 사용자는 서로 통신의 종단점 역할을 한다.

Ⅲ. 문제점 분석

1. MOMAT의 동작 과정

MOMAT은 기본적으로 클라이언트-서버 구조로 디자인 되었고, 이를 바탕으로 구현되어졌다. MOMAT의 초기 동작은 글로벌 변수를 이용하여 데이터베이스로부터 정보를 불러오고 이를 어플리케이션에 적용한다. 그림 4 는 MOMAT의 초기화 과정을 보여준다.

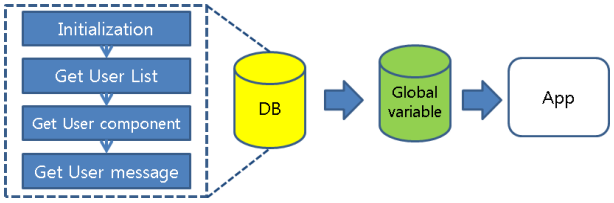

그림 4. MOMAT의 초기화 과정
Fig. 4. Initialization procedure of MOMAT.

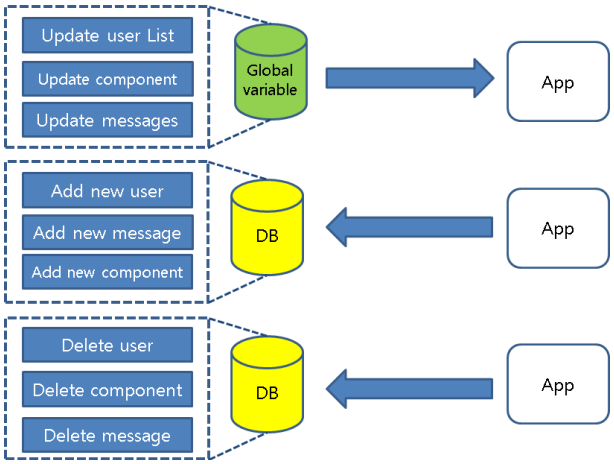

그림 5. MOMAT 데이터베이스 동작 과정
Fig. 5. Database procedure of MOMAT.

이후 데이터베이스의 동작은 메시지나 제공자 / 사용자의 유저의 명령어를 받았을 때 동작한다. 특히 추가 및 삭제 명령어는 데이터베이스에 직접 접근하여 처리한다. 자세한 동작 과정은 그림 5와 같다.

2. 데이터베이스 구조

MOMAT 데이터베이스는 MYSQL을 이용하여 구성된다. 최상위 단계인 프로젝트부터 CSCI, CSC, CSU 단위까지 테이블로서 서로 연계되어있다. 메시지는 토픽과 연계되어 있고, 각각의 특성을 나타내기 위한 필드 테이블과 QoS 테이블과도 연계되어있다. 그림 6은 기존 MOMAT의 데이터베이스 구조를 도식화한 것이다.

이러한 데이터베이스 구조에서 새로운 메시지를 추가하거나 삭제, 제공자 / 사용자로 등록하기 위해서는 매번 메시지 테이블과 CSU 테이블을 검색해야한다. 적은 수의 메시지가 발생할 때는 문제가 되지 않지만, 함정 전투 시스템과 같은 대규모 시스템에서 발생하는 메시지들을 관리할 때는 문제가 발생한다. 특히, 테이블 검색 횟수가 높아짐에 따라 어플리케이션 및 서버의 성능에 영향을 미친다.

테이블 접근횟수가 증가하면서 디스크 입출력 횟수가 증가하면서 이에 따른 지연시간이 발생하고, 대규모로 데이터를 로딩 할 때 병목현상이 발생한다. 또한, MOMAT에서 코드 생성 기능을 이용할 때 다양하고 수많은 인자를 사용하는데 이때 어플리케이션 및 데이터베이스에 부하가 많이 발생한다.

이러한 문제점을 해결하기 위해 본 논문에서는 함정 전투 시스템에 적합한 MOMAT 기반의 데이터베이스

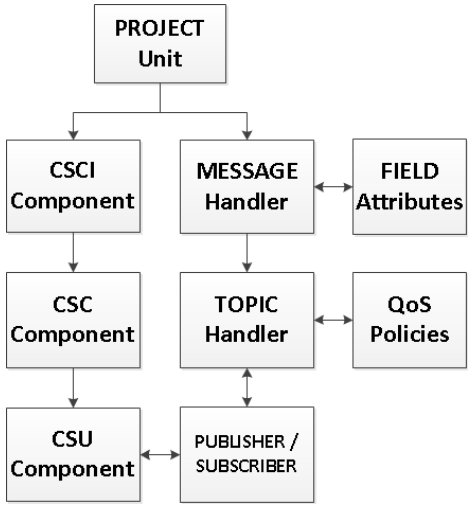

그림 6. MOMAT의 데이터베이스 구조
Fig. 6. The structure of MOSMAT database.

구조 설계 방안을 제안한다. 서로 연관된 테이블을 통합하여 전체 테이블 접근 횟수를 줄이고, 데이터를 보다 효율적으로 관리하기 위함이다.

Ⅳ. MOMAT의 데이터베이스 설계방안

본 논문에서는 MOMAT에서 테이블 접근 횟수 증가에 따른 병목현상 집중 문제점을 파악하고, 이를 해결하기 위한 적절한 데이터베이스 설계 방안을 제안한다.

1. MOMAT의 성능 분석

MOMAT은 서버/클라이언트 네트워크 모델을 기반으로 설계되었다. 수많은 개발자 및 사용자들이 MOMAT을 이용할 때 마다 서버에 접근하면서 데이터베이스를 사용한다.

그림 7은 MOMAT의 네트워크 모델을 보여주고 있다. 중앙 집중형 네트워크 모델로 인해 사용자가 늘어나면 그에 따른 네트워크 및 서버 부하도 선형적으로 증가한다. 네트워크 모델로 인한 트래픽 부하에 관한 문제점은 현재 연구 중에 있다. 이러한 상황에서 기존 어플리케이션인 MDMS와 MOMAT의 성능은 그림 8과 같이 보여준다.

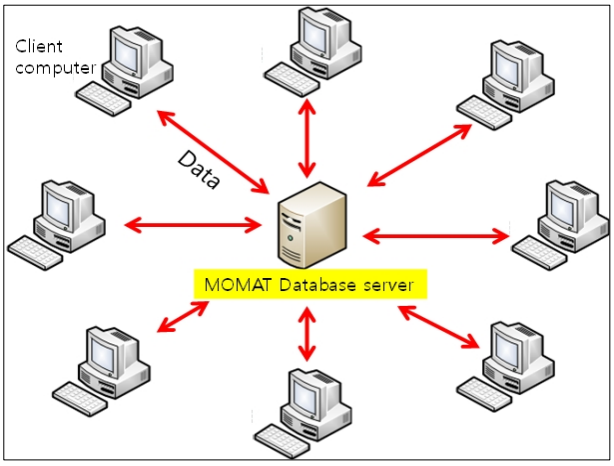

그림 7. MOMAT의 네트워크 모델
Fig. 7. Network model of MOMAT.

2. MOMAT 테이블 구조

데이터베이스의 병목 현상을 줄이기 위해 MOMAT은 MOMAT 데이터베이스에서 테이블 수를 줄였다. 특히, CSCI, CSC, CSU의 테이블은 MOMAT의 모든 명령에 관여되는 테이블이다. 메시지나 토픽이 생산, 삭제, 수정 등의 명령어를 받으면 가장 많은 접근이 이루

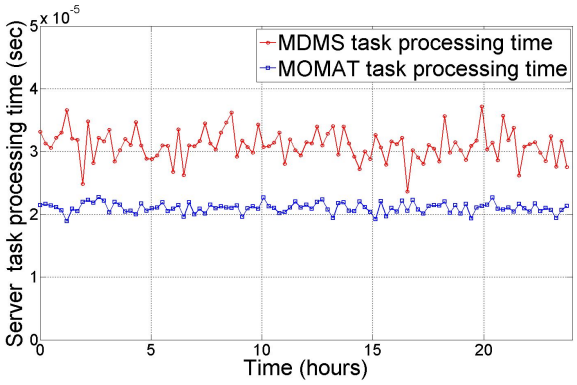

그림 8. MDMS와 MOMAT의 성능분석
Fig. 8. Performance analysis of MDMS and MOMAT.

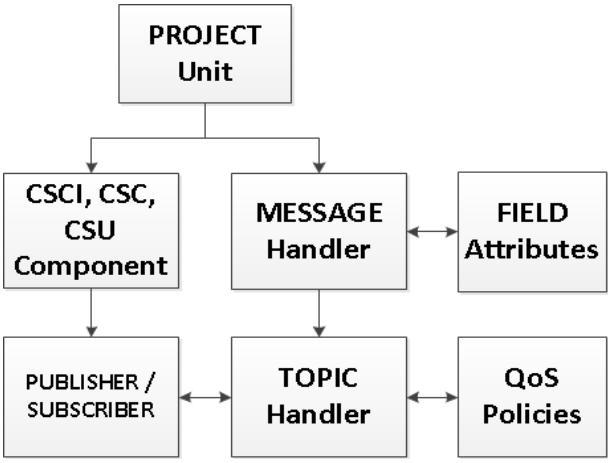

그림 9. 제안하는 MOMAT 데이터베이스 구조
Fig. 9. Proposed database structure of MOMAT.

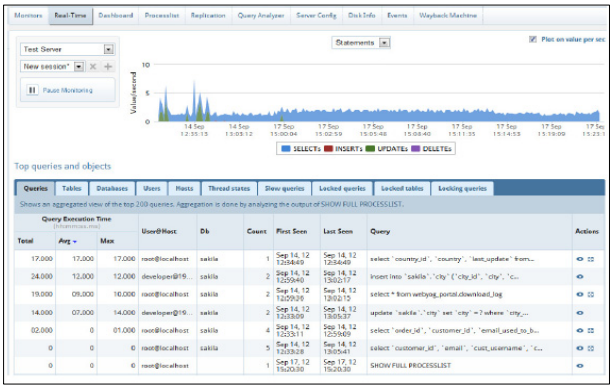

그림 10. MONyog 테스트 소프트웨어
Fig. 10. MONyog test software.

어지기 때문이다. 제안하는 데이터베이스 구조는 그림 9와 같다.

컴포넌트 테이블인 CSCI, CSC, CSU의 테이블을 하나로 묶음으로서 전체 테이블 접근 횟수를 줄였다. 이와 같은 테이블 형태는 코드 생성 기능 실행 시 발생하는 데이터 처리 부하를 줄일 수 있다. 제안하는 데이

터베이스 설계 방안의 가능성을 보여주기 위하여 OPNET modeler 16.0과 MONyog를 이용하여 성능 분석 하였다.

V. 모델링 도구의 성능 분석

1. 실험 환경

제안하는 데이터베이스 테이블 구조를 적용시킨 MOMAT의 성능 분석을 위하여 MONyog를 이용하였다. MONyog는 MYSQL 서버를 모니터링 할 수 있는 소프트웨어로서 다양한 분석 자료들을 실시간으로 제공해 준다. 그림 10은 MONyog 테스트 소프트웨어를 이용하여 MOMAT 클라이언트 노드로부터 서버로 전송되는 데이터양을 나타낸 것이다.

기존 MOMAT 데이터베이스 테이블 구조와 제안하는 테이블 구조의 차이점을 보여주기 위하여 클라이언트와 서버간 송수신되는 데이터양을 비교분석 하였다.

MOMAT은 중앙 집중형 서버 / 클라이언트 모델을 기반으로 개발되었으므로 그림 7의 네트워크 모델을 기반으로 실험이 이루어 졌다. 다양한 조건을 고려해 OPNET modeler 16.0을 이용하여 노드 수 증가에 따른 성능분석을 하였다. 함정 전투 시스템의 트래픽을 표현하기 위하여 포아송 분포를 이용하였고, 노드 수는 20부터 100 까지 순차적으로 증가 시켰다.

2. 성능 분석

그림 11 과 12 는 기존 MOMAT과 제안하는 방법이 적용된 MOMAT으로부터 서버로 전송되는 데이터양에 대한 실험 결과를 보여주고 있다. 각각의 항목에서 제안하는 데이터베이스 설계 방안을 이용한 MOMAT이 기존 MOMAT 보다 현저히 낮은 데이터 전송량을 보여주고 있다. 특히, Byte_received 와 Byte_sent

Global status variables

Name	Value (Jun 13 2013 02:39:09)	Value (03:39:09)	Change
Bytes_received	2257091	2862062	+604971
Bytes_sent	114730483	146188583	+31458100
Com_select	14724	18770	+4046
Com_set_option	7366	9390	+2024
Com_show_databases	7361	9384	+2023
Com_show_master_status	7361	9384	+2023
Com_show_processlist	71	72	+1
Com_show_status	7378	9401	+2023
Com_show_variables	7364	9387	+2023
Connections	931	932	+1

그림 11. MOMAT 노드의 데이터 전송량
Fig. 11. Amount of data from MOMAT node.

Global status variables

Name	Value (Jun 13 2013 05:35:19)	Value (06:35:19)	Change
Bytes_received	204956	883268	+678312
Bytes_sent	10706325	46043447	+35337122
Com_select	1380	5931	+4551
Com_set_option	691	2969	+2278
Com_show_databases	689	2964	+2275
Com_show_master_status	689	2964	+2275
Com_show_processlist	0	2	+2
Com_show_status	690	2965	+2275
Com_show_variables	690	2965	+2275
Connections	3	6	+3

Page 1 of 4 View 1 - 10 of 34

그림 12. 제안된 테이블 구조를 적용한 MOMAT 노드의 데이터 전송량
Fig. 12. Amount of data from MOMAT node that applied by proposed table structure.

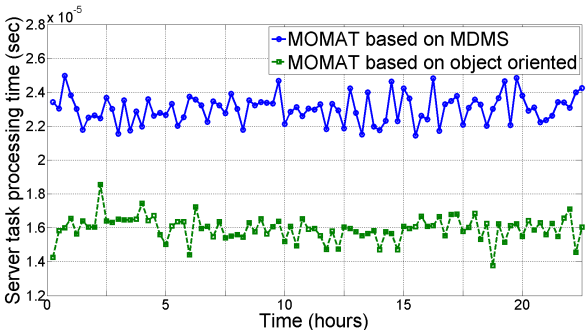

그림 13. 기존 MDMS기반의 MOMAT과 객체지향 기반의 MOMAT간의 서버처리시간 성능분석
Fig. 13. Performance analysis of server task processing time between MOMAT based on MDMS and MOMAT based on object-oriented.

항목에서 제안한 설계 방안이 기존 방법 보다 1/10 정도 낮은 전송량을 보여주고 있다. 이와 같이 전송량이 낮으면 서버 및 어플리케이션에 부하 부담이 줄어듦으로서 보다 나은 성능을 보여 줄 수 있다.

OPNET moderler 16.0을 이용한 시뮬레이션 결과는 그림 13에서 보여주고 있다. MDMS기반으로 구현된 MOMAT의 서버 태스크 처리 시간은 평균 2.3 μs (마이크로초), 최대 2.5 μs, 최소 2.18 μs 를 나타내고 있다. 메시지 지향 데이터베이스 설계 방안을 기반으로 구현된 MOMAT의 서버 태스크 처리 시간은 평균 1.6 μs, 최대 1.9 μs, 최소 1.4 μs 를 보여주고 있다. 시뮬레이션 결과는 제안하는 MOMAT 데이터베이스 설계 방안이 실현가능함과 동시에 기존 설계와 비교하여 보다 안정적이고 효율적인 성능을 보여준다.

그림 14는 MDMS 기반의 MOMAT 설계 방안과 객체지향 기반의 MOMAT 설계 방안간의 함정 전투 시스템의 노드 수 증가에 따른 평균 태스크 처리시간을 보여주고 있다. 두 가지 방법 모두 노드가 증가함으로

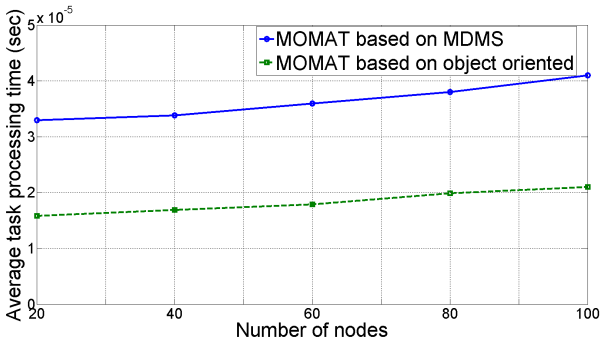

그림 14. 함정 전투 시스템의 노드 증가에 따른 평균 처리시간

Fig. 14. Average processing time to the number of nodes increases in NCS.

서 처리시간이 선형적으로 증가한다. 그럼에도 불구하고, 객체지향 기반 설계 방안으로 구현된 MOMAT은 기존 MDMS 기반 MOMAT 보다 높은 성능을 보여주고 있다.

다각도로 변화하는 전장의 실시간 상황 데이터를 처리해야 하는 함정 전투 시스템은 동시 다발적으로 메시지와 데이터를 처리해야한다. 본 논문에서 제안하는 함정 전투 시스템의 데이터베이스 설계 방안을 적용함과 미적용한 상황에서 데이터 송수신량과 태스크 처리 시간이 더 큰 차이를 보임을 분석을 통해 알 수 있다.

VI. 결론 및 향후연구

본 논문에서는 함정 전투 시스템을 위한 메시지 지향 모델링 도구의 데이터베이스 설계 방안 및 성능향상에 관하여 연구하였다. 기존 MDMS 기반의 MOMAT이 설계된 방법은 데이터베이스 테이블의 접근 횟수가 높아 이에 따른 서버 부하 및 MOMAT 어플리케이션 성능 저하가 초래 되었다. 제안된 실시간 데이터베이스 설계 방안을 이용하여 클라이언트의 서버의 접근 및 엑세스 횟수를 줄임으로서 전송되는 데이터양과 서버부하를 감소시켰다. 제안하는 데이터베이스 설계 방안이 사용가능함을 보여주기 위하여 MONyog과 OPNET modeler 16.0을 이용하여 성능분석을 하였다.

모의실험을 통해 제안된 함정 전투 시스템을 위한 모델링 도구의 데이터베이스 설계 방안은 서버처리시간 측면에서 MOMAT의 성능이 향상됨을 볼 수 있었다. 더불어, 함정 전투 시스템의 메시지들과 데이터들을 보다 효율적으로 관리 할 수 있도록 도움을 주었다.

향후 연구로는 STC DDS와의 호환성 및 그에 따른 성능분석에 대하여 연구할 것이다. 또한, 함정 전투 시스템의 트래픽 부하 감소를 위한 적합한 네트워크 모델에 관하여 연구할 것이다. 각 컴포넌트의 제공자 / 사용자 사이의 통신 연결 여부를 보여주는 기능이 필요하다.

REFERENCES

[1] D. S. Kim, Y. S. Lee, W. H. Kwon, and H. S. Park, “Maximum Allowable Delay Bounds of Networked Control Systems” Elsevier, Control Engineering Practice, vol. 11, nol. 11, pp. 1301 - 1313, Nov. 2003.

[2] D. S. Kim, and S. K. Huh, “Distributed Control Networks of Naval Combat Systems” The Magazine of KIICE, vol. 13, no. 2, pp. 47 - 53, Dec, 2012.

[3] S. J. Ko “Network Centric Warfare to Prepare for Combat Systems Development and Future Direction” Journal of IEIE, vol. 37, no. 11, pp. 1123 - 1134, Nov, 2010.

[4] J. Y. Ryu, and J. H. Park, “A Message Management System for Cooperative Message-based Interface Development” Journal of KISS Computing Practices and Letters, vol. 14, no. 6, pp. 609 - 613, Aug, 2008.

[5] W. Kang, K. Kapitanova, and S. H. Son, “RDDS: A Real-Time Data Distribution Service for Cyber-Physical Systems” IEEE Transactions on Industrial Informatics, vol. 8, no. 2, pp. 393 - 405, May. 2012.

[6] J. Y. Ryu, “Development of Message Oriented Software Modeling and Analysis Tool” Conference of KISS, vol. 38, no. 2, pp. 42 - 43, Nov, 2010.

[7] D. C. Schmidt, and F. Kuhns, “An overview of the Real-Time CORBA specification” IEEE Computer Society, Computer, vol. 33, no. 6, pp. 56 - 63, June. 2000.

[8] W. R. Otte, A. Gokhale, D. C. Schmidt, and J. Willenmsen, “Infrastructure for Component Based DDS Application Development” Proceedings of the 10th ACM International Conference on Generative Programming and Component Engineering, pp. 53 - 62, 2011.

[9] “Data Distribution Service for real-time systems version 1.2” Object Management Group, 1.2 formal / 07 - 01 - 01 edition, Jan. 2007.

[10] G. P. Castellote, “OMG Data-Disribution Service : Architectural Overview” IEEE Proceedings of the 23rd International conference on Distributed

Computing Systems Workshops, pp. 200 - 206, May. 2003.
[11] OPNET modeler 16.0, 2013. http://www.opnet.com/
[12] MONyog v5.6.0-4, 2013. https://www.webyog.com/product/monyog

저 자 소 개

송 경 섭(학생회원)
2012년 금오공과대학교 전자공학 학사 졸업.
2014년 동대학원 IT융복합공학과 석사 졸업.
2014년 3월~현재 삼성탈레스 기반체계그룹 연구원.
<주관심분야 : 네트워크 기반 임베디드 시스템, 함정 제어 통신망, 실시간 전송 기법>

김 동 성(정회원)-교신저자
전자공학회논문지-CI 제 47권 제 6호 참조

최 윤 석(정회원)
1999년 경북대학교 컴퓨터과학과 학사 졸업.
2002년 동대학원 컴퓨터과학과 석사 졸업.
2004년 동대학원 컴퓨터과학과 박사 수료.
2004년 3월~현재 삼성탈레스 기반체계그룹 전문연구원.
<주관심분야 : 미들웨어, 데이터 통신>