JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 29 No. 7, pp. 191-198, July 2024
https://doi.org/10.9708/jksci.2024.29.07.191

A Study on Efficient User Management System of Combat System

Hee-Soo Kim*

*Engineer, Naval R&D Center, Hanwha Systems, Gumi, Korea

[Abstract]

In this paper, we proposes a user management system for efficient operation of the combat system within naval ship. Recently, naval ships have seen performance enhancements through various sensors, features, and continuous system development. This progress in the system has led to an increase in multi-funstion consoles that can manipulate various sensors and features within naval ship, consequently increasing the number of operators for these consoles. Therefore, a user management system that can control and manage multi-function consoles and operators in real-time is necessary for efficient management within naval ship. This paper suggests a user management system that can effectively manage the real-time situation of users accessing multi-function consoles. Additionally, a parallelization method using GPUs to reduce the CPU workload in operating various functions of the combat system is proposed. The proposed user management system has shown a performance improvement where the response time decreased by approximately 82% and the occupancy reduced by approximately 20% compared to the method using CPUs.

▸ **Key words**: Naval Combat Management System, User Management System, GPU Process, Real time

[요 약]

이 논문에서는 함정 내 전투체계 시스템을 효율적으로 운용하기 위한 사용자 관리 시스템을 제안한다. 최근 함정에는 다양한 센서, 기능 그리고 시스템의 지속적인 발전을 통해 성능이 강화되고 있다. 이러한 시스템의 발전은 함정 내 다양한 센서와 기능을 조작할 수 있는 다기능 콘솔의 증가로 이어지며, 이에 따라 다기능 콘솔의 운용자 수도 증가하고 있다. 따라서 함정 내 효율적인 관리를 위해 다기능 콘솔과 운용자를 실시간 통제 및 관리하는 사용자 관리 시스템이 요구된다. 본 논문에서는 다기능 콘솔에 접근하는 사용자의 실시간 상황에 대해 효율적으로 관리할 수 있는 사용자 관리 시스템을 제안한다. 또한, 전투체계 시스템의 다양한 기능을 운용하는 CPU의 부하를 줄일 수 있는 GPU를 이용한 병렬화 방법을 제안한다. 제안한 사용자 관리 시스템은 GPU를 활용한 결과 CPU를 활용한 결과에 비해 응답시간은 약 82%, 점유율은 약 20% 줄어드는 성능을 확인하였다.

▸ **주제어**: 함정 전투 관리 시스템, 사용자 관리 시스템, GPU 프로세스, 실시간성

• First Author: Hee-Soo Kim, Corresponding Author: Hee-Soo Kim
*Hee-Soo Kim (heesoo91124@hanwha.com), Naval R&D Center, Hanwha Systems
• Received: 2024. 04. 19, Revised: 2024. 06. 21, Accepted: 2024. 07. 10.

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

I. Introduction

함정 내 전투체계 시스템은 복잡하게 연결된 여러 센서와 다양한 기능을 통합한 시스템으로, 함정의 안전과 효과적인 작전 수행을 위한 핵심적인 임무를 수행한다. 이 시스템은 함정 주변을 끊임없이 모니터링하며 타겟을 식별하고, 식별된 정보를 바탕으로 위험도를 평가하여, 이에 맞는 적절한 지휘, 무장, 그리고 교전 작전을 수행한다. 전투체계 시스템의 효율적인 사용을 위해, 함정에는 다기능 콘솔(Multi Functional Console, MFC)이 설치된다[1]. 다기능 콘솔을 통해 운용자는 각종 센서의 운용, 함정의 현재 상태 확인을 비롯해 다양한 작업을 수행할 수 있으며, 이는 전투 상황에서의 신속한 의사 결정과 효과적인 대응을 가능하게 한다[1~2].

초기 함정 전투체계의 설계는 상대적으로 제한된 수의 센서와 기능에 초점을 맞췄으며, 이에 따라 필요한 다기능 콘솔의 수도 제한적이다. 초기의 전투체계는 간단한 작전 수행과 기본적인 해상 작전에 필요한 기능들을 중심으로 구성되었기 때문에, 사용자 관리 시스템에 대한 요구가 미미하다. 하지만, 시간이 지나면서 최근 전투 상황의 복잡성이 증가하고, 다양한 센서와 기능이 개발되어 함정에 통합되기 시작하면서 다기능 콘솔의 수가 증가하는 추세를 보인다. 이에 따라 다수의 콘솔을 효율적으로 관리하고 운용하기 위한 사용자 관리 시스템의 필요성이 대두되었다.

사용자 관리 시스템은 중요한 역할을 담당하며, 특히 실시간성과 안전성이 요구된다[15]. 최신 함정들은 다수의 무장과 센서를 동시에 운용, 통제하므로 다수의 사용자가 각자 운용 권한을 가지고 콘솔을 통한 통제를 실시하고 있다[15]. 사용자 관리 시스템의 실패가 발생한다면, 사용자가 권한을 가져야 하는 장비를 통제하지 못하게 되거나 본인의 장비 이외의 장비를 통제하게 되는 등의 오류를 일으키면 이는 작전 실패를 일으키는 원인이 될 수 있다. 따라서, 사용자 관리 시스템은 높은 신뢰성과 함께 실시간으로 응답할 수 있는 능력을 갖추어야 한다. 본 논문에서는 GPU를 이용하여 연산 병렬화를 통해 응답 속도가 30ms 이내 실시간 사용자 관리 시스템을 제안한다.

본 논문은 다음과 같은 구성으로 작성한다. 2장에서는 전투체계와 전투관리체계 시스템 기능에 대한 설명과 연산 병렬화 방법에 관하여 서술한다. 3장에서는 다수의 다기능 콘솔을 관리할 수 있는 새로운 사용자 관리 시스템에 대하여 제안한다. 또한, 다기능 콘솔의 CPU 점유율을 줄이기 위해 GPU를 활용한 방법에 관하여 서술한다. 4장에서는 사용자 관리 시스템 방식의 응답 속도 측정을 통하여 시스템의 성능을 검증한다. 5장에서는 본 논문에서 제안한 시스템 성능 분석 결과와 향후 연구 계획에 관하여 서술한다.

II. Preliminaries

1. Related works

1.1 Naval Combat Management System

함정전투체계는 함정의 두뇌와 같은 역할이다. 함정에 탑재된 다양한 센서들과 센서를 통제할 수 있는 지휘무장통제체계(Combat Fire Command System/CFCS)으로 이루어지며 실시간 전투에 필요한 정보들을 수집 및 분석하는 대규모 시스템이다[1][14]. Fig 1은 지휘무장통제체계의 내부 구성도이며 이는 상황에 따라 함정의 목적이나 특수성에 따라 구성도가 다를 수 있다.

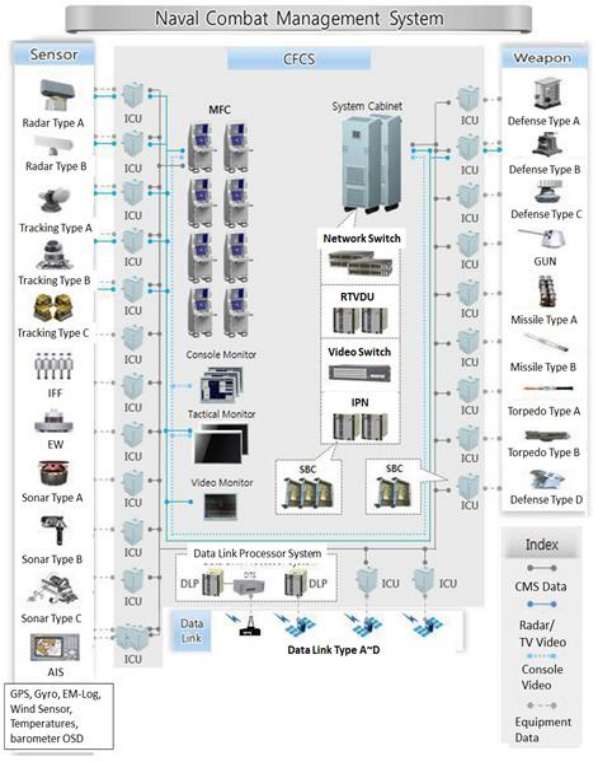

Fig. 1. Combat Fire Command System

지휘무장통제체계는 운용자가 전투체계의 조작, 센서의 상태, 전술 상황을 전시하는 다기능 콘솔과 센서 및 무장을 연동하기 위한 연동단(Interface Control Unit/ICU), 연동 체계 간 데이터를 전송하기 위한 통합 네트워크, 레이더 비디오, TV/IR 비디오를 분배와 전송을 위한 영상분배장치(Radar TV Video Distributor Unit/RTVDU), 데이터를 처리하는 단일 기판 컴퓨터(Single Board

Computer/SBC), 다기능 콘솔, 다양한 센서들로부터 획득한 정보를 처리하기 위한 장치(Information Processing Node/IPN) 등으로 구성되어있다. 함정 전투체계는 군에서 요구하는 생존성과 가용성을 만족하기 위해 다수의 SBC를 이용하여 정보처리장치와 연동단, 영상분배장치를 구성하였다. 이렇게 복수로 구성함으로써 적의 공격으로 일부 장비가 훼손되어 운용할 수 없어도 이중화를 통해 전투체계의 성능을 유지할 수 있도록 설계하였다.

1.2 DDS Communication

DDS(Data Distribution Service)는 실시간 시스템에서 데이터를 공유하기 위한 분산 통신 표준이다. DDS는 객체 지향적이며 이벤트 기반의 통신 방식을 제공하여 여러 시스템 및 장치 간의 데이터 교환을 지원한다. DDS는 고성능 및 안정성이 요구되는 분산 시스템에서 사용되며, 특히 실시간 시스템 및 임베디드 시스템에서 주로 사용된다.

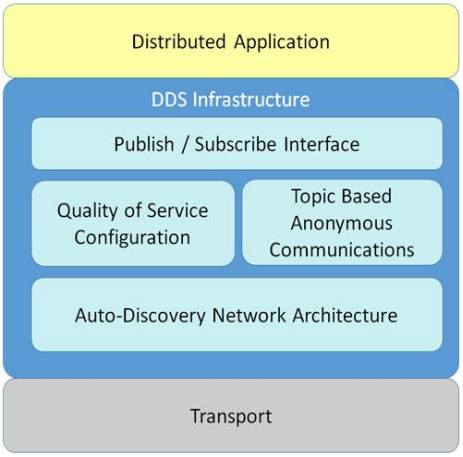

Fig. 2. DDS Infrastructure

DDS는 수신-송신 패턴을 기반으로 하며, 데이터 송신자(Publisher)와 데이터 수신자(Subscriber) 간의 통신을 관리한다. DDS는 분산 통신, 실시간성, 확장성, 유연성, 신뢰성의 특징을 가지기 때문에 함정의 전투체계 시스템에서 쓰인다. Fig 2는 다양한 유형의 응용 프로그램이 서로 통신할 수 있도록 하는 인프라 계층에 대한 구조이며 Fig 3은 DDS의 전반적인 Topic의 흐름에 대하여 보여준다. 함정 DDS가 필요한 내 시스템은 DDS DOMAIN에 연결되어있고 이를 통해 Topic을 송신 혹은 필요한 정보를 수신한다[2~4].

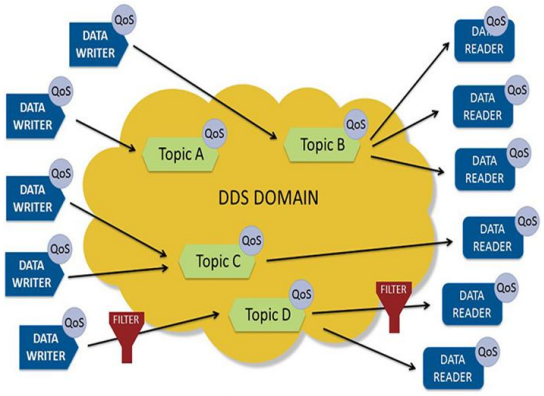

Fig. 3. DDS Topic System

1.3 User Authentication Process For Combat System

전투체계 시스템은 다양한 센서와 전투체계 기능을 사용하기 위해 로그인 시 사용자 인증 프로세스가 요구된다[15]. 이러한 사용자 인증 프로세스는 시스템이 효율적이며 정상적으로 예상대로 작동하는 능력을 의미하는 안정성, 시스템이 손상이나 공격받았을 때 최소 운용을 보장하는 능력을 의미하는 생존성이 요구된다. 안정성과 생존성이 요구되는 이유는 어떠한 상황에서도 프로세스의 오동작과 비가용이 되는 경우는 함정의 위험도와 직결되기 때문이다. 안정성과 생존성 향상을 위하여 인증 프로세스 시스템의 개선과 실시간성을 연구하였다[5]. 개선을 위하여 Fig 4과 같이 개선하였다. 먼저 체계접근통제 소프트웨어 전시 화면을 삭제하며 운용 환경 구성 시 운용자가 필수로 요구되는 대기시간을 줄였고 로그인/로그아웃 시 전시 소프트웨어를 실행/종료 대신 데이터 초기화와 미들웨어 초기화하였으며 이를 통해 함정 전투체계 시스템에 재접속 시간을 단축하여 안정성을 향상하였다. 또한, 함정 전투관리체계 운용 개념을 기본 전투관리체계, 교전 전투관리체계, 통합 전투관리체계로 분류하여 관리하며 교전 전투관리체계는 사용자 인증 전 신속하게 교전 작전을 수행할 수 있도록 하며 생존성을 향상하였다.

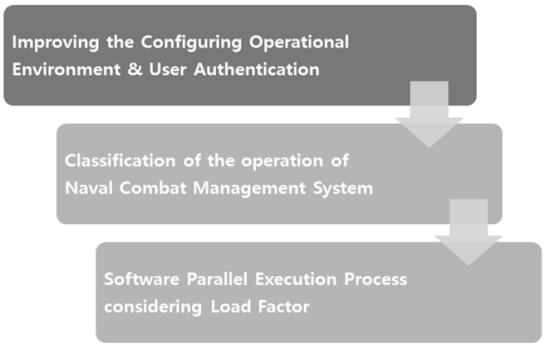

Fig. 4. Improvement Process

이를 통해 로그아웃 후 재접속 시간을 2분 28초에서 42초로 약 74.6% 감소하는 효과를 얻었으며 이는 전투관리체계에서 사용자의 신속한 접속을 도우며 원활한 작전 수행에 도움이 된다.

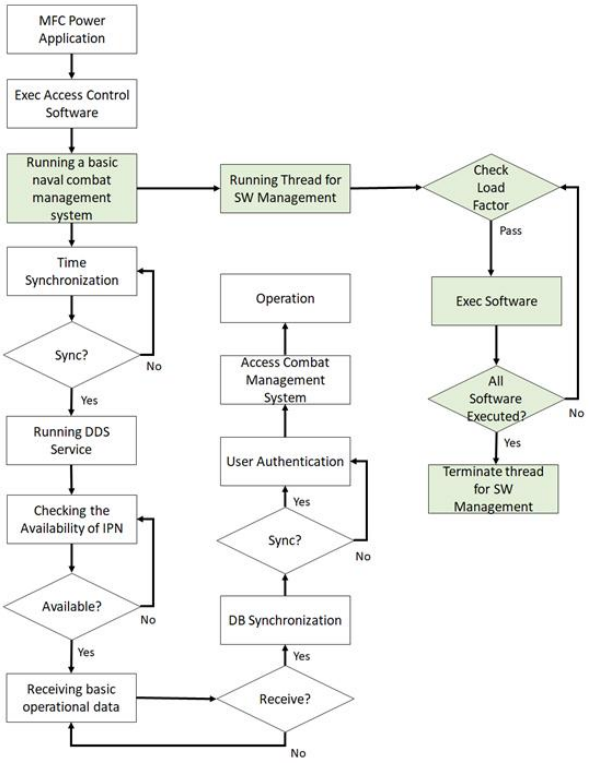

Fig. 5. Flow Chart

1.4 Parallel Processing

전투체계 시스템에는 실시간성이 요구된다. 기존 연구되어있는 시스템의 실시간성을 확보하는 방법은 CPU 혹은 GPU를 활용한 병렬화 방법이 있다[8~11][13]. CPU 병렬화는 ARM 아키텍처에서 제공하는 네온(NEON)(Fig 6) 방식이 있으며 GPU 병렬화에는 CUDA(Compute Unified Device Architecture)를 이용하는 방식이 있다.

네온(NEON)은 ARM 아키텍처에서 제공하는 벡터화(SIMD - Single Instruction, Multiple Data) 기술이다[6][7]. NEON은 하나의 명령어로 여러 개의 데이터를 동시에 처리할 수 있도록 설계되어 있어, 병렬 처리를 효율적으로 수행할 수 있다. NEON은 주로 모바일 장치와 같은 저전력 장치에서 사용되며, 데이터 병렬 작업을 가속화하는 데 유용하다[8][13]. 단 GPU에 비해 코어의 개수가 다소 적은 단점이 있다.

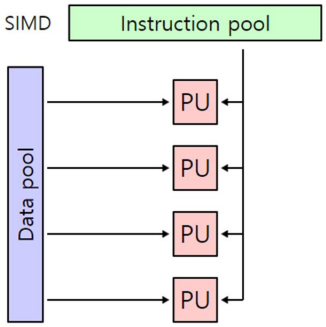

Fig. 6. SIMD Architecture

그에 비해 CUDA는 NVIDIA에서 개발한 병렬 컴퓨팅 플랫폼 및 프로그래밍 모델이다. 이를 통해 NVIDIA GPU를 사용하여 병렬 처리 작업을 수행할 수 있다. CUDA는 GPU의 병렬 처리 능력을 활용하여 고성능 컴퓨팅 작업을 수행하고, 주로 과학 및 공학 계산, 그래픽 처리 등에 사용되며 전투체계 시스템의 가속화를 위해 사용될 수 있다.[9~11]. 전투체계 시스템의 복잡한 연산은 CPU 혹은 GPU를 이용하여 병렬화를 통해 연산 시간을 단축하며 30ms 이내의 응답 속도로 실시간성[10]을 확보할 수 있다.

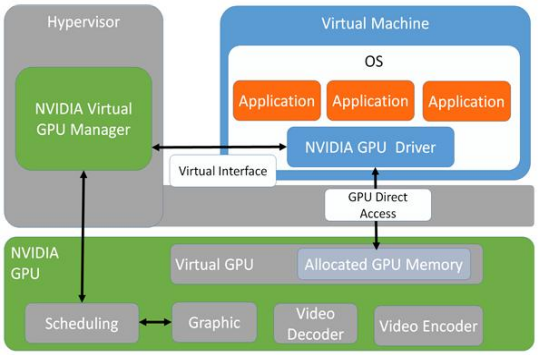

Fig. 7. Virtual GPU Architecture

III. The Proposed Scheme

기존 전투체계 시스템에는 로그인을 위한 사용자 인증 프로세스는 기존 연구가 있지만[5] 로그인 후 다기능 콘솔의 운용자를 위한 사용자 관리 프로세스가 존재하지 않는다. 따라서 본 논문에서는 기존 전투체계 시스템에서 필요한 여러 기능과 다수의 다기능 콘솔을 위한 사용자 관리 시스템을 제안한다. 또한 다양한 전투체계 시스템을 운용하는 CPU의 점유율을 분산하기 위해 GPU를 활용한 병렬화 방법에 대하여 제안한다.

1. User Management System

Fig 8은 콘솔에서 사용자가 로그인 시 사용자 관리 시스템의 전체 흐름도이다. 먼저 각 콘솔에 전원이 켜진 후 계정을 통하여 로그인한다. 그 후 로그인 성공 시 로그인 정보(계정명, 역할, 서비스 정보 등)를 사용자 관리 시스템으로 전달한다. 사용자 관리 시스템에서 전달받은 로그인 정보를 통해 함정 내 설치된 여러 콘솔에서 동일한 계정으로 로그인하여 사용 중인지 확인한다. 일반적으로 로그인 시스템은 동일 계정 접속을 허용하지 않지만, 전투체계 시스템의 특수한 경우로 인해 상황에 따라 동일 계정 로그인을 허용하거나 혹은 허용하지 않거나 사용자의 요구사항에 맞게 선택할 수 있도록 구성한다. 타 콘솔에서 동일한 계정으로 사용 중인 경우 기존에 사용 중인 콘솔로 동일 계정 접속 경고를 알려준다. 또한 로그인 한 콘솔과 같은 계정으로 동시 접속 허가할지 확인한다. 동시 접속을 허가하지 않는 경우 새롭게 로그인한 콘솔은 강제로 로그아웃한다. 동시 접속을 허가하는 경우 서비스를 동시에 사용할지 여부를 확인하며 동시 사용 여부에 따라 모든 서비스는 활성화/비활성화한다. 비활성화된 서비스의 경우 모든 조작은 불가능하며 단순히 전시만 한다.

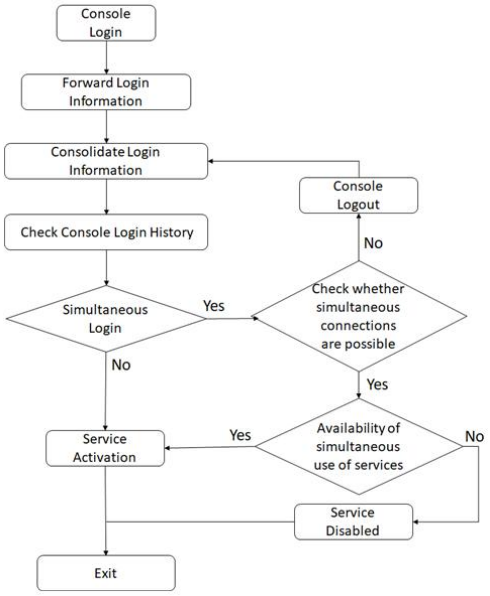

Fig. 8. User Management Method Flow Chart

Fig 9의 콘솔의 전원을 끄거나 사용자가 로그아웃 시 사용자 관리 시스템의 흐름도이다. 먼저 로그아웃이 수행되면 해당 정보를 사용자 관리 시스템에 전달된다. 사용자 관리 시스템에서는 로그아웃 한 사용자가 사용한 최근 서비스에 관하여 확인한다. 최근 사용한 서비스가 있는 경우 사용 정보를 사용자 관리 시스템에 동기화한다. 동기화하는 이유는 추후 각종 서비스의 히스토리를 남기기 위함이다. 해당 정보를 시스템에 동기화 후 로그아웃 한 콘솔의 로그인 정보를 사용자 관리 시스템에서 삭제한다. 이러한 흐름도를 통해 여러 대의 다기능 콘솔의 사용자 및 과거 사용한 서비스에 대한 히스토리를 관리한다.

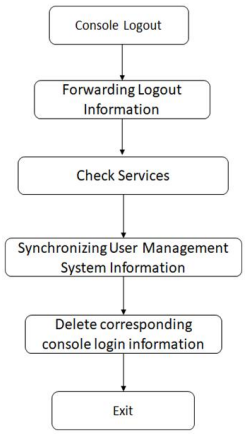

Fig. 9. Logout Flow Chart

2. GPU Parallel Computing Process

전투체계 시스템은 어떤 상황에서도 모든 응용 프로그램은 실시간 동작해야 하며 본 논문에서 제안하는 사용자 관리 시스템 역시 어떠한 상황에서도 실시간성이 보장되어야 한다. 이를 위해 대규모 시스템에는 한정적인 컴퓨터 자원에 대한 관리가 필요하다. 특히 CPU나 메모리의 부하율이 높아지는 경우 시스템의 지연이 발생하거나 최악의 경우 전투체계 시스템이 모두 멈추게 된다. 이러한 경우를 대비하기 위하여 다기능 콘솔의 전원이 켜지는 순간 항상 백그라운드에서 실행되는 기능은 상황에 따라 CPU 혹은 GPU에서 동작하도록 개발할 필요가 있다. 이에 본 논문에서는 사용자 관리 시스템의 GPU를 활용한 병렬화 방법에 대하여 제안한다.

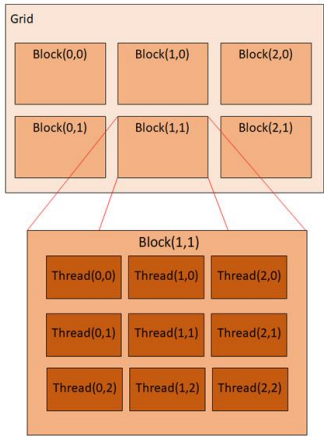

Fig. 10. GPU Architecture

NVIDIA GPU를 이용한 CUDA는 Fig 10과 같이 구성되어있다. 하나의 GPU는 하나의 Grid를 가지며 하나의 Grid에는 다수의 Block을 가지고 있으며 Block 내부에는 여러 개의 Thread로 구성된다[12].

병렬화를 위하여 NVIDIA의 GTX 1080, GTX 1060, GTX 960 GPU를 이용하였다. 이는 1024개의 Block을 사용할 수 있으며 하나의 Block에는 1024개의 Thread를 이용할 수 있으며 Fig 11과 같이 다기능 콘솔 하나당 하나의 Block에 할당하여 콘솔이 접속 시 해당 Block에서 연산을 수행하도록 설계하였다. 제안하는 가속화 시스템을 이용하면 여러 대의 다기능 콘솔을 동시에 운용하며 생기는 병목현상을 해결할 수 있다.

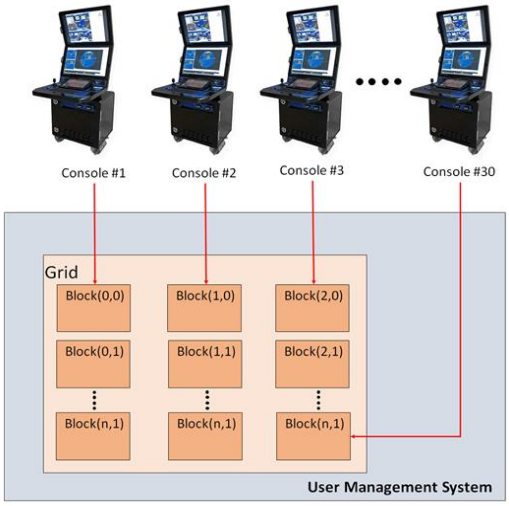

Fig. 11. Console and Block Allocation Method

또한, Fig 12와 Fig 13과 같이 사용자 관리 시스템의 콘솔 정보 통합을 위한 프로세스를 각 콘솔에서 할당된 Block과 Block의 내부 Thread를 이용하여 연산 최적화를 수행하였다.

IV. Test and Evaluation

Table 1. PC Performance

PC	OS	CPU	RAM	GPU
#1	Linux 20.04	Intel i5-11400 2.6Ghz	16GB	NVIDIA GTX 1080
#2	Linux 20.04	Intel i5-9400 2.9Ghz	8GB	NVIDIA GTX 1060
#3	Linux 20.04	Intel i5-7500 3.4Ghz	8GB	NVIDIA GTX 960

본 논문의 사용자 관리 시스템의 성능을 검증하기 위하여 함정에서 사용하는 방식과 유사하게 시뮬레이션을 구현하였다. 객관적인 성능 평가를 위해 사용한 PC는 Table 1와 같다. 총 3대의 PC를 사용하였으며 CUDA를 이용하기 위해 GPU는 NVIDIA의 제품을 사용하였다.

시뮬레이션을 통해 가상으로 생성한 다기능 콘솔을 사용자 관리 시스템으로 연결하여 로그인 정보를 주기적으로 보내고 돌아오는 응답 시간을 측정하였다. Block 내 Thread는 배리어 동기화를 사용하여 작업을 조율하였으며 임계값 5ms로 사용하여 Thread의 동기화를 위한 최대 대기시간을 제한하였다. 대기시간이 임계값 5ms을 넘어가는 경우 모든 Thread를 동기화하며 Thread가 무한대기에 빠지는 경우를 사전 차단하였다. 임계값은 다양한 테스트를 통해 Thread의 평균 소요 시간을 분석하여 지정하였다. 그 후 다양한 함정 전투체계에 유동적으로 대응하기 위해 사용자로부터 함정의 다기능 콘솔의 대수, GPU 종류, 각 서비스에 할당하는 Thread 수를 입력받는다. 이를 활용하여 Block과 Thread를 유동적으로 할당하며 다수의 함정에 적용할 수 있도록 사용자 관리 시스템을 구성하였다.

Table 2는 총 13대의 다기능 콘솔이 동시에 로그인하는 극단적인 상황에서 사용자 관리 시스템에서 CPU를 이용한 방법과 GPU의 1개 Block을 이용한 방법에 대한 지연 시간 결과이다.

Table 2. CPU and GPU Run Time

PC	Use CPU	Use GPU Thread
#1	3.2ms	1.2ms
#2	3.9ms	1.9ms
#3	7.1ms	4.3ms

GPU의 Block 내부 1개 Thread를 이용하는 방법은 CPU를 이용하는 방법에 비해 평균 약 51%의 지연시간을 단축할 수 있었다. 이 실험을 통해 GPU가 장착된 다기능 콘솔에서는 CPU의 점유율 관리와 지연 시간 단축을 위해 가능한 기능에 대하여 GPU의 적극적인 활용이 필요함을 확인하였다. Fig 12, Fig 13은 GPU의 Block 내부에 있는 Thread를 이용하는 방법에 대한 그림이다. Fig 12은 서비스와 Thread를 1:1로 할당하여 병렬화하는 방법이며 Fig 13는 서비스와 Thread를 1:4로 할당하여 병렬화하는 방법이다. 할당하는 방법은 상황에 따라 유동적으로 가능하다. 사용하는 시스템과 환경에 테스트를 통해 사용할 필요가 있다.

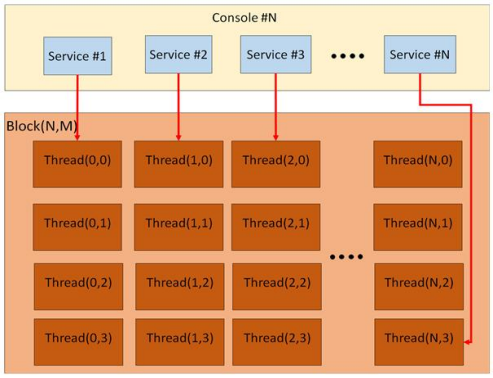

Fig. 12. Matching Method of Service #1

Table 3의 Method #1은 Fig 12의 매칭 방법을 이용한 결과이며 Method #2는 Fig 13의 할당 방법을 이용한 실험 결과이다. 검증 결과 두 방법 모두 Table 2에 비해 응답시간이 단축된 것을 볼 수 있다. 하지만 서비스와 Thread를 1:4로 할당하는 방법이 연산 가속화에 조금 더 효율적이다. 1:1로 할당하는 경우 일부 Thread에 병목 현상이 발생하며 연산 시간에 지연이 생기며 시스템 응답에 지연이 발생하였다.

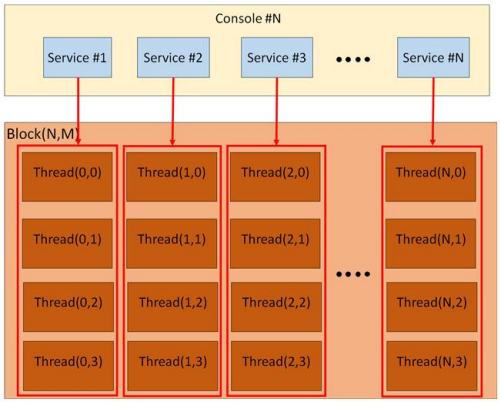

Fig. 13. Matching Method of Service #2

Table 3. Run Time of Methods

PC	Method #1	Method #2
#1	0.7ms	0.4ms
#2	1.2ms	0.8ms
#3	3.1ms	2.1ms

Fig 14는 하나의 서비스에 대하여 할당된 Thread의 개수를 변경하며 제안한 알고리즘의 수행시간을 측정한 결과이다. PC #1의 기준으로 1개의 Thread를 할당한 경우 약 0.7ms, 4개의 경우 약 0.4ms, 15개의 경우 약 0.38ms를 나타낸다. Thread의 개수와 수행시간이 비례하지 않는 것을 검증하였으며 본 논문에서 제안한 사용자 관리 시스템에서는 하나의 서비스에 4개의 Thread를 할당하는 것이 가장 효율적인 방법임을 확인하였다. Thread를 불필요하게 많이 할당한 경우 일부 Thread가 연산을 수행하지 않고 대기를 하는 경우가 발생하였다. 따라서, Thread의 수와 연산 시간이 비례하지 않는다. 많은 Thread를 할당하여도 연산 시간을 단축할 수 있는 부분이 없기에 많은 수의 Thread 할당은 비효율적임을 확인하였다.

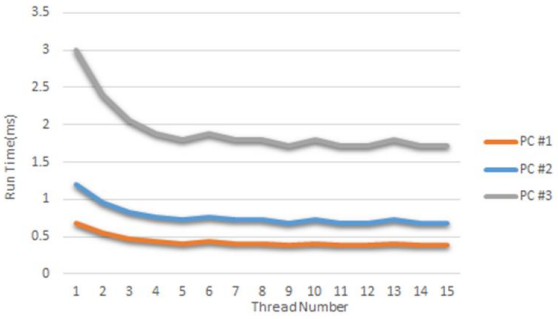

Fig. 14. Run Time according to Thread

Table 4는 CPU를 이용한 사용자 관리 시스템의 응답시간과 Fig 13과 같이 하나의 서비스에 4개의 Thread를 할당하는 방법의 응답시간을 최종적으로 비교한 결과이다. 그 결과 CPU를 이용한 Original Method는 PC별로 각각 약 3.2ms, 3.9ms, 7.1ms가 소요되며 GPU를 이용한 Propose Method는 약 0.4ms, 0.7ms, 1.8ms로 CPU를 이용한 방법에 비해 평균 5.6배 빠른 속도를 보였다. GPU의 성능에 따라 차이는 있을 수 있지만 병렬화를 수행한 경우 CPU를 이용한 방법에 비해 지연 시간의 단축을 확인하였다.

Table 4. System Environment

PC	Original Method	Propose Method
#1	3.2ms	0.4ms
#2	3.9ms	0.7ms
#3	7.1ms	1.8ms

Table 5는 사용자 관리 시스템의 CPU를 이용한 방법과 GPU를 통해 병렬화 한 경우 CPU의 점유율을 분석한 결과이다. 가상의 다기능 콘솔 13대를 사용자 관리 시스템으로 동시 접속하였을 때 CPU를 이용한 경우 평균 약 34%의 점유율을 보였다. 그에 비해 GPU를 이용한 방법 중 서비스 Thread를 1:1 매칭 한 경우 평균 약 15%의 점유율을 보이며 1:4 매칭 한 경우 역시 평균 약 16%의 점유율을 보였다. 이를 통해 Thread의 할당 방법과 CPU의 점유율은 무관한 것을 확인하였다. 또한, GPU를 이용하며 실시간성을 추가로 확보할 수 있었고 CPU의 부하를 줄이는

것에 도움이 되는 것을 확인하였다.

Table 5. CPU Utilization

Original Method(CPU)	Propose Method(1:1)	Propose Method(1:4)
34%	15%	16%

V. Conclusions

최근 함정의 전투체계에는 많은 센서와 다양한 기능이 추가되고 있다. 이에 따라 여러 대의 다기능 콘솔이 설치되고 사용자 역시 증가하며 늘어나는 콘솔의 사용자를 관리할 수 있는 시스템이 필요하다. 이에 본 논문에서는 전투체계의 사용자 관리 시스템에 대하여 제안하였다. 또한, GPU를 활용하여 다기능 콘솔의 성능을 극대로 사용할 수 있는 병렬화 방법에 대하여 제안하였다.

본 논문에서 제안한 방법을 활용하는 경우 함정 내 모든 콘솔의 상태에 대하여 실시간 관리할 수 있으며 콘솔 간의 현재 통제 상황에 대하여 실시간 공유가 가능하다. 이러한 유기적인 사용자 관리 시스템은 전투체계 시스템을 더 효율적이며 직관적으로 운영하는 데 큰 도움이 될 것이다. 제안한 방식을 검증하기 위해 가능한 다양한 상황에 대하여 가상의 다기능 콘솔을 생성하여 시스템의 성능을 테스트하였다. 향후 다양한 함정에 사용자 관리 시스템을 적용하여 실제 발생할 수 있는 다양한 상황에 대하여 추가적인 실험 및 시스템 보완을 통하여 좀 더 효율적이며 안정적인 시스템 구축에 관한 연구를 진행할 계획이다. 또한, GPU가 없는 상황을 대비하여 CPU를 통한 알고리즘 최적화 방안에 관한 연구를 진행할 계획이다.

REFERENCES

[1] S.M. Kwon, S.M. Jung, "Virtualization based high efficiency naval combat management system design and performance analysis", Journal of the Korea Society of Computer and Information Vol. 23, No. 11, pp. 9-15, Nov 2018. DOI: https://doi.org/10.9708/jksci.2018.23.11.009

[2] DDS, https://www.omg.org/omg-dds-portal/

[3] P. C. Gerardo, B. Farabaugh, and R. Warren. "An introduction to DDS and data-centric communications." Real-Time Innovations 2005.

[4] A. Alaa, and D.K. Kim. "Tailoring DDS to smart grids for improved communication and control." 2016 5th International Conference on Smart Cities and Green ICT Systems, IEEE, 2016. pp. 1-6.

[5] J.H. Im. "A Study on the Design of System Access Control Software For the Improvement of the Stability and Survivability of Naval Combat Management System", Journal of The Korea Society of Computer and Information Vol. 28 No. 12, pp. 137-145, December 2023. https://doi.org/10.9708/jksci.2023.28.12.137

[6] W. Wenjie, et al. "SMCOS: Fast and parallel modular multiplication on ARM NEON architecture for ECC." Information Security and Cryptology. pp. 531-550, August 2021

[7] H.J. Seo et al. "Efficient arithmetic on ARM-NEON and its application for high-speed RSA implementation." Security and Communication Networks, pp. 5401-5411, 2016

[8] H.J. Seo et al. "Montgomery modular multiplication on ARM-NEON revisited." Information Security and Cryptology-ICISC, pp. 328-342, December 2014

[9] K.S. Song, “A Study of Feasibility and Performance Analysis of VDI based on GPU Acceleration for Naval Combat System”, Journal of The Electronics and Information Engineers, Vol. 58 No. 11, pp. 1078-1085 https://doi.org/10.5573/ieie.2021.58.11.86

[10] Garland, Michael, et al. "Parallel computing experiences with CUDA." IEEE micro, pp. 13-27. 2008

[11] Luebke, David. "CUDA: Scalable parallel programming for high-performance scientific computing." IEEE international symposium on biomedical imaging: from nano to macro, pp. 836-838, 2008

[12] Ghorpade, Jayshree, et al. "GPGPU processing in CUDA architecture." arXiv preprint arXiv:1202.4347, 2012

[13] https://ko.wikipedia.org/wiki/SIMD

[14] Arciszewski, Henryk FR, Tjerk E. De Greef, and Jan H. Van Delft. "Adaptive automation in a naval combat management system." IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, pp. 1188-1199, 2009

[15] Kerpez, Kenneth J., et al. "Software-defined access networks." IEEE Communications magazin, pp. 152-159, 2014

Authors

Hee-Soo Kim received the B.S., M.S degrees in Computer Engineering from Kyungpook National University, Korea, in 2015, 2017, respectively. He is interested in Combat System Software, interface control unit and data processing algorithm.