ACE 네트워크 프레임워크를 이용한 고효율성 게임서버※

박성준[+○], 추교성[*○], 박창훈[+○]
호서대학교 게임공학과[○]
SungE@maiet.net*, {sjpark+, chpark+}@hoseo.edu

High Efficient Game Server using ACE Network Framework

Sungjun Park[○], Kyo-Sung Choo[○], Chang-Hun Park[○]
Dept of Game Engineering, Ho-Seo University[○]

요 약

본 논문은 오랜 기간 다양한 분야에서 개발되어 온 공개 네트워크 라이브러리인 ACE를 사용하여 게임 서버에 접목 시켜 보았다. ACE 네트워크 라이브러리는 고성능 실시간 통신 라이브러리와 어플리케이션 개발에 집중되어 있고 방대한 기능을 제공하고 있다. 본 연구에서는 ACE의 여러 기능 중에 게임 개발에 필요한 부분을 논리적으로 재구성하여 최적화 하였고, 재구성한 라이브러리의 검증을 위해 실제 배틀넷 서버를 구현하였다. 실험방법으로는 배틀넷 서버와 테스트 클라이언트를 설정하여 접속 요청 테스트와 데이터 전송 테스트를 수행하였다. 실험 결과로서 검증된 네트워크 라이브러리인 ACE를 사용하여 온라인 게임 개발이 가능하다는 결과를 얻었다.

ABSTRACT

In this paper, we propose a game server using public network library ACE, which has been developed in various fields for a long time. ACE network library has been considered not only in the area of high efficient real-time communication library but also in the area of application development, and it provides various facilities. We logically reorganized the part, which is necessary to develop games, among various functions of ACE and optimized it, and developed real battlenet server using verify the reorganized library. As the method of experiment, the battlenet server and test client were set and interface request test and data electrical transmission test were conducted. As the result of the experiment, the conclusion that it is possible to develop games by using ACE, which is verified network library, has been obtained.

Keyword : ACE, Game Server, Battlenet, Proactor Pattern

접수일자 : 2008년 11월 11일
심사완료 : 2008년 12월 21일
※ 이 논문은 2008년 정부(교육과학기술부)의 재원으로 한국학술진흥재단의 지원을 받아 수행된 연구임(KRF-2008-331-D00557)

1. 서 론

현재 온라인 게임은 지속적으로 발전하고 있다. 국내 시장의 경우 PC게임 시장은 온라인 게임이 거의 대부분을 차지하고 있고, 국외에서도 마찬가지로 온라인 게임에 대해 많은 관심을 가지고 있다. 그 대표적인 예로는 블리자드사가 개발한 WOW(Word Of Warcraft)가 있다. WOW는 전 세계적으로 서비스를 하고 있으며 많은 유저들이 플레이 하는 게임이다. 우리나라 게임 역시 국내 서비스에서 해외 서비스로 지속적인 진출을 하고 있으며, 지금은 해외 서비스가 개발 시점부터 고려되고 있다. 이렇듯 게임의 트랜드가 온라인 중심으로 개발되고 있는 가운데, 네트워크의 성능은 온라인 게임의 성장에 있어서 매우 중요한 부분을 차지하게 되었다. 온라인 게임에 있어서 네트워크 기능에 대해 높은 성능 처리와 안정적인 관리가 병행되지 않으면 게임은 쉽게 다양한 문제가 발생하여 서비스 할 수 없는 상황에 이르게 된다. 즉, 네트워크와 관련된 게임 개발은 그 크기에 상관없이 아주 중요한 부분이다.

일반적으로 온라인 게임에서 네트워크 기능은 별도의 모듈로 가장 먼저 개발 하게 된다. 네트워크 모듈이 안정적으로 개발 된 후에 게임과 관련된 기본적인 기능부터 개발이 진행 되거나, 혹은 네트워크 모듈 개발 진행과 병행으로 게임 로직 개발이 진행 된다. 병행으로 진행 될 때는 네트워크와 연동되지 않은 제한적인 부분만 진행될 수 있기 때문에 개발이 제한적으로 될 수밖에 없다. 이렇듯 네트워크 모듈은 온라인 게임 개발의 시작이면서 전체 게임 개발에 있어서 기초가 되는 부분이다. 네트워크 모듈을 안정적으로 개발하기 위해서는 다음과 같은 세 가지 사항을 고려해야 한다. 첫째, 같은 기능에 대해서 반복적인 개발을 피한다. 둘째, 기능을 모듈화 하여 복잡한 로직은 관련된 범위로 최소화 하여 복잡도를 낮춘다. 셋째, 검증된 모듈은 재사용 한다. 위의 세 가지 사항을 모두 고려한다면 검증된 네트워크 라이브러리를 사용하는 것이 좋은 방법이 될 수 있다.

게임 개발에 있어서 다른 분야를 본다면 렌더링 엔진 같은 경우는 이미 라이브러리를 사용하여 개발하는 것이 일반화 되어 가고 있으며, 이것을 잘 이용하여 퀄리티가 높은 게임을 개발하고 있다. 물리 엔진 분야 역시 라이브러리를 사용하는 것이 일반화가 되어 가고 있다. 물리 엔진 같은 경우는 렌더링 엔진에 포함 되는 경우도 있다. 이렇게 게임 개발에 있어서 다른 분야들은 라이브러리를 사용 하는 것이 일반화가 되어 있지만 네트워크 분야는 아직 라이브러리를 직접 사용하는 경우는 미비하며 참조 모델로 사용하는 경우가 일반적이다.

본 논문에서는 게임 개발의 네트워크 분야에 있어서 검증된 라이브러리를 사용하여 위에서 제시한 3가지 고려사항을 충족하면서 윈도우에서의 게임 개발에 사용할 수 있는 네트워크 라이브러리를 개발하였다. 본 논문에서 사용한 검증된 네트워크 라이브러리는 ACE (Adaptive Communication Environment) 라이브러리이다. ACE는 여러 분야에서 오랜 기간 동안 사용되어 왔고 방대한 기능과 프레임워크를 지원해 주는 범용적인 네트워크 라이브러리이다.

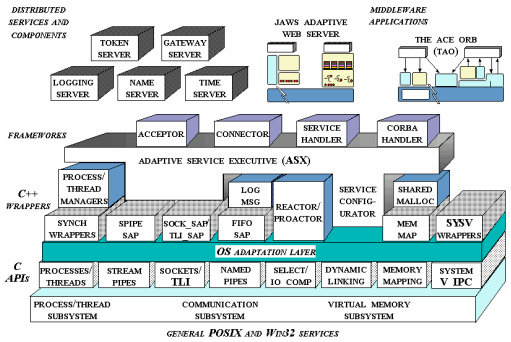
본 논문에서는 ACE 네트워크 라이브러리에 대해 2장에서 소개하고, 3장에서는 ACE라이브러리를 이용한 네트워크 모듈 개발과 배틀넷 서버 구현에 대해 다루었다. 4장에서는 성능 및 테스트를 다루고 마지막으로 5장에서 결론 및 향후 연구 방향을 제시한다.

2. 관련 연구

2.1 ACE Network Library

ACE (Adaptive Communication Environment)는 동시 처리 방식의 많은 핵심 패턴들을 구현한 객체 지향 프레임워크이다.[1,2] OOP를 바탕으로 한 라이브러리 이므로 객체 지향 프로그래밍에 익숙한 개발자라면 쉽게 익숙해 질 수 있고 사용할 수 있다. ACE는 고성능 실시간 통신 서비스와 어

플리케이션을 개발 하는 데에 집중화 되어 있다. 이것은 프로세스간 통신, 이벤트 다중 수신, 명시적 동적 링킹, 동시 처리 방식의 객체 지향 네트워크 어플리케이션과 서비스의 개발을 단순하게 만들어 준다.



[그림 1] ACE Framework

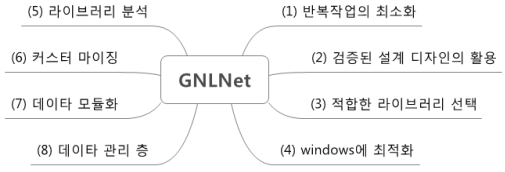
ACE의 장점으로서 4가지가 있다. 첫 번째로 이식성의 증가다. ACE 컴포넌트들은 하나의 운영체계상에서 동시처리방식의 네트워크 어플리케이션들을 작성하기 쉽게 해주며, 수많은 다른 운영체계로 그것을 포팅하는 작업을 손쉽게 해준다. 두 번째로 소프트웨어 품질의 증가이다. ACE 컴포넌트는 통신 소프트웨어의 핵심 품질요소 즉, 융통성, 확장성, 재사용성, 모듈화 등을 향상시키기 위해 많은 핵심 패턴을 사용해서 디자인되었다. 세 번째로 효율과 예측력의 증가이다. ACE는 넓은 분야의 어플리케이션 QoS 요구를 지원하기 위해 조심스럽게 디자인되었다.[3] 이런 요구에는 딜레이에 민감한 어플리케이션을 위한 낮은 지연(latency), 대역폭에 민감한 어플리케이션을 위한 높은 성능, 실시간 어플리케이션을 위한 예측 능력 등이 있다. 마지막으로 ACE는 The ACE ORB (TAO)에서 사용된 재사용가능한 컴포넌트와 패턴들을 제공한다. TAO는 실시간 시스템과 고성능에 최적화된 CORBA의 오픈소스 표준을 따르는 구현물이다.[4] 따라서 협력적인 미들웨어 솔루션을 제공할 목적으로 ACE와 TAO는 서로 궁합이 잘 맞도록 구현되어있다.

게임서버는 동시에 많은 네트워크 요청을 처리해야 한다. ACE는 이 부분을 IOCP를 사용함으로 동시 I/O요청을 효율적으로 처리해 준다. 또한 복잡한 IOCP관리를 프레임웍으로 지원해 주고 있기 때문에 복잡한 로직 구현에 대해 발생할 수 있는 문제를 해결해 줄 수 있다. 본 논문에서는 ACE의 이러한 장점를 기반으로 게임 서버에 적용하려고 한다. ACE를 이용한 게임 제작은 국외의 경우 ATRIARCH[14] 온라인게임이 있고, 국내에서는 게임서버에 적용된 사례가 공식적으로 없다. 그러나 현재 많은 게임 업체에서 관심을 가지고 적용하려는 시도를 보이고 있다.

3. 게임 서버 시스템

3.1 ACE를 이용한 네트워크 모듈

본 논문에서 개발한 ACE 기반의 게임 네트워크 라이브러리 (GNLNet : Game Network Library Net)는 게임 개발에 있어서 다양한 문제점이 발생될 수 있는 부분들에 대해 아래와 같은 8가지 부분에 대해 고려하여 설계하고 구현하였다.



[그림 2] GNLNet 설계 모델

3.1.1 반복 작업의 최소화

반복 작업은 코드의 품질을 저하시키는 원인 중 하나이다. 라이브러리를 사용함으로 반복 작업을 최소화 하여 반복 작업 중 발생할 수 있는 문제점을 해결함으로서 코드의 품질을 높이고, 불필요하게 소모되는 개발 리소스를 줄일 수 있다.

3.1.2 검증된 설계 디자인의 활용

설계 디자인은 아주 중요한 부분이고 소프트웨

어 개발의 기초 공사라 할 수 있다. 만약 설계 디자인이 잘 못 되면 개발하는 동안 지속적인 문제를 발생시킬 수 있다. 또한 한번 정해진 디자인이 개발 도중에 문제가 있다고 판단되면 그것을 수정하는데 들어가는 비용이 크기 때문에 다양한 분야에 걸쳐 오랜 기간 동안 검증된 설계 디자인을 사용하는 것이 무엇 보다 중요하다. 이러한 검증된 설계 디자인을 이용하여 특정 응용 어플리케이션에 적용하려고 할 때 가장 중요하게 고려해야 할 작업은 전체 구성을 분석하는 작업이다. 전체적인 설계 디자인의 분석 작업을 통해 처음부터 디자인하는 수고를 덜어 주고, 향후 특정 응용 어플리케이션에 맞도록 재구성 하는 데에 큰 도움이 된다.

3.1.3 적합한 라이브러리 선택

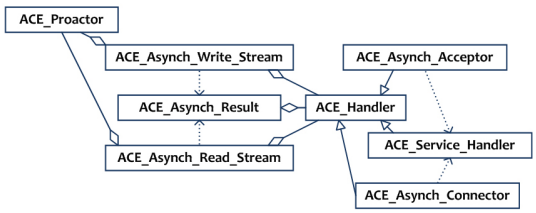
본 논문에서는 개발한 게임 네트워크 라이브러리 제작을 위해 ACE를 사용하였다. ACE 네트워크 라이브러리는 다양한 네트워크 분야에서 범용적으로 사용 되고 있는 검증된 라이브러리이다. ACE는 지원하는 기능 또한 방대하여 필요한 기능을 다시 구현할 필요가 거의 없다. 관련 정보는 웹과 서적, 포럼을 통해 얻을 수 있고, 지속적인 개발로 업데이트 되고 있다.

3.1.4 windows에 최적화

라이브러리를 사용하는 경우에 있어서 고려해야 할 사항 중의 하나는 라이브러리 자체가 범용적으로 사용되기 위해 만들어졌기 때문에 어떤 특정 영역의 응용 어플리케이션을 위해서는 그 영역에 맞도록 최적화 하는 단계가 필요하다는 점이다. 특히, 게임 관련해서는 게임 개발이 많은 윈도우 플랫폼에서 최적의 성능을 낼 수 있도록 최적화하는 작업이 필수적이다. 본 논문에서 가장 중요한 핵심 기술 중의 하나가 바로 윈도우즈 플랫폼에 최적화 하도록 재구성 하는 부분이라 할 수 있다.

3.1.5 라이브러리 분석

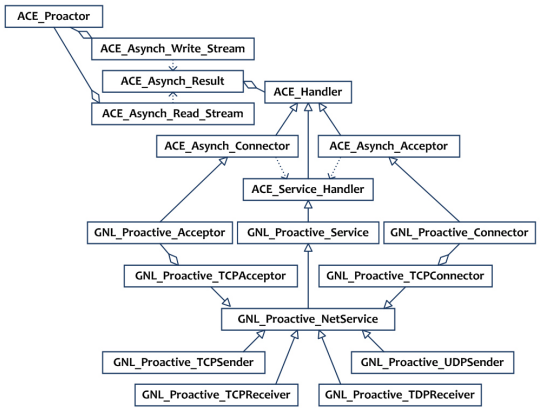
게임 서버는 게임 중 동시 접속자가 많기 때문에 많은 수의 동시 접속자 처리를 원활하게 처리할 수 있어야 한다. 해결 방업으로는 비동기 입출력(IOCP)을 사용할 수 있어야 하며 효율적으로 처리할 수 있는 논리적인 네트워크 구조가 있어야 한다. ACE 네트워크 라이브러리는 기본적으로 Proactor패턴[5,6,7](ACE_Proactor)을 프레임워크로 지원해 주고 있으며 추가적으로 필요한 기능들은 클래스로 직접 구현할 수 있도록 되어 있다. ACE 네트워크 라이브러리가 게임 서버에 적합한 아주 중요한 기능 중의 하나가 바로 Proactor 패턴의 지원이라고 할 수 있다.



[그림 3] ACE Proactor Framework

3.1.6 커스터 마이징

ACE는 게임 서버로서의 필요한 네트워크 관련 대부분의 기능을 프레임워크와 라이브러리에서 지원 하고 있기 때문에 게임 서버 구현 시 많은 부분의 작업을 줄일 수 있다. 클래스 모델에선 많은 클래스들이 추가 되었지만 대부분 ACE라이브러리를 재사용한 부분이고 실제 로직이 작성된 부분이 50% 미만이다. 커스터마이징 단계에서 ACE 라이브러리를 재사용하여 대부분의 복잡한 로직과 클래스 구현을 최소화 할 수 있다.



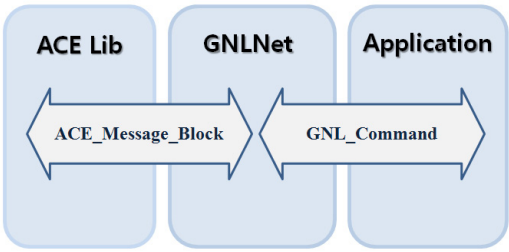
[그림 4] 커스터 마이징 한 클래스 모델

3.1.7 데이터 모듈화

네트워크로 전송될 수 있는 데이터는 바이트단위의 스트림 형태로 되어 있다. 이 구조는 관리가 어렵고 실수가 발생하기 쉽다. 또한 스트림 구조는 라이브러리에서 관리 할 수 없기 때문에 어플리케이션에서 잘못된 데이터를 넘겨주더라도 검증 할 수 있는 방법이 없다. 그래서 네트워크로 전송되는 데이터를 모듈화 하여 어플리케이션 단계에서 데이터 객체를 사용함으로서 데이터 객체에서 재공해주는 안전한 인터페이스를 통해서 데이터를 구성하게 한다. 또한 데이터 객체는 네트워크 라이브러리와 연동되는 복잡한 로직과 예외 처리를 내부로 숨김으로서 잘못된 로직으로부터 라이브러리가 불안전해 지는 것을 막을 수 있다.

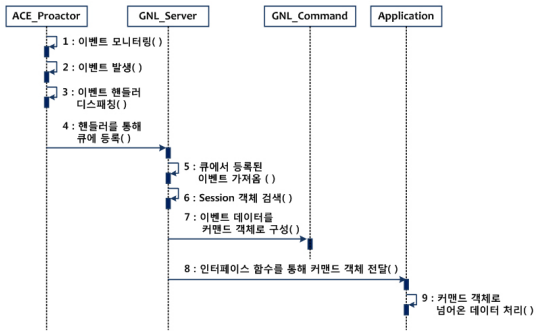
3.1.8 데이터 관리 층

데이터 관리 층에서는 데이터 객체가 네트워크로 전송되기 위해서는 스트림 형태로 변환하는 작업을 한다. 반대로 네트워크에서 전송된 데이터를 관리하며 수신된 데이터는 커맨드로 구성해 주는 작업을 수행한다. 또한 데이터 객체와 ACE라이브러리의 연동을 관리하는 역할도 담당한다.



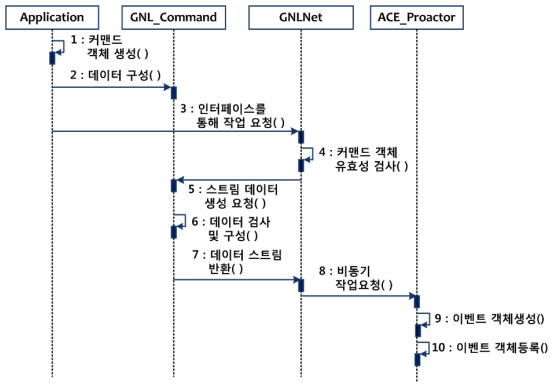
[그림 5] 라이브러리 데이터 흐름도

GNLNet은 [그림 5]에서 보는 바와 같이 ACE 라이브러와 어플리케이션 사이에 하나의 계층으로 존재하고 있다. GNLNet 은 데이터 객체와 최소한의 유저 인터페이스로 연동이 되기 때문에 ACE 라이브러리의 변경 사항이나 GNLNet 라이브러리의 수정 사항이 어플리케이션까지 영향을 주는 것을 막고 있다. 즉, ACE 라이브러와의 의존성을 최소화 하여 변경이나 최적화를 쉽고 안정적으로 할 수 있도록 하였다.



[그림 6] 어플리케이션까지의 데이터 처리과정

[그림 6]은 네트워크에서 방생한 이벤트가 ACE_Proactor에서 Application으로 전달되는 과정을 나타내고 있고, [그림 7]은 Application에서 ACE_Proactor까지 요청한 데이터가 전달되는 과정을 보여 주고 있다.

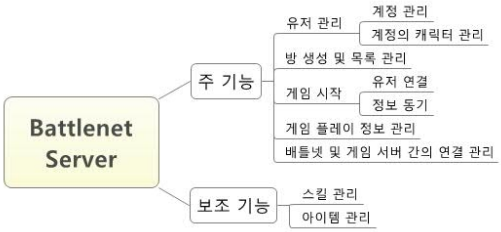


[그림 7] ACE까지의 데이터 처리과정

GNLNet 라이브러리 구현의 가장 큰 특징은 Application에서 라이브러리와 관련된 코딩을 하지 않고 사용 하는데 있다. Application은 커맨드 객체를 생성하고 구성한 후 GNLNet에 요청하는 것으로 모든 작업이 완료 된다. 또한 이벤트를 받는 것 또한 GNL_Server가 재공해주는 인터페이스 함수를 통해서 커맨드 객체를 단순히 받고 있다. 커맨드 객체역시 라이브러리와 연동되는 로직을 내부로 숨기고 있고 데이터를 등록, 가져오는 인터페이스만을 제공하여 라이브러리 내부와 관련된 부분에 접근하지 못하도록 하고 있다. 이렇게 함으로서 라이브러리와 어플리케이션의 의존성을 최소화 하고 어플리케이션에서 라이브러리 복잡한 내부 로직에 관여할 필요성을 제거 하여 사용의 편리성과 어플리케이션에 의해 라이브러리가 불안전해 지는 것을 막을 수 있어 라이브러리의 안전성을 확보할 수 있다.

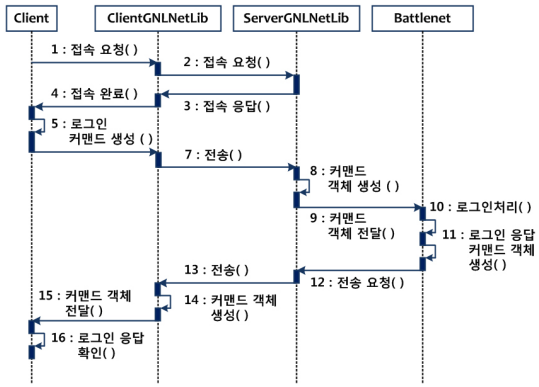
3.2 배틀넷 시스템

본 논문에서는 ACE 네트워크 라이브러리를 사용하여 게임 개발에 최적화된 네트워크 라이브러리를 개발하였다. 따라서 본 논문에서 개발한 네트워크 모듈이 게임 서버 개발에 사용하기 적합한 구조인지를 검증하기 위해 베틀넷 서버를 구현하였다. 베틀넷 서버 구현에 있어서 필요한 요구 사항은 다음과 같다.



[그림 8] 배틀넷 서버 시스템 요구사항

3.2.1 시스템 구성



[그림 9] 배틀넷 서버 접속 및 로그인 과정

- 접속 관리 : GNLNet을 사용하여 정상적인 접속과 접속 종료 처리가 되어 어플리케이션에 접속 이벤트가 통보되고 처리 되어야 한다.
- 로그인 처리 : 계정 생성 및 계정 확인을 작업하며, 계정 확인이 성공적으로 이루어지면 서버는 유저 객체를 생성하고 등록한다. 등록된 유저 객체는 접속이 종료 될 때까지 관리 되며 게임에 필요한 정보를 가지고 있다.
- 유저 객체 관리 : 로그인을 성공한 유저 정보를 관리하는 객체로 게임 정보, 아이템 등을 가지고 있으며, 게임을 플레이 하면서 얻는 정보(Level, Point, Kill/ Death Count)등도 관리 한다.
- 방 관리 : 방의 생성, 삭제, 목록을 관리 한다.
- 게임 관리 : 방에 있는 유저들이 게임을 시작할 수 있는 상태인지 관리 한다.

3.2.2 객체 관리 정책

본 논문에서 개발한 GNLNet을 이용하여 베틀넷 서버를 구현할 때 생성되는 객체들에 대한 효율적인 관리 방법으로 객체 포인터의 소유권과 객체 관리자 정책을 사용하고 있다. 객체가 생성 되면 각 객체 관리자에 등록된다. 등록 시 객체는 서버에서 고유한 ID를 할당해 주고, 베틀넷에서는 이것을 UID라고 정의하고 사용한다. 서버는 객체의 정보가 필요할 때 마다 객체 관리자에서 UID를 가지고 찾게 된다. 또한 객체는 다른 객체의 포인터를 가지고 있을 수 없다. 이는 포인터 관리가 잘못되었을 경우 잘못된 포인터로 서버가 죽는 경우를 줄일 수 있다. 그렇기 때문에 만약 다른 객체를 알고 있어야한다면 서버가 할당해준 객체의 UID를 가지고 있고 필요한 시점에 객체 관리자에서 찾아서 사용할 수 있다.

4. 실험 및 결과

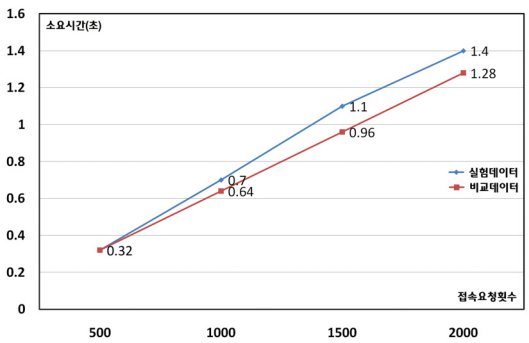
본 논문에서는 ACE를 사용하여 만든 라이브러리의 성능을 확인하고 온라인 게임에서 사용할 만큼의 성능을 지원할 수 있는지를 확인하기 위해 접속 요청 테스트와 데이터 전송 테스트를 실험하였다. 테스트 환경으로 서버는 GNLNet라이브러리를 사용하여 구축한 배틀넷 서버를 사용 하고, 클라이언트는 GNLNet 라이브러리를 사용하여 별도의 테스트용 클라이언트를 만들었으며 테스트를 위해 최소한의 기능만을 구현하였다. 테스트 클라이언트의 기능은 한꺼번에 많은 요청을 시도할 수 있고, 한꺼번에 많은 데이터를 전송할 수 있도록 하였다.

4.1 접속 요청 테스트

첫 번째 실험은 접속 요청 테스트이다. 게임 서버에서 게임 서버에서 부하가 많은 부분 중 하나로 초 당 접속 처리 능력을 테스트 한다. 테스트 방법은 접속 요청과 접속 종료를 무한히 요청하는 테스트 클라이언트를 설정하고, 배틀넷 서버에 접속, 접속 종료를 요청하도록 하였다. 그리고 테스트 클라이언트는 5개로 실행해서 동시에 배틀넷에 요청하도록 하였다. 배틀넷 서버는 초당 접속 처리를 출력하도록 하였다.

[표 1] 접속 테스트 수치 테이블

접속 요청 횟수	소요시간(초)
500	0.32
1000	0.7
1500	1.1
2000	1.4



[그림 10] 접속 테스트 결과 그래프

배틀넷 서버는 초당 평균 1400개의 네트워크에서 발생한 접속 요청을 처리해 주었다. 보통 한 대의 서버가 동시 접속자 수를 5000명 미만으로 잡고 있으며, 한 번에 몰린다 하더라도 인증 작업에 걸리는 시간이 대 부분이며 네트워크 부하로 인해서 접속이 늦어지는 것은 극히 적을 것이다. 그렇다면 초당 1400개의 접속 처리는 충분한 성능이라고 할 수 있다.

4.2 데이터 전송 테스트

두 번째 실험으로써 네트워크 라이브러리의 기본적인 기능으로 데이터 전송 능력을 초당 처리량으로 테스트 한다. 데이터 크기는 다음과 같다.

[표 2] 테스트 데이터 크기

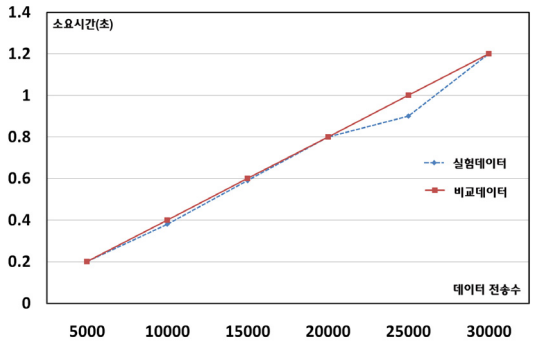
데이터 크기	
Header	2 Byte
Command ID	8 Byte
DataType + UID	1 + 8 = 9 Byte
DataType + int	1 + 4 = 5 Byte
DataType + string	1 + 12 = 13 Byte
Total	37 Byte

테스트 방법으로는 클라이언트에 미리 50000개의 커맨드를 만들어 놓고 한번에 50000개씩 서버로 보낸다. 서버는 받은 커맨드에서 데이터를 가져오는 것 까지 해서 1초에 몇 개의 커맨드를 처리하는 지를 출력하도록 하였다.

[표 3] 데이터 전송 수치 테이블

데이터 전송 수	소요시간(초)
5000	0.2
10000	0.38
15000	0.59
20000	0.8
25000	0.93
30000	1.2

실험 결과로서 서버는 초당 25000대 까지 처리가 가능하였다. 수치적인 면만 본다면 동시 접속자 수가 5000이라 가정한 서버에서 모든 유저가 거의 동시에 대략 초당 4개 정도의 커맨드를 보낼 수 있다고 볼 수 있다. MMO와 같이 분산 서버로 운영이 되는 게임은 한 서버의 동시 접속자수가 많지 않으며 주로 게임 로직이 많은 부분을 차지하기 때문에 실제 처리되는 수는 테스트 수치에 많이 미치지 못한다고 할 수 있다. FPS 와 같이 초당 보내는 커맨드가 많은 게임이라고 하더라도 동시에 게임을 플레이 하는 유저수가 많지 않다. 보통 20명 내외인 것은 감안 하면 초당 600개 정도의 커맨드 처리량을 보여주는데 이는 사람이 입력할 수 없는 수치이다.



[그림 11] 데이터 전송 결과 그래프

5. 결 론

ACE 라이브러리는 여러 분야에서 범용적으로 사용되고 있는 네트워크 라이브러리다. 그렇기 때문에 안정성과 범용성을 구현하기 위해서 사용하는 목적보다 무거울 수 있다. 그런 부하는 대부분이 하드웨어적인 요인이 대부분이라고 할 수 있다. 하지만 ACE 라이브러리가 개발된 후 오랜 시간이 지났고 하드웨어 역시 상당한 발전을 하였다. 또한 앞으로도 계속 발전해 나갈 것이고 이에 따른 라이브러리의 무거운 부분 역시 상당수 감소될 것이라고 예상한다. 이는 개발에 사용되는 기술을 보면 알 수 있는 부분이다. 초기에는 성능을 크게 고려하여 하드웨어에 가까운 기술을 사용하였지만, 하드웨어의 지속적인 발전으로 지금은 성능이 조금 떨어진다 하더라고 사용하기 편하고 안정적인 기술을 사용하려는 경향이 있다. ACE라이브러리를 사용한 것도 마찬가지라고 할 수 있다. 분명 최적화해서 만드는 코드보다 불필요한 코드도 많고 필요 이상으로 구조가 클 수 있다. 하지만 그 차이는 기술이 발전할수록 더욱 좁아 질 것이다. 그것은 테스트에서 충분히 나왔다. 오히려 라이브러리의 약간의 성능을 높이는 것 보다 라이브러리를 사용하는 프로그램의 로직을 최적화 하는 것이 전체적인

성능을 높이는 방법이 될 것이다. 그렇기 때문에 네트워크 라이브러리 개발역시 다른 분야처럼 검증된 라이브러리를 사용하여 개발할만한 가치가 충분히 있다고 판단된다. 개발 초기에 많은 비용이 드는 설계 디자인과 비동기 처리에 대한 복잡한 로직에 대한 구현 및 테스트를 감소시켜주는 것은 그만큼 라이브러리를 사용하는 관점에 좀 더 집중하여 개발할 수 있기 때문에 같은 기간에 더 좋은 품질의 어플리케이션을 만들 수 있게 되다.

아직 ACE라이브러리를 사용한 GNLNet은 최적화해야 하는 부분이 많이 남아 있다. 현재 연구는 ACE라이브러리를 사용해서 네트워크 라이브러리를 개발할 만한 가치가 있는지에 대한 연구였다. 오프젝트 풀이나 ACE 이벤트를 처리하는 방식에 대한 최대화 등의 문제가 아직 남아 있다. 또한 네트워크의 부하가 심할 때 이것을 어떻게 관리를 해 줄 것이지에 대한 문제와 보안에 대한 부분도 미 구현으로 남아 있다. 그리고 단순한 클라이언트 서버 한대의 구조가 아닌 클라이언트들과 서버군의 구조에서도 사용할 만큼의 충분한 성능을 발휘할 수 있도록 라이브러리의 개발 확장이 남아 있다. 향후 연구에는 이러한 문제점들을 해결하여 보다 효율성이 높고 안정적인 게임 서버 구현에 최적화된 네트워크 라이브러리를 구현할 것이다.

참고문헌

[1] D. C. Schmidt, “ACE : An Object-Oriented Framework for Developing Distributed Applications”, USENIX Association, April, 1994.

[2] D. C. Schmidt, “The ADAPTIVE Communication Environment An Object-Oriented Network Programming Toolkit for Developing Communication Software”, Sun User Group conference, 1993.

[3] D. C. Schmidt, “An Arcitectural Overview of the ACE Framework - A Case Study of Successful Cross-platform Systems Software Reuse”, USENIX, 1998

[4] D. C. Schmidt, A. Gokhale, T. ;Harrison, and G. Parulkar, “A High-Performance Endsystem Architecture for Real-time CORBA”, IEEE Communications Magazine, vol.14, Feb. 1997.

[5] D. C. Schmidt, “Acceptor and Connector : Design Patterns for Initializing Communication Services”, in Pattern Languages of Program Design. Addison-Wesley, 1997

[6] D. C. Schmidt, “Reactor : An Object Behavioral Pattern for Concurrent Event Demultiplexing and Event Handler Dispatching,” in Pattern Languages of Program Design, pp.529-545, MA:Addison-Wesley, 1995

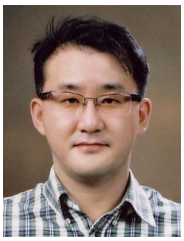
[7] T. Harrison, I. Pyarali, D. C. Schimidt, and T. Jordan, “Proactor-An Object Behavioral Pattern for Dispatching Asynchronous Event Handlers”, in The 4th Pattern Languages of Programming Conference, Sep. 1997.

[8] D. C. Schmidt, “An Object Creational Pattern for Connecting and Initializing Communication Services”, Addison-Wesley, 1997

[9] D. C. Schmidt, Stephen D. Huston, “C++ Networking, Volume 1 : Mastering Complexity with ACE and Patterns”, Paperbook, 2003.

[10] D. C. Schmidt, Stephen D. Huston, “C++ Networking, Volume 2 : Systematic Reuse with ACE and Frameworks”, Paperbook 2003.

[11] D. C. Schmidt, Tim Harrison, and Ehab Al-Shaer, “Object-Oriented Components for High-speed Network Programming”, USENIX, 1995
http://www.cs.wustl.edu/~schmidt/ACE.html
http://www.joinc.co.kr/modules/monwiki/wiki.php/Site/ACE
http://atriarch.com



박성준(Sungjun Park)

1997 호서대학교 컴퓨터공학과 학사
1999 건국대학교 컴퓨터공학과 석사
2005 건국대학교 컴퓨터공학과 박사
2006~(현) 호서대학교 게임공학과 조교수
한국콘텐츠학회 편집위원
한국게임학회 종신회원

관심분야 : HCI, VR, Computer Vision, Bioinformatics

박창훈(Chang-Hun Park)

1995 단국대학교 전자계산학과 학사
1997 단국대학교 전자계산학과 석사
2003 고려대학교 컴퓨터학과 박사
2004 ~ 2006 동경대학교 특별연구원
2006 ~ (현) 호서대학교 게임공학과 조교수
한국게임학회 종신회원

관심분야 : VR, Serious Games, HCI

추교성(Kyo-Sung Choo)

2008 호서대학교 게임공학과 학사
2003 GUNZ Online Game 개발

관심분야 : Game Server, Game DB, Game AI