Journal of Korea Multimedia Society Vol. 7, No. 2, February 2004(pp. 241-250)

GVM 개발환경에서 모바일 온라인 컨텐츠를 위한 프로토콜

김 승 훈†

요 약

본 논문에서는 모바일 온라인 컨텐츠를 위한 이질적인 분산환경에서 단말기간 컨텐츠의 전역상태일치를 보장하고, 통신의 흐름을 조절하는 기능을 가진 서버와 단말기간의 응용계층 프로토콜을 제안하였다. 제안된 프로토콜에서는 단말기의 흐름제어를 위한 부담을 줄이고, 컨텐츠의 내용을 서버가 유지 보관할 필요가 없다. 제안된 응용계층의 프로토콜을 이용하여, CP들은 그들의 컨텐츠를 응용 프로세서로 쉽게 개발할 수 있다. 특히 가입자 수의 증가에 따라 서버규모도 쉽게 확장 가능하다.

Protocol for Mobile On-line Conents in GVM Development Environment

Seung-Hoon Kim†

ABSTRACT

In this paper, an application layer protocol between server and client is proposed for mobile on-line contents in heterogeneous distributed environments. The proposed protocol guarantees the global state consistency in the distributed environments and controls the flow of communication. The protocol reduces the overhead of handsets for flow control and does not need to maintain state information for contents. CPs, by using the proposed scalable protocol, can easily develop their contents as application processes.

Key words: mobile internet(모바일 인터넷), on-line game server(온라인 게임서버), GVM

1. 서 론

온라인 게임 등 온라인 컨텐츠는 유선 인터넷에서는 물론 모바일 인터넷에서도 중요한 응용으로 자리잡고 있다. 그러나 관련업계에서의 필요성에서도 불구하고 모바일 인터넷에서 온라인 컨텐츠를 지원하기 위한 연구, 특히 서버와 단말기간의 프로토콜에 관한 연구, 단말기들의 이질적인 환경을 고려한 단말기간 컨텐츠 상태 일치 보장을 위한 연구 등은 그다지 활발하다고 할 수 없다[1,3,9,10,11,14]. GVM(General Virtual Machine)은 모바일 온라인 컨텐츠를 개발하기 위한 대표적인 개발 환경 중 하나로 국내업체에서 개발되어 현재 가장 많이 사용되고 있다[4-8]. 그러나 현재 GVM 개발환경은 물론 다른 개발환경을 이용한 온라인 컨텐츠는 그다지 많다고 할 수 없다. 본 논문에서는 모바일 인터넷 컨텐츠를 개발하는 CP들을 지원하기 위하여 GVM 개발환경에서 간단하지만 쉽게 확장 가능한 서버와 단말기간의 프로토콜을 개발하였다. 즉, 모바일 인터넷 환경에서 온라인 컨텐츠를 제공하고자 하는 CP들이 그들의 프로토콜을 개발하는 과정에서 참조함은 물론, 본 연구 결과물을 확장하여 자신들의 컨텐츠에 활용할 수 있도록 하였다.

모바일 온라인 컨텐츠에서 단말기와 서버간의 모

※ 교신저자(Corresponding Author) : 김승훈, 주소 : 충청남도 천안시 안서동 산29번지(330-714), 전화 : 041)550-3481, FAX : 041)550-3490, E-mail : edina@dankook.ac.kr
접수일 : 2003년 11월 14일, 완료일 : 2003년 12월 31일
† 정회원, 단국대학교 전자컴퓨터학부 조교수
※ 본 연구는 2003년 (주)신지소프트의 지원으로 연구되었습니다.

바일 데이터 송수신 능력도 중요하지만 컨텐츠를 처리하기 위한 단말기의 처리 능력도 중요하다. 즉, 모바일 온라인 컨텐츠에서의 전역상태일치(global state consistency)를 보장하기 위하여 단말기와 서버간의 통신능력의 편차는 물론이고 수신된 데이터를 처리하기 위한 단말기간의 처리능력의 편차도 고려되어야 한다. 따라서 일반적인 유선 인터넷의 온라인 컨텐츠보다 더욱 편차가 큰 이질적인 분산환경이라 할 수 있다.

GVM 개발 환경에서는 단말기와 서버간의 소켓 통신은 연결지속능력과 신뢰도에 있어서 비교적 양호한 통신 품질을 보이고 있는 것으로 조사되었다. 그러나 유선 온라인 컨텐츠에서와 같은 정도의 활발한 활동을 보이는 컨텐츠를 모바일 온라인으로 지원하기에는 다소 어려움이 있는 것으로 조사되었다. 또한 현재 시중에 서비스되는 단말기를 조사한 결과 컨텐츠를 운영하기 위한 단말기의 처리능력 편차는 상당히 크다. 따라서, 특히 노후한 단말기의 데이터 처리능력을 상회하는 속도로 데이터를 계속 수신할 경우, 단말기에서는 수신된 데이터의 일부를 손실하게되며, 이는 컨텐츠 상태의 불일치를 초래하게 됨을 의미한다. 현재 GVM SDK에서는 컨텐츠에서 통신 속도를 조절하기 위한 기능이 제공되고 있지 않다[4-8]. 따라서 컨텐츠마다 별도로 통신 속도 조절 기능을 구현하여야 한다. 또한 이 과정에서 컨텐츠 상태일치를 위하여 처리 속도가 빠른 단말기로부터 발생된 데이터를 유실함이 없이 다른 단말기들에게 전달되어야 한다. 서버에서 단말기의 데이터를 유지 보관할 경우 이러한 문제를 해결할 수 있으나 서버에서 현재 운영되는 많은 컨텐츠를 위한 데이터를 모두 유지 보관한다는 것은 상당한 부담이 된다. 이와 같이 컨텐츠 상태일치를 보장하면서 통신속도조절을 위한 기능을 각 CP들이 설계하여 구현한다는 것을 각 CP 들에게 상당한 부담이 된다.

본 논문에서는 서버에서 각 단말기의 컨텐츠 처리 상황을 파악하여 컨텐츠 운영에 참가한 단말기들 중에서 가장 느린 단말기의 처리 능력이 포화상태에 도달하였을 경우 다른 단말기들의 송신을 일시 중지시키는 흐름제어(flow control) 기능을 프로토콜에 도입하였다. 컨텐츠를 처리하기에 바쁜 단말기들에게 흐름제어를 위한 판단을 하는 부담을 경감시키도록 하여 컨텐츠 처리에 전념하도록 함으로써 결과적으로 컨텐츠 처리를 더욱 빠르게 할 수 있는 효과를 얻는다. 또한 서버에서 자체적인 판단으로 각 단말기의 송신을 일시 중지시키는 간단한 제어 프로토콜을 도입함으로써 컨텐츠 상태일치를 보장하기 위하여 단말기의 데이터를 서버에서 저장할 필요가 없다. 따라서 서버의 부하가 경감되고 결과적으로 더욱 많은 컨텐츠 운영을 지원할 수 있게 된다.

본 논문에서는 위와 같이 컨텐츠의 전역상태일치를 위한 기능은 물론, 로긴 기능, 채널선택기능, 룸 개설, 선택 및 폐쇄기능, 게이머가 살아서 참여하는 active한 게임 참여 및 단순 관람 형태의 inactive한 게임 참여 등 모바일 인터넷에서의 온라인 컨텐츠를 위한 일반적인 기능을 위한 단말기와 서버간의 응용계층 프로토콜을 구현함으로써, CP들은 컨텐츠라는 응용 프로세서 개발에만 전념할 수 있도록 하였다. 또한 컨텐츠 사용자 수가 증가하더라도 프로토콜은 물론 서버구조의 근본적인 변화가 없는 것이 서버의 유지 보수에 유리하다[12-14]. 이를 위하여 데이터베이스를 구축하고 활용하여 사용자 단말기의 증가에 따라 추가적인 서버의 도입을 쉽게 하도록 하는 확장 가능한 구조로 설계, 구현하였다. 본 논문의 결과는 GVM 개발 환경을 사용하는 일부 CP들에게 제공되어 컨텐츠 개발에 사용되고 있으며, 앞으로도 계속 그 기능을 보완할 예정이다.

본 논문은 다음과 같이 구성되어 있다. 제 2절은 제안된 프로토콜을 정의하기 위한 서버 및 클라이언트의 상태를 정의하고 서버 구현에 필요한 데이터베이스 구축을 정의한다. 제 3절에서는 상위 및 하위 프로토콜과의 인터페이스를 위한 서비스 프리미티브를 정의한다. 또한 서버와 단말기 동등 프로토콜(peer protocol)간의 통신을 위한 PDU(protocol data unit)도 정의한다. 제 4절에서는 단말기의 컨텐츠 처리 속도에 근거한 단말기의 송신 발생을 제어하는 방식의 흐름 제어를 제시한다. 제 5절에서는 제안한 프로토콜의 서버 및 클라이언트의 상태전이도를 제시한다. 또한 제안된 프로토콜을 구현하고 실제로 적용한 동작 예제를 제 6절에 보였으며, 마지막으로 결론과 앞으로의 연구를 제 7절에서 유도하였다.

2. 서버 및 클라이언트의 상태정의 및 데이터베이스 구축

본 절에서는 제안하는 프로토콜을 정의하기 위하여, 서버 및 클라이언트의 상태 정의와 서버간의 통

신 및 컨텐츠 사용자의 데이터를 관리하기 위한 데이터베이스 구축을 정의한다.

2.1 서버의 종류 및 상태

본 논문에서는 모바일 온라인 컨텐츠를 위한 서버와 단말기의 프로토콜만을 제안하며, 서버 구조의 설계를 다루지는 않는다[13]. 이는 서버구조의 설계에 독립적인 프로토콜을 제안하여 CP와 컨텐츠에 적합한 대부분의 서버 구조에서도 사용 가능하게 하기 위함이다. 대부분의 서버 구조에서 로긴 기능을 수행하는 로긴 서버와 이후의 처리를 하는 연결 서버(connection server)가 필요하다. 본 논문에서도 기능상 두 서버를 가정하였으며 앞으로 로긴 서버를 게임서버라고 호칭하며, 연결 서버를 채널서버라 호칭한다.

두 서버의 상태는 다음과 같다. 즉 클라이언트는 게임서버에 접속할 경우 CONNECTED상태가 되며 이후 채널을 선택할 경우 채널 서버로 연결이 이루어진다. 이후 룸을 선택하고 컨텐츠 실행에 참여하는 상태 등을 거치게 된다. 표 1에 서버 상태를 요약하였다. 서버의 상태는 사실 서버에서의 단말기의 상태를 의미한다.

표 1. 서버 상태 요약

서버	상태	설명
게임 서버	WAIT	대기상태
	CONNECTED	로긴상태
채널 서버	CHANNEL_JOINED	채널 선택후 대기실에 있는 상태
	ROOM_ENTERED	룸에 들어가 컨텐츠 실행 대기상태
	ACTIVE_RUNNING	룸에서 컨텐츠 실행에 참여하는 상태
	INACTIVE_RUNNING	룸에서 컨텐츠 실행을 관전하는 상태

2.2 데이터베이스 구축

컨텐츠 운영에 필요한 데이터를 유지 관리하기 위하여, 또한 게임서버와 채널서버간의 통신을 위하여 데이터베이스를 사용한다. 데이터베이스를 이용한 통신을 사용하는 이유는 작은 규모의 서버구조는 물론 분산 대용량의 서버 구조에서도 별다른 수정 없이 모두 사용 가능한 통신 수단이기 때문이다. 유선 인터넷 온라인 컨텐츠와는 달리 모바일 인터넷 온라인 컨텐츠는 서버의 운영규모가 소규모인 경우가 많다. 본 논문에서는 MySQL 데이터베이스를 이용하였으나 대부분의 관계형 DBMS 는 모두 사용 가능하다.

컨텐츠 운영에 필요한 사용자의 데이터를 저장하기 위한 테이블인 userInfo는 id, 현재 서버에 연결상태를 나타내는 flag는 물론 컨텐츠 운영을 위한 임의의 필드들로 구성된다. 이 테이블은 클라이언트의 게임서버 접속, 종료(Disconnection), 및 중지(Abort)는 물론, 채널서버에서의 비정상적인 중지(Abort) 시에도 접근이 필요하다.

채널서버의 정보를 저장하기 위한 테이블인 channelInfo는 채널서버의 IP주소 및 포트번호, 현재 채널서버에서 서비스중인 클라이언트의 수 등으로 구성된다. 이 테이블은 클라이언트가 게임서버에서 채널서버로 재접속을 할 때, 즉 상태가 CONNECTED에서 CHANNEL_JOINED로 바뀔 때 채널서버의 주소를 조회할 때 주로 사용한다. 따라서 서버의 동작 중에 동적으로 새로운 채널서버의 추가가 가능하다. 또한 게임서버와 채널서버의 임시데이터를 전달하기 위하여 Channel# 테이블을 사용한다. 보안의 목적으로 게임서버에서 채널서버로 재접속할 클라이언트의 명단을 전해주기 위하여 주로 사용한다. 그림 1에 데이터베이스의 테이블 구조와 게임서버와 채널서버의 테이블 접근을 보인다.

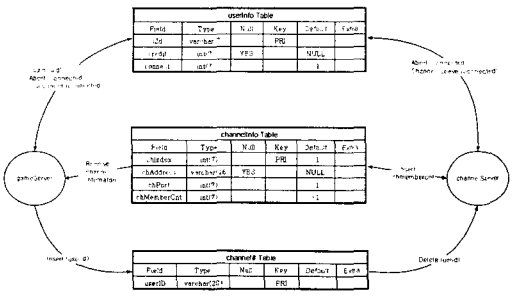

그림 1. 데이터베이스 테이블 및 서버와 관계

2.3 단말기 클라이언트의 상태

단말기에서의 클라이언트의 상태는 서버의 상태보다 복잡하며, 이는 클라이언트가 채널선택 및 룸 생성/선택을 하고, 그 결과를 기다리는 임시 상태가 필요하기 때문이다. 클라이언트의 상태는 표 2에 요약되어 있다. 채널 및 룸에 들어가기 위한 기능은 물

표 2. 단말기에서의 클라이언트 상태 요약

종 류	상 태	설 명
초기상태	Idle	대기상태
게임서버 연결부터 채널서버 재연결까지	Connecting	게임서버에 연결 요청상태
	Connected	게임서버에 연결된 상태
	ChannelChoicing	채널서버 선택 요청상태
	ChannelChoiced	채널서버 선택 확인상태
채널서버 재연결 관리	ChannelJoining	채널서버에 재연결 요청상태
	ChannelJoined	채널서버에 재연결된 상태
	ChannelLeaving	채널서버에서 나가기 요청상태
	ChannelLeaved	채널서버에서 나가기 확인상태
룸 개설 및 선택 관리	RoomCreating	새로운 룸 만들기 요청상태
	RoomChoicing	기존 룸 선택 요청상태
	RoomEntered	새로운 룸/기존 룸에 들어간 상태
	RoomLeaving	룸에서 나가기 요청상태
컨텐츠 운영	ActiveRunning	컨텐츠 운영에 참가하는 상태
	InactiveRunning	컨텐츠 운영에 참가하지 않고 관람하는 상태

론, 채널 및 룸에서 나가서 다른 채널 및 룸을 선택하기 위한 기능을 제공하기 위한 상태도 포함된다. 또한 컨텐츠 운영에 적극적으로 참여하는 상태와 컨텐츠 운영에는 참여하지 않고 관람하는 상태도 클라이언트의 상태에 포함되어 컨텐츠의 유형에 따라 활용할 수 있다.

3. 서비스 프리미티브 및 PDU 정의

본 논문에서는 서버와 단말기간의 온라인 컨텐츠를 지원하기 위한 프로토콜을 응용계층(application layer) 프로토콜로 정의한다. 일반적으로 계층 N-프로토콜은 하위계층인 (N-1)-계층 프로토콜로부터 서비스를 제공받으며, 상위계층인 (N+1)-계층 프로토콜에 서비스를 제공한다. 상하위 계층간의 서비스 요청은 서비스 프리미티브(service primitives)의 집합을 이용하여 명시하며, 프리미티브의 종류는 request, indication, response, 및 confirm이 있다[2]. 본 논문에서 단말기의 온라인 컨텐츠는 응용 프로세서(application process)로 정의한다. 응용 프로세서는 단말기의 컨텐츠이므로 서버에서는 응용 프로세서가 존재하지 않는다. 응용 계층 프로토콜은 세션 혹은 표현 계층 프로토콜의 지원을 받는다.

응용 프로세스는 컨텐츠 수행 중에 응용계층 프로토콜에 서비스를 요청하기 위하여 AP_XXX_REQUEST 프리미티브를 사용한다. 또한 응용 프로세스는 응용계층 프로토콜로부터 AP_XXX_INDICATION 프리미티브를 통하여 서비스를 제공받는다. 따라서 사실 CP들이 구현하여야 하는 것은 이들 프리미티브이다.

단말기의 응용 계층 프로토콜은 AL_XXX_REQUEST 프리미티브를 사용하여 하위 계층에 서비스를 요청한다. 이 요청은 서버의 응용 계층 프로토콜에 전달되어 AL_XXX_INDICATION 프리미티브가 된다. 서버의 응답이 필요한 경우, 서버의 응용계층 프로토콜은 AL_XXX_RESPONSE 프리미티브를 통하여 응답하게 된다. 이 응답은 단말기의 응용 계층 프로토콜에 전달되어 AL_XXX_CONFIRM 프리미티브가 된다. 비슷하게 서버의 응용 계층프로토콜도 AL_XXX_REQUEST 프리미티브를 사용하여 하위 계층에 서비스를 요청한다. 이 요청은 단말기의 응용 계층 프로토콜에 전달되어 AL_XXX_INDICATION 프리미티브가 된다. 표 3에 응용계층 프로토콜과 세션계층 프로토콜간 프리미티를 요약하였다. 표에서 빈칸은 프리미티브가 존재하지 않음을 의미한다.

단말기와 서버의 응용 계층 프로토콜은 AL_PDU_XXX를 송수신 한다. 즉, 실제 전송을 위하여 하위 계층으로 프리미티브를 통하여 서비스를 요청하고, 하위 계층으로부터 서비스를 제공받지만, 동등

표 3. 프리미티브 요약

Primitive	REQUEST	INDICATION	RESPONSE	CONFIRM	설명
AL_JOIN	×	×	×	×	단말기의 게임서버 접속
AL_CHANNEL_LIST	×	×			게임서버의 채널리스트 응답
AL_CHANNEL_CHOICE	×	×	×	×	단말기의 채널 선택
AL_CHANNEL_JOIN	×	×			단말기의 채널서버로 재접속
AL_CHANNEL_LEAVE	×	×	×	×	단말기의 채널 나가기
AL_ROOM_LIST	×	×			채널서버의 개설 룸 상황
AL_ROOM_CREATE	×	×			단말기의 룸 생성
AL_ROOM_CHOICE	×	×			단말기의 룸 선택
AL_ROOM_LEAVE	×	×			단말기의 룸 나가기
AL_ROOM_USER	×	×			채널서버의 룸내의 멤버상황
AL_START_GAME	×	×	×	×	단말기의 컨텐츠 시작
AL_DATA	×	×	×	×	단말기의 컨텐츠 데이터
AL_KILLED	×	×	×	×	단말기의 컨텐츠 데이타
AL_END_GAME	×	×			채널서버의 게임종료 알리기
AL_ABORT	×	×	×	×	단말기의 접속 중단
AL_DISCONNECT	×	×			단말기의 접속 종료

프로토콜간에는 PDU를 전송하게 된다. 단말기의 응용 프로세서와 응용 계층, 서버의 응용계층 및 응용계층 간 PDU 등의 관계를 그림 2에 보인다.

4. 흐름제어

현재 시중에 서비스되는 단말기의 데이터 전송능

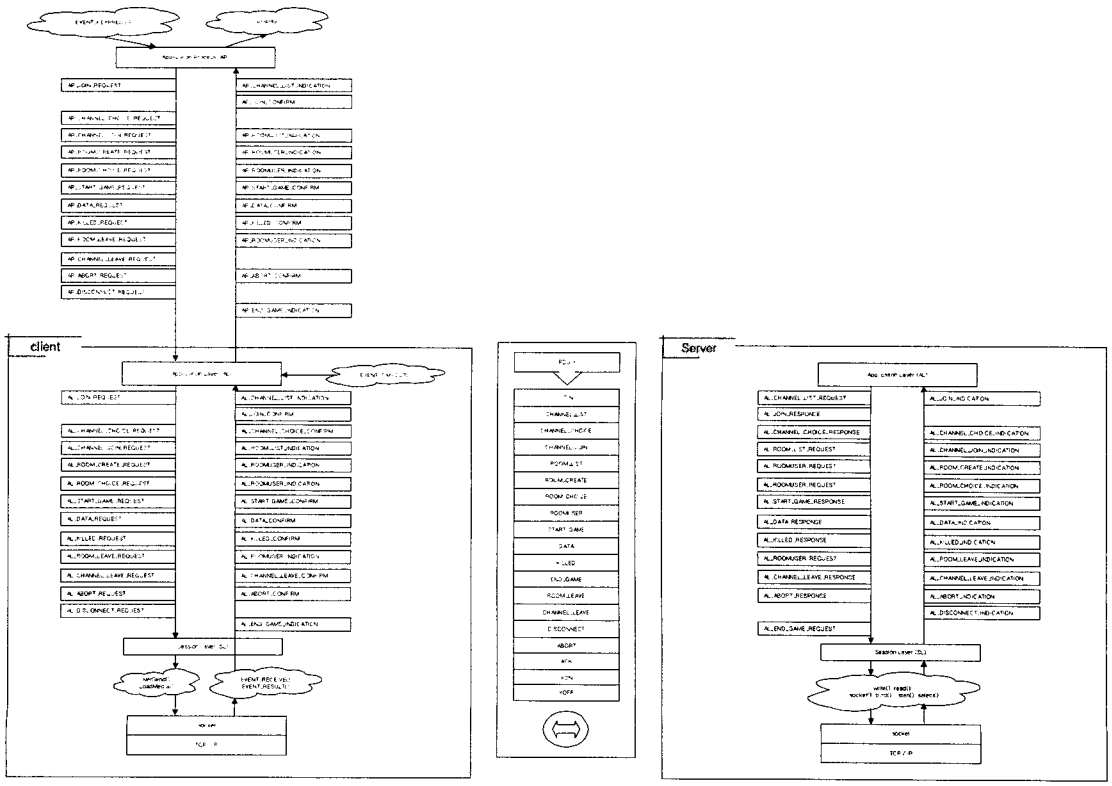

그림 2. 응용 프로세서, 응용계층 프로토콜, 및 PDU

력은 물론 컨텐츠 처리능력 편차는 상당히 크다. 따라서 비교적 성능이 좋은 단말기로부터의 데이터 발생을 제어하지 않는다면, 일부 단말기에서는 수신된 데이터를 처리할 수 없게 되어 컨텐츠 상태의 불일치를 초래할 수 있다. 현재 GVM SDK에서는 컨텐츠에서 통신 흐름 속도를 조절하기 위한 기능이 제공되고 있지 않다. 본 논문에서는 단말기로부터 발생한 데이터의 유실이 없는 통신 흐름제어 기능을 응용계층 프로토콜에 도입하여 CP들의 응용 프로세서를 개발을 쉽게 하였다.

본 논문에서 두 가지 기준을 가지고 흐름 제어를 제안하였다. 첫째, 단말기는 흐름 제어를 위한 부담을 가능한 한 적게 가지도록 한다. 이는 단말기, 특히 성능이 낮은 단말기가 컨텐츠 처리에 주력할 수 있도록 하기 위함이다. 둘째로 서버는 컨텐츠의 내용을 유지 보관하는 등 컨텐츠 운영에 참여하지 않도록 한다. 컨텐츠에 따라서 그 상태를 추적 보관할 필요가 있는 경우도 있으나 일반적인 모바일 컨텐츠의 경우 그 내용의 중요성이 크지 않고, 또한 CP들이 운영할 서버의 성능이 그다지 좋지 못하기 때문이다.

본 논문에서는 서버에서 각 단말기의 컨텐츠 처리 상황을 파악하여 컨텐츠 운영에 참가한 단말기들 중에서 가장 느린 단말기의 처리 능력이 포화상태에 도달하였을 경우 다른 단말기들의 송신을 일시 중지시키는 통신제어 기능을 프로토콜에 도입하였다.

서버가 각 단말기의 컨텐츠 처리 상황을 판단하는 방식은 슬라이딩 윈도우 프로토콜과 비슷하다[2]. 컨텐츠를 운영하며 단말기는 데이터를 송신하게 된다. 서버는 수신된 데이터에 일련번호를 부여하여 현재 컨텐츠에 참여한 모든 단말기에 방송한다. 각 단말기들은 수신된 데이터들을 버퍼에 보관하며 차례로 처리하게 되며 이때 마지막으로 처리한 데이터의 일련번호를 LastProcessed로 저장한다. 단말기는 Last Processed 정보를 서버에 전달하기 위하여 별도의 ACK를 송신하거나 다른 데이터에 piggyback시킬 수 있다. 서버는 단말기로부터 LastProcessed 정보를 받을 경우 해당 단말기가 현재 어떤 데이터까지 처리하였는지 알 수 있다. 이제 단말기의 Last Processed 정보와 이미 송신된 데이터의 일련번호의 차가 일정 크기를 넘을 경우 해당 단말기의 큐가 포화상태에 도달하였다는 것을 서버가 판단할 수 있다.

서버가 단말기로부터 송신을 일시 중지시키는 방식은 X-ON/X-OFF 프로토콜과 비슷하다[2]. 즉 포화 상태에 이른 단말기가 발생하였다고 서버가 판단할 경우 X-OFF를 모든 단말기에 송신한다. 단말기는 X-OFF를 수신할 경우 더 이상의 데이터를 발생시키지 않도록 함으로써 데이터 송신을 일시 중지한다. 서버는 포화 상태에 이른 단말기가 그 상태에서 벗어났다고 판단할 경우 X-ON을 모든 단말기에 송신한다. 단말기는 X-ON을 수신할 경우 데이터의 발생을 허용한다.

이와 같은 서버주도의 제어를 통하여 단말기로부터 발생하는 데이터를 유실함이 없이 데이터의 흐름을 제어할 수 있다. 단말기의 처리상황을 서버에서 파악하기 위한 개요를 그림3(a)에 보이며, 포화 상태에 이른 단말기가 있을 경우 단말기의 데이터발생 제어의 개요를 그림 3(b)에 보인다.

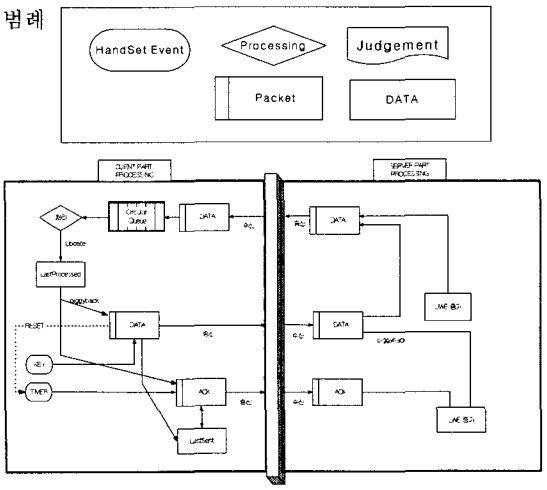

그림 3(a). 흐름제어-단말기 처리상황 파악

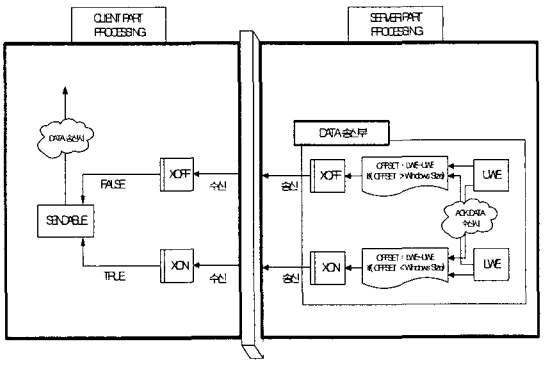

그림 3(b). 흐름제어-단말기로부터 데이터발생 제어

그림 3. 흐름제어

5. 서버 및 클라이언트의 상태전이도

본 논문에서 제안한 서버 및 클라이언트의 모바일 온라인 컨텐츠를 위한 서버 및 클라이언트의 프로토콜에 대한 상태전이도(state transition diagram)를 각각 그림 4와 그림 5에 보인다. 표에 나타난 상태는 2절을 참고하고, 프리미티브는 3절을 참조하기 바란다.

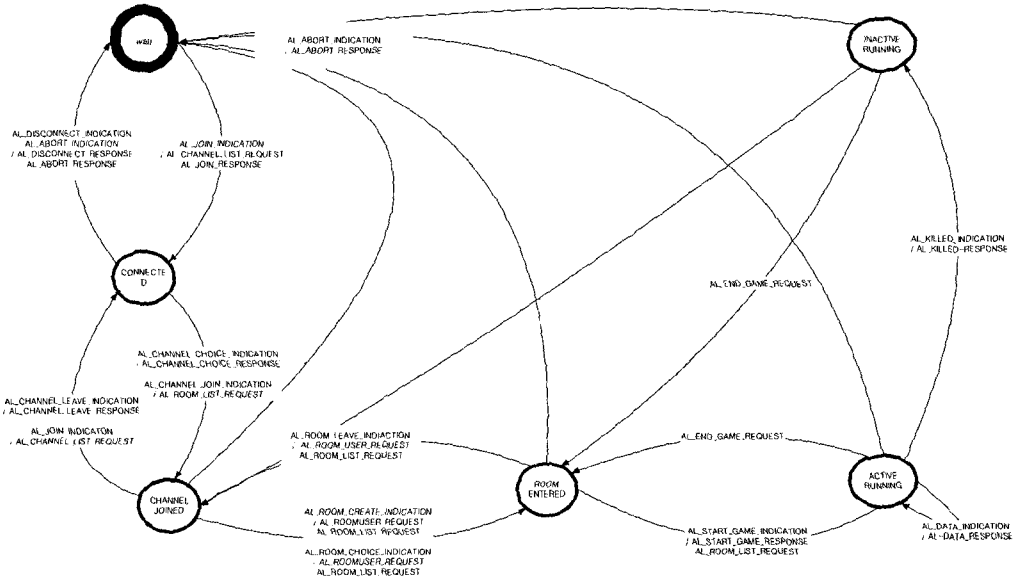

그림 4. 서버의 상태전이도

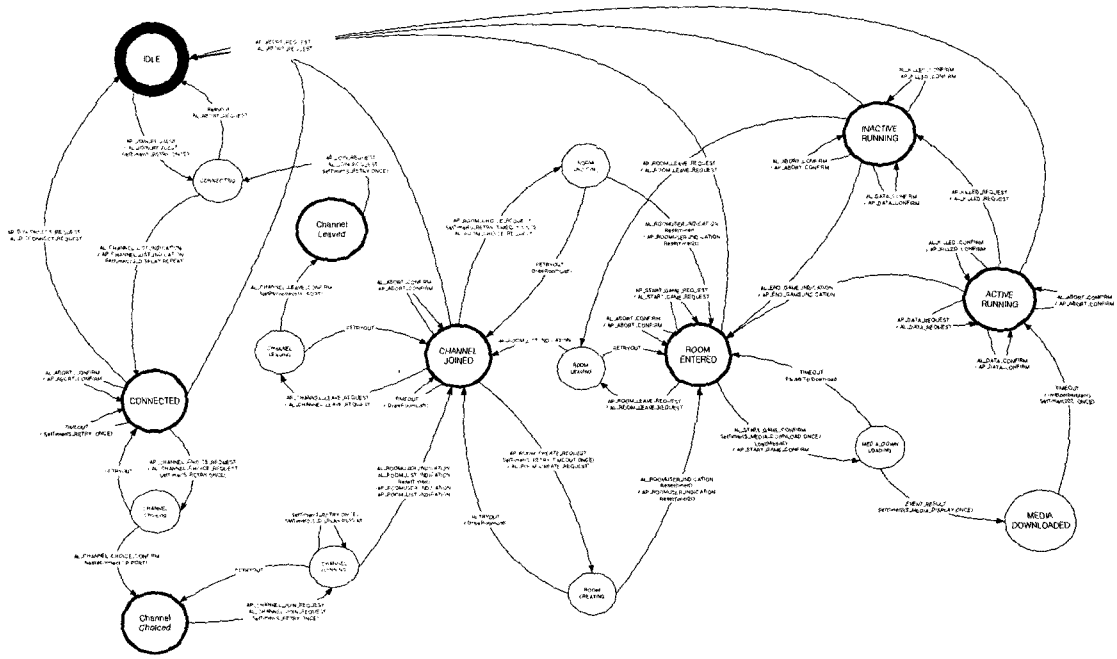

그림 5. 단말기의 상태전이도

6. 프로토콜 구현 및 응용프로세서 동작 예제

제안한 응용계층 프로토콜은 C언어를 사용하여 Linux 및 Unix 환경에서 구현하였다. 구현한 프로토콜의 검증을 위하여 BomberMan이라는 온라인 게임을 응용프로세스로서 온라인 환경에 맞게 수정하여 사용하였다. BomberMan은 2인에서 3인이 온라인으로 실행하는 컨텐츠이다.

응용프로세서의 실행은 그림 6에 보인다. 그림 6(a)는 단말기가 게임서버에 접속한 후 채널서버를 선택

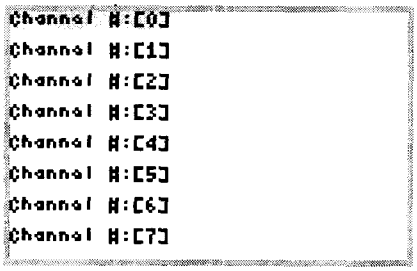

그림 6(a). 채널선택 화면

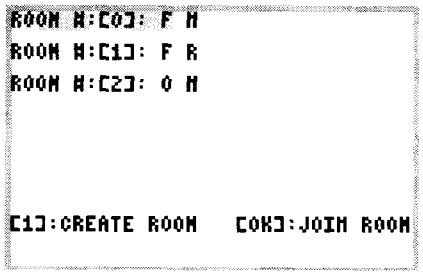

그림 6(b). 채널서버 대기실 화면

그림 6(c). 게임룸 개설 화면

그림 6(d). 게임룸 선택 화면

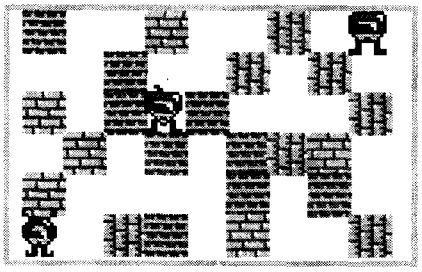

그림 6(e). 게임 시작 화면

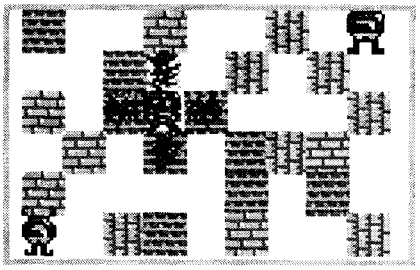

그림 6(f). inactive 상태로 전이

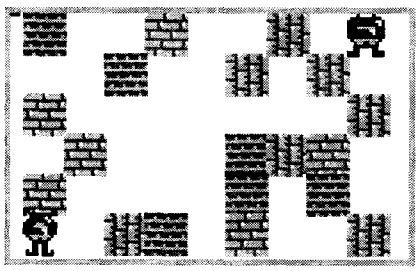

그림 6(g). 2 active/1 inactive 상태의 화면

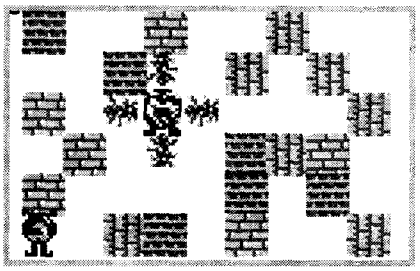

그림 6(h). 최소 게임가능 인원미만으로 변하는 화면

하는 화면이다. 그림 6(b)는 채널을 선택한 후 채널서버에 재접속하여 룸을 생성 혹은 선택하기 위하여 대기실에서 대기하는 화면이다. 현재 3개의 룸이 개설된 상황을 보이고 있다. 그림 6(c)는 새로운 룸을 생성한 후의 화면이고, 그림 6(d)는 기존의 룸을 선택한 후의 화면이다. 그림 6(e)는 임의의 게이머가 게임실행 명령을 송신한 후 서버가 모든 게이머에게 실행명령을 전달한 후 게임이 시작된 화면이다. 이제 게이머들은 active한 상태로 게임에 참여한다. 그림 6(f)-(g)는 임의의 active한 게이머가 active한 상태에서 inactive한 상태로 변화하는 상황을 보여주는 화면이다. Inactive한 게이머는 게임에는 참여하지 못하나, 나머지 게이머들의 상태를 계속 관람할 수 있다. 그림 6(h)는 게임수행도중 최소 게임가능인원인 2명 미만일 경우를 보여준다. 서버는 게임 종료명령을 모든 active/inactive한 게이머에 전달하고, 이에 따라 게이머들은 새로운 게임시작을 대기하는 상태로 변하게 되어 그림 6(d)의 화면과 동일하게 된다.

임의의 CP가 BomberMan과 같은 모바일 온라인 컨텐츠를 개발하기 위하여, 온라인 컨텐츠를 응용 프로세서로 구현한 후, 응용 프로세서와 응용 계층 프로토콜간의 프리미티브만 구현하면 된다. 즉 미리 정의된 인터페이스에 따라 응용계층 프로토콜에 서비스를 요구하고, 정의된 형식에 따라 응용계층 프로토콜로부터 제공되는 서비스를 이용하게 된다. 물론 컨텐츠의 특성상 새로운 기능이 요구되기는 경우도 있으나 제안된 프로토콜에서 쉽게 수용 가능하다.

7. 결 론

본 논문에서는 GVM 개발환경을 이용한 모바일 온라인 컨텐츠를 위하여, 서버와 단말기간의 프로토콜을 제안하였다. 제안한 프로토콜은 간단하고, 가입자 수의 증가에 따라 서버규모가 쉽게 확장 가능한 구조로 구현이 가능하다. 따라서 모바일 컨텐츠를 개발할 CP들이 그들의 컨텐츠를 위한 자신의 프로토콜을 개발하는 과정에서 참조함은 물론, 본 연구 결과물을 확장하여 자신들의 컨텐츠에 직접 활용할 수 있도록 하였다.

본 논문에서는 모바일 온라인이라는 이질적인 분산환경에서 단말기간 컨텐츠의 전역상태일치를 보장하고, 통신의 흐름을 조절하는 기능을 가진 서버와 단말기간의 응용 계층 프로토콜을 제안하였다. 제안된 흐름제안은 두 가지 기준으로 설계되었다. 첫째 기준은 단말기는 흐름 제어를 위한 부담을 가능한 한 적게 가지도록 한다. 둘째 기준은 서버는 컨텐츠의 내용을 유지 보관하는 등 컨텐츠 운영에 참여하지 않도록 한다. 본 논문에서는 서버에서 각 단말기의 컨텐츠 처리 상황을 파악하여 컨텐츠 운영에 참가한 단말기들 중에서 가장 느린 단말기의 처리 능력이 포화상태에 도달하였을 경우 다른 단말기들의 송신을 일시 중지시키는 흐름제어 기능을 프로토콜에 도입하였다.

그동안 온라인 게임 서버의 아키텍처에 관한 연구는 있었으나[11,12], 특히 모바일 온라인 컨텐츠를 지원하기 위한 서버와 클라이언트의 프로토콜을 제시한 연구는 찾아보기 힘들었다. 효율적인 게임 이벤트 데이터 전송을 위한 새로운 전송 프로토콜인 게임전송프로토콜(GTP)를 제안한 연구[9,15]는 있었으나, 이는 전송계층의 프로토콜로 기본적으로 소켓통신을 사용하는 국내의 모바일 인터넷에 사용하기 어렵다. 또한 현재 다양한 성능을 보이는 단말기간 흐름제어를 위하여 기존 유선네트워크에서의 일반적인 흐름 제어를 사용하기에는 어려움이 많다. 본 논문에서는 특정 온라인 컨텐츠에 특화된 부분과 공통적으로 컨텐츠에서 요구하는 부분을 분리하는 Layering 개념을 갖는 응용계층의 프로토콜을 제시하였으며 간단하지만 쉽게 적용가능한 흐름 제어를 제안하였다. 본 프로토콜은 응용계층의 프로토콜로 다른 게임서버 프로토콜과 호환하지 못한다는 단점이 있다. 이를 위하여 SIP 프로토콜을 적용한 연구도 계획중이다.

제안된 프로토콜을 기술하기 위하여 서버 및 단말기의 상태, 서버간의 통신을 위한 데이터베이스 구조, 서비스 프리미티브 및 PDU 등을 정의하였다. 또한 단말기와 서버의 상태전이도를 제시하였다. 제안된 프로토콜은 컨텐츠의 다양한 요구를 반영할 수 있는 구조로 확장되기 위하여 계속 연구될 예정이다.

제안된 응용계층 프로토콜은 C언어로 Linux 및 Unix 환경에서 구현되었으며, 검증을 위하여 BomberMan이라는 응용프로세스를 사용하였다.

참 고 문 헌

[1] E. Cronin, B. Filstrup, AR.Kurc and S. Jamin, "An Efficient Synchronization Mechanism for

Mirrored Game Architectures," *NetGames 2002*, April 16-17, 2002.
[2] Fred Halsall, *Data Communications, Computer Networks and Open Systems*, Addison-Wesley, 1996.
[3] J. Farber, "Network Game Traffic Modelling," *NetGames 2002*, April 16-17, 2002.
[4] GVM SDK Overview, 신지소프트, 2003.
[5] GVM SDK User's Guide, 신지소프트, 2003.
[6] Mobile C Library Function Reference, 신지소프트, 2003.
[7] Mobile C Programming Guide, 신지소프트, 2003.
[8] Mobile C Reference Guide, 신지소프트, 2003.
[9] Sangheon Pack, Eunsil Hong, Yanghee Choi, Jong-Sung Kim, and K. Ko, "Game Transport Protocol: A Reliable Lightweight Transport Protocol for Massively Multiplayer On-line Games(MMPOGs)," *ITCOM 2002*, Boston, USA, pp.83-94, July, 2002
[10] "모바일 게임시장 및 개발동향," 한국정보처리학회, 정보처리, 제9권 제3호, 2002. 5.
[11] "온라인 게임 개발 현황," 한국정보처리학회, 정보처리, 제9권 제3호, 2002. 5.
[12] 고동일, "온라인 게임 네트워크/서버 기술," Taff System, 기술문서.
[13] 신동원, "온라인 게임 서버 아키텍처의 연구," KGDA(Korea Game Developer Association) 네트워크분과, 연구논문.
[14] 오재환, "온라인 상의 고려할 내용들," KGDC 2002.
[15] 홍은실, 백상현, 박일규, 김종성, 고동일, 최양희, "게임 전송 프로토콜(GTP): 효율적인 게임 이벤트 데이터 전송을 위한 새로운 전송 프로토콜," JCCI, 통신 정보 합동 학술대회, April, 2002.

김 승 훈

1985년 2월 인하대학교 전자계산학과 졸업
1989년 8월 인하대학교 전자계산학과 석사
1998년 2월 포항공과대학교 전자계산학과 박사
1998년 3월~2001년 8월 상지대학교 컴퓨터정보공학부 조교수
2001년 9월~현재 단국대학교 전자컴퓨터학부 조교수
관심분야 : 멀티미디어통신망, 멀티미디어응용