Journal of the Korea Industrial Information Systems Research Vol. 20 No.6, Dec. 2015 :11-20
http://dx.doi.org./10.9723/jksiis.2015.20.6.011 ISSN:1229-3741

슬롯머신 회계 프로토콜 (SAS-G2S) 컨버터의 설계 및 구현

(Design and Implementation of Slotmachine Accounting Protocols (SAS-G2S) Converter)

김 상 민[1], 박 현 준[2], 안 희 준[3]*
(Sangmin Kim, Hyunjoon Park, and Heejune Ahn)

요 약 본 연구에서는 카지노 슬롯머신의 회계관리에 사용되고 있는 기존의 SAS 프로토콜을 사용하는 단말기들을 근래에 정의된 인터넷기반의 G2S 프로토콜 서버와 연동하기위한 SAS대G2S 변환 미들웨어시스템을 설계하였다. 두 프로토콜이 갖는 차이점, 폴링방식과 메시지방식, 바이너리 포맷과 XML 포맷, 매칭되지 않는 메시지 등의 문제들을 해결한다. 소프트웨어 구조는 브리지 패턴을 바탕으로 하고 프로토콜을 커넥터 패턴을 바탕으로 하고, 리눅스환경에서 C++와 Python 언어로 구현하였다. 1GMHz-512MB 정도의 사양의 임베디드 시스템의 시작품으로 제품화되어, 2015년 공인 정합성 및 안정성을 테스트를 시험을 통과하였다.

핵심주제어 : 카지노, 슬롯머신, 회계, 프로토콜 컨버터, SAS, G2S

Abstract This paper describes design and implementation experience of SAS-G2S slotmachine accounting protocol middleware system that converts the legacy binary format SAS (Slot Accounting System) protocol with servers with recent standards G2S protocol. The paper examines the difference of two protocols such as link control, message format, and parameters. The converter architecture uses bridge and connector patterns and implemented in C++ and Python mixed language on Linux environment. The prototype system uses a 1GMHz-512MB linux machine and has passed Korean official protocol compatibility and performance test in 2015.

Key Words : Casino, slotmachine, accounting, Protocol Converter, SAS, G2S

1. 서 론

슬롯 머신회계 프로토콜 및 시스템은 국내외 카지노 (해외 라스베가스, 마카오 및 국내 강원랜드, 세븐럭 등)에서는 슬롯머신의 매출 및 부가 서비스, 조작방지 등 (Table 1)을 지원한다. 국내외 미주, 아시아 등에서는 현재 SAS(Slot Accounting System) 프로토콜[1]을 주로 사용한다. SAS는 IGT (International Gaming Technology)에 의하여 1980년대 중반에 개발되어 여러 가지 기술적 한계가 있어, 2000년대 중반부터 GSA

* Corresponding Auther : heejune@snut.ac.kr
본 연구는 미래창조과학부 및 정보통신기술진흥센터의 해외ICT 전문인력활용촉진사업의 연구결과로 수행되었음" (IITP-2015-R0134-15-1030)지원에 의해 연구되었음.
Manuscript receivrd October 29, 2015 / revised November 24, 2015 / accepted December 2, 2015
1) (주) 몽태랑, 본 연구는 서울과학기술대학교 석사과정 중 연구되었음, 제1저자
2) (주) 몽태랑, 제2저자
3) 서울과학기술대학교 전기정보공학과, 교신저자

(Gaming Standards Association) 주도로 인터넷 기반의 G2S(Game to System) 프로토콜[2]이 정의되어 있다.

Table 1 Core functions of slot accounting protocols

Communication	Manage Communication State between an EGM and a Host.
Fund Transfer	transfer fund from and to a EGM and host (or Player Account).
Metering	tracking Meter values in an EGM about Accounting.
Event Handle	manage Events generated in an EGM.
Player Tracking	Track a Player playing a game.
Authentication	system authentication for a Regulator.

SAS기반 시스템은 서버와 게임머신 사이에 SMIB(Smart Interface Board)라는 인터페이스 장치를 사용 (Fig. 1)하여 SMIB와 게임머신 사이에는 표준 SAS 프로토콜을 이용하지만, SMIB와 서버사이에는 SMIB 제조사에 따른 비표준 프로토콜을 이용하여 SMIB제조사의 의존성 및 호환성에 문제가 있다. 반면 최근 정의된 G2S 프로토콜 (Fig. 2)은 인터넷 표준을 준용하므로 제조사의 의존성 없이 시스템 간 하나의 표준 프로토콜로 묶을 수 있다.

이러한 장점에도 불구하고, 국내외 현장에서는 기존에 설치된 SAS기반 시스템을 SAS 게임머신을 G2S기반의 신식 머신으로 교체하기 위하여 필요한 비용과 그 과정에서 발생할 수 있는 서비스의 불안정성 등이 전환을 어렵게 하고 있다.

본 논문에서 이러한 호환성 문제를 해결하기 위한 방안으로 기존의 SAS 프로토콜과 신규 프로토콜인 G2S를 변환하는 장치를 설계하고 구현하였다. 그림 2에서 보는 바와 같이 카지노는 이를 이용해 기존의 SAS 게임머신의 객장환경에서도 게임머신의 교체 없이 G2S 기반의 시스템을 구축하고 장점을 활용할 수 있고, 과도기적인 기간 동안 카지노의 점차적인 G2S 머신교체를 가능하게 하여 카지노의 비용부담을 줄여줄 수 있다.

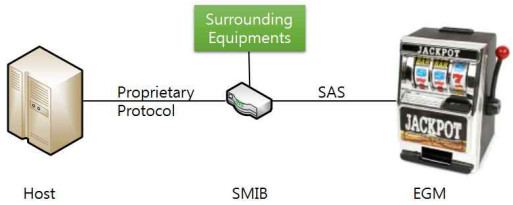

Fig. 1 Using a SMIB, Routine Network Based SAS

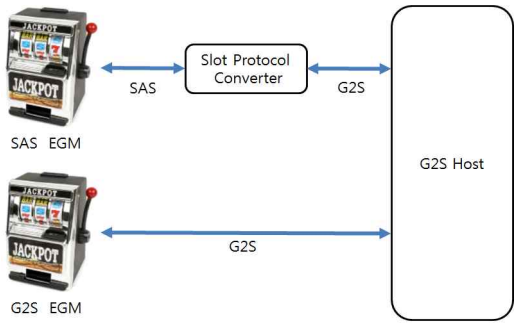

Fig. 2 Using a Slot Protocol Converter, Network Based SAS and G2S Mixed.

2. 시스템 요구사항 분석

슬롯 프로토콜 컨버터 디자인을 위해서는 SAS와 G2S 프로토콜의 비교를 통한 기능을 대응시키고, 차이점을 적절히 보상하여야 한다.

2.1 SAS 프로토콜

호스트 (SMIB)와 게임머신은 19200bps의 RS-232C방식의 시리얼 통신을 물리계층의 방식으로 사용하는데, 특이한 점은 8 비트의 데이터에 한 비트의 웨이크-업(wake-up)비트)을 추가하여 9bit 통신하는 점이다. 이 웨이크-업 비트는 호스트가 메시지의 시작을 표시하는 용도로 첫 번째 바이트를 1로 설정하는데 사용된다. 링크 구조는 SMIB와 게임머신간 1:N 스타구조 사용하고 링크제어방식으로 마스터/슬레이브(master/slave)를

사용한다. 머신들은 1~127의 주소값으로 구분된다.

사용되는 메시지의 포맷은 Fig. 3과 같다. 호스트는 마스터로서 GP(general poll)과 LP(long poll)의 두 가지형태의 메시지로 폴링(polling) 방식을 사용한다. 호스트는 머신의 주소만으로 구성된 1바이트의 GP을 주기적으로 머신 전송되어 머신의 상태(이벤트)를 수집한다. 에에 대한 응답으로 머신은 1바이트의 상태정보가 담긴 예외(exception) 코드를 반환한다. 또한 호스트는 게임머신에게 최대 255 바이트의 LP을 전송하여 명령을 하고, 머신은 대응하는 처리결과를 응답으로 반환한다. LP과 응답 메시지내의 파라메터는 BCD(Binary-coded decimal), ASCII 그리고 바이너리 포맷을 사용한다.

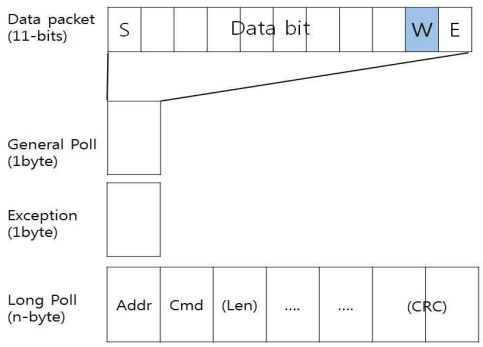

Fig. 3 SAS Message Format

2.2 G2S 프로토콜[2]

G2S는 전송방식으로 인터넷 프로토콜을 메시지 포맷으로 XML SOAP 방식을 [7, 8, 9] 사용한다. G2S게임 머신과 G2S 호스트는 N:N 통신이 가능하다. 여러 상대 시스템과의 통신과 다양한 명령의 처리에 요구되는 시간들이 다르고 독립적인 것을 고려하여, 요청 메시지와 응답 메시지는 '비동기' 방식을 사용한다. 이러한 목적으로 메시지의 전달을 담당하는 메세지 계층과 명령을 처리하는 응용계층으로 구분되어 있으며, 계층사이에는 인-바운드와 아웃-바운드 큐가 존재한다. Fig. 4는 G2S 응답 메시지의 처리과정을 보여주는 메시지 흐름도이다.

G2S는 응용계층은 기능 단위로 구분된 클래스(class)로 모델된다. 이 클래스 모델이 시스템에 생성되면 이를 디바이스(device)라고 한다. 기능들은 이 클래스의 명령들로 정의되는데 메시지 포맷은 XML형태를 따른다. Table 2는 G2S 메시지 중 간단한 경우의 예를 보이고 있다.

2.3 SAS와 G2S 프로토콜의 비교

Table 3에 SAS와 G2S 프로토콜의 차이점을 비교하여 정리하였다. SAS는 폴링방식의 순차 동기적인 마스터/슬레이브 통신 프로토콜이며, G2S는 메시지의 전송과 처리가 분리된 비동기적인 서버/클라이언트 통신방법을 택하고 있다. 이러한 구조적인 차이로 인해 두 프로토콜은 게임머신에서 발생하는 이벤트에 관한 정보를 가지고 오는 과정에서 차이를 보인다.

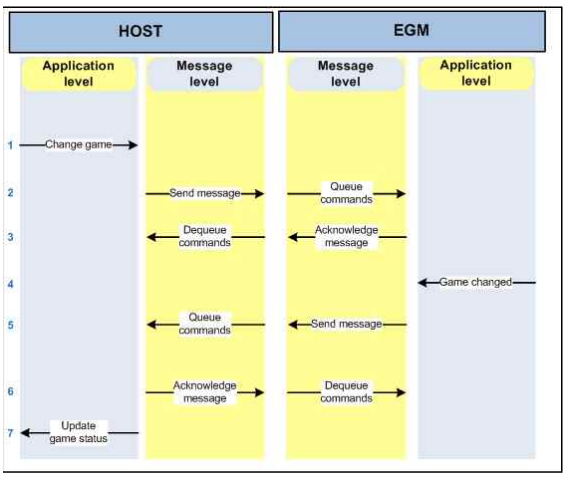

Fig. 4 Process Procedure of a G2S Message[2]

Table 2 G2S message format

G2S_body	G2S_ack
<g2sMessage> <g2sBody> <anyClass> <anyCommand /> </anyClass> <anyClass> <anyCommand /> </anyClass> </g2sBody> </g2sMessage>	<g2sMessage> <g2sAck /> </g2sMessage>

Table 3 Comparison SAS and G2S protocols

Category	SAS	G2S
transport	RS-232	Internet protocol
architecture	Master-Slave Polling	Peer to Peer Asynchronous
server interface	Requires a SMIB	Directly Support
player peripheral	Connect to SMIB	EGM itself with GDS
message encoding	Binary	XML
reliable delivery	Implied Ack	Explicit Ack

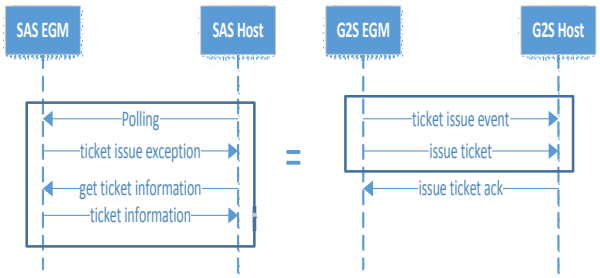

Fig. 5 Protocol Comparison G2S and SAS

게임머신에서 티켓이 발급되는 경우, SAS와 G2S프로토콜 모두 게임머신에서 발급한 티켓에 대한정보를 호스트에게 알리도록 명시하고 있으며, 정보의 실제 내용 또한 매우 유사하다. 그러나 통신 방식은 Fig. 5와 같이 다르다.

SAS

① 게임머신이 고객에게 티켓을 발급
② 호스트(SMIB)는 게임머신에게 일반 poll을 전송(polling)
③ 게임머신이 티켓 발급 exception 응답
④ 호스트가 티켓발급정보를 위해서 ticket info 명령 전송
⑤ 게임머신이 티켓발급 정보를 요청 응답

G2S

① 게임머신이 티켓을 발급
② 게임머신은 티켓발급 관련 정보를 포함한 메시지를 함께 발생
③ 호스트가 확인응답 회신

두 프로토콜 방식의 차이로 인하여, G2S의 티켓발급 ②절차를 위해서는 SAS 티켓 발급절차의 ②, ③, ④을 조합해야 한다. Fig. 6에서 변환기 내부적으로 이 티켓발급 절차 상태를 가지고 있어야 하며 하나의 절차가 끝나기 전까지 모든 메시지의 정보를 저장하고 있어야 한다.

이 변환 과정을 구현하기 위해서는 슬롯 프로토콜 변환기는 여러 개의 SAS메시지들을 내부적으로 하나의 '세션'으로 묶어 관리할 수 있는 기능이 필요하다. 이를 이용해서 연속되는 메시지 속에서 연관된 메시지들을 구분하여 메시지 변환을 위한 올바른 정보를 수집할 수 있다. 일부 메시지들은 하나의 G2S 메시지를 생성하기 위해서 하나 이상의 SAS 요청과 응답이 필요로 하는 경우도 있다.

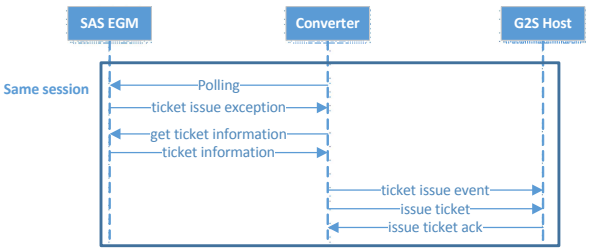

Fig. 6 Session Bundle during Converting Protocol

① 머신이 고객에게 티켓을 발급
② 컨버터(호스트)는 GP를 전송
③ 게임머신은 티켓 발급 SAS 예외를 전달
④ 컨버터는 티켓발급 G2S 이벤트를 준비
⑤ 컨버터는 티켓발급정보를 얻기 위해서 게임머신에게 요청 메시지 전송
⑥ 게임머신은 티켓발급관련 정보를 요청에 대한 응답으로 전송
⑦ 컨버터는 게임머신으로부터 받은 응답에서 원하는 정보 추출
⑧ 컨버터는 대기 시켜놓은 티켓발급 G2S 이벤트에 관련-데이터를 호스트의 설정에 따라 포함을 결정하고 G2S 호스트에게 전송
⑨ 컨버터는 티켓발급 관련 정보를 담은 메시지를 G2S 호스트
⑩ G2S 호스트가 수신확인응답

SAS 프로토콜에서는 티켓발급과 펀드 전송에 관련된 트랜잭션 로그만을 지원하는 반면 G2S에

서는 다양한 트랜잭션 로그를 지원한다. G2S에서 요구하는 트랜잭션을 추적하기 위해서는 컨버터 내부를 지나는 SAS와 G2S 메시에서 정보들을 추출하고, 로그 정보를 생성할 수 있는 기능이 구현되어야 한다. 아래 표는 프로토콜별 지원하는 트랜잭션 로그를 정리하였다.

SAS의 예외 메시지는 1바이트의 바이너리 코드만을 포함하는 간단한 메시지이지만 G2S의 이벤트는 이벤트 코드와 동시에 연관-데이터(affected Data)를 포함할 수 있어야 한다. 연관-데이터는 이벤트의 발생으로 인해 변경될 게임머신의 상태 값들이다. SAS의 예외 코드는 매우 제한적인 정보만을 제공한다. 따라서 슬롯 프로토콜 컨버터에서 SAS 예외 메시지를 이용해서 G2S의 연관 데이터 구현하기 위해서는 Table 4와 같이 추가적인 정보수집을 통하여 이벤트에 포함시켜야한다.

Table 4 Comparison SAS Exception and G2S Event

SAS Exception	G2S Event
1Byte Exception Code	G2S Event Code + MeterInfo(Op) + TransactionInfo(Op) + StatusInfo(Op)

3. 시스템 설계

전체 시스템 구성은 다음 Fig. 7과 Table 5에 제시된 것과 같다. 저자들이 2014년에 G2S엔진을 C++기반으로 개발 [10]한 적이 있으나, 최근 IT 업체들의 개발자들의 추세가 C/C++에서 Java/Python으로 전환되고 있으며, Java의 경우는 유료라이센스 정책으로 인하여 제품 단가 등이 증가될 수 있는 점을 반영하여 Python 기반으로 설계하였다. 구조상 프로토콜 변환기는 '브리지 패턴'과 '커넥터 패턴'을 사용 [11]하며, 구조적으로 3가지 모듈로 구성하였다 (Fig. 8). SAS 머신과 연결된 SAS 프로토콜의 링크를 담당하는 SAS 엔진과 프로토콜 변환과정과 메시지가 처리되는 컨버터와 G2S 호스트와 붙어 G2S 프로토콜의 링크를 담당하는 G2S 엔진 있다. 각각의 모듈들은 쓰레드 단위로 구성되어 있으며 이 쓰레드들은 이벤트 기반의 큐를 통해서 통신한다. 평소 이들 쓰레드들은 큐로부터의 이벤트를 기다리며, 큐로부터 이벤트가 발생하였음을 알리는 I/O가 발생하면 큐에서 메시지를 꺼내 처리하는 방식으로 동작한다.

슬롯 프로토콜 변환기는 SAS와 G2S로부터 각각 바이너리와 XML 형태의 메시지를 받아 Json 형태로 변환하여 내부 처리한다(Table 6). Json 메시지는 Python의 자료구조인 사전(dictionary)와 상호변환이 용이해 다루기 쉽다는 장점이 있다. 여기서 SAS 바이너리 메시지를 Json 형태로 변환한 형태를 SAS-유사, G2S XML 형태의 메시지를 Json으로 변환한 메시지를 G2S-유사 형태라 칭하겠다.

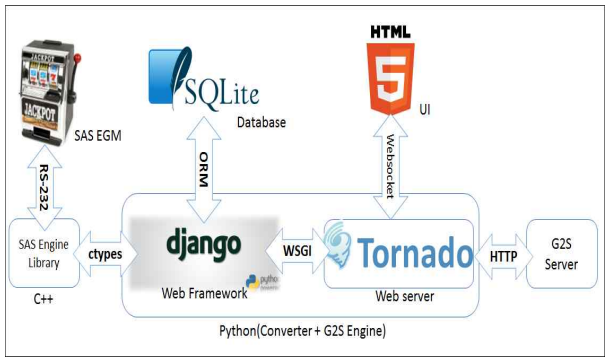

Fig. 7 End to end Slot machine Accounting System Architecture

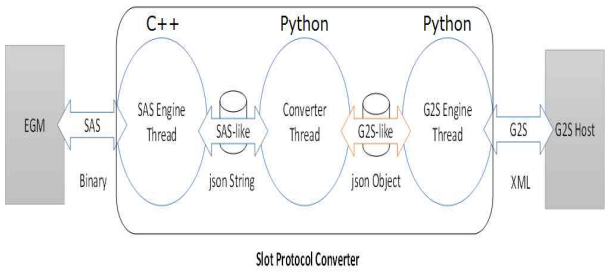

Fig. 8 The proposed slot protocol converter architecture

파이썬의 쓰레딩 정책은 GIL(global interpreter lock)을 사용하고 있어 멀티코어를 사용함에도 불구하고 다중-쓰레딩에서 싱글-쓰레딩보다 낮

은 성능을 보여준다. 즉 코드상에서는 쓰레딩을 사용하고 있지만 파이썬 인터프리터는 한 번에 하나의 쓰레드만을 실행시켜 쓰레딩의 이점을 이용하지 못하며 문맥전환(context switch)의 오버헤드만 발생시킨다[3]. SAS프로토콜은 메시지 전송을 위한 시간 요구조건이 미리세크단위로 비교적 실시간 성을 요구하고 있다. 이는 파이썬으로 구현된 다중-쓰레딩 시스템에서 예상치 못한 병목현상으로 프로토콜 정합성에 문제를 제기할 수 있다. 이러한 영향을 줄이기 위해서 SAS 엔진은 C++를 이용하여 구현하여 동적 라이브러리화 하였고 파이썬코드에서 SAS 엔진 라이브러리를 불러 사용할 수 있도록 ctype[4] 패키지를 사용하였다.

Table 5 Input, output message format for modules

Module	Input	Input format	Output	Output format
SAS Engine	SAS Message	Binary	SAS-like Mesasge	Json
Converter	SAS-like Message	Json	G2S-like Message	Json
G2S Engine	G2S-like Message	Json	G2S Message	XML

3.1 SAS 엔진

슬롯 회계 프로토콜 변환기에게 SAS 엔진은 Table 6에 명시한 바와 같은 SAS의 링크계층의 기능을 수행한다. 이 커넥터 엔진을 통하여 SAS 기능을 추상화하고 있으며, SAS 프로토콜의 정의에 의해서 SAS 엔진은 SAS기반의 머신과 연결되어 게임머신을 제어, 모니터링하고, 또는 컨버터 브리지로부터의 부터의 명령어를 처리한다.

Table 6 Function List of the SAS Engine

SAS Engine for SAS Protocol Link
Loop Break Condition
Wake-up Mode
Polling
Inter-byte time, No-Response time
Implied Ack
Re-transmit(3-Times)

이때 인터페이스에 사용되는 포맷은 SAS의 바이너리 포맷대신 Json형태의 메시지의 디코딩을 사용하여 사람이 읽을 수 있는 형태로 변환된다. 이 과정은 jsonCPP라이브러리를 기본으로 단위 데이터타일을 변환하고, SAS 메시지 디코더/인코더 클레스를 정의하여 구현하였으며, 그 변환 예를 Table 7에 제시하였다.

Table 7 Example of Decoding Binary SAS Message to Json

Binary Format SAS Message	Json Format SAS Message

0x01 0x74 0xFF 0x00
0x00 0x00 0x1A 0xF9
	{
	"lockTimeout": 0,
	"lockCode": 255,
	"command": 116,
	"address": 1,
	"type": "SAS_longPoll",
	"transferCondition": {
	"transferFromGamingMachineOk": 0,
	"transferToGamingMachineOk": 0,
	"bonusAwardToGamingMachineOk": 0,
	"transferToPrinterOk": 0
	}
	}

3.2 G2S 엔진

Fig. 9와 같이 G2S 엔진 커넥터는 G2S 프로토콜 부분을 담당한다. 전송방식으로 HTTP/SOAP을 사용하며 G2S 호스트와 통신하고, 컨버터와 G2S-유사메시지 포맷으로 인터페이스 한다. G2S엔진의 G2S에서 정의하는 메시지 계층의 역할을 주로 담당하며 응용 로직은 컨버터에서 담당한다.

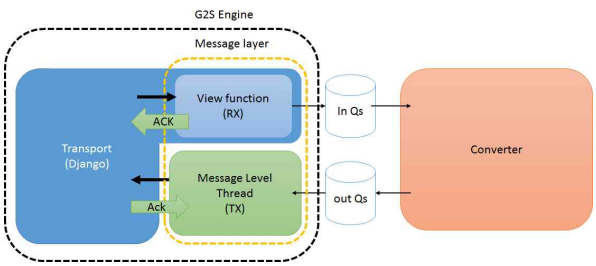

Fig. 9 G2S Engine Structure

3.3 컨버터

컨버터는 파이프라인(pipeline)과 이벤트 큐 기반의 구조로, 큐는 SAS 커넥터, G2S 커넥터 및 기타 사용자 및 시스템 이벤트 큐로 구성된다. 컨버터와 커넥터들 간의 통신은 Fig. 10에 보인 바와 같이 컨버터 쓰레드는 큐에 메시지가 들어오면 해당 큐에서 메시지는 가져와 파이프라인에게 순차적으로 전달한다. 파이프라인 구조는 메시지 처리에 사용되는 단계를 정형화하고, 여러 가지 기능에 대하여 공통적인 처리 흐름구조를 갖도록 고안되었다. 이 파이프 단위 처리자들은 전달받은 메시에 대해서 자신의 역할을 한 후, 다음 파이프 모듈에게 전달한다.

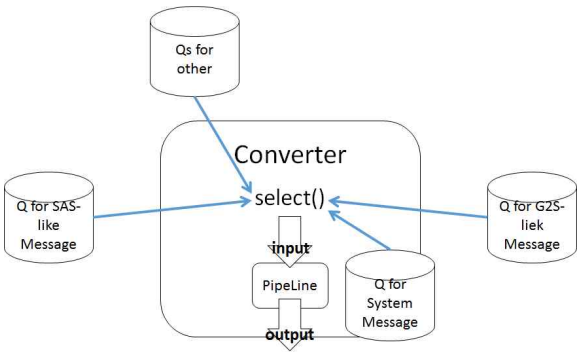

Fig. 10 Event driven structure

컨버터는 유사 G2S 메시지를 생성하기 위해서 G2S의 클래스단위로 SAS의 기능들을 세부적으로 나누어 관리한다. Fig. 11에서 절차적으로 표현한 바와 같이 기능적으로 나누어진 클래스들은

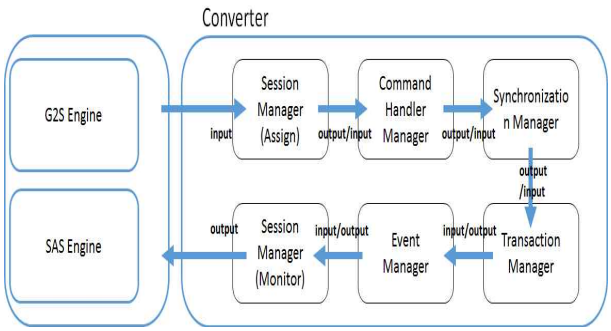

Fig 11 Converter pipeline Structure

트랜잭션, 이벤트, 메터 그리고 동기화과 같은 기능적인 측면으로 다시 나누어지며, 나누어진 세부 기능들은 파이프 모듈들에게 기능별로 분류되어 관리되고 사용된다.

파이프라인은 Fig. 12와 같이 각 클래스에 따라 다르지만 공통적인 절차를 따르는 처리 단계를 정형화하기 위한 구조로 명령어 처리 관리자(command handler manager)모듈, 동기화 관리자(synchronization manager)모듈, 트랜잭션 관리자(transaction Manager) 모듈, 이벤트 관리자(event manager)모듈 그리고 세션 관리자(sessions manager)모듈로 이루어져 있다. 각각은 다음과 같은 세부기능을 담당한다.

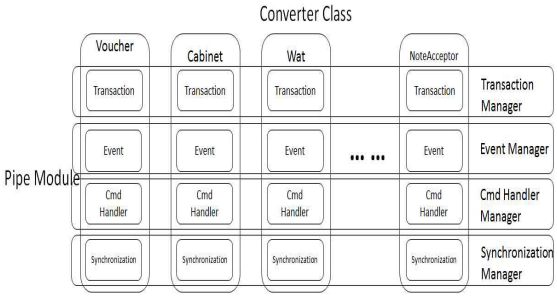

Fig. 12 pipeline modules in the converter

1) 세션 관리자: SAS와 G2S메시지의 변환은 대부분 1:N의 관계로 다수의 SAS메시지가 하나의 G2S 메시지로 변환이 이루어지는 경우가 대부분이다. 세션관리자는 프로토콜 변환과정에서 연속되는 메시지들 중에서 연관된 메시지들에게 하나의 세션식별자를 부여하여 다른 모듈들이 구분하여 처리할 수 있도록 하였다.
2) 동기화 관리자 : 동기화 매니저는 SAS 기반의 게임머신 정보를 읽어와 G2S 클래스 프로파일 형태로 변형하여 저장한다. 이 절차는 SAS 통신상태가 오프-라인에서 온-라인으로 변경되는 매순간 이루어진다. 이는 SAS 통신이 오프-라인이 되었다는 것은 연결된 머신을 다른 것으로 교체하였거나 설정 값을 바꾸었을 가능성을 제기하기 때문이다.
3) 명령어 처리 관리자: SAS와 G2S 엔진으로

부터 전송된 메시지를 처리하고 새로운 메시지를 반환한다. 이 반환되는 메시지는 프로토콜의 변환과정에서 필요한 메시지일 수도 있고 변환이 완료된 메시지일 수도 있다. 이 모듈은 SAS 엔진이나 G2S 엔진으로부터 온 메시지에 대한 처리로 변형된 새로운 메시지 또는 메시지에 대한응답을 생성한다. 이렇게 생성된 메시지는 다음 파이프 모듈에게 사용되거나 필터 될 수 있다.

4) 이벤트 관리자 : 이벤트 관리자는 G2S또는 SAS 엔진과의 상호작용으로 발생한 G2S 이벤트들을 관리하는 역할을 한다. SAS의 엑셉션과 G2S의 가장 큰 차이는 G2S는 이벤트가 발생함에 따른 변경되는 게임머신의 상태 값들을 실시간으로 이벤트와 함께 전송한다는 점이다. 아래는 이벤트 관리자의 SAS의 익셉션을 G2S의 이벤트로 변환하여 반환하는 과정이다.

① SAS 예외이벤트에 입력
② SAS 예외이벤트에 해당하는 G2S 이벤트 코드 찾음
③ G2S의 이벤트 코드에 따라 변경이 되었을 상태정보를 수집하는 SAS 명령어를 반환, 모든 정보가 수집될 때까지 G2S 이벤트 반환을 대기
④ 모든 상태값들이 수집되면 수집된 상태값을 포함한 G2S 이벤트를 반환

5) 트랜잭션 관리자: G2S와 SAS는 돈 거래와 관련된 절차에서 게임머신이 일정양의 거래정보를 유지하고 있기를 요구한다. 그러나 G2S와 비교하여 SAS프로토콜이 지원하는 정보가 제한적이기 때문에 트랜잭션 관리자는 따로 정보를 수집하고 데이터베이스에 저장한다.

4. 시스템 평가

4.1 시스템 동작 테스트

슬롯 프로토콜 변환기를 테스트는 SAS의 6.02을 지원하는 게임머신과 레드블루(RadBlue)에서 제공하는 G2S 테스트 서버인 RGS (RadBlue G2S Scope)를 이용하였다[6]. RGS는 EGM 개발자를 위한 테스트 서버로, G2S 매뉴얼이 언급하고 있는 모든 클래스들을 지원하며 정상 서버와 같이 동작을 한다[5]. 슬롯 프로토콜 변환기는 카지노 객장의 특성상 정지 없이 24시간 동작 가능해야 한다. 이를 시뮬레이션 하기 위해서 슬롯 프로토콜 컨버터는 RGS와 실제 게임머신과 연결하여 객장의 일반적인 객장의 상황을 반영하여 간헐적으로 총 500회 게임을 실시하고 로그값을 확인하였다.

4.2 성능 테스트

하드웨어는 미국 임베디드 커스텀업체인 BoundaryDevice 사의 Nitrogen6_Lite(ARM Solo Processor CortexA9 @ 1Ghz, 514 MB DDR3)를 사용하였다 (Table 8). 메모리의 사용량은 로그파일 크기의 증가에 따른 추가적인 메모리 사용량으로 파악된다. 메시지 처리속도는 호스트의 연속되는 메시지 속에서 슬롯 프로토콜 컨버터의 메시지 처리속도를 측정하였다. 전송하는 메시지의 개수를 10-50개까지 변화를 주었다. 연속되는 메시지 10개를 처리하는 속도는 대략 10초정도 소요되었으며, 메시지의 개수의 따라 정비례하게 시간이 소요되었다 (Fig. 13).

Table 8 Resource usage according to time

	CPU	Memory
idle	3~9%	42MB
message process	30~33%	42MB
busy state	70~80%	42MB
after 24Hours		
idle	3~9%	61MB
message process	30~33%	61MB
busy state	70~80%	61MB

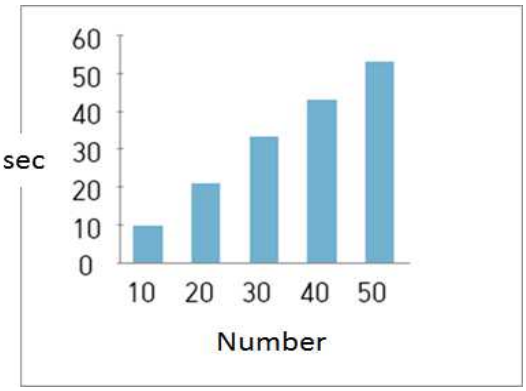

Fig. 13 performance for sequential message processing (with message count)

5. 결 론

본 연구에서는 SAS대G2S 프로토콜 변환 미들웨어를 설계하면서 두 프로토콜이 갖는 차이들을 해결하였다. 소프트웨어 구조는 브리지 패턴으로 프로토콜은 프로토콜을 커넥터 패턴을 바탕으로 하여 설계하였고, 리눅스환경에서 C++와 Python 언어를 사용하여 구현하였다. 1 GHz-512MB 정도의 사양의 임베디드 시스템의 시작품으로 제품화되어, 2015년 각종 정합성 및 안정성에대하여 테스트기관의 시험을 통과하였다 [12].

현재 국내에 다수의 대형 신규 카지노 도입이 예상된다. 인천송도에는 2개 이상의 카지노가 결정되었으며, 제주도에도 중국 관광객들을 위한 카지도의 신규허가가 예상된다. 이러한 상황에서 세수와 사행성을 감독하는 카지노의 슬롯머신 운영 시스템은 100% 외산 제품에 의존하고 있어, 국산화가 시급하다[13].

본 논문에서 컨버터에 필요한 많은 주요문제들을 점을 해결하였고, 상용화작업에 들어갔기는 하지만, 여전히 두 프로토콜의 근본적인 차이에 의하여 지원이 불가능한 부분도 존재한다. 그 예로 슬롯머신의 게임도중 보너스의 개념으로 발생할 수 있는 두 번째 게임(Secondary Game)은 G2S 프로토콜에 의해서 이벤트가 발생하고 매터정보가 추적되지만 SAS 프로토콜은 두 번째 게임에 대해서 전혀 언급하고 있지 않다. 이러한 부분의 추가 연구와 개발이 필요하다.

References

[1] Gaming Standards Association, “Slot Accounting System Protocol-Version 6.02”, Nov. 2005.

[2] Gaming Standards Association, “G2S Message Protocol v2.0.3-Game to System”, 2009.

[3] Python Standard Library, "threading Higher-level threading interface", 2015

[4] Python Standard Library, "ctypes; A foreign function library for Python" 2015

[5] Jake VanderPlas, "Why Python is Slow: Looking Under the Hood", 2015

[6] RadBlue Inc., "G2S : Why a standard protocol really better", 2015

[7] K.-H. Park, S.-J. Yoo, Y. Sohn, W. Lee, S.-Y. Heo, "An home-delivery service management system using OMA DM agent," Journal of the Korea Industrial Information System Society, Vol. 13, No. 2, pp. 8-17, 2008.

[8] K. Jung, J. S. Lee, Y.-W. Kim, "Design and Implementation of tiny UDP/IPv6 Protocols for Sensor Networks", Journal of the Korea Industrial Information System Society, Vol. 13, No. 4, pp. 73-82, 2008.

[9] I.-G. Oh, J.-I Lee, "A Study for Vulnerability of Security of the UPnP Home-Networking", Journal of the Korea Industrial Information System Society, Vol. 12, No. 2, pp. 30-36, 2007.

[10] S.-M. Kim, H. Ahn, "Open source-based G2S (game to system) engine design and implementation", Contemporary Engineering Sciences, Vol. 7, 2014, no. 22, 1171-1179.

[11] Gamma, Erich, et al. Design patterns: elements of reusable object-oriented software. Pearson Education, 1994.

[12] Monterang Inc. "Desgin and Implementation of Casino accounting protocol converting system," Final Report, Small and Medium

Business Adminstration, 2015.
[13] S.-M. Kim, "Design of International Standard Protocol(SAS to G2S) Converter for Slot Machine," Master Thesis, Seoul National University of Science and Technology, Aug. 2015.

김 상 민 (Sangmin Kim)

- 정회원
- 서울과학기술대학교 전기정보공학과 공학사
- 서울과학기술대학교 전기정보공학과 공학석사
- (주) 몽태랑 기술연구소 선임연구원
- 관심분야 : 카지노전산시스템, 임베디드 소프트웨어

박 현 준 (Hyunjoon Park)

- 정회원
- 경희대학교대학교 호텔경영대학 이학사
- (주) 몽태랑 대표이사
- 관심분야: 카지노전산시스템, 소음중화 시스템

안 희 준 (Heejune Ahn)

- 정회원
- KAIST 전기및전자공학과 공학박사
- 서울과학기술대학교 정보통신대학 전기정보공학과 정교수
- 관심분야: 인터넷 컴퓨팅, 임베디드 소프트웨어