Implementing Value-added Application Platform for NGN Server

Chun-hui YUAN*, Yu ZHANG and Xiang-yu LIU

Department of Graduate Management, Space Engineering University, Beijing, China

*Corresponding author

Keywords: NGN, VAP, SIP Server, Media Server, API, APP.

Abstract. The next generation network (NGN) is the core of Soft switch technology, it can provide the network architecture including voice, data, video and multimedia services. This paper presents a design method of Value-added Application Platform (VAP) based on Linux operating system for constructing NGN application server, its core is software. Based on this platform, developers can quickly and in a low cost way to develop a variety of value-added services. As rapid business development platform, VAP encapsulates a highly efficient and flexible layer of C++ API interface, so it is more easily and quickly to develop the server business.

Introduction

An IP Media Server is an important part of the next generation network (NGN) solutions [1]. It is located in the business layer network, providing the required media resources and services for a variety of services in the next generation network. These services are meeting bridge resources, create and control, IVR, recording, various formats for audio and video streaming media, synthesis, transformation, and so on.

At present, most of the IP media servers are based on a dedicated hardware platform, and media services software needs to be run on a specialized DSP hardware. Unable to interconnect, the cost is also high [2]. VAP has a very good solution to this problem, the hardware of the NGN application server based on Intel, using host media process technology for media processing, running on the Linux platform, with IP multimedia processing capabilities and sip call functions. VAP provides all the SIP signaling processing functions, and block the SIP signaling in the processing of detail, ordinary C++ programmers do not need to know the SIP are completed can call control and processing. In this way, the developers just focus on business logic when creating the business design and development. In addition, its flexible networking program allows developers to deploy their applications according to the current status of the network and business needs.

VAP System Structure

VAP is developed for the construction of NGN application server based on Linux operating system. It is a pure software middleware. It provides media processing functions, such as: playback, recording, mixing, generation and detection of DTMF tone, various media format conversion (G.723, G.729, G.711).

Value-added Application Platform has the functions of IP multimedia processing power and SIP call. VAP is embedded a SIP Server for call control and processing base on SIP. As rapid business development platform, VAP encapsulates a highly efficient and flexible layer of C++ API interface, it can easily develop a variety of "media rich" value-added services, it is mainly composed of SDK kernel and service, like the media server and the application server. The system structure is shown in Figure.1:
**System Composition and Logical Relation**

The VAP system consists three parts: SIP Server, Media Server, and API.

SIP server is responsible for handling SIP signaling and managing Media Server resources, as well as providing a C++ API interface for the logical business. Media Server provides voice processing services, such as: IVR, playback, recording, meetings and so on. The logical service handles call logic and applies to the SIP server through the API interface [3]. SIP server accepts Media Server and App business registration, manages Media Server in port mapping and accepts App port request events. As shown in Figure 2.

VAP contains the complete SIP server, Media Server and API programming interfaces. Through the API interface VAP provides a wealth of call control, media processing functions and port registration control functions to the business layer. Port registration and call control are fully open to the business layer by VAP through the API interface. The VAP internal structure diagram is shown in Figure 3.
SIP Server

SIP Server is based on the SIP call control system, through the API layer to the upper layer of the business, providing call events, receiving calls from the business control instructions, and completing the call control function of value-added services. From the business point of view, SIP Server is actually a sip based programmable soft switching platform [4].

SIP Server is responsible for the basic call processing process, and APP Server is responsible for the business logic. SIP server can shield the SIP protocol for App. One under the SIP Server can hang more APP Servers, realization of multiple business call, having different business logics completely separated. App server can be developed according to requirements.

Sip Server based on the soft switching architecture, using SIP as the control protocol. It transfers a voice via RTP, and provides the call control and other functions. It uses control access layer devices to complete the call succeeded, including call control, business, business-exchange, resource management, user authentication, SIP proxy and so on. Also SIP Server as an application server, open to business developer’s application development interface (API). It uses the SIP protocol register method to accept terminal registration. Uses SIP info method to interactive port state, PTT message, VOX message and terminal system, uses the RTP/RTCP protocol [5] to interactive voice media streaming and the terminal system. SIP Server message interaction is shown in Figure.4.

![Figure 4. SIP Server message interaction diagram.](image)

Media Server

Media Server provides recording/playback and conference resources, consisting of a monitor monitoring process, a control signaling control process, and six basic function processes. Monitor process is responsible for the entire system startup monitoring, as well as the allocation of resources, not responsible for business and media processing.

Control process is responsible for the business process, and through the call to complete the basic function of the business process, control process only deal with the signal, all the media flow through this process, the basic functions of the process to complete the process.

There are two FIFO communications between the control process and the process of each function, corresponding to a table, the mode of communication between them is Table+FIFO form.

A FIFO (FIFO1) is responsible for the task allocation process to the basic function of control process, the basic functions of the FIFO1 process monitoring, and processing the message from FIFO1, the results through another FIFO (FIFO2) is sent to the control process.

The basic function of the process from the control process to receive the task assignment, the media stream for the most basic treatment (such as: receive RTP, send RTP, etc.). According to the instructions of the steamtables, the media stream is sent directly to the next basic function process to continue processing, and will be detected by the event fifo2 sent to the control process, the control process for further processing. The architecture of the whole system is shown in Figure.5.
VAP API

VAP contains the complete API programming interface. VAP can provide a wealth of call control, media processing functions, and port registration control function to the business layer through the API interface. Port registration and call control are fully open by VAP through the API interface [6].

API class library provides the main five base classes, they are TPort, TSession, TMediaServer, TConference and TEvtHandler, they are used to build business logic. The library also provides four tools: TPhrase, TPlayctrl, TRecordConfCtrl and TRecordCtrl, the four class is mainly used for construction for IVR voice prompts, radio broadcasting control, conference recording control and recording control.

TSession provides port related services requested, TMediaServer provides media server requests, TConference provides conference requests, TEvtHandler as a bridge between the VAP core and the business event notification, provides event notification trigger for the business logic.

The system uses a single thread asynchronous state machine to realize the mode, business through TEvtHandler derived class to receive the event notification of VAP, after processing, return the results or the behavior to VAP by calling TPort TMediaServer/TConference or its derived class methods, then drive VAP to make the desired action of the business. Similar to the windows platform based on MFC programming model. As follows:

```c
while(!EndProcessFlag){
  if(pEvtHandler->processEvent()==0){
    printf("processEvent()error,Reconnect to SDK, errno = %d\n", errno);
    pEvtHandler->disconnectFromSDK();
    printf("Connect to SDK retry...");
    while(1){
      if(!pEvtHandler->connectToSDK(inet_addr(spIP), spPort)){
        printf(".");
      sleep(2);
      }else{
        printf("\n");
        break;
      }
    }
  }
  printf("
");
}
```

ProcessEvent () is a function of handling events. The business must continue to call the function to ensure that the event can be processed in time, so as not to lose. In this thread that deals with an
asynchronous state machine, all IO operations, must be stripped off and placed in another thread or process.

App Business Operation Mode

The communication between APP businesses and media server VAP kernel is achieved through the class library. Service logic requests VAP kernel services (such as calling, connecting the port to the media server, IVR, voice broadcast, to creating a meeting and so on), by calling the base class library (TPort, TSession, TMediaServer and TConference). The kernel will report the implementation of the service request to the service through the event notification, when an event notification is generated in the VAP kernel, VAP will automatically call the corresponding event handler of the TEvtHandler class.

For example, in the business of a three party conference, it is usually converted from a two party call. When both parties invite third parties to enter into a joint discussion, either party can create a meeting of the three party by specifying the DTMF. At this point, the port that triggers the three party conference will be given a "business session". The port then calls the service request to create the conference. When the conference event is successfully created, the port state is update to "join the conference". This will trigger the port to execute TConference::addPort join the conference. In other words, when you receive a conference event (to create success), locate the port where the request is to be created. Define the following structures.

```c
typedef struct reqid_to_sessionid {
  int reqID;
  SessionID sessionId;
} ;

int setReqidToSessionid(int reqID, const SessionID* pSessionID);
int getSessionidFromReqID(int reqID, SessionID* pSessionID);

CMyPort: trigger port
int reqID=CMyEvtHandler::getReqID();
SessionID sessID;
sessID.id=getID();
sessID.indexInPort=indexInPort;
setReqidToSessionid(reqID,&sessID); // g_mediaServer.createConference(reqID,&g_conference[0], CONFERENCE_TYPE_3WAY,3,3,3600);

Locate the trigger port of the conference in the event handler:
void T3WCEvtHandler::OnConferenceEvt(unsigned int reqID, TConference *conference,EVT_CONF_OPR_PARAM &param){
  switch(param.errorCode){
    case MSERR_NOERROR: {
      switch(param.result){
        case COR_CONF_NEW_OK: {
          SessionID sessID;
          memset(sessID,0,sizeof(SessionID));
          getSessionidFromReqID(reqID, &sessID);
          CMyPort *pPortA = (CMyPort*)getPortByID(sessID.id);
          ...
          break;
        } ...
      }
    }
  }
}
```

It can be concluded that the key to the development of business on the VAP is to understand the work of the API class library, and according to the business needs design the business state machine. By inheriting the API class library and the business needs of the auxiliary class implement businesses. The main body and framework of the business is the business state machine, and it is mainly driven by the TEvtHandler. The operation of the business state machine depends on the implementation of the
TEvtHandler and TSession. The whole business development model is the transaction driven development model in the object-oriented environment.

Conclusions

The next generation network (NGN) is a new public telecommunication network based on IP technology, which integrates voice, data, video and other services. Soft switch technology is the core technology of NGN; VAP draws on the advantages of IP, ATM and TDM to form a layered, fully opened architecture. Its core idea is software, through the software to achieve the original switch control, connection and business processing functions [7] [8]. The entities are connected and communicated through standard protocols. It is convenient to provide business quickly in NGN. It is achieved the integration of network and business, ensuring the openness and compatibility.

As rapid business development platform, VAP encapsulates a highly efficient and flexible layer of C++ API interface, so that the application of APP business development more easily and quickly. Compared with the hardware based IP media server, the IP media processing function provided by VAP is more powerful, and has the advantages of low operating cost, wide coverage, and so on. Therefore, the research and development of the system has great economic benefits and good social benefits.

Reference