showed in figure 2-5.

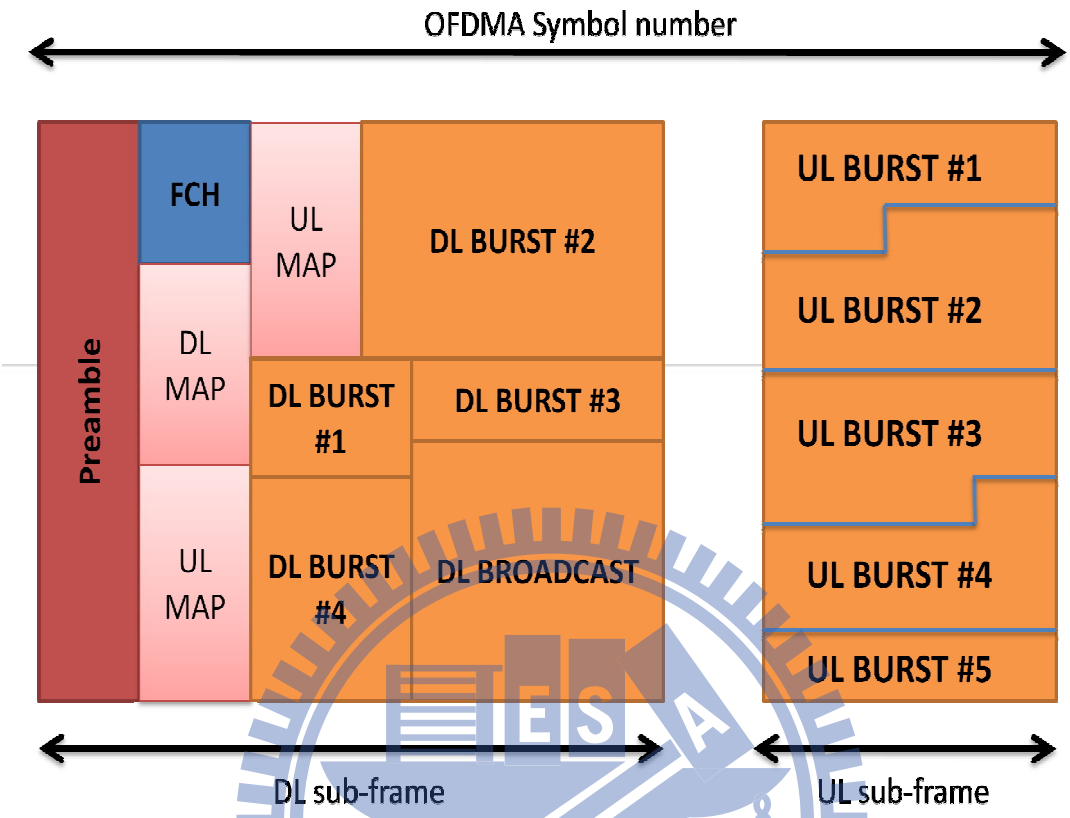

Figure 2-5 OFDMA symbol

2.3.3 Code Modulation And Transmission Rates Of PHY

The WIMAX system supports many kinds of modulation coding scheme. It also uses the channel quality feedback indicator. That is, the SSs can tell the BS about the channel condition, so the BS can estimate the channel to make some adjustment and to use the proper modulation scheme.

The system supports several modulation types : (1) Quadrature Phase Shift Keying (QPSK), (2) 16-state Quadrature Amplitude modulation (16-QAM), (3) and 64-state Quadrature Amplitude modulation (64-QAM). And the system also supports several coding schemes : (1) Convolution Code (CC), (2) Low Density Parity Check Code (LDPC), (3) Block Turbo Code (BTC), and Convolution Turbo Code (CTC).

The above are collected in table 2-2.

Because the WiMAX PHY is very flexible, the transmission rate is different with different parameters and channel conditions.

		DL	UL
Modulation		QPSK,16QAM,64QAM	QPSK,16QAM,64QAM
Code Rate	CC	1/2, 2/3, 3/4, 5/6	1/2, 2/3, 5/6
	CTC	1/2, 2/3, 3/4, 5/6	1/2, 2/3, 5/6
	Repetition	x2, x4, x6	x2, x4, x6

Table 2-2 code modulation

CHAPTER 3

AN INTRODUCTION TO THE EMULATION SYSTEMS

3.1 Introduction to the design background

Verification of the communication protocol and the performance analysis is important for communication system design. If we know the pros and cons of an architecture we design and how it performs on some transmission, we can improve the design. Therefore it really helps a lot to find an effective way to evaluate the system. In this thesis, we propose a new design approach to the evaluation system, or what we call, an "Emulation system".

In previous work, system evaluation is usually done by simulation. The difference between emulation and simulation is that in simulation, we only verify the protocol, or the mathematical model. We don`t really do the transmission. But in our system, we provide a verification system that can test the whole protocol in real environment, which we call the MAC emulator, or MAC emulation system. We run the whole system in real medium instead of merely run the algorithm on the software, such as the Matlab. Therefore, emulation systems have been researched [7] [8].

There is also an emulation system called NS2, but it is based on the network layer. But what we now focus on is the MAC layer. Because no prior arts on MAC emulation system, therefore to develop a useful system architecture and to implement is what we want to do.

Also, previous research on verification is usually on the software level. But in our emulation system, we design the details of the system. That is, simulation system often skips the data plane, but what we do is to

add the data plane. So we can evaluate the system performance more accurate because we run it through real environment not merely channel model.

3.2 A brief introduction of the design concept

In this thesis, we will propose a new design methodology. In the whole design, we will propose the complete architecture and the basic idea. We design the WiMAX system, which is composed of the base station (BS) and many end users, or subscriber stations (SSs). The base station is the centralized controller, and serves many SSs. The whole system is designed in object oriented concept, as showed in figure 3-1.

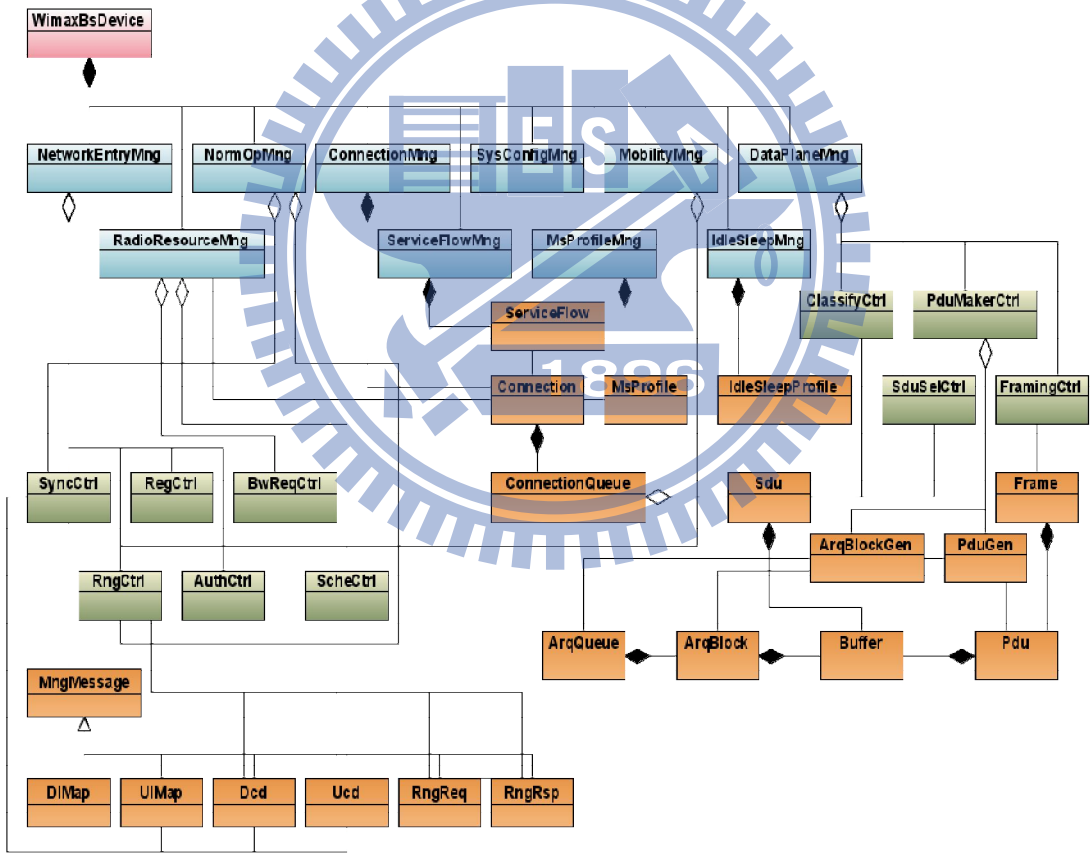

Figure 3-1 Object-Oriented MAC System

In our design, we separate each function unit into different modulus, which would be more easily implemented and maintained. Also, it can achieve more abstraction between layers. In our design, we mainly focus

on the BS side, we provides most of the BS mechanism, such as the data unit handler, and the scheduling unit. In MS side, it only works as a receiver and handles all the packets received and then packing them.

The data flow between the BS and MS is also considered. We don`t implement the channel environment because we use the real medium. The flow between BS and MS is showed in figure 3-2.

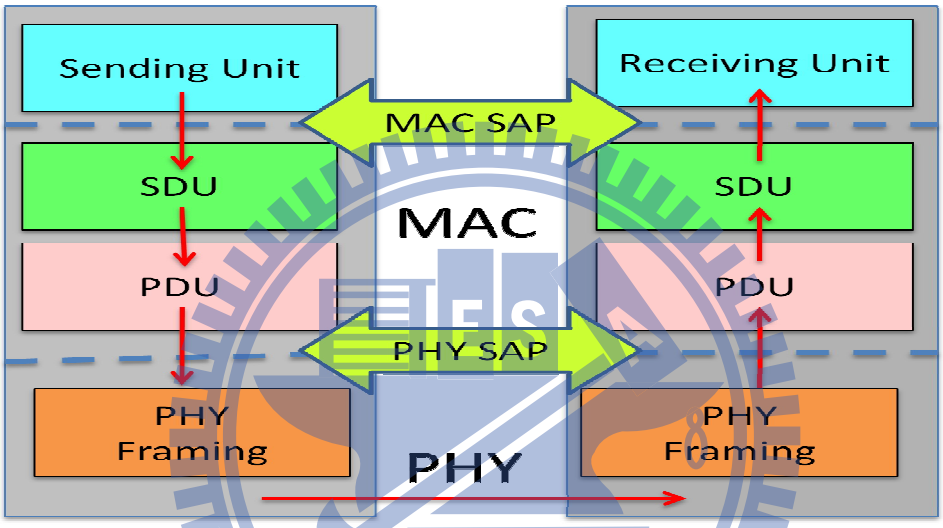

Figure 3-2 The data flow between BS and MS

We only implement the architecture of the MAC layer. Packets from upper layer is called SDU or service data unit, and then it will be processed into PDU, through the fragmentation and packing mechanism introduced in chapter 2. Then the MAC PDU, or MPDU will be transmit to PHY layer.

Other than the architecture, we also implement the system with the C programming language. The details of implementation will be discussed in the next chapter. This chapter only describes the design concept and the architecture. The main architecture of BS and MS is discussed below.

3.3 An introduction to the BS design architecture

In our architecture, we divide the whole BS system into control plane and data plane, where the control plane will process the control signals and data plane only processes data flow. The control plane will also decide how to utilize the radio resource and also how to use it effectively. Also, inside the system is also a traffic generator, which generates packets from the upper layer. Because we only implement MAC layer, upper layer behavior is complete in the traffic generator. The architecture is provided in figure 3-3.

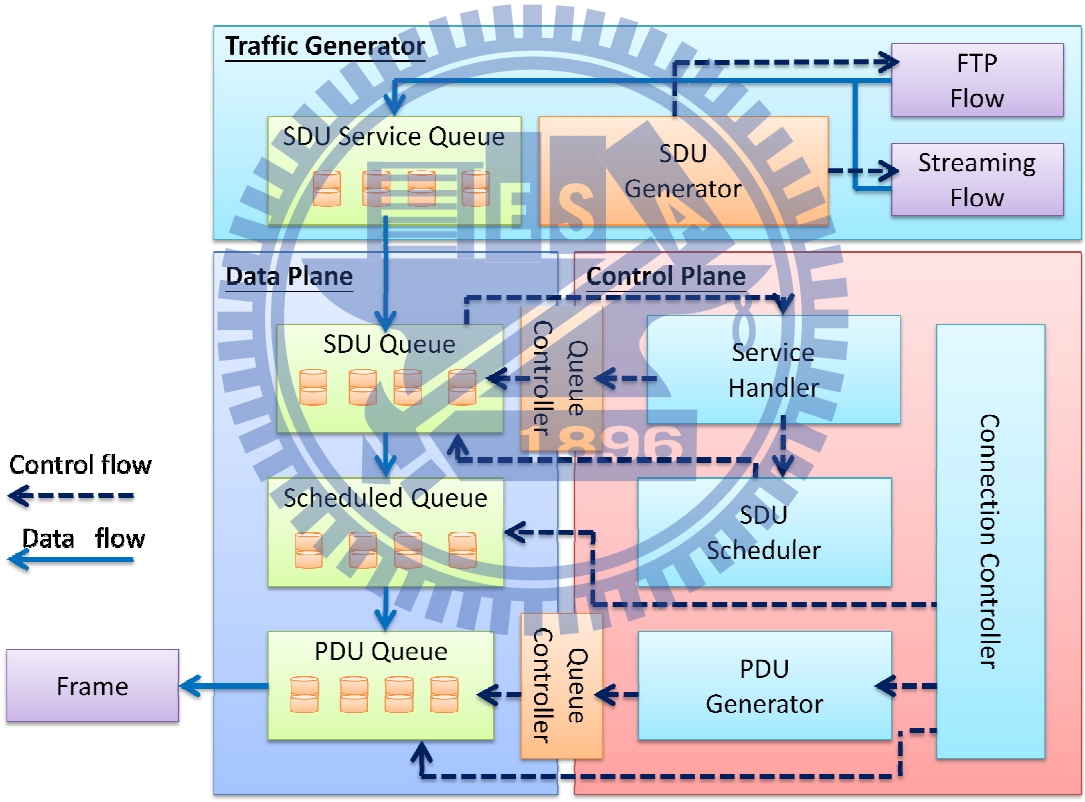

Figure 3-3 The BS architecture

We will introduce the detailed architecture in details in the following subsections.

3.3.1 An introduction to the detail architecture

In this design, we divide the BS system into two parts. The first part

is the traffic generator, which will generate traffic packets. It could generate different types of packets, such as the video streaming traffic, the ftp traffic, the HTTP traffic and the VoIP traffic. But in this thesis, we only implement the video streaming and the ftp traffic. The second part is the main BS system that implements the WiMAX BS protocol. But the traffic generator could be eliminated, because we introduce the event trigger design manner and the multi-thread technique.

In the traffic generator, the SDU generator component uses raw data to generate SDUs and store these packets in the SDU service queue. These packets in the queue will be transmitted to BS. In our system, the data packets are stored using the queue data structure.

A picture of this is showed in figure 3-4.

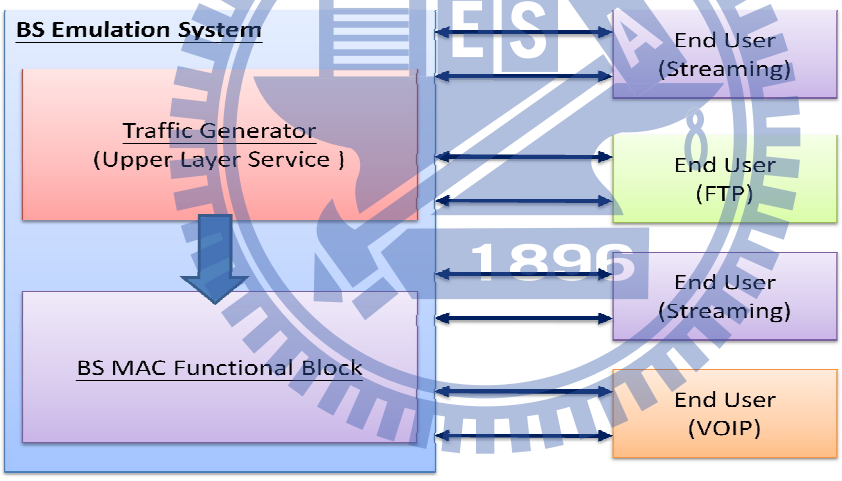

Figure 3-4 the BS Emulator

3.3.2 Description of the traffic generator component

As description above, this component is used to generate traffic flows. In this thesis, we only implement the ftp traffic and the video streaming traffic. The video is sent in a real time manner.

In this component, we input raw data and divide them into SDUs that could act as the upper layer service for MAC. After making them into

SDU packets, we then store in the Queue data structure. We label each SDU with different attributions to classify them, such as the connection ID (CID), the packet length and the traffic type.

And then we send the traffic flow to the BS side using the TCP/IP protocol with inter process communication (IPC). Then the BS module will receive the packets and then process it.

3.3.3 Description of BS module

In this architecture, we main focus on the BS modulus. We introduce the object oriented programming concept. This way the system could be easily maintained and enhanced.

We divide the BS architecture into data plane and control plane. Data plane controls data flow and control plane is responsible for control signal exchange and implement control mechanism, for example, the auto repeat request (ARQ) mechanism, the scheduling mechanism, the fragmentation and packing mechanism, the resource management and the network entry control mechanism.

In the following two subsections, we discuss the architecture we design for the data plane and the control plane.

3.3.4 Description of the data plane

The data plane is responsible of data management. In this design, we store data in the queue data structure. The definition of queue and the operation we can do with it is described as follows. We give a brief introduction of queue because we store most of our data in queue.

A picture of queue is showed in figure 3-5.

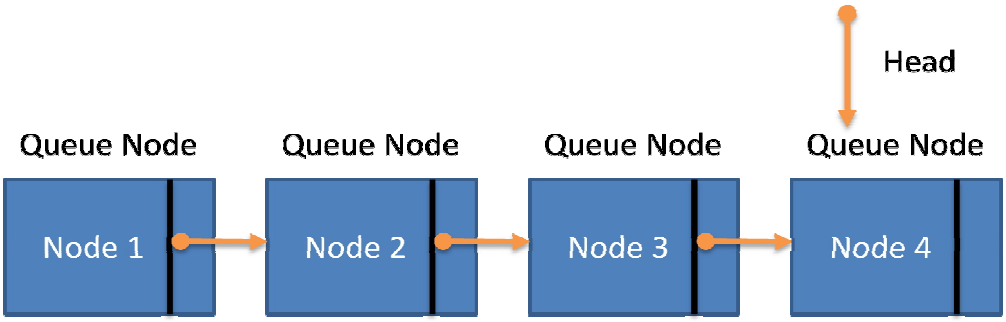

Figure 3-5 The queue data structure

As showed in figure 3-5, the queue may contain several nodes. The queue has a head pointer which points to the starting node of the queue. Each node points to its next. Therefore, once we have the starting point, we can access the whole queue.

Therefore the queue is in first in first out manner (FIFO). The operations that we can do on the queue data structure are to pop elements out the queue, and also to push elements into the queue.

In our system, most of the SDUs and PDUs are stored in the queue structure, in SDU queue and PDU queue separately. In data plane, we mainly focus on the processing of data, for example, the fragmentation and packing of SDUs into PDUs. But the controlling signal of fragmentation and packing is in control plane. The data plane only does the actions instead of deciding the whole idea.

Other than the packing and fragmentation of SDUs, the data plane also generates the frames and put them into the queue. We have three kinds of queues, they are SDU queue, PDU queue and the MAC frame queue.

3.3.5 Description of the control plane

The control plane controls the resource management, the control mechanism, such as scheduling, ARQ and queue management. In our

proposed architecture, there are four functional units, they are the service handler, the SDU handler, the PDU generator and the connection controller. We will introduce each part as follows.

The service handler will manage the traffic data, maintain its information, such as the packet size, the traffic type and the connection ID. It waits for data flow from the upper layer, once service packets come, it constructs them into SDUs and then push them into the SDU queue.

In this architecture, we implement the queue controller that implements the operation on the queue, such as push and pop operation, so that we can add or delete elements.

The SDU scheduler schedules SDU in the SDU queue, we can implement our own algorithm, or use traditional methods such as round robin, first come first serve and early deadline first. In this unit, the scheduler will schedule the queue, and then sorts them. After completing sorting, the sorted SDUs will be pushed into the scheduled queue.

We will always transmit packets from the scheduled queue.

The PDU generator gets elements from the scheduled queue, and then it will make SDUs into PDU based on the channel condition and then decides the appropriate frame size, and then it will consider whether the SDU is too large or too small. If the SDUs are too small and the channel condition is good enough, then the PDU generator will pack these SDUs together. Then the frame will be transmitted to the physical layer by the connection controller described below.

Just as its name, the connection controller will handle connection and maintain the information of connection. It will record the target address, set the connection and then transmit frame.

3.4 An introduction to the MS design architecture

In this thesis, we mainly focus on the BS mechanism, but the subscriber station (SS), or the mobile station (MS) is also implemented as well. We design the MS side to receive data as an end user. And the performance evaluation is done on the MS side.

Because whether an application performs well should be considered from user side. Because we hope not only to analyse the numeric data but also to see how an application works. Therefore beside numeric results, we also introduce real applications.

An illustration of the MS structure is showed in figure 11. A discussion about this structure is as follows.

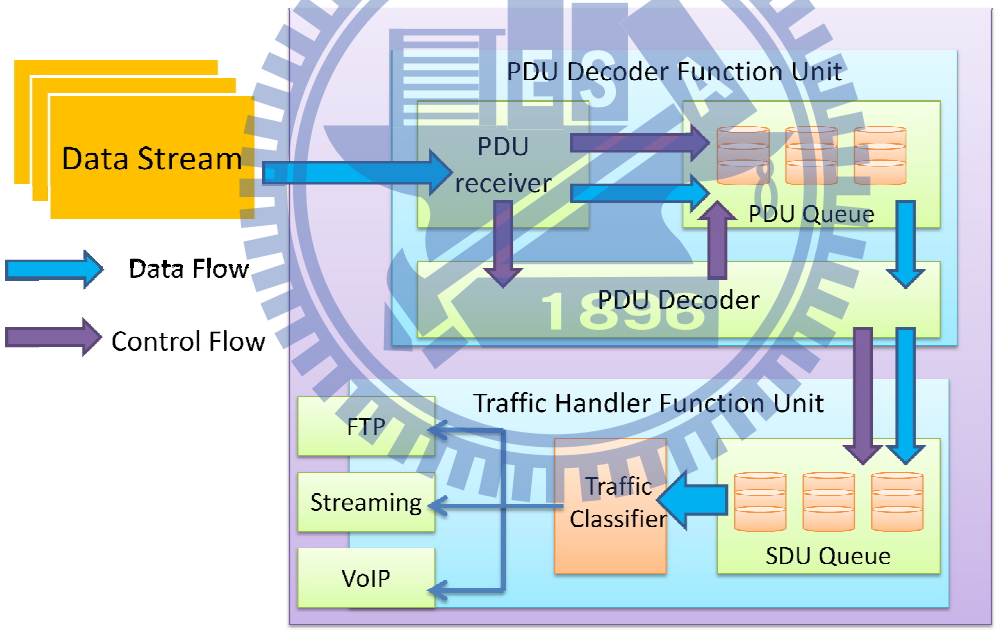

Figure 3-6 the MS architecture

The MS architecture contains several parts. The PDU receiver is a functional unit which receives PDU packets, and then decode them. We decode the header information, and then get the PDU data stream, then make these data into queue structure and push them into the PDU queue.