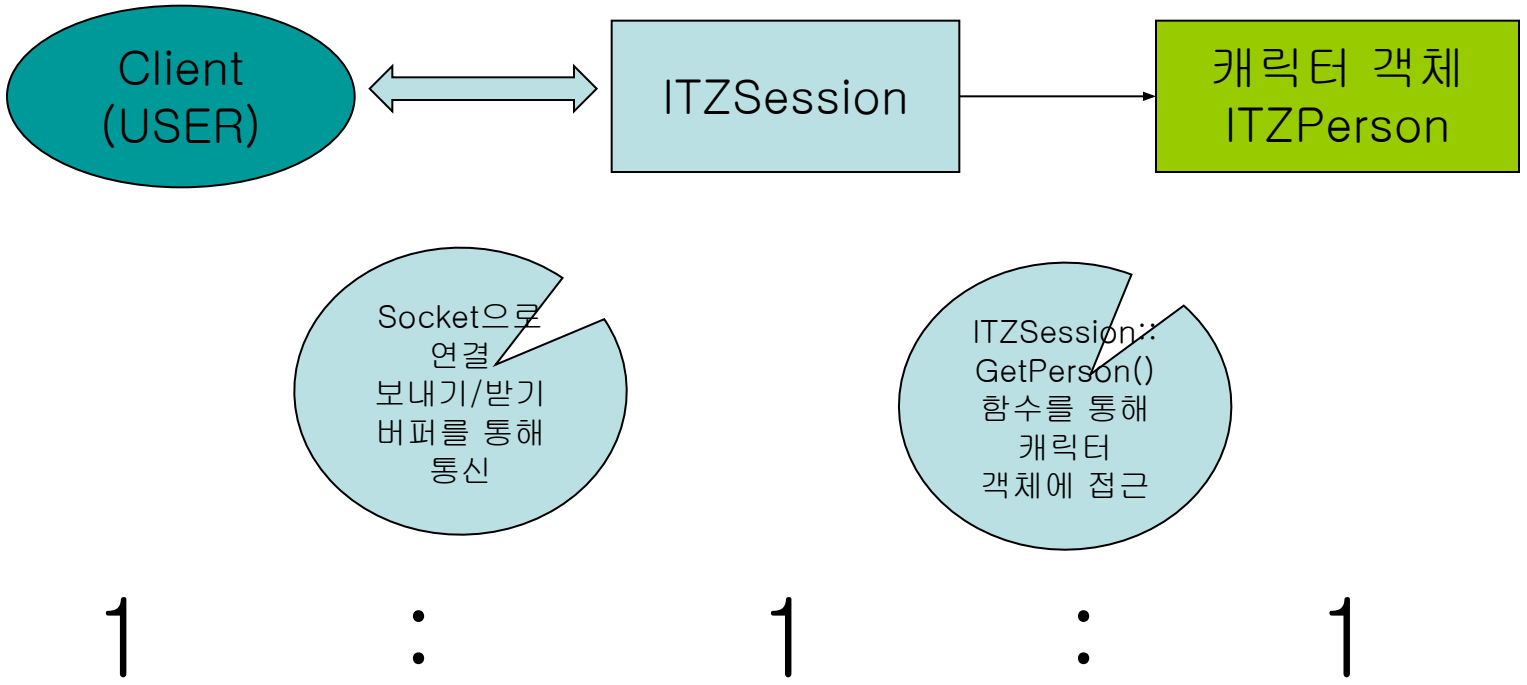
네트웍 라이브러리 설명서

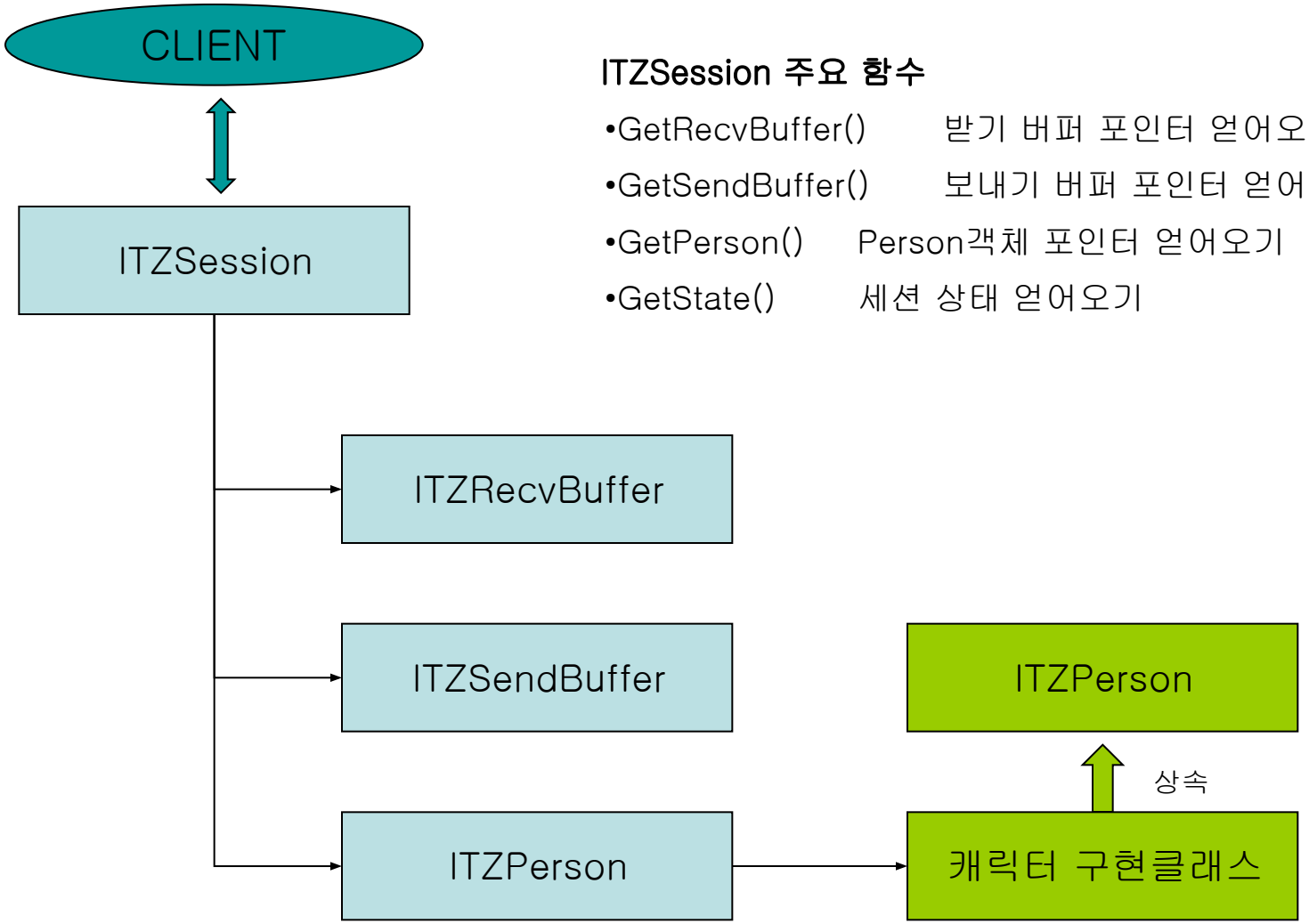
이제완

클라이언트 - 세션 - 캐릭터

라이브러리의 사용에서 ITZSession객체의 포인터만 있으면 송수신 버퍼를 통해 세션 객체와 연결된 클라이언트와 통신할 수 있고, 세션 객체와 연결된 캐릭터 객체에 접근 할 수 있다



ITZSesson 구조



ITZSession 주요 함수

- •GetRecvBuffer() 받기 버퍼 포인터 얻어오기
- •GetSendBuffer() 보내기 버퍼 포인터 얻어오기
- •GetPerson() Person객체 포인터 얻어오기
- •GetState() 세션 상태 얻어오기

SESSION의 상태

Session의 상태는 ITZSession::GetState() 함수를 호출하여 알 수 있다

SESSION_STATE_NONE	세션이 활성화 되지 않음
SESSION_STATE_CONNECTED	세션의 연결이 성공함 패킷 송수신 가능 상태는 아님(단지 연결만…)
SESSION_STATE_ESTABLISHED	세션이 완벽하게 연결됨 정상적인 패킷 송수신 가능 상태
SESSION_STATE_DISONNECTED	접속이 끊어짐 서버의 경우 클라이언트의 로그 아웃 처리가 필요하고, 클라이언트의 경우 서버로의 재접속이 필요한 상태

보내기/받기 버퍼 클래스 멤버함수

ITZRecvBuffer

CreateRecvBuffer(SI32 size) :버퍼 생성
ClearBuffer() :버퍼 비우기
GetRecvParam(…) :iocp호출을 위한 인자 추출
Completion(SI32 siRecvSize) :iocp완료처리

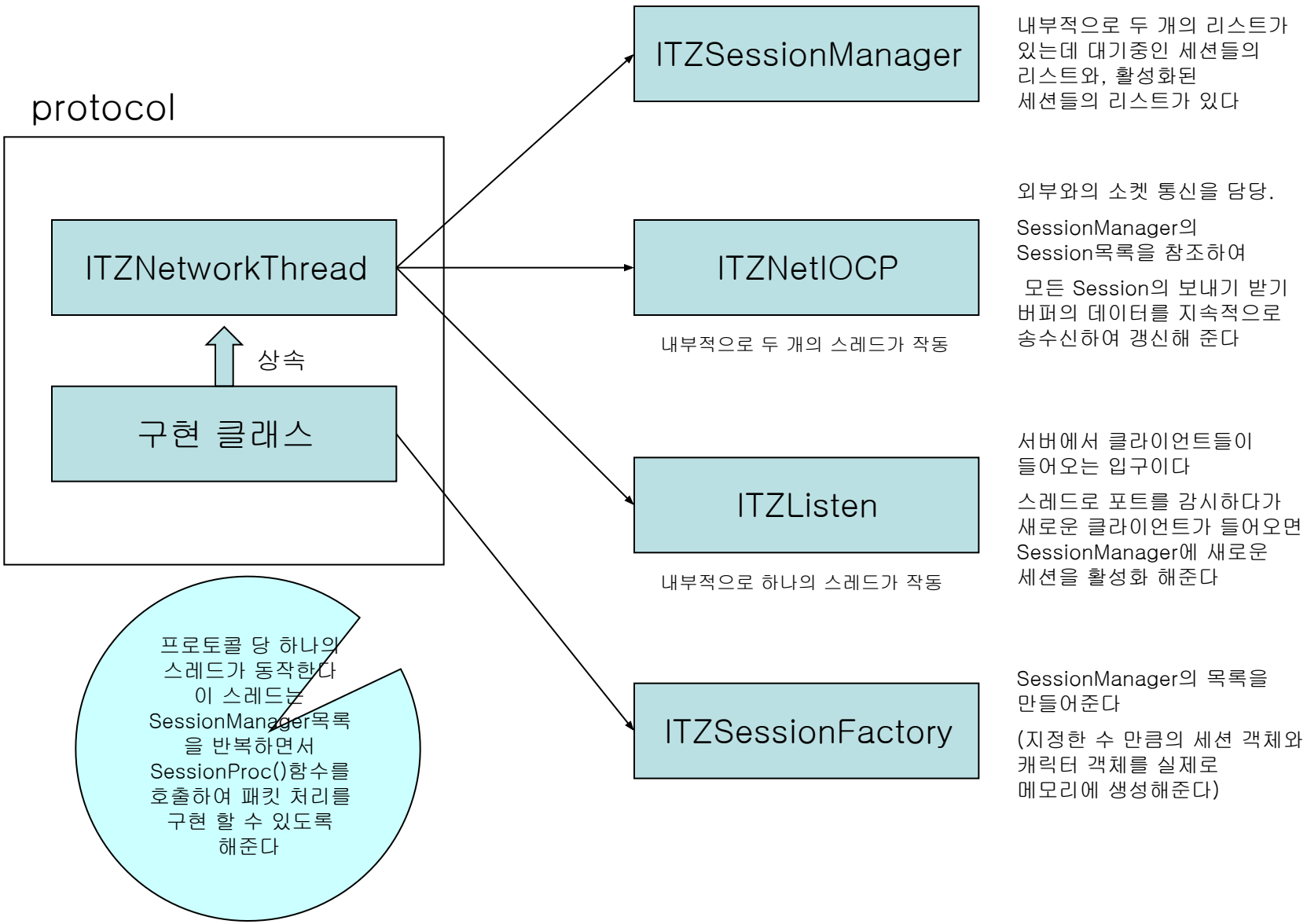
주로 사용하는 함수
GetFirstPacket() :첫번째 패킷 포인터를 얻어온다
(패킷이 하나도 없으면 NULL반환)
RemoveFirstPacket() :첫번째 패킷을 버퍼에서 지운다

ITZSendBuffer

CreateSendBuffer(SI32 size) :버퍼 생성
ClearBuffer() :버퍼 비우기
GetSendParam(…) :iocp호출을 위한 인자 추출
Completion(SI32 siSentBytes) :iocp완료처리
IsReadyToSend() :보낼 데이터가 있는지 여부

주로 사용하는 함수
Write(char *pPacket) :패킷을 버퍼에 쓴다

서버 프로토콜 구성


protocol
ITZNetworkThread
상속
구현 클래스
ITZSessionManager
ITZNetIOCP
내부적으로 두 개의 스레드가 작동
ITZListen
내부적으로 하나의 스레드가 작동
ITZSessionFactory
프로토콜 당 하나의 스레드가 동작한다
이 스레드는 SessionManager목록을 반복하면서 SessionProc()함수를 호출하여 패킷 처리를 구현 할 수 있도록 해준다
Session들의 목록이다
내부적으로 두 개의 리스트가 있는데 대기중인 세션들의 리스트와, 활성화된 세션들의 리스트가 있다
외부와의 소켓 통신을 담당.
SessionManager의 Session목록을 참조하여
모든 Session의 보내기 받기 버퍼의 데이터를 지속적으로 송수신하여 갱신해 준다
서버에서 클라이언트들이 들어오는 입구이다
스레드로 포트를 감시하다가 새로운 클라이언트가 들어오면 SessionManager에 새로운 세션을 활성화 해준다
SessionManager의 목록을 만들어준다
(지정한 수 만큼의 세션 객체와 캐릭터 객체를 실제로 메모리에 생성해준다)

ITZNetworkThread의 스레드 처리 과정

thread

```
while( 클래스가 종료 할 때 까지 반복 ) {

    pClass->Update();
    // 가상 함수 Update()를 호출. 패킷 처리 외에 매 프레임당 처리해야 할 것들을 Update()에서 구현하자


    for( SessionManager의 목록에 등록되어 활성화 된 Session객체 수 만큼 반복 )    // 연결된 모든 세션들을 처리
    {

        if( pSession->GetState() == SESSION_STATE_DISCONNECTED ) {
        // 현재 세션이 클라이언트와 연결이 끊어졌다면,,

        pClass->LogOut( pSession );
        // 가상 함수 LogOut을 현재 세션을 넘겨서 호출한다. (로그아웃처리)

        pClass->GetSessionManager()->DelSession( pSession );
        } else {
        // 현재 세션이 정상적으로 클라이언트와 연결되어 있다면

        ITZNetworkThread->SessionProc( pSession );
        // 가상 함수 SessionProc를 현재 세션 포인터를 넘겨서 호출한다. (패킷처리)

        }
    }
}
```

*** pClass는 스레드가 생성될때 넘겨받은 ITZNetworkThread객체의 포인터**

프로토콜 구현하기I

•하나의 프로토콜이 구동되기 위해서는
ITZNetworkThread를 상속 받은 프로토콜 구현 클래스와, 그 멤버로써
ITZSessionFactory, ITZSessionManager, ITZNetIOCP 객체가 반드시 필요하고,
서버의 경우에는 ITZListen 객체도 필요하다

•ITZNetworkThread 클래스는 기본적으로
ITZSessionManager, ITZNetIOCP, ITZListen 객체를 멤버로 가지고 있다.
따라서 구현 클래스에서 별도로 선언해 줄 필요가 없으며, 객체의 획득은
GetSessionManager(), GetNetIOCP(), GetListen() 함수를 이용한다.
물론 각 객체들의 초기화는 구현 클래스에서 직접 해주어야 한다.

•ITZSessionFactory 객체는 구현 클래스에서 직접 선언하고 초기화해 주어야한다

프로토콜 구현하기II

구현 클래스 작성 - ITZNetworkThread를 상속 받아서 클래스를 작성한다

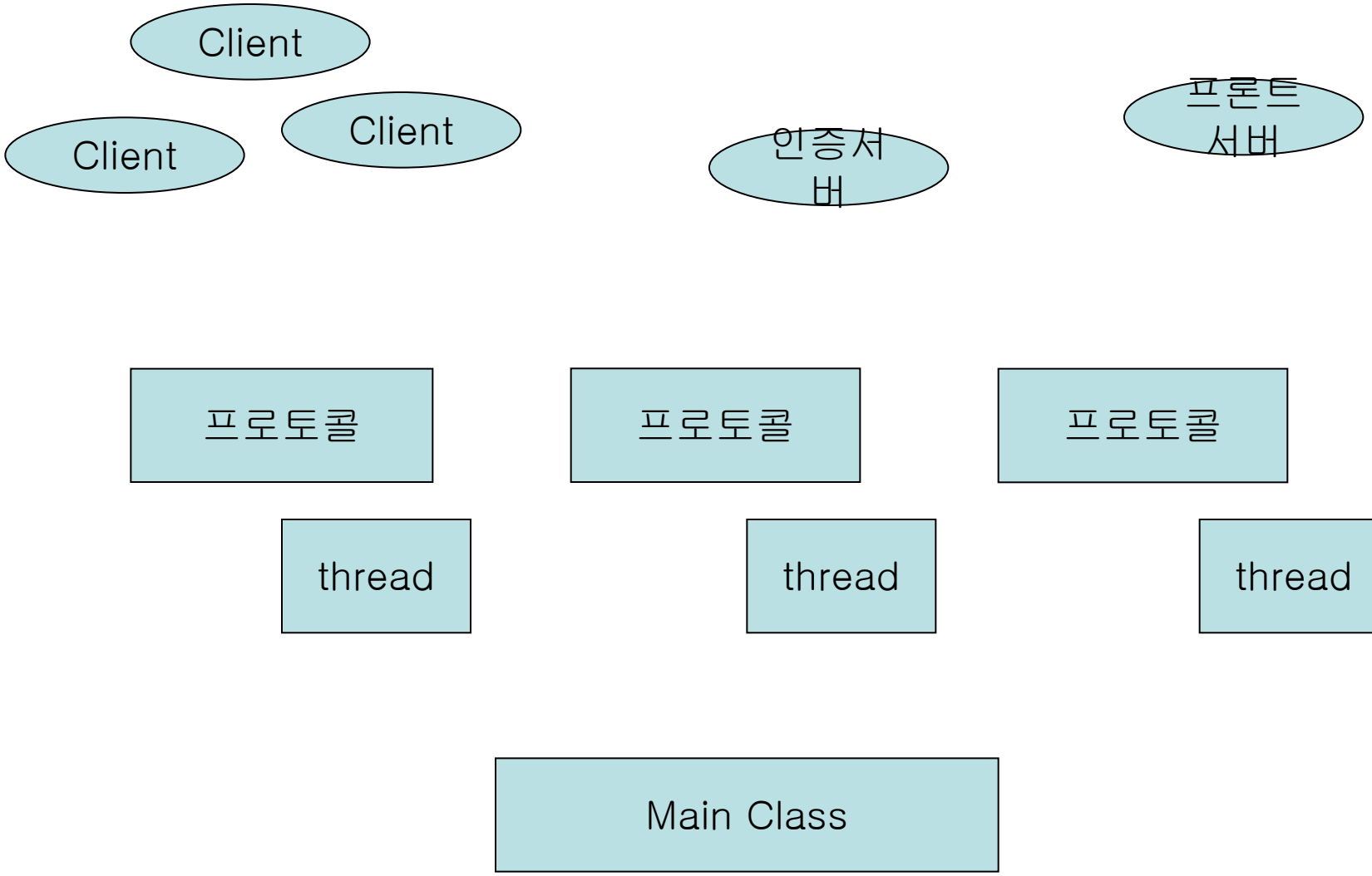
구현 클래스에 멤버로 SessionFactory객체를 생성한다

초기화

생성자 또는 초기화 함수에서 구현해야 할 것 들

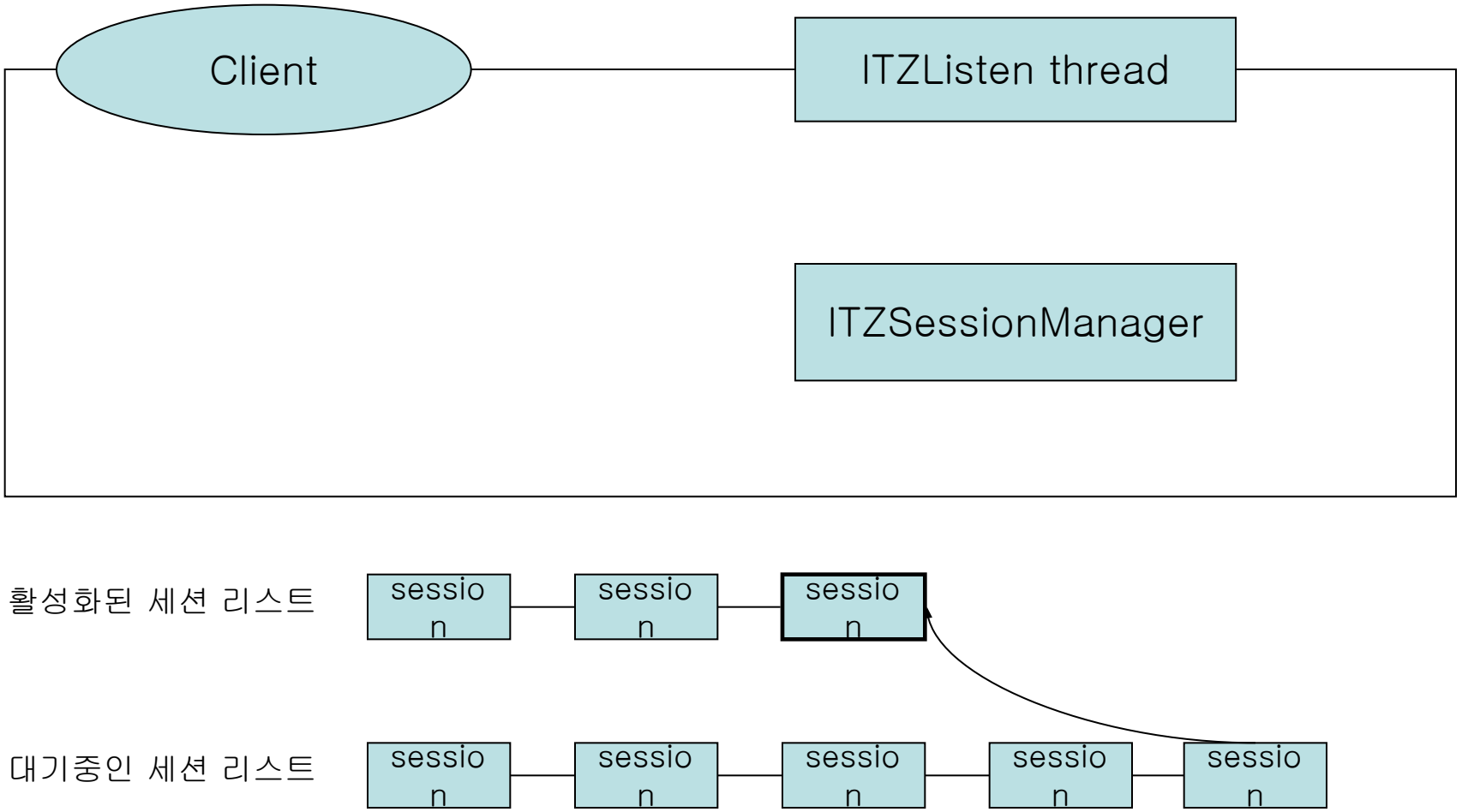
제일 먼저 SessionFactory를 초기화한다.

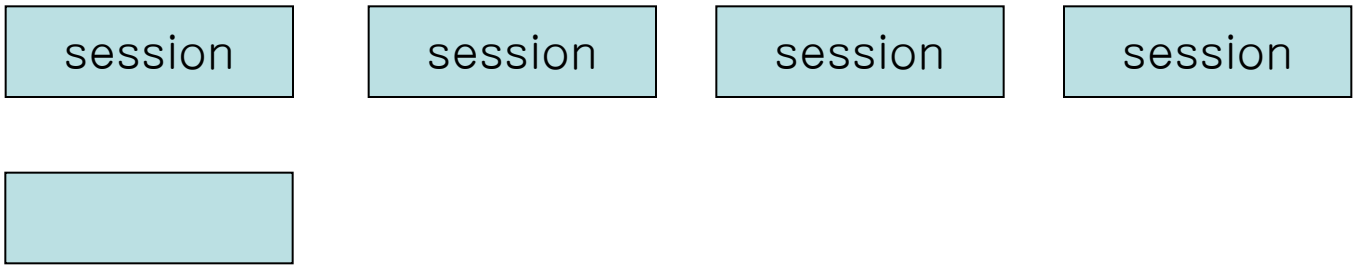
```
m_SF.CreateFactory( GetSessionManager(), 1000, 30000, 30000 );
```


Client
Client
Client
인증서버
프론트서버
프로토콜
프로토콜
프로토콜
thread
thread
thread
Main Class

클라이언트 접속 과정

1. 클라이언트가 서버의 열린 포트로 접속을 시도한다
2. 서버의 ITZListen객체가 클라이언트의 접속을 감지하고


Client
ITZListen thread
ITZSessionManager
활성화된 세션 리스트
session
session
session
대기중인 세션 리스트
session
session
session
session
session


session
session
session
session

SessionFactory가 무슨일을 하는가?

서버와 클라이언트의 세션 활성화