**MFC 서버-클라이언트 프로그램**

**개발 노트**

**<9.25 금>**

1. 1대1 서버-클라이언트 MFC 프로그램 구현

**<9.28 월>**

1. 서버 소켓 Thread 클래스 추가

2, 클라이언트 스레드 버튼, 기능 추가

3. 클라이언트 스레드 정상작동 테스트

4. 클라이언트 스레드를 활용한 send기능 추가

5. 서버 Recv 테스트

6. 리스트에 로그 출력 기능 추가

**<9.29 화>**

1. 서버 스레드를 활용한 send기능 추가

2. 메시지 규칙 적용

3. 클라이언트 Recv 테스트

4. SEQ, ACK 로그 창 분리

5. ACK 생성 조건 적용 및 ACK 전송

**<10.5 월>**

1. 스레딩 속도 증가 시 데이터 결합 현상 해결

- 데이터가 결합되면 SEQ또는ACK 단위로 나누어 처리하는 코드 추가

2. 전송 속도 테스트

- 최대 속도 : 15ms ~ 16ms

3. 클라이언트 독립성 평가

- 하나의 소켓으로만 통신하던 버그 해결

**<10.6 화>**

1. ACK 에러 검사 코드 추가

- Error Log 형식( ACK ERROR : ack=n, recv\_msgs=m)

- 의미 : ack n부터 m-1까지 Error (n<m, n,m은 양의 정수)

2. 에러 발생 시 내부 ack 보정 코드 추가

3. 서버(소켓) 및 클라이언트 생성 초기 SEQ 1~5만큼이 생성되지 않는 문제 해결

- 서버(소켓) 및 클라이언트 최초 생성 후 500ms 이후에 send스레드 동작.

4. n개의 스레드 suspend 이후 Resume에서 n-1개의 스레드만 깨어나는 문제 해결

- loop 수정

**<10월 7일 수>**

1. 디버깅

**<발견된 버그>**

1. 클라이언트 종료 시 서버 쪽 “엑세스 위반” Exception

* m\_ptrclientsocket 해제 후 접근(send) 문제
  + 종료 시 스레드 suspend로 접근 못하게 막음

1. 서버 종료 시 서버 메모리 누수 발생

* CSocketThread, CSendThread, CSocket 메모리 누수

\*\* 클라이언트 종료 시 4, 5번 둘 중 하나 발생

1. 클라이언트 종료 시 클라이언트 쪽 메모리 누수 발생

* Send스레드, socket스레드

1. 클라이언트 종료 시 클라이언트 쪽 “중단점을 트리거 했습니다” Exception

* Second chance assertion failed
* 클라이언트 개수 만큼 발생

**<10월 8일 목>**

1. 디버깅

<버그>

1. 클라이언트 종료 시 CClientSocketThread 100byte 메모리 누수
2. 서버 종료 시 CSocket 56bytes 메모리 누수

* \*m\_pServerSocket 메모리 반납으로 해결

**<10월 12일 월>**

1. 디버깅
   1. 클라이언트 / 서버 종료 시 중단점 트리거 문제
      1. 원인불명
      2. 일부해결, 항상 발생 => 간헐적 발생(빈도 적음)
   2. 스레드 동작 중 Log 리스트 박스 스크롤 시 팅김 현상
      1. 스레드 suspend, resume 버튼 및 기능 추가(suspend 이후 로그 확인)
   3. 클라이언트 종료 시 CClientSocketThread 104byte 메모리 릭
      1. 원인 : socketThread 소멸 안됨, 소멸이 안되는 원인을 모르겠음

**<10월 13일 화>**

1. 비정상적인 수신 상황에 대한 대처 및 ACK검사 방법
   1. 의문1) 데이터가 마치 페이로드처럼 전송되는데 중간 잘릴 수가 있나?
      1. 테스트 결과 끝부분은 잘리더라

**<10월 16일 금>**

1. 데이터가 길어질 때 끝이 깨지는 현상 발견
   1. 전송 메시지의 프로토콜 포맷을 변경하여 깨지는 현상 catch, Log출력
2. ACK에 대해서만 오류를 검사하는 문제
3. 데이터 결합에 따른 정보 손실 문제
   1. 데이터의 중간이 잘려서 다른 데이터랑 붙을 수가 있는지 의문

**<10월 19일 월>**

1. Check\_ack + Check\_seq 단계 추가
   1. 1:0:text:seq\_num:1001:2:0:tex 같이 다음 데이터가 합쳐져서 읽을 경우 (**읽기 오류**)
      1. Seq\_num자리에 아무것도 없으므로 에러 출력
      2. 에러 출력으로 인해 손실된 데이터 감지 가능
   2. 1:0:text:seq\_num:1췤췤췤췤헼 1:0:같이 읽을 경우 (**읽기 오류?, 전송 오류?)**
      1. 전송 데이터 길이 150자 이하에서는 에러 無(200자 이상에서 에러 확인)
         1. 심한 인터럽트가 걸리면 에러 발생
      2. Seq\_num와 1001(SCSA\_num)과 같은 데이터는 key data
      3. Key data부분에 데이터 손실 시 에러 출력
      4. 만약 데이터의 중간((ex)1:0:te--:seq\_num:1001)에도 손실이 일어난다면 에러 감지 불가
   3. 수신이 안된 경우, 전송 실패(**전송 오류**)
      1. TCP/IP 프로토콜에 의해 재전송 될 것임
      2. 단, seq,ack 번호가 일치 하지 않게 됨. 따라서 ERROR 출력
2. ~~에러 상황 해결 알고리즘 필요~~
3. ~~재전송 요청 방법(Server에서 -> Client 경우) (Like Selective Repeat)~~
   1. ~~Server에 Batch parameter 설정~~
      1. ~~Batch마다 Server의 모든 send스레드 정지, Client에게 재전송 시작 메시지 전송~~
         1. ~~Client는 에러 발생한 번호를 알고 있다가 재전송 요청~~
         2. ~~But 재전송 시 데이터는 어떻게 기억?~~
            1. ~~Server에 데이터 Vector 생성~~
            2. ~~Server에서 send하는 모든 데이터를 벡터에 저장~~
            3. ~~batch만큼 보내면 재전송 요청에 해당하는 데이터 전송~~

~~But 또 오류면?~~

~~Client는 에러 발생 번호 리스트에서 정상 재전송된 번호 삭제~~

~~리스트가 모두 비워지면 요청 끝 메시지 전송~~

~~요청 끝 메시지 받으면 send 스레드 재개~~

* ~~전송 속도가 느려짐 ∵ batch마다 스레드 정지 and 재전송 과정~~
* ~~하나라도 재전송이 실패하면 무한 루프의 가능성~~
* ~~모든 error에 대해 재전송 요청이 완료된 시점에 error number가 남아있으면(Server쪽에서 요청 못받음error) 다시 재전송 요청~~

1. ~~Stop and Wait 방법~~

* ~~송신 독립성을 보장할 수 없음(응답에 종속)~~

<10월 20일>

1. ~~재전송 요청 방법 코드화~~

<10월 21일>

1. 에러 해결 알고리즘 필요 없음
2. 파일 저장 기능

<10월 22일 목>

1. 전송 스레드 제어 쪽 코드 수정
   1. Epoch와 출력 주기 설정 오류 현상
   2. Epoch가 0일 때 무한대 설정
2. 저장 단위 설정 기능추가
3. 저장 경로 설정 기능추가
4. 저장 주기 설정 기능추가
5. 파라미터 설정 값, 날짜 저장 기능 추가
6. 서버 UI 업데이트
7. ~~HEAP: Free Heap block --- modified at --- after it was freed 문제 해결~~
8. 클라이언트 UI 업데이트
9. 클라이언트 서버와 같은 기능 추가

<10월 23일 금>

1. 저장 단위가 커지면 프로그램이 다운되는 현상 해결
   1. q접근 크리티컬 섹션 범위 조정
2. 워크 스레드 사용
3. 크리티컬 섹션 사용

<10월 27일 화>

1. 저장경로 설정 오류 수정
2. 파일 다이얼로그로 저장경로 설정 기능 추가
3. 디버깅
   1. ~~클라이언트 종료 시 서버 다운 문제 해결(heap free후 재 접근 문제)~~
      1. 간헐적으로 발생
   2. 클라이언트 종료 시 서버 측 저장 스레드가 소멸하는 문제 해결
   3. 클라이언트 종료 시 서버 측 변수 값 초기화 오류 해결

<10월 28일 수>

1. 16시간 lines=120000(lines), 저장주기=4500(ms) 테스트 결과
   1. 동작 상태 => 정상
   2. 저장 상태 => 정상 (252개, 2.6기가바이트)
   3. CPU 점유율 => 평균 20%
   4. 클라이언트 종료 => 비정상(중단점 트리거, 힙이 손상) => 해결
   5. (서버)연결 해제 시=> 정상
   6. 서버 종료 => 정상
2. 서버, 클라이언트 해제된 힙 접근 문제 해결
3. 클라이언트 send스레드 메모리 릭해결
   1. Sendthread suspend 삭제
4. 클라이언트 CSocketThread 메모리 릭 해결
   1. cThread[i]->PostThreadMessage(WM\_QUIT,0,0); 스레드 종료 메시지
5. 서버 CSocketThread 메모리 릭 해결
6. 클라이언트 종료 시 서버 다운 문제 해결
   1. CMFCServerDlg::OnClientClose 수정
7. ~~저장단위 값 제약조건 설정 (20만 이상 설정 금지)~~
   1. ~~20만 이상에서 프로그램다운 현상(q의 Lock관련) 발생~~
8. 서버CSocket 메모리 릭 해결
   1. delete m\_pServerSocket;
9. 서버, 클라이언트g\_Lock 메모리 릭 해결
10. 테스트

<10월 29일 목>

1. 연결 중에 서버 종료 시 서버 비정상 종료 문제 해결
2. 글자 수 2000개이상(2048bytes) 테스트 (4개의 클라이언트)
   1. ACK는 최대한 짧고 간단하게 송신
      1. 16만 lines
         1. 저장 => 정상
         2. 송수신 => 저장 스레드 동작 시 데이터 손실(저장 스레드 부하 때문인 듯)
      2. 10만 lines
         1. 저장 => 정상
         2. 송수신 => 정상
         3. 저장 단위 limit를 10만으로 조정.
         4. CPU 점유율 평균 20%미만, 저장 시 60%

<10월 30일 금>

1. 테스트
   1. 10 lines, 2000byte 이상(2000자 이상), 16시간 테스트
      1. 송수신 => 정상
      2. 저장 => 정상
      3. 데이터 오류 => 없음
      4. CPU 점유 평균 25%
2. 개발 보고서 작성

<11월 2일 월>

1. 테스트 결과 종합
   1. 저장 공간 부족 예외처리(OS error information : 112) 해결
   2. 프로그램 이상 없음.
2. 개발 보고서 마무리 및 검토
3. 프로젝트 파일 및 보고서 제출

개발 끝.