

實際設計一款 BBS 難度不高，但是在於電腦通訊過程需要處理許多封包錯誤或是例外狀況，且 TCP 需要一邊 send 一邊 recv 才會確保每次收到的詩一個封包，設定過程相當繁瑣。

Sever Side 設計：

初始畫面：直接將目錄 `article.txt` 的內容讀入，做成陣列儲存。如此可以節省每次收到目錄要求時要開啟、讀檔等的繁瑣步驟一次處理，若有變動再重新寫入就好。

```
Connection Successful.  
Article Number: 3  
ARTI: Cat  
ARTI: Rabbit  
ARTI: Fourth  
Waiting...
```

讀取文章列表畫面：收到要求目錄指令(>~\$readArti)時，server 端回傳"Server Start Read Article"表示 server 端已經接收到目錄指令，client 回復 ACK 後，server 回傳文章數量，再收到 client 的 ACK 後，接下來就是 server 端正式回傳目錄。而在初始化時已將檔案列表全部讀入，直接將暫存的檔案回傳即可。

```
Waiting...  
Accepted  
Client IP is : 127.0.0.1  
Receive 11 byte(s): >~$readArti  
Server Start Read Artical  
Article Number: 3  
[1]Cat  
[2]Rabbit  
[3]Fourth
```

選取、回傳文章畫面：使用者回傳文章編號後，在佔存的文章列表找出相對應的標題，並到資料夾 `article` 下搜尋，找到後就將文章回傳，因為 `fgets` 讀到 EOF 時會回傳 NULL 指標，便可利用此作為 loop 迴圈條件，送方看到 EOF 時傳 EOFA(~>\$EOF)作為結束傳輸的訊

息。如此可以不必為每次傳輸都使用 ACK 確認，也沒有長度上限。

```
Waiting...
Accepted
Client IP is : 127.0.0.1
Receive 13 byte(s): >~$selectArti
Wait For Selected Article
Select: "Cat" Open "./article/Cat.txt"
Hello Cat!Waiting...
```

上傳檔案：若收到上傳新文章檔案的指令(>~\$UploadArti)時，回應”Server is Ready to Receive Title”到 client 端，，程式先將新文章標題新增到暫存的文章列表，然後將文章數加一接收 client 端回傳的文章編號便在 article 資料夾下創建檔案(檔名：標題.txt)，創建完後便回應 client ”Start Upload Article”，client 端便開始用迴圈上傳檔案，方式和 server 端讀檔傳送文章給 client 端方式仿若，送方看到 EOF 時傳 EOF(~>\$EOF)作為結束傳輸的訊息。如此可以不必為每次傳輸都使用 ACK 確認，也沒有長度上限。傳送完後就將 Article.txt 複寫更新成暫存的内容。

```
Accepted
Client IP is : 127.0.0.1
Receive 13 byte(s): >~$uploadArti
Server is Ready to Receive Title
./article/Fourth.txt
fileName: Fourth.txt
f: Fourth
title: Fourth
artiList[artiNum]: Fourth
Start Upload Article.
This is Fourth.

End of Uploading
Waiting...
```

Client Side 設計：

初始畫面：簡單歡迎畫面，可選擇 localhost 或是自行輸入 IP

```

Hello!! Wellcome to BBS
-----
Which one do you want to connect
[1]Local Host 127.0.0.1
[2]Type an IP
1
Local Host Address Setted Up.

```

讀取文章列表畫面：當使用者輸入“read”指令時，發出要求目錄指令(>~\$readArti)時，server 端回傳“Server Start Read Article”表示 server 端已經接收到目錄指令並回傳文章數量，接下來 server 端正式回傳目錄。

```

1
Local Host Address Setted Up.
Please Input "read" or "upload" for Reading Articles or Upload an Article
read
Server Start Read Artical, Article Amount: 3
[1]Cat
[2]Rabbit
[3]Fourth

```

選取、回傳文章畫面：使用者輸入並回傳文章編號，server 將文章回傳，Send SA 表示 client 端已送出讀入文章的請求，Uploading 則是表示正在傳輸文件，而迴圈設計方式則相同於 server 端回傳文章的方式，用 EOFA 終止條件。

```

Please Input The Index of Article that Wanted to Read.
1
Read Articl [1]

Send SA
Wait For Selected Article
Hello Cat!

```

上傳檔案：client 傳送指令(>~\$UploadArti)，接下來便要求輸入要傳輸的檔案名，輸入完後 server 便在 article 資料夾下以該名創建檔案並同時作為該文章標題，創建完後便回應“Start Upload Article”，client 端便開始用迴圈上傳檔案，方式和 server 端讀檔傳送文章給 client 端方式仿若，使用 EOFA。

```

Please Input "read" or "upload" for Reading Articles or Upload an Article
upload
Please Input the File Name You Want to Upload.
Fourth.txt
Server is Ready to Receive Title
Start Upload Article.
This is Fourth.

```

遇到的問題：

因為希望收方每次僅收到一種指令內容，以防混亂，所以一般在送出下一個訊息前會先要求讀取接收器緩存中的資料，而一直到下一次傳送資料才讀取緩存中資料是為了避免 recv

一直等到新資料送進來後才會結束動作執行下一行程式，為了節省時間，便放到需要傳下一筆資料時才檢查緩存中是否有未讀的資料，避免消耗過多時間。