Web을 이용한 사용자간 정보공유

이 정 률
jrlee@suwon-c.ac.kr
소속기관 : 충북대학교 전자계산학과 석사과정
주소 : 충청북도 청주시 흥덕구 개신동 충북대학교

Key Word: Push, Socker, Component

요 약

인터넷상에서 사용자가 그룹을 이루어서 같이 정보를 공유하고, 다른 페이지에서 정보를 얻고자 할 때 그룹화된 다른 사용자들도 같은 정보를 서버로부터 제공받을 수 있는 방법을 제시하고자 Push와 Socket, 컴포넌트 기술을 살펴보고 Push와 컴포넌트를 이용하여 사용자들이 그룹을 이루어서 정보를 공유할 수 있는 시스템 설계 및 구현에 대해 설명한다.

현재 인터넷은 대부분의 사람들이 인터넷의 컨텐트(content)를 당겨옴(pulling)으로서 그것에 접근한다. 브라우저를 사용하여 웹을 서핑할 때, 링크를 클릭할 때마다 브라우저는 웹 서버에 링크와 관련된 페이지를 요청(pull)하고, 이를 다운로드 받아 브라우저에서 보여준다.

하지만, push의 경우에 서버는 클라이언트에서 요청이 들어올 때까지 기다리지 않고 클라이언트에서 받아달라고 요청하거나 세팅한 컨텐트가 준비되었을 때, 서버가 그것을 자동적으로 클라이언트에 전달하여 줌으로 사용자는 자신이 원할 때에 자동적으로 전달된 컨텐트를 볼 수 있다.

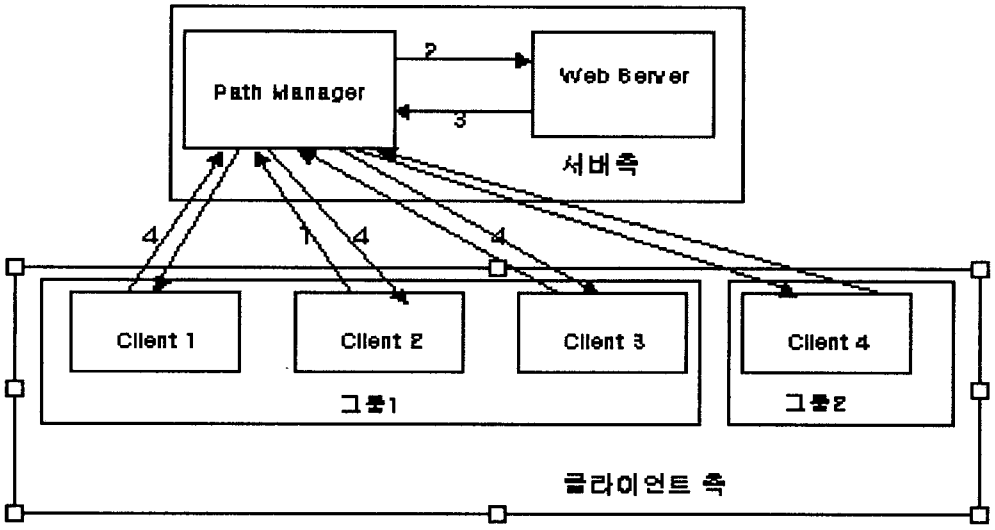
Push의 경우에는 등록된 사용자들에게만 정보가 제공되는 멀티캐스트의 성격을 지니는데 대부분 뉴스/정보를 전달하려는 목적으로 현재 한정되어 제공되고 있으나, 한정적으로 Push 기술을 사용하고 있다. 그러나 서버에서 클라이언트에게 계속해서 데이터를 전송해야함으로 인터넷 자체 트래픽 문제를 유발시키고 있다. 또한, 업체들의 프로그램간에 호환성이 없고 정보를 특정 시간에 전달해주는데 만약 하루에 몇 번씩 자료 전달을 받게 된다면 네트워크에 큰 부담을 준다.

Socket을 이용하여 웹 서버를 구축했을 경우 서버측은 WEB 서버를 통해 클라이언트와 접촉하면서 요구되는 정보를 DB 서버로부터 가져와 클라이언트에게 보내주거나 또는 클라이언트가 보낸 정보를 DB 서버에 저장한다. 이것은 클라이언트들이 정보를 공유하면서 같이 이동하기 위해 필요로 하는 서버의 Method 방식에 따라서 클라이언트 변수 정보를 유지할 수가 있거나 그렇지 못할 수 있다.

컴포넌트 기술은 삽입 컴포넌트(Pluggable Component) 개념을 지원한다. 단일 구조 소프트웨어와 달리, 컴포넌트는 잘 정의된 목적을 가진 소프트웨어이기 때문에 바이너리 수준에서 재사용 가능하고 유지보수 또한 용이하다. 일반적으로 동적 링크 라이브러리(DLL 파일)나 실행 파일(EXE 파일)인 소프트웨어 컴포넌트는 COM 클래스로 만들어지고, 이 클래스의 인스턴스는 COM 객체로 지칭된다. 이 COM을 이용하면 한 컴포넌트에서 소스 코드 없이도 다른 컴포넌트를 재 사용할 수도 있다. COM은 기존의 우수한 몇몇 기술들을 통합한 것이라고 할 수 있다.

기존의 Push 방법은 서버에서 클라이언트에게 계속해서 데이터를 전송해야함으로 인터넷 자체 트래픽 문제를 유발시키고 있고 Socket을 이용한 경우에는 서버의 Method 방식에 따른 클라이

언트 변수 정보 유지가 달라져 문제가 발생하게 된다. 연구에서는 변경된 정보를 등록된 클라이언트에게 보내주는 Push와 Socket의 단점을 보안하기 위한 COM을 이용하여 다수의 클라이언트들이 그룹을 이루어서 정보를 공유하면서 이동하도록 한다.



[그림 1] 서버와 클라이언트의 구성

[그림 1]의 서버 측은 Path Manager와 Web Server로 구성되어 있고 Path Manager에 System Manager Component가 있다. Path Manager는 다수의 클라이언트들이 서로 정보를 공유하면서 다른 페이지로의 이동시 현재 클라이언트들이 서버에 접속한 시점과 그 시점에 해당되는 페이지에 대한 정보 내용들을 저장한다. 또한 서버로 접속하는 클라이언트들 중에 같이 이동하려고 하는 클라이언트들을 그룹화 시킨다. 이때 클라이언트들의 IP 주소들을 이용하여 몇 개로 그룹화된 클라이언트들을 구별한다. 이같은 Path Manager를 통해 클라이언트들이 정보 공유를 하면서 동행할 때, Socket을 사용하였을 때 문제가 되었던 Web Server의 메소드 방식, 즉 Get, Post 방식을 고려하지 않아도 된다. [그림 1]의 서버와 클라이언트의 이벤트 발생 과정을 보면 다음과 같다.

[이벤트 발생 순서]

1.클라이언트는 서버 측의 Path Manager로 접속하여 자신의 변수 정보와 그룹을 등록한다.
2.Path Manager는 클라이언트로부터 온 이벤트를 Web Server로 보낸다.
3.Web Server는 이벤트를 처리한 컨텐츠를 Path Manager에 보낸다.
4.Path Manager는 클라이언트들의 경로 정보, 그룹 정보 등을 이용하여 변경된 컨텐츠를 받고자하는 클라이언트에게 보낸다.

Path Manager는 클라이언트들이 브라우저를 통해 서버로 접속하려고 할 때 객체를 생성하고 서버를 동적으로 활성화한다. 이것은 클라이언트가 서버와 계속적으로 연결하여 변경된 신호를 기다리는 Socket의 문제점을 해결해 준다. 그리고 클라이언트가 계속 접속을 하고 있는지 아니면 접속을 중단하였는지를 알기 위해서 클라이언트와의 연결을 관리한다. Path Manager가 동적으로 활성화되고 객체를 생성하였다면 클라이언트들의 현재 서버의 접속 페이지에 대한 정보를 알아야 한다. Socket을 이용하였을 경우 발생되는 문제점인 Web Server의 메소드 방식을 고려하지 않고

서 클라이언트들의 페이지들에 대한 정보를 얻을 수 있다. 현재 페이지들에 대한 정보는 link_Url을 통한 link selection 변수로 얻을 수 있다. 이것은 Push 와 Socket을 사용했을 때 클라이언트들의 정보변수를 넘겨주는 과정에서 발생되는 서버 메소드 방식에 따른 문제점을 해결한다. 즉 서버가 Get, Post 방식인지 고려하지 않아도 된다는 것이다.

```
When Client try to conneting web server,
{
  if not UnkSite
      Message is "pUnkSite is NULL"
    else
   {
      website_reference = pUnkSite;
      if is website_reference
      {
       Create Window and Show Window
         // Connect to the browser in order to handle events.
         if (!ConnectionManager())
           Message is "Failure sinking events from WebBrowse
       }
   }
}
```

[알고리즘 1] Path Manager의 클라이언트 브라우저 인식 처리

```
When PathManager's Invoke method with Client pages Information parameter
{
  Get Client's WebBrowse LocationURL;
   switch (Client pages Inoformation parameter)
   {
      case link selection
          {
            Get the link_URL and register client page information to Path Manager
            Message is "Before the Navigator"
          }
     case selected page downloaded
          {
            Path Manager send the other clients to selected link page
            Message is "Complete the Navigator"
          }
    }
}
```

[알고리즘 2] 클라이언트의 페이지 정보 처리

본 연구에서는 Push 기술과 COM 기술을 사용해 클라이언트에게 능동적으로 정보를 전달하는 시스템을 설계하였다. 연구에서 제안한 방법은 한 서버에서 클라이언트들이 그룹을 지어서 다른 페이지로 정보를 공유하면서 같이 이동을 할 수 있지만, 그룹 내에 있는 하나의 클라이언트가 다른 서버로 페이지를 이동하였을 때는 같이 동행하고 있던 다른 클라이언트들이 다른 서버로 이동할 수가 없게 된다. 이런 단점을 향후 보안하고, 또한 보안 부분을 고려하여 웹 상에서 클라이언트들이 그룹을 지어서 같이 동행하고자 하는 경우 적용할 수 있을 것이다.