HTML5 기반 하이브리드 SNS 시스템 구현

허태량*, 창지민*, 민유경*, 임승호*
*한국외국어대학교 디지털정보공학과
e-mail:slim@hufs.ac.kr

Implementation of HTML5-based Hybrid SNS Systems

Tae-Ryang Huh*, Ji-Min Chang*, You-Kyung Min*, Seung-Ho Lim*
*Dept of Digital Information Engineering, Hankuk University of Foreign Studies

요 약

네이티브 어플리케이션의 한계를 극복하기 위해 웹 서비스를 이용한 Hybrid Application이 개발되었다. Hybrid Application은 단말기의 센서, 카메라 등 기기의 기능을 Application이 지원하고 내부 구조와 인터페이스, 데이터 전송 등에 Web기술을 이용하여 기존 네이티브 어플리케이션의 단점을 보완하고 장점을 합한 것이다. 웹 기반으로 구현 되었기 때문에 다양한 기기와 해상도에 적용이 가능하고 문제 발생 시 웹과 동일하게 실시간 업데이트가 가능하다. 본 논문에서는 HTML5과 하이브리드 어플리케이션 기반의 SNS 메신저를 설계하고 구현하였다. SNS의 게시판 기능과 메신저를 결합하여 커뮤니케이션을 강조한 어플리케이션을 구현하여, one-source-multi-platform SNS 시스템을 구축하였다.

1. 서론

스마트폰은 다양한 운영체제에 의해서 운영되며, 각 운영체제의 어플리케이션은 안드로이드는 java, IOS는 오브젝트 씨를 이용하여 각자에 맞는 개발환경을 이용하여 구현하였다. 각각 운영체제에 맞게 개발된 이 어플리케이션을 네이티브 어플리케이션이라 부른다. 네이티브 어플리케이션들은 동일한 기능을 여러 운영체제에 구현하는데 시간, 비용 등 많은 자원을 소모한다.

이것을 해결하기 위해 개발된 것이 하이브리드 어플리케이션이다. 하이브리드 어플리케이션은 웹 기술을 통해 내부 구조와 인터페이스를 구성하고 카메라, 단말기의 센서 등 웹으로는 지원하지 않는 단말기의 인터페이스를 사용할 수 있다. HTML과 CSS, 자바스크립트로 구성된 웹 기술을 어플리케이션으로 패키징하여 어플리케이션을 실행하는 형태로 만드는 것이다. 웹 기술을 이용해 하나의 개발 언어로 여러 가지 플랫폼에 적용시키는 One Source Multi-Platform 이라는 장점을 가지고 있다. 별도의 플러그인 없이 플래시나 동영상 등을 구동할 수 있다. 또한 웹 서비스를 기반으로 구현되기 때문에 업데이트와 수정이 실시간으로 가능하다는 장점을 가지고 있다.

One Source Multi-Platform 이라는 장점을 가진 하이브리드 어플리케이션이지만 네이티브 어플리케이션에 비해서 그 속도나 구성의 측면에서 부족하다는 단점이 있다. 해당 운영체제에 맞게 개발된 네이티브 어플리케이션에 비해서 반응속도의 차이가 분명히 존재한다는 것이다.

본 논문에서는 현재 널리 사용되는 스마트폰의 스펙을 기반으로 HTML5 기반의 하이브리드 어플리케이션을 사용할 때의 성능 및 만족도를 알아보도록 한다. 즉, HTML5 기반 하이브리드 SNS-Messenger 시스템을 구현해보고, 사용자수를 증가해 가면서 SNS 어플리케이션 시스템의 수행시간, 통신 시간등을 측정해 봄으로써 하이브리드 어플리케이션의 사용성을 알아보도록 한다.

2. HTML5 기반 하이브리드 SNS 시스템

본 논문에서는 하이브리드 어플리케이션을 이용하여 SNS와 메신저의 특징을 결합한 두가지 기능을 한번에 가지고있는 어플리테이션을 구현하였다.

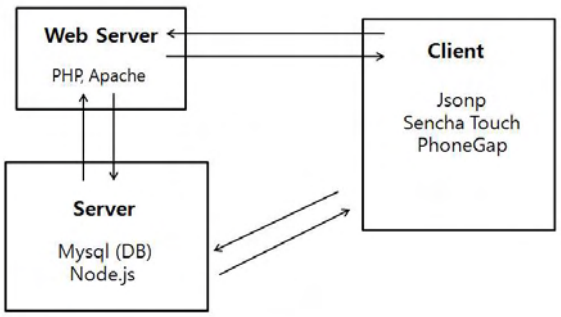

그림 1 구현된 어플리케이션의 구성도

그림1은 구현된 어플리케이션을 구성하고 있는 Frame work과 통신방식을 보여주고 있다. 어플리케이션은 Web framework로 SechaTouch를 사용하였고 PhoneGap을 Hybrid Framework으로 구성했다.

어플리케이션은 클라이언트로서 원하는 기능을 요청하기 위해 JSonp를 이용해 각자 기능을 담당하는 PHP로 데이터를 전송한다. PHP를 이용하기 위해서는 Web Server를 구현해야한다. Web Server는 Apache를 이용해 구현하였다. 서버들은 각 기능에 맞게 구현 되어있는데 어플리케이션은 해당 기능의 서버로 데이터를 전송해야한다. 서버는 전송 받은 데이터를 이용해 사용자를 확인하고 게시판의 글의 양을 조절한다. 그리고 데이터베이스에서 필요한 자료를 가져오거나 검색하여 Callback으로 어플리케이션으로 데이터를 전송한다. 어플리케이션은 받아온 데이터를 이용해 리스트에 출력해준다.

어플리케이션의 채팅통신방식은 Node.js를 이용하였다. 기존의 네트워크 프로그램들은 동기 방식으로 Thread를 여러개 만들어 일을 처리 하기 때문에 쓰레드가 많아질수록 메모리 사용량이 증가하지만 Node.js는 이벤트 기반 비동기 방식으로서 대규모 네트워크 프로그램을 개발하기에 적합하고 쓰레드를 하나만 사용하고 이벤트를 사용하므로 빠른 속도로 일을 처리할 수 있다.

2.1 클라이언트 구조

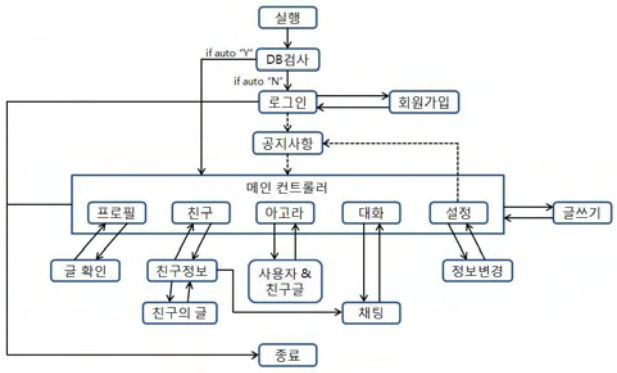

그림 2 어플리케이션의 실행 구성도

그림2는 어플리케이션 클라이언트의 실행 구성을 보여준다. 어플리케이션의 실행 순서로는 어플리케이션을 실행하면 자동로그인을 확인하는 어플리케이션 내의 데이터베이스를 검사하여 자동로그인을 확인한다. 자동 로그인을 하게 되면 로그인 화면을 거치지 않고 메인컨트롤러 화면으로 이동하게 된다. 자동 로그인을 선택하지 않았을 경우에는 로그인창으로 이동할 수 있다. 처음 접속한 사용자는 로그인 화면에서 회원가입으로 이동, 어플리케이션을 사용할 수 있다.

로그인을 하면 메인컨트롤러로 이동하게 되는데 메인컨트롤러는 아래의 Tab의 형태로 프로필. 친구, 아고라, 대화와 설정 창으로 구성되어있다. Tool bar로 어느 페이지에서든 글을 쓸 수 있도록 글쓰기 메뉴를 구현하였다. 글쓰기 메뉴를 선택하면 게시글을 업로드 할 수 있다. 게시글을 갱신하는 Refresh 버튼이 있다. 어플리케이션을 실행한 후 첫 화면인 프로필 메뉴는 회원의 정보와 프로필 사진이 나오는 창과 회원이 작성한 글을 볼 수 있는 창으로 이루어져있다. 작성된 글을 터치하면 해당 글을 볼 수 있는 팝업창이 나타난다.

친구 메뉴는 회원의 프로필 사진, 상태 글 그리고 접속여부를 확인 할 수 있다. 친구의 리스트를 터치하면 친구의 프로필이 나타난다. 친구의 프로필에서 글 목록으로 이동하여 해당 친구가 작성한 글을 확인할 수 있다. 또한 대화하기 버튼으로 친구에게 1:1 대화를 실행 할 수 있다.

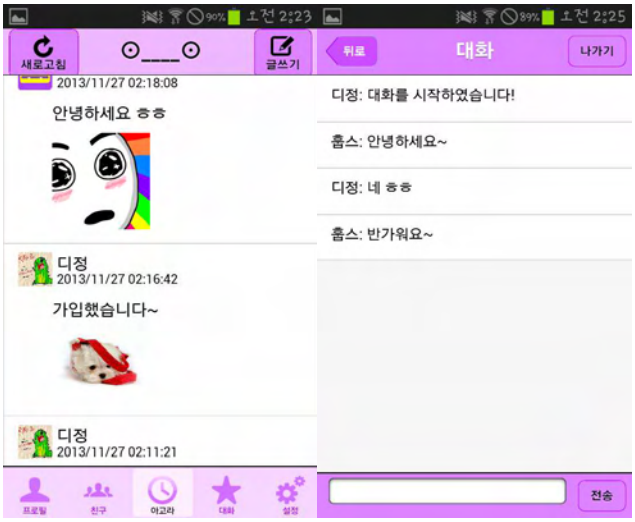

그림 3 게시판 페이지 그림 4 대화창

SNS의 두가지 주요한 기능인 공유 게시판과 Chatting에 대한 구현을 그림 3과 4에 나타내었다. 그림3은 아고라 메뉴로서 아고라는 게시판을 구현한 것으로서 리스트에 제목만 나오고 클릭해서 들어가는 게시판이 아닌 SNS의 출력방식을 사용하였다. 사용자가 업로드한 글뿐만 아니라 사용자와 등록되어 있는 친구들이 업로드한 글까지 볼 수 있는 메뉴이다. 사용자가 글을 업로드 하면 리스트에 자동으로 추가가 된다. 프로필 메뉴와 같이 리스트의 글을 터치하면 해당 글을 볼 수 있는 새로운 화면이 나타난다. Chatting 메뉴는 모든 사용자가 입장 할 수

있는 전체 채팅과 사용자가 이전에 다른 사용자와 채팅을 했던 채팅을 두 가지로 나눈다. 친구 목록에서 1:1 대화를 실행하면 리스트에 추가가 된다. 대화창에 들어가 있지 않고 다른 메뉴를 실행중이라면 Toast 알림이 나타난다. Toast알림에는 답장하기라는 버튼이 있는데 이 기능으로 바로 답장 할 수 있도록 구현하였다.

2.2 서버 구조

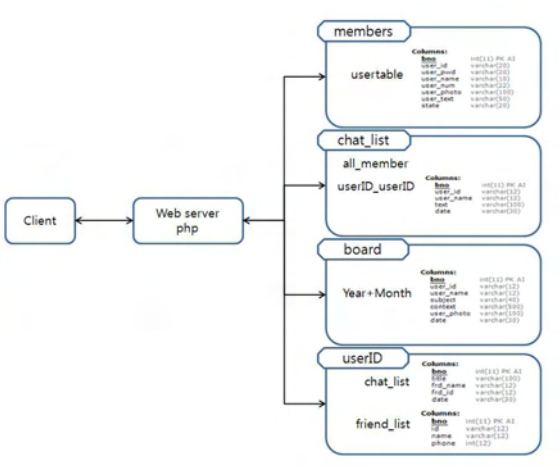

그림 5 데이터베이스의 구성

Server는 Apache로 구현된 Web Server로서 어플리케이션에서 어떤 기능을 수행하고자 할 때 어플리케이션에서 각 기능별 담당하는 부분으로 데이터를 전송하고 그것을 수행한다.

Database는 사용자들의 개인정보를 가지고 있는 Database members에 Table인 usertable, 게시판의 정보를 모아놓는 Database board에 연도와 월의 약자로 이루어진 Table로 이루어져 있다. 게시판에 글을 쓸 때 현재날짜를 읽어 연, 월 별로 정리하는 이유이다. 그리고, 채팅 목록을 모아놓은 Database 로서 chat_list Table이다. 전체채팅인 all_member Table과 1:1 대화의 사용자의 정보를 가지고 있는 Table로 이루어져있다. 마지막으로 UserID로 이루어진 Database는 사용자들이 회원 가입 때 생성되는 개인 Database이다. 사용자가 속해 있는 채팅목록을 가지고 있는 Chat_list와 친구목록의 정보를 가지고 있는 friend_list Table로 이루어져있다. 이와 같은 서버 및 Databae 구성을 통해서 클라이언트의 요청을 처리하고 응답을 주고받는다.

3. 실험

본 논문에서 구현한 하이브리드 어플리케이션의 사용자수 및 사용 콘텐츠에 따른 응답속도를 측정하는 실험을 진행하였다. 직관적인 시간측정을 위해 어플리케이션에서 중요한 두 기능인, 콘텐츠의 관리 및 네트워트 통신에 대한 실험을 통하여 객관적인 실험 측정지표를 알아보았다.

먼저 콘텐츠의 양에 따른 속도의 차이가 얼만큼 차이가 나는지 알아보기 위해 게시판에 출력되는 글의 양을 10개, 50개, 100개, 500개, 1000개로 변화시켜 가면서 응답속도 측정 실험을 하였으며, 두 번째로 네트워크 전송 양에 따른 응답속도를 알아보기 위해서 통신 횟수에 따른 응답 속도를 측정해 보았다. 두 결과를 각각 아래 테이블에 정리하였다.

Contents	10	50	100	500	1000
response time(s)	1.25	1.49	1.73	2.99	5.08

communications	10	50	100	500	1000
response time(s)	0.5	0.65	0.78	1.4	2.2

표 1 하이브리드 어플리케이션 실험 결과

실험결과에서 알 수 있는 바와 같이, 통신에 비해서 콘텐츠를 출력하는데 많은 시간이 소모되고 있으며, 하이브리드 어플리케이션은 UI 처리에 많은 처리 비용이 드는 것을 확인할 수 있다. 이러한 부분들은 웹 표준과 모바일 시스템의 성능 향상에 따라서 향후 개선될 수 있을 것으로 기대된다.

4. 결론

하이브리드 어플이케이션은 다양한 플랫폼에서도 하나의 소스를 이용해 구현가능하다는 장점이 있으나, 성능의 한계를 가지고 있다. 본 논문에서는 SNS Messenger 시스템을 HTML5 기반의 하이브리드 어플리케이션으로 설계하고 구현해보고, 실제 그 성능을 실험적으로 측정해봄으로써 현실적인 상황을 살펴보려고 하였다. 향후, 이러한 하이브리드 어플리케이션이 설계 및 구현, 실험을 바탕으로 다양한 응용 시스템의 적용으로 확장할 예정이다.

참고문헌

[1] HTML5 Instruction - W3Schools, www.s3cshools.com/html/html5_intro.asp

[2] PhoneGap, "PhoneGap License", Phonegap.com, www.phonegap.com
[3] 신용권, "모바일 웹앱: HTML5,센차터치2,제이쿼리,폰갭을 이용한 하이브리드앱 개발", 스마트미디어, May, 2012
[4] Jsonp, "Doug Crockford "Google Tech Talks: JavaScript: The Good Parts"". 7 February 2009.
[5] Apache, http://www.apache.org