

**Open Source SW Project**

**Final Report**

**Team5 : B1G4**

**좌석 혼잡도 기반 경로 추천 서비스**

Submitted By

20162406 이설희

20174470 전희수

20172030 김윤희

20163453 현도연

20161344 허정우

June 22 2019

**목차**

1. 프로젝트 소개
   1. 주제
   2. 필요성
2. 관련 기술, 서비스, 오픈소스 소프트웨어 소개
   1. 차별성
   2. 오픈소스 커뮤니티에 미칠 효과
3. 구현물 설명
   1. 아키텍쳐
   2. Workflow / Usage Scenario
   3. 핵심 기술 설명
      1. 버스 데이터 파싱
      2. 혼잡도 데이터 분석
      3. 버스 경로 추천 알고리즘
      4. 안드로이드
      5. 서버
      6. 지하철 경로 추천 알고리즘
   4. Implementation 이후로 개선된 부분
4. 성능 평가, 비교, 최종 구현물 데모
5. Github repository
   1. 최종 commit 수, contributor 수, issue / pull request 수
   2. 다른 학생들이 등록한 issue, pull request에 대한 평가 (부적절한/불필요한 contribution은 아닌지) + 이들을 어떻게 처리했는지
   3. 이외에 얼마나 Github repository를 잘 활용했는지 보여줄 것
6. 기타
   1. 업무분담
7. **프로젝트 소개**
   1. **주제**

이번 프로젝트의 주제는 ‘혼잡도 기반 경로 추천 서비스’ 이다. 기존의 여러 경로 추천 서비스는 보통 최단거리, 혹은 최단시간 소요의 길을 안내한다. 하지만 일부 사람들의 경우, 한 장소에서 다른 장소로 이동할 때에 최단거리 혹은 최단 시간이 우선순위가 아닌 경우도 있다. 몸이 불편하거나 이동이 불편한 교통약자, 혹은 짐이 많거나 피곤에 지친 사람들의 경우이다. 이들은 단순히 빠르게 이동하기보다는 대중교통에서 앉아갈 수 있기를 우선적으로 바라게 된다. 그런데 현재 혼잡도에 기반하여 최대한 앉아서 이동할 수 있도록 길을 추천해주는 서비스는 존재하지 않는다. (‘차별성’ 에서 기존의 서비스와의 비교를 통해 상세하게 다룰 예정이다) 따라서 이 주제로 해당 프로젝트를 진행하게 되었다. 혼잡도와 환승 횟수, 최대 시간 등을 고려하여 사람들이 최대한 편하게 길을 찾을 수 있도록 도와주고, 이 서비스를 최종적으로는 모듈화하여 다른 프로젝트나 서비스에도 쉽게 추가하고 실행할 수 있도록 구현할 예정이다

* 1. **필요성**

먼저, 신체적으로 활동에 제약을 받는 교통 약자는 대중교통을 이용한 이동이 쉽지 않다. 한국교통장애인 협회의 전국 교통약자 현황 통계자료에 따르면, 장애인 250만명, 65세 이상 노인 650만명, 9세 이하 유아 460만명을 포함하여 임산부, 보행 불편자, 신체 약자 등 전체 인구의 약 33%(1,700만명)에 해당하는 비율이 이동제약자인 것으로 확인되었다. 비록 이동에 어려움이 있는 교통약자라도 대중교통 이용은 불가피하다. 실제로 지역 내 외출 시 주로 이용하는 교통수단 실태 조사 결과 임산부(58.9%)는 버스, 고령자는 지하철(46.4%)의 빈도가 가장 높았고 장애인은 버스(24.6%), 지하철(22.4%)을 비슷한 수준으로 이용함을 확인할 수 있었다.

그리하여 교통약자의 이동권 문제를 해결하기 위해 다양한 서비스 및 지원정책들이 꾸준히 제안되고 있으며 각종 시스템이 등장하였다. 대표적인 몇 가지 예로는 교통약자 승차지원시스템과 버스 예약 어플리케이션이 있다. 교통약자 승차지원시스템은 버스 정류소에 설치된 단말기로부터 버스번호, 교통약자 유형 등의 대기 상황을 입력하면 운전기사가 출입문 개방 및 도착 알림 등을 제공하는 서비스로 현재 도입을 앞두고 있다. MyBus는 “한국교통약자버스이용협동조합”에서 제공하는 버스 예약 어플리케이션으로 장애인들을 대상으로 운영되고 있는 시스템이다. 그럼에도 대중교통 내부의 혼잡도와 관련하여, 최대한 앉아 이동할 수 있도록 경로를 제공하는 서비스는 아직 존재하지 않았다.

뿐만 아니라, 해당 서비스는 교통 약자가 아닌 직장인 또는 대학생의 현대인에게도 꼭 필요하다. 간단히 주변만 둘러보아도 피곤한 몸을 이끌고 혼잡한 대중교통에 몸을 실은 채 언제쯤 자리가 날까 눈치를 보는 수많은 현대인이 흔히 보인다. 뿐만 아니라 짐이 많은 사람, 몸 컨디션이 좋지 않은 사람, 피로에 지친 시민 등 조금이라도 여유 있는 대중교통에서 편안히 이동하고자 하는 모두가 필요로 하는 서비스임을 알 수 있다. 이와 관련한 사용자 니즈를 만족시키고자 2017년 5월부터 서울 시내 버스에서 실시간 버스 차내 혼잡도 안내서비스를 제공하고 있다. 버스 정류소에서, 그리고 어플리케이션에서 실시간 운행되는 버스의 좌석 현황을 확인할 수 있는데 이는 직관적인 혼잡도 이해를 돕지만 실제적으로 혼잡도 고려한 경로추천까지는 이어지지 않는다는 한계가 있었다.

따라서, 해당 프로젝트는 혼잡에 취약한 교통 약자와 지친 현대인에게 여유로운 대중교통에서 되도록 앉아갈 수 있도록 하는 서비스를 제공할 것이다. 이에 추가적으로, 해당 프로젝트가 성공적으로 진행되었을 때 사람들의 이동 경로를 기존의 최단 경로에서 혼잡도 기반 경로로 분산한다면 부분적인 혼잡도 해소까지도 기대할 수 있으므로 더욱 필요성이 부각된다.

1. **관련 기술, 서비스, 오픈소스 소프트웨어 소개**
   1. **차별성**

이미 출시되어 있는 여러 어플리케이션과의 비교를 통해 차별성을 알아보았다

* + 1. 지하철 종결자, 카카오지하철, 또타지하철, 지하철매니저

: 해당 앱들이 제공하는 기능은 대표적으로 실시간 지하철 도착 정보, 하차 알람 기능, 출발시간 기준 / 도착시간 기준 경로 검색, 최소 환승 또는 최소 시간으로 경로를 검색하는 기능이다. 이 경우 대중교통 중 지하철 이용만 고려했다는 점과, 혼잡도를 고려하지 않고 단순히 최소 시간만 생각했다는 점이 특징이다

* + 1. 지하철 혼잡도(바글바글), 지하철혼잡도알리미, R-Subway (<https://mplatform.seoul.go.kr/w/contest/award/2016/wnpz/selectWinner.do>)

: 지하철 혼잡도 통계 값을 제공하는 앱이다. 이 경우 대중교통 중 지하철 이용만 고려했다는 점과, 혼잡도는 제공하지만 구체적인 길 찾기 기능은 제공하지 않는다는 단점이 있다. 또한 앱 자체에 오류가 많다는 후기들이 있었다. R-Subway의 경우 역시 지하철 이용만 고려했지만 부가적으로 통계적인 혼잡도 정보와 역의 사고 여부를 제공한다는 특징이 있다.

* + 1. 전국 스마트 버스, 카카오버스, busconditionServer(<https://github.com/aerain/busconditionServer>)

: 실시간 버스 위치정보와 좌석 혼잡도 현황, 하차 알람 및 일반 경로 찾기를 지원하는 앱이다. 이 경우 대중교통 중 버스 이용만 고려했다는 점과 혼잡도 정보를 제공 만 하고 이를 고려한 경로 추천까지는 해주지 않는다는 한계가 있다.

* + 1. 서울교통포털, 실시간 교통정보, Jamco(<https://github.com/carolrizzi/Jamco>)

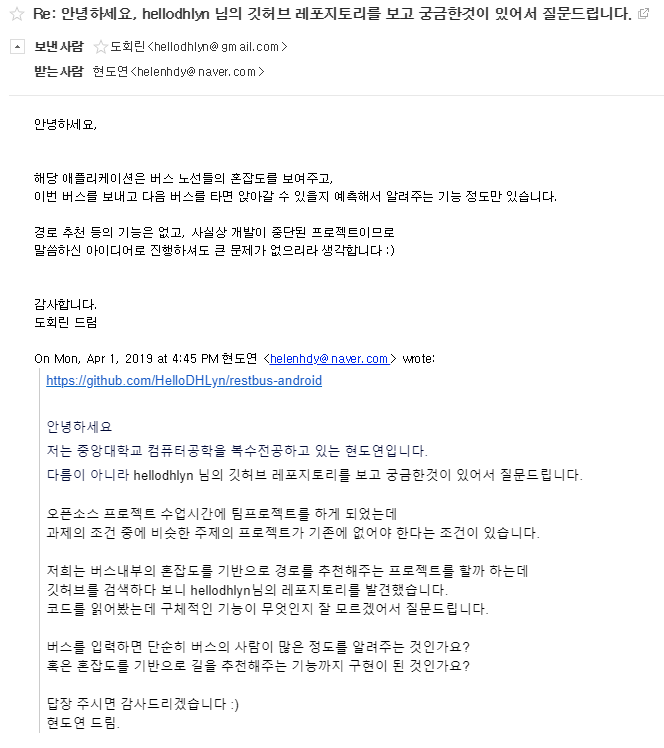
: 주변 지하철 및 정류소를 조회하고, 경로 검색 및 현재 일시 기준 도로통제정보를 확인, 통계 값에 기반한 교통예측정보 확인이 가능한 앱이다. 서울교통포털 앱은 버스와 지하철을 통합하여 길찾기를 최소시간으로 제공하지만 앱 자체가 너무 느리고 오류가 많다는 후기들이 있었다. 또한 실시간 교통정보 앱과 Jamco의 경우 대중교통 사용보다는 자가용 사용에 초점이 맞추어진 도로 별 혼잡도 제공 어플리케이션이었다.

* + 1. 카카오맵, 네이버 지도, 구글 맵 (일반 길찾기 앱)

: 주변 지하철, 정류소 조회 기능과 길찾기를 지하철, 버스, 택시, 도보 등 다양한 경로로 제공하는 앱이다. 버스와 지하철, 도보, 택시 등의 여러 수단을 제공하며 혼잡도 정보는 버스만 제공하고, 경로 찾기는 수단별로 최소 시간 경로들을 제공하였다. 버스 앱과 마찬가지로 혼잡도 정보가 단순히 제공되기만 하고 이를 고려한 경로 탐색은 이루어지지 않았다.

* + 1. Restbus(<https://github.com/HelloDHLyn/restbus-android>)

: github에 있는 해당 프로젝트의 경우, 버스의 혼잡도 정보를 이용한 길찾기 프로젝트였는데 정확히 어떤 기능을 구현하였는지 알 수 없어 해당 레퍼지토리의 소유자에게 이메일을 보내 우리 조의 프로젝트와의 차별성을 확인해 보았다. (자세한 내용은 아래 사진 첨부)



통합해 보았을 때 기존의 여러 앱 중 단순히 최단 시간이 아니라, 최대한 오래 앉아서 편안히 이동할 수 있도록 혼잡도 기반 길 찾기를 제공하는 앱은 없었다. 이 점에 차별성을 두어 본 프로젝트를 진행하고자 한다.

* 1. **오픈소스 커뮤니티에 미칠 효과**

기존의 길찾기 (최단경로 기반)의 경우 다른 여러 프로젝트에 활용할 수 있도록 api가 오픈되어 있다. 마찬가지로 해당 프로젝트 역시 모듈화시켜 이식할 수 있게 제작하고 오픈소스 커뮤니티에 공개한다면, 다른 프로젝트 진행 시에 사람들이 사용할 수 있을 것이다. 단순히 시간을 우선순위로 한 경로들이 아니라, 사용자에 맞게 혼잡도를 우선적인 필터로 고려할 수 있으므로 많은 교통 약자와 피곤에 지친 현대인을 생각해 보았을 때 많은 수요가 예상된다.

1. **구현물 설명**
   1. **아키텍쳐**

ㅇㅇㅇ

* 1. **Workflow / Usage Scenario (사용방법 적어도 좋을 듯)**

ㅇㅇㅇ

* 1. **핵심 기술 설명**
     1. **버스 데이터 파싱**

희수,윤희

* + 1. **혼잡도 데이터 분석**

설희, 도연

* + 1. **버스 경로 추천 알고리즘**

1,2방법대로 데이터를 파싱 하여 우리는 경로 추천 서비스를 위해 필요한 데이터를 모두 얻을 수 있었다. 그렇다면 이제 어플리케이션의 동작 순서에 따라, 어떤 알고리즘을 사용하여 경로를 추천하는지 설명한다.

|  |
| --- |
| [어플리케이션 순서]  1. 출발지와 도착지의 X,Y좌표를 통해 걸어갈 수 있는 거리의 주변 정류소를 API를 통해 찾는다.  2. 출발 정류소와 도착 정류소 사이의 경로를 API를 통해 탐색한다.  3. 나온 모든 경로들 중 최단경로를 우선적으로 저장한다.  4. 실시간 API를 사용한 알고리즘을 통해 앉아갈 수 있는 경로를 저장한다.  5. 최단경로와 앉아갈 수 있는 경로를 모두 사용자에게 보여준다. ( 3번 + 4번) |

1. 출발지와 도착지의 X,Y좌표를 통해 걸어갈 수 있는 거리의 주변 정류소를 API를 통해 찾는다.

어플리케이션을 처음 실행하면, 사용자에게 출발지 및 도착지를 키워드 기반으로 입력 받는다. 그리고 이 입력 받은 정보를 ‘mapAPI’ package안의 searchByAddress 클래스와 SearchByKeyword 클래스를 통해 API를 사용하여 각각 WGS84기준의 좌표로 변환한다. 그 뒤에 사용자가 ‘길찾기’ 버튼을 누르면, 내부적으로 알고리즘이 진행된다.

2. 출발 정류소와 도착 정류소 사이의 경로를 API를 통해 탐색한다.

사용자가 위치한 좌표를 기준으로, 걸어갈 수 있는 거리의 주변 정류소를 찾는다. 700m 를 기준으로 찾으며, 제공되는 API를 사용하였다. 또한, 현재 위치에서 걸어가는 거리를 최소화하기 위해 가장 가까운 정류소를 출발지를 기준으로 최대 3개, 도착지를 기준으로 최대 2개를 찾아 저장한다. 그 이후, 그 정류소들 간의 경로를 모두 찾아낸다. 이 부분은 SearchRoute 클래스를 통해 api를 호출하여 구현했으며, 나온 경로들을 모두 ArrayList<ArrayList<String>> 의 형태로 저장한다. 예를 들어, 출발 정류소 주변에 2개의 정류소가 있고, 도착 정류소 주변에 1개의 정류소가 있을 때, 우리는 2\*1번의 API 호출을 통해 2\* N개보다 더 많은 경로를 모두 찾아낸다. 이 모든 과정들은 searchPath 클래스를 통해 구현한다.

3. 나온 모든 경로들 중 최단경로를 우선적으로 저장한다.

하지만 이 경로들은 모두 최단시간을 기반으로 추천된 경로이며, 우리가 원하는 혼잡도 기반의 경로들이 아니다. 그러므로 이 많은 경로 중 최단경로 하나만을 저장한다. 그리고 나머지 경로중 실시간 API 사용하여 찾아낸 앉아갈 수 있는 경로를 저장한다.

4. 실시간 API를 사용한 알고리즘을 통해 앉아갈 수 있는 경로를 저장한다.

실시간 API를 통해 경로를 얻어내는 방법은 다음과 같다. 나에게 다가오는 버스가 혼잡한지를 판단할 수 있는 기준은 그 버스의 재차인원이다. 즉, 재차인원이 적으면 적을수록, 특히 그 버스의 의자 개수보다 적다면 나는 100%의 확률로 앉을 수 있다. 하지만 현재 내가 찾은 재차인원이, 실제로 버스를 타는 지점까지 도착했을 때 까지 유지된다는 보장이 없다. 즉, 지금 내가 타려는 버스가 나한테 오기까지 몇 명을 더 태우고 몇 명이 하차할지는 지금 알 수 없는 미래의 일이다. 따라서 이 어플리케이션에서는 CurrentLocationXY 클래스를 통해 실시간으로 나의 좌표를 계속 확인한다. RecommendPath클래스의 isNearestStop 메소드를 통해 시간이 흐르다가 내가 환승할 지점 전 정류소에 도착했을 때를 찾는다. 그 때 실시간 api를 사용하여 나에게 다가오는 버스들의 재차인원을 확인한다. 그리고 그 버스가 내가 있는 위치에 도착할 때까지의 재차인원 변화 추이를 앞에서 계산한 혼잡도 통계치를 사용하여 예측한다. 이 과정을 통해 사용자가 앉을 수 있는 가능성이 가장 큰 버스를 추천한다.

5. 최단경로와 앉아갈 수 있는 경로를 모두 사용자에게 보여준다. ( 3번 + 4번)

처음에 추천된 경로를 통해 사용자가 앉았고, 앞으로도 환승이 없는 경우에는 어플리케이션의 목적을 달성한 샘이므로, 종료한다. 하지만, 앉음 유무와 상관없이 환승이 있는 경우에는, 원래의 환승 지점보다 더 혼잡도가 낮은 환승 지점을 추천하여 사용자가 앉을 확률을 높여주어야 한다. 이 때, 우리는 RecommendPath클래스의 calcTotalCongestionInPath 메소드를 통해 기존의 환승 지점을 포함하여 사용자가 이동할 전 정류장 중 통계 혼잡도가 가장 낮은 환승 지점을 탐색한다. 통계 혼잡도는 앞서 설명한 혼잡도 데이터를 이용한다. 그렇게 혼잡도가 가장 낮은 지점을 탐색하면 앞서 사용한(4번) 실시간 API 알고리즘을 통해 경로를 찾고 그 경로를 실시간으로 사용자에게 추천하여 더 나은 환승을 유도한다.

* + 1. **안드로이드**

희수, 설희

* + 1. **서버**

정우,도연,윤희

* + 1. **지하철 경로 추천 알고리즘**

정우

* 1. **Implementation 이후로 개선된 부분**

서버 //도연언니 부분 이어서

3. 안드로이드

안드로이드에서는 TCPClient라는 클래스를 통해 서버와 통신한다. 먼저, 사용자에게 입력받은 좌표를 서버에게 전달하고, 그에 상응하는 결과값을 받는다. 결과값이 하나의 긴 string값인 것을 고려하여 TCPClient클래스 안에 convertMsg 라는 메소드를 구현하였다. 이 메소드는 서버로부터 전달받은 결과값을 사용이 용이하게 가공하는 역할을 맡는다. 변환한 결과값은 ResultOfServer라는 클래스 안에 저장한다. 이 클래스안에는 경로에 대한 결과값이 두가지로 나누어서 저장되는데, 각각 환승 지점, 소요시간, 출발 및 도착지를 저장한 arraylist와 경유 정류장 전부를 저장한 arraylist이다. 이 클래스를 view와 연결하여 사용자에게 보여준다.

1. **성능 평가, 비교, 최종 구현물 데모 (스크린샷같은것도 넣자)**

ㅇㅇㅇㅇ

1. **Github repository**
   1. **최종 commit 수, contributor 수, issue / pull request 수**

Data parsing repository

126 commits, 5 contributors (team member), no issue and pull request

Application repository

175 commits, 8 contributors, 5 issues, 11 pull requests

* 1. **다른 학생들이 등록한 issue, pull request에 대한 평가 (부적절한/불필요한 contribution은 아닌지) + 이들을 어떻게 처리했는지**

|  |
| --- |
| Issu2 1번과 3번을 등록한 학우가 2번 Pull Request를 해주었다.  <Issue>위치 권한 미설정 시 앱 튕김 현상 #1([**https://github.com/b1g4/Application/issues/1**](https://github.com/b1g4/Application/issues/1) **)**  <Issue> 출발지 미 기입시 튕김 현상 #3 (<https://github.com/b1g4/Application/issues/3> )  <Pull Request> 위치 권한 부여, 출발지 검색 오류 처리 #2 (<https://github.com/b1g4/Application/pull/2> )  위치 권한 설정을 하지 않아 사용자가 위치 권한을 거부하면 앱 튕김 현상이 일어나고 출발지를 기입하지 않고 길 찾기 버튼을 누르면 앱이 종료되는 버그가 있었다. 이를 학우분께서 몇 줄의 코드를 추가해 해결해 주었고 이를 merge하여 Android를 개선했다. |
| <Issue> 최근 검색 목록 기능 추가 #4 (<https://github.com/b1g4/Application/issues/4> )  아래 pull request와 같이 설명 |
| <Issue> Add the keyword function #7 (<https://github.com/b1g4/Application/issues/7> ) |
| <Issue> 주소 검색 시 지역 범위에 대한 범위 설정 #9 (<https://github.com/b1g4/Application/issues/9> )  ~~기존 Proposal에는 길 찾기 개발 범위를 서울 시내로 한정했다. 하지만 구현하는 과정에서 미처 개발을 하지 못했다. 이는 추후에 수정할 계획이다.~~ |
| <Pull Request> 최근 검색 목록 기능 추가 #5 ( https://github.com/b1g4/Application/pull/5 )  안드로이드의 클라이언트 부분의 기능 중, 최근 검색 목록 기능을 추가하고자 하는 pull request를 확인하였다. 이는, 우리 어플리케이션의 특성상, 사용자가 검색했던 경로를 또 검색하는 경우가 적지 않게 있을 수 있고, 자주 가는 장소에 대해 시각화 해주는 이로운 기능이라고 판단하여 코드 리뷰 및 merge를 진행하였다. 코드를 직접 clone하여 확인했을 때 별다른 이상이 없었으나, 마스터 브랜치로 바로 merge하기 전에 ‘pull requests’ 라는 새로운 브랜치로 안전하게 merge하고자 pull request 수정을 요청하였고, 이후 merge를 완료했다. |
| <Pull Request> 즐겨찾기 기능 추가 #6 (<https://github.com/b1g4/Application/pull/6> ) |
| <Pull Request> Find the Best path by using keyword. #8 (<https://github.com/b1g4/Application/pull/8> ) |
| <Pull Request> find thd best spot by using keyword #10 (<https://github.com/b1g4/Application/pull/10> ) |
| <Pull Request> Revert "find thd best spot by using keyword" #11 (<https://github.com/b1g4/Application/pull/11> ) |
| 풀리퀘 받은 사람이 알아서 링크 넣고 설명달고 할 것!!!!!!!!!  옮기다가 귀찮아서 관뒀으니까 깃허브 필수로 확인할 것!!!!!!!!!! |

* 1. **이외에 얼마나 Github repository를 잘 활용했는지 보여줄 것**

ㅇㅇㅇ

1. **기타**
   1. **업무분담**

기본적으로 모든 업무는 팀원이 함께 진행하며, 진행할 때 메인으로 맡는 사람을 정해 분담하였다.

|  |  |
| --- | --- |
| **역할** | **팀원** |
| 팀장 | 이설희 |
| 개발자 | 데이터 파싱 - 김윤희, 전희수  데이터 분석 - 이설희, 현도연  버스경로 구현- 김윤희, 허정우, 현도연  지하철 구현 - 허정우  안드로이드 구현 - 전희수, 이설희  서버구축 - 김윤희, 현도연, 허정우 |
| 문서작업 | Proposal report- 전희수(메인), 김윤희, 이설희, 허정우, 현도연  Final report - 김윤희, 이설희. 전희수, 허정우, 현도연 |
| 발표자 | Proposal - 전희수  Final - 김윤희 |

3-a 현도연

3-b 이설희

4 허정우

5-a 김윤희

5-c 전희수

