2012 年度　卒業論文
「青春１８きっぷの旅を支援する電子時刻表の制作」

担当教員　渡辺　恭人
学籍番号　０９４０１７８
中村　勝久

提出日：2013 年 1 月 24 日

目次

１．背景・目的

１－１背景

JR 線の普通列車が一日乗り放題となる「青春１８きっぷ」というものがある。これを利用して短から中期間の旅行に出かける者は多く、毎年約 50 万枚の売り上げ実績があるようだ。（注 1 http://seisyun.tabiris.com/uriage.html 2012 年 10 月 29 日）購入者の中には、「時刻表とにらめっこをしながらプランを練るのも旅の醍醐味だ」と思う者が多いように感じるが、他方で、旅慣れていない者の中にはそういった作業が苦手、あるいは嫌いだという層も一定数いると思われる。筆者個人の意見としては、あの大きくて分厚い時刻表を旅先で利用すること自体、一苦労だと感じてしまう。時刻表を用いたアナログな調べ方が苦手という人の為に、ネット上でも様々なサービスが展開されているが、そういった物の多くは「乗り降りする駅やその時間が明確に決まっている人の手助けをする」というのが本来の用途であるため、長距離を移動し、その所々で観光をするような旅行には向かない。また、現地の人々との触れ合いや、そこから得た情報を元に観光や食事を楽しむという旅行の醍醐味を活かすためにも、すべての行程を機械任せにした旅ではなく、ある程度の「行き当たりばったり性」を残した旅のスタイルを提案したい。

１－２　目的

国内旅行に出かける時、何よりも速さを重視するのであれば「飛行機」を利用する事が多いだろう。仮に千葉から名古屋までの旅行の場合、千葉県の成田国際から愛知県の中部国際空港までを 1 時間 10 分（注 2)という速さで移動することができる。しかしその分費用も掛かるのが飛行機であり、前述のチケットの場合、往復割引を適用しても片道 16400 円（注 2)を支払うことになる。飛行機以外で速さ重視の旅をするのであれば「新幹線」が良いだろう。千葉駅から名古屋駅までを移動する場合、普通列車の乗継も含めて 2 時間 55 分（注 3）の時間を要するが、飛行機とは違い空港から観光地までの移動もスムーズに行えるため、ロスが少なく効率的な上、掛かる費用も 10910 円（注 3）と割安だ。しかし、これら二つの移動方法にもデメリットはある。それは「一度乗ったら目的地に到着するまで降りられない」という点だ。移動時間を極力短縮して目的地での観光に専念するというのも一つの旅行モデルだが、今回私が提案したいのはもっと自由な旅だ。自由度を重視するのであれば「自動車」という選

択肢もあるが、あまりに長距離の移動は体力的にも厳しく、高速道路の利用料金や燃料費などもバカにならない。仮に千葉駅から名古屋駅までを移動した場合 12829 円（注 3）の費用が必要になる。そこで注目したのが「青春１８きっぷ」である。JR 線の普通列車が一日乗り放題となる青春１８きっぷには、一日の間に「どこでも」「何度でも」電車を乗り降りすることができるという特徴があり、その時の気分や状況などで自由に途中下車をして観光や食事を楽しむことができる電車の旅にはうってつけのサービスと言える。そして、移動に電車を利用することで、出発点から目的地までの道程においても、その土地土地の景色や味覚、そして住民との触れ合いを終始リラックスした状態で楽しむことができる。一度の旅行で多くの町の魅力や生活に触れることができるというのはとても贅沢なことであり、そこで得たものが次の旅へのモチベーションとなることも多々ある。千葉駅から名古屋までを例にした場合、7 時間 24 分（注 4）という時間を要する移動にはなるが、一人での利用ならば一日当たりの交通費も 2300 円（注 4）と格安に抑える事が可能であるため、時間に余裕があるのなら是非とも利用してもらいたいサービスだ。さて、そんな自由度の高いサービスである青春１８きっぷだが、その旅の行程をすべて機械任せにしていたのではせっかくのメリットが活かせない。しかし、あの大きくて分厚い時刻表を持ったまま全国を渡り歩くのは骨であるし、情報量の多い時刻表から必要な情報だけを抜き出すという点も玄人向けの楽しみ方と言える。加えて、発着駅や経由する駅を文字でしか知ることが出来ず、自分がどのような経路を通って旅行をしているのかがわかりづらいため、旅の支援ツールとして使用するには魅力に欠けると考えている。

そこで本研究では、「乗り降りする駅や経由するルートまで自分で決めることが出来る」という電車の旅の良さを残しつつ、時刻表の持ち運びやページ間を行ったり来たりするような手間を省くことのできる仕組みを構築することで、
旅行に慣れていない人をはじめ、「観光したいスポットがいくらかしか決まっていない」、また「そこに至るまでの行程が判然としていない」といった人達が気軽に電車の旅を楽しめる手助けとなることを目的とする。

（注 2　飛行機の運賃試算には「JAPAN AIRLINE」の国内線空席・運賃検索を使用 http://www.jal.co.jp/ ）

（注 3　自動車を利用した場合の移動時間と費用の試算には「NAVITIME 自動車ルー

ト案内」を使用 http://www.navitime.co.jp/ ）

（注 4　電車を利用した場合の移動時間と運賃の試算には「ジョルダン乗換案内」を使用 http://www.jorudan.co.jp/norikae/）

１－３　本論文の構成

本論文の 2 章では、ネット上で利用される既存の乗換案内サービス、路線情報提供サービスの分析について述べる。3 章では 2 章の問題点を整理し解決へのアプローチを行い、１８きっぷを使用した旅の支援モデルについて述べる。4 章では 3 章で提案したことを元に設計を行い、5 章で 4 章の設計に基づき実装について述べる。6 章では 5 章で実装した１８きっぷを使用した旅の支援環境の動作確認、及び評価と考察を行う。7 章では本研究のまとめと今後の課題を述べる。以上が本論文の構成である。

２．理想とするシステムと既存のサービス

本章では筆者の考える理想的なシステムと、現在展開されているウェブ上のサービスについて述べる。

２－１　理想とするシステム

指定した駅（自分がこれから電車に乗ろうとしている駅）から発車する電車がどこの駅まで運行するのかを地図上に表示することのできるシステムを構築したい。乗り換える駅ごとに時刻表を引く感覚で「次の電車でどこまで行けるのか」を調べさせることで、電車を利用した旅の楽しみ、面白みを損なわずに乗り換え等の手助けをすることができる。さらに、自分がどのような経路を辿って旅をしてきたのかということを視覚的に把握できることで、旅の思い出を記録にも記憶にもより深く残せる。

２－２　既存のサービス

・Yahoo ロコ　路線情報

図　２－２（１）：Yahoo!ロコ 路線情報

・サービス名：Yahoo!ロコ 路線情報

・URL：http://transit.loco.yahoo.co.jp/

・提供者：Yahoo! JAPAN

・概要：「Yahoo!ロコ 路線情報」とは、ポータルサイトを運勢する Yahoo! JAPAN が提供している無料の交通情報サービスである。

・機能・項目：「経路、運賃検索」、「電車の時刻表検索」、「駅情報・駅周辺情報の提示」、「路線図の表示」、「運行情報の掲示」

・使い方：トップ画面から乗換案内か時刻表検索かを選択し、調べたい駅名を入力して検索する。

・特徴：歩行速度の指定や、使用する交通機関を細かく設定した検索が可能となっている。電車の乗り換え案内サービスの他、時刻表の閲覧や駅内の設備情報、駅周辺の飲食店・宿泊店情報なども網羅している。

・長所・短所：「空路」、「フェリー」、「高速バス」の使用など、利用設定を細かく指定できる点が長所であるが、デフォルトでは全ての使用を考慮した結果が出力されるため、それらの交通機関を使用したくない場合には設定の手間が増えるという短所もある。

・ekitan

図　２－２（２）：ekitan

・サービス名：ekitan

・URL：http://ekitan.com/

・提供者：株式会社駅探

・概要：「ekitan」とは株式会社駅探が運営する無料の交通情報サービスである。

・機能・項目：「乗換案内」、「終電・始発案内」、「バス・電車・新幹線・飛行機の時刻表検索」、「運行情報の掲示」「駅情報・駅周辺情報の提示」「路線図の表示」

・使い方：トップページに表示されている乗換案内のフォームに「乗車駅」と「下車駅」、必要であれば「経由駅」を入力して経路検索を実行する。それ以外の機能を利用する場合には、トップページ上部か左のメニューから必要な機能を選んで利用する。

・特徴：通常の乗り換え案内やバス時刻表の閲覧機能の他、スマートフォンや携帯向けアプリケーションが充実している点が特徴。個人使用を目的としたサービスの他、法人向けのサービスやシステムの販売も行っている。

・長所・短所： 経路別の所要時間や乗換回数などがシンプルに一覧表示されるため比較が容易であるという点が長所といえる。「利用する交通機関」において、JR 線も私鉄も地下鉄もすべて同じアイコンで表示されるため、一見しただけでは利用する路線がわからないという点は短所である。

・ジョルダン

図　２－２（３）：ジョルダン

・サービス名：ジョルダン 乗換案内

・URL：http://www.jorudan.co.jp/norikae/

・提供者：ジョルダン株式会社

・概要：「ジョルダン乗り換え案内」とは、ジョルダン株式会社が運営する無料の路線情報サービスである。

・機能・項目：「乗換案内」、「定期代の検索」、「バス・電車・新幹線の時刻表検索」「運行情報の掲示」、「駅情報、駅周辺情報の提示」、「路面図の表示」、「旅行予約」

・使い方：トップページに表示されている「乗換案内かんたん検索フォーム」に「出発駅」と「到着駅」を指定して経路、運賃等を検索する。その他の機能を利用する場合には画面左のメニューから必要な機能を選択して利用する。

・特徴：定期代検索や地図を利用した検索機能の他、大手乗り換え案内サービスの中で唯一、青春１８きっぷ検索（青春１８きっぷの使用を前提とした検索）を導入しており、これを使用した場合には JR 線以外を使用した経路は選択されない。また、ユーザーによって投稿された多数の運行情報が常に表示されており、電車の遅延情報などのリアルタイムな情報を入手することができる。

・長所・短所：「到着時間順」、「出発時刻が遅い順」、「所要時間順」、「乗換回数順」、「安い順」などのように経路検索の条件を変えることができる。また、経路を検索した後から「5 分後の出発時間」、「10 分後の出発時間」を調べられる等、状況に応じた柔軟な対応ができる点がジョルダンの長所である。

２－３　既存のサービスと理想のサービスの相違

前項で挙げた既存のサービスは、乗り換え案内の検索精度や細かな機能の違いを除けば、いずれもサービス内容自体に大きな差異は無い。またこれらの乗り換えアルゴリズムは非常に正確なもので、一個人が制作したプログラムでは及びもしないと思われる。しかし、上記のサービスでは本研究の目的としている点を全て満たすことはできない。なぜなら、本研究で制作するプログラムは自由な電車の旅を実現するため、「乗り降りする駅は人間が一つ一つ決めていく」という手順が必要になるからだ。したがって、乗換経路の全てをプログラムが決める既存のサービスでは満たせない点が出てくる。また、電車が走る区間を視覚的に表示する機能、そして旅の軌跡（電車の乗り

換えの軌跡）を記録、表示するという機能を搭載している乗り換え案内サービスは存在しないため（発着駅を指定した場合を除く）、その点は本研究独自のプログラムで再現することになる。

３．　解決法

本章では 2 章で挙げた既存のサービスでは満たせない課題点についての解決方法を考えるとともに、本研究の到着点についても検討していく。

３－１　問題点の整理

第２章でも述べたとおり、既存の乗り換え案内サービスでは本研究の目的を達成することはできない。では具体的に何が問題なのか以下に述べる。

第１に、「ネットの乗り換え案内サービスでは乗換経路の選択が簡単すぎる」という問題。第２に「時刻表を旅先に持参すること、旅先で使用することは難しい」という問題。そして第３に、「乗り降りする駅の情報を文字でしか知ることができない」という問題だ。

第１の問題点について、各種乗換案内サービスで目的地までの経路を検索した場合、「出発点→A 駅→B 駅→C 駅→D 駅→目的地」という結果が出る。複数のルートを提示してくれるサイトもあるが、前述の検索結果が「出発点→A 駅→B 駅→C 駅→D 駅→E 駅→目的地」や「出発点→A 駅→C 駅→D 駅→目的地」となるだけで、結局、機械が全ての経路を決めている事に変わりはない。今回の研究で筆者が実現したいシステムは「出発点→（A 駅 orB 駅 orC 駅）」という検索結果が得られるシステムなのだ。

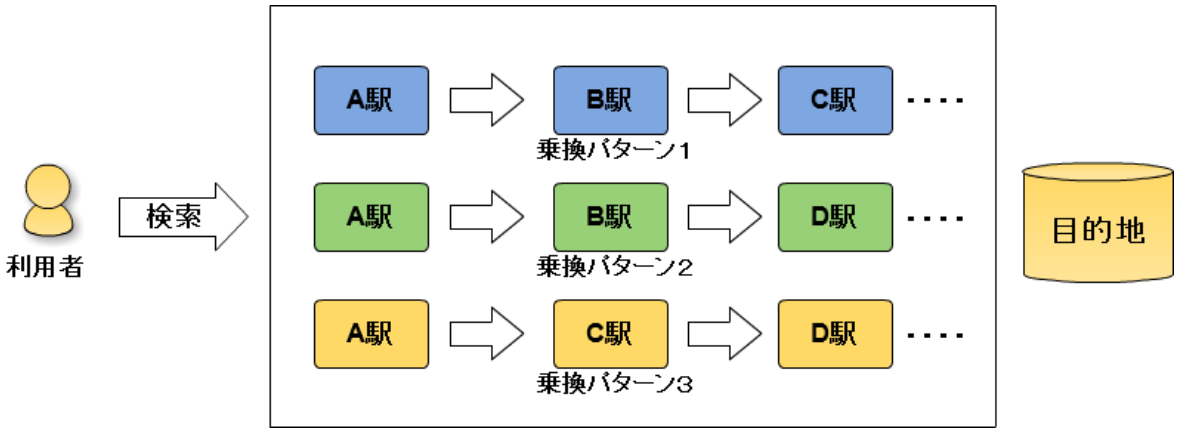

図３－１（１）：既存のサービス

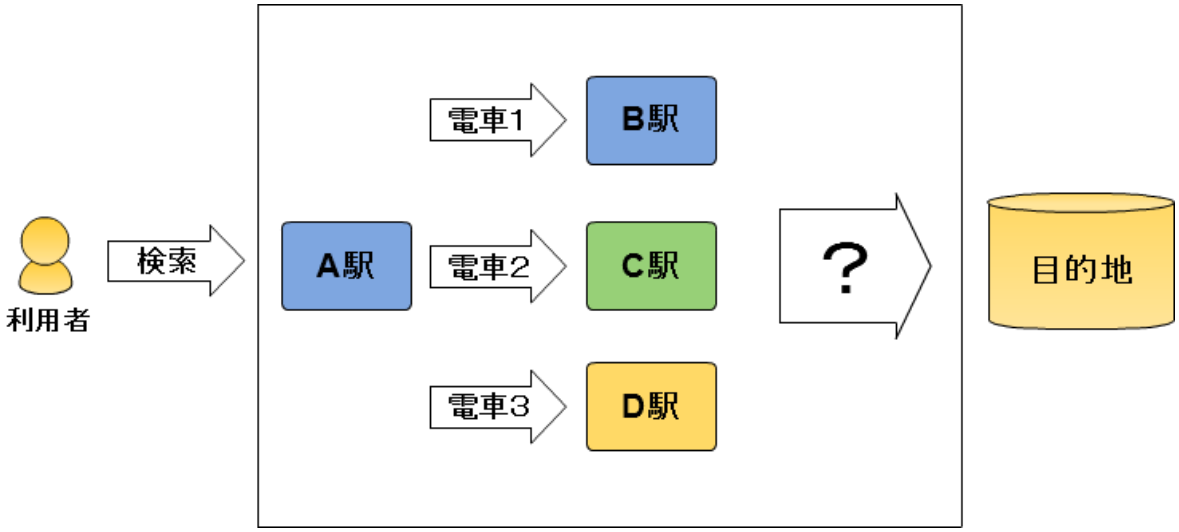

図３－１（２）：理想のサービス

電車の旅とバスツアーは異なる。すべての行程を決められたままにただこなしていくのは電車の旅として面白みに欠けるため、乗る電車や降りる駅などは自分で選んでいきたい。であれば、紙媒体の時刻表を持参して旅に出ればよいとも考えられるが、旅先で時刻表を逐一使用するのは骨であり、慣れない者にとっては気軽に手を出しづらいツールでもあるということが第２の問題に繋がる。

そして第３の問題について、旅行の際、初めから訪れることが決まっている観光スポットについて何も調べずに出発するという人は少ないだろう。しかし、今回想定しているのは「行き当たりばったりの旅」である。その時の気分でぶらりと途中下車した駅について十分にリサーチが出来ていなかったとしても不思議はない。聞きなれない地名であれば、そこが日本のどのあたりなのか明確にわからないこともあるだろう。しかし、それでは観光の思い出も薄れてしまう。それは旅行の楽しみを大きく損ねていると言える。

次節では以上の問題点から解決へのアプローチを行う。

３－２　解決の方針

２章で列挙した各種乗換案内サービス、あれらは「早く」、「無駄なく」電車を乗り換えるサービスとしては非常に便利で優れたものだ。しかし、今回の目的を達成するためには、効率的というだけのサービスでは魅力に欠ける。旅の実感を高めるために

も、目的地までの行程について人間が考える、悩む、そういった一手間があってしかるべきだ。そこで本研究では、旅の支援を目的としたサービスを「旅行の楽しみ、面白みを確保するための最低限の機能のみ」で構成することで、問題の解決を目指す。

４．　設計

本章では前章で述べた既存サービスの問題点と解決の方針に基づき、構成機能と構成要素について検討する。

４－１　構成機能

本環境の機能と構成（図４－１）を図に示す。

・駅名の検索

プルダウンメニューないしはテキストによる検索形式をとることで簡略化し、本来時刻表を使用する際に起こるような手間を省く。指定した駅名の検索には WebAPI を利用する。

・駅情報の受け取り

指定の駅に関する情報（駅名、緯度経度、時刻表）とそこから発車する電車の情報（路線名、発着駅、発着駅の緯度経度）をデータベースから取得する。

・行き先に合った電車を表示

取得したデータを元に、行き先の合致した電車を地図上に表示させる。

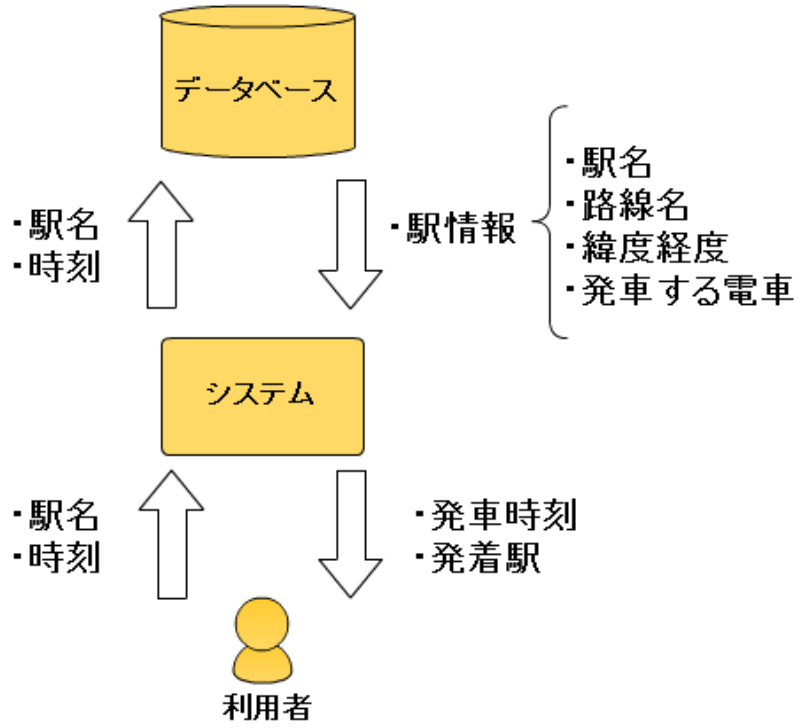

図４－１：構成機能図

４－２　構成要素

本環境を実際に使用する手順を想定しながら、必要な要素と構成（図４－２）について検討、設計を行った。これを示す。

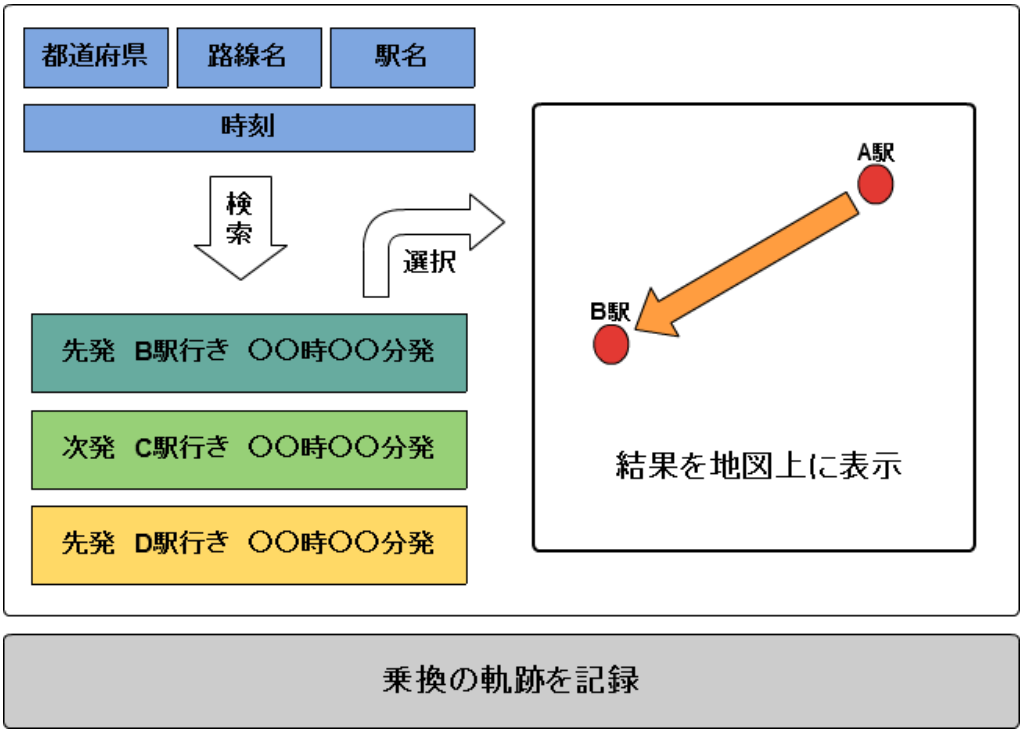

図４－２：構成要素図

① 最寄り駅から利用する電車を検索する

電車を利用する際にまず行うことは、自分が今現在いる駅、あるいは最寄り駅から発車する電車について調べることだ。この時、時刻表は持ち運ばないことが前提である為、スマートフォンやタブレット等から利用できる形態で実装する必要がある。

② 発車時刻、終着駅によって利用する電車を選択する

乗車する電車を選択する際、基準となる点は大きく二つ、「発車時刻」と「終着駅」だ。何時何分発でどこの駅まで移動できる電車なのかがわからなければ、選択のしようもない。この時、どの電車に乗るのかを自分で選択する余地を与える為に、「先発Ａ駅行き」、「次発Ｂ駅行き」、「次々発Ｃ駅行き」など複数の選択肢を提示する必要があ

る。

③ 好きな駅で下車

電車での移動になるため、好きな駅で自由に途中下車することができる。しかし、今回は普通列車にしか乗ることができない青春１８きっぷを利用した旅行を想定しているため、一日当たりの移動効率を考慮するならば、基本的には終点まで乗車することが望ましい。

④ 自分が通ってきた旅の道程を再確認する

自分が乗車した駅、下車した駅が日本のどのあたりに位置する駅なのかということを視覚的に把握できることで、その場所での観光や思い出などがより深く記憶され、旅の実感が高まる。

その後は電車を利用する度に①から④までの工程を繰り返す。このように、乗換駅ごとに電車を検索することで、「利用する駅や乗換経路を自分で見つけて選ぶ」という楽しみを実現することができる。

５章では以上の機能の実装について述べる。

５．　実装

本章では青春１８きっぷを利用した旅の支援システムの実装について述べる。

５－１　実装環境

本環境は以下の環境において実装を行った。

５－２　駅名の検索

本節では本環境の利用者が特定の駅名を選択、検索するまでの過程を述べる。駅名の選択にはプルダウンメニューを採用、その実装には「駅データ.jp」http://www.ekidata.jp/ から提供されている WebAPI を利用する。API は「application programming interface」の略で、アプリケーションの開発者が、他のハードウエアやソフトウエアの提供している機能を利用するための手法である。

注１　http://itpro.nikkeibp.co.jp/article/Keyword/20070829/280600/ 2012 年 12 月 20 日

また、駅名の検索フォーム、及び検索フォームを構成する API の設置にはプログラミング言語の PHP を用いる。

５－２－１　検索フォーム

goo_search.php は都道府県・路線名・駅名を順次選択することで乗車する駅を決定、そのデータを後述の goo_line.php に送信する。使用したプログラム（表５－３－１）を以下に示す

表５－２－１　「goo_search.php」

```
<html>
<head>
<link rel="stylesheet" href="default.css" type="text/css" />
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>都道府県から沿線/駅表示プルダウン表示</title>


<script type="text/javascript"><!--
var xml = {};
function setMenuItem(type,code){
```

```
var s =
document.getElementsByTagName("head")[0].appendChild(document.createElement("script"));
s.type = "text/javascript";
s.charset = "utf-8";

var optionIndex0 = document.form.s0.options.length;
var optionIndex1 = document.form.s1.options.length;

if (type == 0){
for ( i=0 ; i <= optionIndex0 ; i++ ){document.form.s0.options[0]=null
for ( i=0 ; i <= optionIndex1 ; i++ ){document.form.s1.options[0]=null}
document.form.s1.options[0] = new Option("----",0);
if (code == 0){
document.form.s0.options[0] = new Option("----",0);
}else{
s.src = "http://www.ekidata.jp/api/p/" + code + ".json"; }
}else{
for ( i=0 ; i <= optionIndex1 ; i++ ){document.form.s1.options[0]=null}
if (code == 0){
document.form.s1.options[0] = new Option("----",0);
}else{
s.src = "http://www.ekidata.jp/api/l/" + code + ".json";
}
}
xml.onload = function(data){
var line = data["line"];
var station_l = data["station_l"];
if(line != null){
document.form.s0.options[0] = new Option("----",0);
```

```
for( i=0; i<line.length; i++){
ii = i + 1 var op_line_name = line[i].line_name;
var op_line_cd = line[i].line_cd;
document.form.s0.options[ii] = new Option(op_line_name,op_line_cd);
}
}
if(station_l != null){
document.form.s1.options[0] = new Option("----",0);
for( i=0; i<station_l.length; i++){
ii = i + 1
var op_station_name = station_l[i].station_name;
var op_station_cd = station_l[i].station_cd;
document.form.s1.options[ii] = new Option(op_station_name,op_station_cd);
}
}
}
}
// --></script>
<script type="text/javascript"><!--
function SendSubmit()
{
document.form.target = "menu";
document.form.method = "GET";
document.form.action = "goo_line.php";
document.form.submit();

}
// --></script>
</head>
<body>
```

```
<img src="train.gif" alt="電車" align="right"/>
<h1>乗車する「駅」・「路線」・「電車」を選択してください</h1>
<form name="form">
<select name="pref" onChange="setMenuItem(0,this[this.selectedIndex].value)">
<option value="0" selected>-----
<option value="1">北海道
<option value="2">青森県
<option value="3">岩手県
<option value="4">宮城県
<option value="5">秋田県
<option value="6">山形県
<option value="7">福島県
<option value="8">茨城県
<option value="9">栃木県
<option value="10">群馬県
<option value="11">埼玉県
<option value="12">千葉県
<option value="13">東京都
<option value="14">神奈川県
<option value="15">新潟県
<option value="16">富山県
<option value="17">石川県
<option value="18">福井県
<option value="19">山梨県
<option value="20">長野県
<option value="21">岐阜県
<option value="22">静岡県
<option value="23">愛知県
<option value="24">三重県
<option value="25">滋賀県
```

```
<option value="26">京都府
<option value="27">大阪府
<option value="28">兵庫県
<option value="29">奈良県
<option value="30">和歌山県
<option value="31">鳥取県
<option value="32">島根県
<option value="33">岡山県
<option value="34">広島県
<option value="35">山口県
<option value="36">徳島県
<option value="37">香川県
<option value="38">愛媛県
<option value="39">高知県
<option value="40">福岡県
<option value="41">佐賀県
<option value="42">長崎県
<option value="43">熊本県
<option value="44">大分県
<option value="45">宮崎県
<option value="46">鹿児島県
<option value="47">沖縄県
</select>
<select name="s0" onChange="setMenuItem(1,this[this.selectedIndex].value)">
<option selected>----
</select>
<select name="s1">
<option selected>----
</select>
<input type="button" value="検索" onclick="SendSubmit()" />
```

```
</form>
</body>
</html>
```

５－３　時刻表データの取得

本節では検索フォーム欄で指定された駅名を元に、実際の時刻表データを取得するまでの過程を述べる。また、時刻表データの取得は goo 路線の時刻表欄 http://transit.goo.ne.jp/timetable/area7.html からの Web スクレイピングによって実現する。Web スクレイピングとは、Web サイトから Web ページの HTML データを収集して、特定のデータを抽出、整形し直すことである。注１: http://www.sophia-it.com/content/Web%E3%82%B9%E3%82%AF%E3%83%AC%E3%82%A4%E3%83%94%E3%83%B3%E3%82%B0　2013 年 1 月 16 日

５－３－１　データの受け取りと路線検索

goo_line.php は、goo_search.php から受け取ったデータを元に駅データ.jp に駅情報をリクエストし、その駅の「駅名」と「駅コード」を受け取る。受け取った駅名をキーワードに、goo 路線の時刻表欄からデータを抽出し、キーワードに一致した駅とそこから運行しているすべての路線を一覧で表示する。また、これがクリックされた場合には、選択された駅・路線の URL と受け取った駅コード(乗車駅のコード)を後述の goo_tt.php に引き渡す。使用したプログラム（表５－３－１）を以下に示す

表５－３－１　「goo_line.php」

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html lang="ja">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>路線選択</title>
</head>
<?php
```

```
extract($_GET);
$response = simplexml_load_file("http://www.ekidata.jp/api/s/$s1.xml");
foreach ($response->station as $station) {
        $station_name = $station->station_name;
        $station_cd = $station->station_cd;
}
$base = "http://transit.goo.ne.jp/";
$query = "tconfirm.php?an=7&st_name=";

$stn = $station_name;
$stn_enc = urlencode($stn);
//駅名検索 URL 設定
$req1 = $base . $query .$stn_enc;

//echo    $req1 . "<br>";
define(INIT_URL , $req1);
$use_proxy = 0;
$proxy_host = 'proxy.example.com';
$proxy_port = 8080;
$no_proxy = array(
    'localhost',
    '127.0.0.0/8',
);
include('proxy.php');

function get_html($url){
    $rdata = http_request($url);
    $data = mb_convert_encoding($rdata['data'],"utf-8","auto");
    return ($data);
}
```

```
function dump_nodes_title(tidyNode $node, &$urls = NULL) {
	$urls = (is_array($urls)) ? $urls : array();
	if(isset($node->id)) {
		if($node->id == TIDY_TAG_H2) {
			if (isset($node->attribute['class'])){
				if (stristr ($node->attribute['class'] ,"tr02") !==FALSE){
					$urls[] = array("title" => $node->child[0]->value ,"link"
=>$node->attribute['h2']);
				}
			}
		}
		if($node->id == TIDY_TAG_A) {
			if (isset($node->attribute['href'])){
				if (stristr ($node->attribute['href'] ,"line_code") !==FALSE){
					$urls[] = array("title" => $node->child[0]->value ,"link"
=>$node->attribute['href']);
				}
			}
		}
	}
	if($node->hasChildren()) {
		$count = 0;
		foreach($node->child as $c) {
			if(isset($c->id)) {
				if($c->id == TIDY_TAG_DIV){
					if (isset($c->attribute['class'])){
					if (stristr ($c->attribute['class'] ,"category") !==FALSE){
					break $count++;
					}
```

```
                        }
                                } elseif($c->id == TIDY_TAG_DL){
                                if (isset($c->attribute['class'])){
                                if (stristr ($c->attribute['class'] ,"ranking") !==FALSE){
                                break $count++;
                                }
                        }
                                }
                        dump_nodes_title($c, $urls);
                }
        }
    }
    return $urls;
}

$data = get_html(INIT_URL);
$config = array('indent' => TRUE,
                 'output-xhtml' => TRUE,
                 'wrap' => 200);
$tidy = tidy_parse_string($data, $config, 'UTF8');
$tidy->cleanRepair();
$urls = dump_nodes_title($tidy->body());
echo '<div align="left">';

foreach ($urls as $item){
    if (strpos($item['title'], "駅") !== false ) {
                echo "<hr>".$item['title'].'<br>';
        } elseif (strpos($item['title'], "線") !== false || strpos($item['title'], "]") == false) {
                echo "    " . $item['title'].'<br>';
        } else {
```

```
                    echo "<ul><li><a href='goo_tt.php?query=" . $item['link']."&name=".
$station_cd. "' target='main'>" . $item['title']."</a></li></ul>";
            }
}
echo '</div>';

?>
</body>
</html>
```

５－３－２　時刻表データの表示とデータの送信

goo_tt.php は、goo_line.php から受け取った URL を元に、goo 路線の時刻表欄から時刻表データを抜き出し、一日に運行している電車の発車時刻と行き先を一覧で表示する。また、行き先の横に設置した「地図」ボタンがクリックされた場合、goo_line.php から受け取った駅コード(乗車駅のコード)と、選択された行き先のデータを後述の goo_map.php に送信する。使用したプログラム（表５－３－２）を以下に示す

表５－３－２　「goo_tt.php」

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html lang="ja">
            <head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>時刻表表示</title>
</head>
<?php
mb_language("uni");
mb_internal_encoding("utf-8");
mb_http_input("auto");
mb_http_output("utf-8");
```

```
mb_regex_encoding('UTF-8');

extract($_GET);

$base = "http://transit.goo.ne.jp/";
//$query =
"/timetable/area_code/7/line_code/%C1%ED%C9%F0%C0%FE%B2%F7%C2%AE%A1%A6%B
2%A3%BF%DC%B2%EC%C0%FE/stationcode/%C1%ED%C9%F0%C0%FE%B2%F7%C2%AE
%A1%A6%B2%A3%BF%DC%B2%EC%C0%FE-120019/";

$req1 = $base . $query;
define(INIT_URL , $req1);

$use_proxy = 0;
$proxy_host = 'proxy.example.com';
$proxy_port = 8080;
$no_proxy = array(
        'localhost',            // localhost
        '127.0.0.0/8',          // loopback
);
include('proxy.php');

function get_html($url){
        $rdata = http_request($url);
        $data = mb_convert_encoding($rdata['data'],"utf-8","auto");
        return ($data);
}

function dump_nodes_tt(tidyNode $node, &$urls = NULL) {
        $urls = (is_array($urls)) ? $urls : array();
```

```
    if(isset($node->id)) {
        if( $node->id == TIDY_TAG_TH ){
            $urls[] = str_replace( array( chr(10) , chr(13) ),"",$node->child[0]->value);
                }

        if( $node->id == TIDY_TAG_TD ){
            $urls[] = str_replace( array( chr(10) , chr(13) ),"",$node->child[0]->value);
        }

        if($node->id == TIDY_TAG_DIV) {
            if (isset($node->attribute['class'])){
                if (stristr ($node->attribute['class'] ,"fs12") !==FALSE){
                    //$hoge = str_replace( array( chr(10) , chr(13) )," ",$node->child[1]->value);
                    $urls[] = $node->child[1]->value;
                }
            }
        }

    }
    if($node->hasChildren()) {
        foreach($node->child as $c) {
            dump_nodes_tt($c, $urls);
        }
    }
    return $urls;
}

if ($query != ''){
        $data = get_html(INIT_URL);
```

```
$config = array('indent' => TRUE,
                'output-xhtml' => TRUE,
                'wrap' => 200);
$tidy = tidy_parse_string($data, $config, 'UTF8');
$tidy->cleanRepair();
$tt = dump_nodes_tt($tidy->body());

$kyou = $tt[0];
$js = array();
array_shift($tt);
$hanrei = $tt[count($tt)-1];

$hanrei = trim($hanrei);
$tmp = explode("    ", $hanrei);
array_shift($tmp);

$stn = array();

for($i = 0; $i<count($tmp); $i++){
        $tmp0 = mb_split("＝", $tmp[$i]);
        $nam0 = $tmp0[0];
        $nam1 = mb_substr($tmp0[1], 0, -1);
        //echo $nam0 . "->" . $nam1 ."<br>";
        $stn[] = array("ryaku"=>$nam0 , "name" =>$nam1);
}

array_pop($tt);
echo "<br>";

for($i = 0; $i<count($tt)/2 ; $i++){
```

```
                $h = $tt[$i*2];
                $tmp0 = explode(">",$tt[$i*2+1]);
                for ($j=5; $j<count($tmp0); $j+=9){
                        $tmp1 = explode("<",$tmp0[$j]);
                        $dest = $tmp1[0];
                        if ($dest == "") break;
                        $tmp2 = explode("<",$tmp0[$j+4]);
                        $m = $tmp2[0];
                        $js[] = array("hour"=>$h , "minute" =>$m , "dest"=>$dest);
                }
        }

        echo '<table>';
        echo '<tr><td colspan="3" align="center">'.$kyou.'</td></tr>';
        echo '<tr><th>時</th><th>分</th><th>行先</th></tr>';
        foreach($js as $item){
                foreach($stn as $s){

                        if ($s['ryaku'] === $item['dest']){
                                $eki_name = $s['name'];
                                //echo "ryaku=" . $s['ryaku'] . "  dest=". $item['dest'] .
"<br>";
                                break;
                        } else {
                                $eki_name = $item['dest'];
                        }
                }
        echo '<form action="goo_map.php" target="map" method="GET" >';
        echo '<tr><td>'.$item['hour'].'</td><td>'.$item['minute'].'</td><td><input type="hidden"
name="dest" value="' . $eki_name . 駅 .'"><input type="hidden" name="start" value="' .
```

```
$name .'">'.$eki_name.'<input type="submit" value="地図"></td></tr>';
            echo '</form>';


            }
            echo '</table>';
} else {
            echo '路線を選択してください';
}
?>
</body>
</html>
```

５－４　地図の表示

goo_map.php は、これまでに受け取ったデータを元に、その電車が日本のどこからどこまでを走る電車であるのかを地図上に表示する。地図の表示には「Google」http://www.google.co.jp/から提供されている Google Maps JavaScript API V3 を使用。緯度経度の算出には、５－２節で述べた webAPI と、「Geocoding.jp」 http://www.geocoding.jp/ が提供している座標検索 API を利用した。使用したプログラム（表５－４）を以下に示す

表５－４　「goo_map.php」

```
<!DOCTYPE html "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
   <head>
      <meta http-equiv="content-type" content="text/html; charset=utf-8"/>
      <title>地図</title>
      <?php
      extract($_GET);
$s1 = $start;
```

```
$goal = $dest;
                $response = simplexml_load_file("http://www.ekidata.jp/api/s/$s1.xml");
                $response_b                                                    =
simplexml_load_file("http://www.geocoding.jp/api/?v=1.1&q=$goal");
                foreach ($response->station as $station) {
                        $station_name = $station->station_name;
                        $lat = $station->lat;
                        $lon = $station->lon;
                foreach ($response_b->coordinate as $coordinate) {
                        $lat_b = $coordinate->lat;
                        $lon_b = $coordinate->lng;
        }
}
?>
    <script type="text/javascript"
      src="http://maps.google.com/maps/api/js?sensor=false"></script>
    <script type="text/javascript">
      var x = <?php echo $lat; ?>;
      var y = <?php echo $lon; ?>;
      var a = <?php echo $lat_b; ?>;
      var b = <?php echo $lon_b; ?>;
    function initialize() {
      var latlng = new google.maps.LatLng(x,y);
      var opts = {
        zoom: 9,
        center: latlng,
        mapTypeId: google.maps.MapTypeId.ROADMAP
      };
      var map = new google.maps.Map(document.getElementById("map_canvas"), opts);
```

```
    var PlanCoordinates = [
        new google.maps.LatLng(x,y),
        new google.maps.LatLng(a,b),
    ];
    var trainPath = new google.maps.Polyline({
        path: PlanCoordinates,
        strokeColor: "#FF0000",
        strokeOpacity: 0.5,
        strokeWeight: 7
    });
        trainPath.setMap(map);
    }
    </script>

  </head>
  <body onload="initialize()">
          <p>「<?php echo $station_name; ?>駅」発ー「<?php echo $goal; ?>」行き</p>
    <div id="map_canvas" style="width:610px; height:400px"></div>

  </body>
</html>
```

５－５　トップページ

本環境の全体構成は以下（図５－５－１）の通りである。

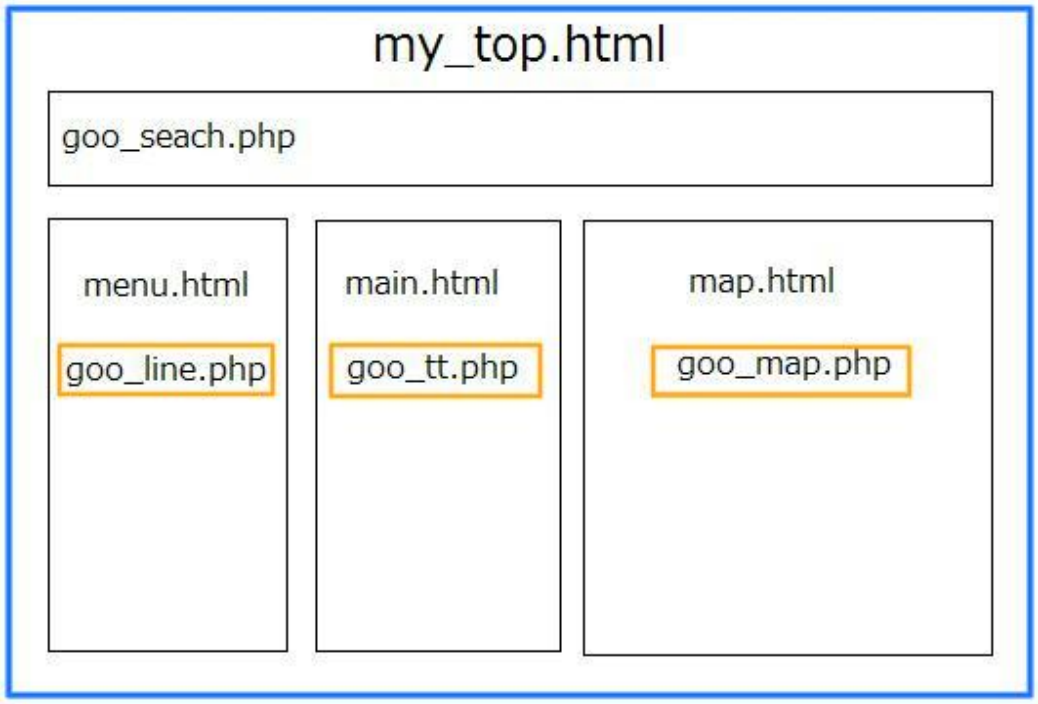

図５－５－１：トップページの全体構成

トップページは my_top.html、menu.html、main.html、map.html、goo_search.php、goo_line.php、goo_tt.php、goo_map.php、default.css で構成される。

・my_top.html はトップページ内にフレームを複数作成し、検索フォームである goo_search.php を上部に、フォームからデータが送信されるまでの待機画面である menu.html、main.html、map.html をそれぞれ下部左、下部中央、下部右に呼び出す。

・default.css は goo_search.php のレイアウトを定義する。

・ goo_line.php は goo_seach.php のフォームからデータが送信された際に画面下部左のフレームに表示される。

・ goo_tt.php は goo_line.php からデータが送信された際に画面下部中央のフレームに表示される。

・ goo_map.php は goo_tt.php からデータが送信された際に画面下部右のフレームに表示される。

以上より、トップページ（図５－５－２）は構成される。

使用したプログラム（表５－５）を以下に示す

図５－５－２（１）：トップページ画面（乗車駅選択前）

図５－５－２（２）：トップページ画面（乗車駅選択後）

表５－５　「my_top.html」

```
<!DOCTYPE html>
<html lang="ja">
<head>
        <meta charset="UTF-8">
```

```
        <title>toppage</title>
</head>
<body>
        <iframe name ="searh" src = "goo_search.php" width="1240"
height="100"></iframe></iframe><br clear="all" />

        <iframe name ="menu" src = "menu.html" width="300" height="800"></iframe>

        <iframe name ="main" src = "main.html" width="300" height="800"></iframe>

        <iframe name ="map" src = "map.html" width="625" height="800"></iframe>
</body>
</html>
```

表５－５　「menu.html」

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
        <meta charset="UTF-8">
        <title>menu</title>
</head>
<body>

<p>１．駅を選んでください</p>

</body>
</html>
```

表５－５　「main.html」

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
	<meta charset="UTF-8">
	<title>main</title>
</head>
<body>

<p>２．路線を選んでください</p>

</body>
</html>
```

表５－５　「map.html」

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
	<meta charset="UTF-8">
	<title>map</title>
</head>
<body>

<p>３．電車を選んでください</p>

</body>
</html>
```

表５－５　「default.cssl」

```
body {
```

```
        background-color: #F5F5F5;
 }
h1 {
        font-family: "Times New Roman", Times,serif;
        font-size: 20px;
        margin-bottom: 5px;
}
```

６．評価と考察

本章では青春１８きっぷを利用した旅の支援環境の動作確認、及びそれを元にした評価と考察を行う。

６－１　機能評価

本節では動作確認を含めた機能評価を行う。尚、動作の確認はトップページ、駅名検索フォーム、路線検索ページ、時刻表検索ページ、地図ページの順番で行う。

・トップページのプログラムの動作確認を行う。

トップページ（図６－１（１））には５－５節で述べたプログラムが正しく動作し、画面上部のフレームには駅名の検索フォーム、画面下部のフレームには路線表示欄、時刻表表示欄、地図表示欄それぞれの待機画面が表示されている。

図　６－１（１））：トップページの動作確認

・駅名検索フォームの動作確認を行う。

駅名検索フォーム（図６－１（２））では５－２－１項で述べたプログラムが正しく動作し、都道府県、路線名を選択することで駅名を絞り、選択した駅の情報を路線検索プログラムに送信している。動作確認では神奈川県を走る JR 東海道本線（東京～熱海間）の中から熱海駅を選択した。

図　６－１（２）：駅名検索フォームの動作確認

・路線検索ページの動作確認を行う。

路線検索ページ（図６－１（３））では５－３－１項で述べたプログラムが正しく動作し、駅名検索フォームで選択された駅から運行している路線名を表示している。動作確認では「東海道本線（東日本)」、「東海道本線（東海)」、「ムーンライトながら」、「伊東線」、「湘南新宿ライン」、「東海道・山陽新幹線」、「踊り子」、「リゾート踊り子（東京)」、「リゾート踊り子（立川)」、「リゾート踊り子（高尾)」、「スーパービュー踊り子」、「サンライズ出雲」、「サンライズ瀬戸、「サンライズゆめ」の１５路線がヒット、その中から東海道本線（東海）を選択した。

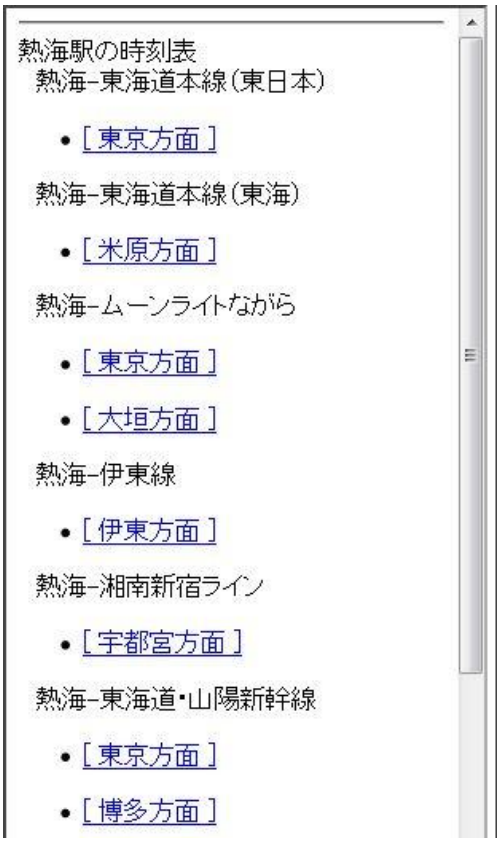

図　６－１（３）：路線検索ページの動作確認

・時刻表検索ページの動作確認を行う。

時刻表検索ページ（図６－１（４））では５－３－２項で述べたプログラムが正しく動作し、路線検索ページで選択した路線の時刻表が表示されている。動作確認では検索した日時、電車が発車する時刻、行き先、そしてその横に地図を表示するためのボタンが表示された。

図　６－１（４）：時刻表検索ページの動作確認

・地図ページの動作確認を行う。

地図ページ（図６－１（５））では５－４節で述べたプログラムが正しく動作し、乗車駅と終着駅を結んだ線を描画した地図を表示している。動作確認のため行き先を変えて実験を行った。「熱海発・沼津行き」（図６－１（５））、「熱海発・静岡行き」（図６－１（６））、「熱海発・浜松行き」（図６－１（７））の結果をそれぞれ図で示す。

図　６－１（５）：地図ページの動作確認（沼津行き）

図　６－１（６）：地図ページの動作確認（静岡行き）

図　６－１（７）：地図ページの動作確認（浜松行き）

６－２　比較評価

本節では青春１８きっぷを利用した旅の支援環境と既存のサービス（２－２節）との比較評価を行う。

－乗換案内サービスとしての比較

日常生活の中で乗換案内サービスを利用する際に重要となる点は「早さ」と「正確さ」、そして「わかりやすさ」だ。一度の検索で全ての情報が得られる早さ、複数の路線を経由するルートであっても分単位の乗換を可能にする正確なアルゴリズム、どの駅でどの電車に乗り換えるのかを一目で理解できる簡潔さ、こういった要素が重要になる日常のシーンでは、上記全ての機能を兼ね備えている既存のサービスを利用する方が便利だ。比較的短い距離の移動であり、乗車する駅と降車する駅が明確に決まっているのであれば、本環境を積極的に活用するメリットは少ない。

－旅の支援ツールとしての比較

日常から離れた旅行という場面、その中でも青春１８きっぷを利用した自由な旅においては、利便性や効率以上に「娯楽性」が求められる。具体的には、一つ一つの行程を人間が考えて選択していける自由度の高さや、旅の実感を高める独自の魅力が必

要だ。その点において、アナログな時刻表を扱う楽しさを手軽にどこからでも楽しめる環境や、電車が走る区間を地図に表示するなど独自の機能を備えた本環境は、青春１８きっぷを利用した旅の支援ツールとして、既存のサービスよりもより魅力的なものに仕上がったと言える。

６－３　考察

「自由な電車の旅を気軽に楽しめる手助けとなる」、「旅の実感を高め、思い出をより深く残せる仕組みを構築する」という本研究の目的は、乗り換える駅ごとに時刻表を引く感覚で使用できる本システムの環境と、乗車する電車が日本のどのあたりを走るのか視覚的に把握できる地図機能の実装により達成することができた。しかし、「特急」・「快速」・「寝台」などの列車種別が表示されず、路線情報を提供するサービスとして問題となる点や、４章で述べた「乗換の軌跡を記録する」という機能が実装できなかった点などの課題も残った。尚、発車時刻を指定しての時刻表検索に関しては、アナログな時刻表を扱っている時の感覚をより感じてもらうために実装を見送った。

７．　まとめと今後の課題

本章では本研究のまとめと、前章でも述べた今後の課題について詳しく述べる

７－１　まとめ

電車の旅に出かける際、旅慣れていない者にとって便利なツールがネット上の乗換案内サービスである。しかし、すべての行程を機械任せにした旅ではなくある程度の「行き当たりばったり性」を残した自由な旅を実現するためには、目的地までの経路がすべて提示される既存のサービスでは不十分だ。この問題に対して本研究では、旅の支援を目的としたサービスを「旅行の楽しみ、面白みを確保するための最低限の機能のみ」で構成することで解決を目指した。システムモデルの設計後、PHP・HTML・JavaScript 等のプログラミング言語で実装したプログラムによって問題の解決に成功したが、いくつかの課題も残った。

７－２　今後の課題

今後の課題についてこれまでの考察に基づき述べる。

―路線情報提供サービスとしての問題

駅名の検索時と時刻表の検索時、その両方で路線を選択する必要であるという点が問題だが、旅の支援ツールとして、自分がどの県のどの路線を利用しているのか再認識してほしいという思いがあったために現状の形態をとった。

―青春１８きっぷを利用した旅を支援するサービスとしての問題

「特急」・「快速」・「寝台」などの列車種別が表示されない、また、青春１８きっぷで利用できる路線・列車なのか否かの判別ができていない点が問題だ。また、旅の実感を高めるために「乗換の軌跡を記録する」機能を実装できることが望ましい。これらを今後解決すべき課題とする。

参考文献

[1] 谷崎竜『青春 18 きっぷパーフェクトガイド 2010-2011』イカロス出版

[2] 青春 18 きっぷ探検隊『青春 18 きっぷで愉しむ鉄道の旅』小学館

[3] goo 路線

http://transit.goo.ne.jp/

[4]Yahoo!路線情報

http://transit.loco.yahoo.co.jp/

[5]ekitan

http://ekitan.com

[6]ジョルダン

http://www.jorudan.co.jp/norikae/

[7]JAPAN AIRLINES

https://www.jal.co.jp/

[8]NAVITIME

http://www.navitime.co.jp/

[9]駅データ.jp

http://www.ekidata.jp/

[10]Cacoo

https://cacoo.com/

[11]AjaxTower

http://www.ajaxtower.jp/

[12]Geocoding

http://www.geocoding.jp/

[13]Crystal-Creation

http://www.crystal-creation.com/

[14]青春１８きっぷ研究所

http://seisyun.tabiris.com/syoyo.html

[15]旅行手段を考える

http://www.thelonelyh.com/

[16]グーグルマッピィ

http://www.google-mapi.com/

[17]WEB 系専門学生

http://drmn0202.blog122.fc2.com/blog-entry-17.html　2010 年 3 月 25 日

[18]PHP GO

http://phpgo.info/　2013 年 1 月 18 日

[19]reCatnap

http://tips.recatnap.info/　2013 年 1 月 11 日

謝辞

本研究を行うに当たり、ご指導を頂いた渡辺恭人准教授に感謝したします。研究の進め方に始まり、論文の構成、プログラミングに至るまで親身にご指導いただきまして、ありがとうございました。特にプログラミングについては多大なお力添えをいただき、機能不足で魅力に欠けるものだった私の研究を現在の段階にまで押し上げていただきました。感謝の念でいっぱいです。本研究を卒業論文として提出出来たことは、これから IT と密接に関わる道に進む私にとって、一つの自信に繋がりました。ここに至るまで、研究のテーマや方向性がぶれる事もありましたが、その都度、よりより方向に導いていただけたおかげで、ここまで研究を進めることができました。最後に、私の卒業論文に関わって下さった方全員にもう一度感謝を述べさせて頂き謝辞とさせていただきます。本当にありがとうございました。