

## 地図情報と連携した歴史年表の開発 —Google Map の利用—

中津 檣男\* 原田 邦彦\*\*

\*情報教育講座

\*\*愛知教育大学大学院学生

## A Chronological Table Overlaid with Geographical Data —Utilization of a Google Map—

Narao NAKATSU\* and Kunihiro HARADA\*\*

\*Department of Information Sciences, Aichi University of Education, Kariya 448-8542, Japan

\*\*Graduate Student, Aichi University of Education, Kariya 448-8542, Japan

### 第1章 はじめに

本研究は、年代ごとに、その時代に発生した歴史的な事件・出来事を地図と連携して表示するシステムを作成することである。

世界地図の上に歴史的な出来事・事実を表示することによって、その時代その時代の世界の全体像を捉えることができる。すなわち、歴史的な史実と地理情報を関連づけて記憶するための手段の開発が本研究の目的である。

世界史や日本史など過去の歴史を勉強するとき、特定の地域（ヨーロッパならずともヨーロッパ史、中国ならずとも中国史）を中心として年代を過去から現代へ進めて勉強していくので、縦のつながり（時代・国の移り変わり）には強くなるのだが、その時代時代の横のつながり（隣の国や貿易の相手国など）はどうしても印象が弱くなってしまう。国の位置情報もあやふやになってしまうため知識の定着が難しい。そこで地域別でも時代別でも全体的にしかも効率よく勉強するにはどうすればいいかと考えた時、地図と併せて歴史が見られたら便利なのではないかと考えた。

日本史と世界史の年表を併記して、日本と世界のつながりがわかるように工夫された年表も存在し、それを Web 化したものが存在する<sup>[1]</sup>が、現在の歴史年表では、どこで何の事件が起きたかという地理情報が国名だけで表示されるため、国の位置関係を正確に把握していなければならない。そこで、歴史の年表を地図の中で見られるようにすれば、歴史上の事件・出来事の地理的な関係やつながりを同時に把握・理解できる

のではないかと考えた。これが研究を始めるきっかけとなった動機である。

地図としては、地図画像の利用が考えられるが、地図画像は拡大・縮小すると形が崩れる、スクロールが難しいなどの問題がある。ヨーロッパ史を考える場合は、ヨーロッパを中心とした見方が必要である。そこで本研究では Google Map を地図として利用し、その上に史実を表示することにした。Google Map を利用したシステムとして日本地図に城のアイコンを配置し、アイコンをクリックすればその城の説明が表示される Web ページがある<sup>[2]</sup>。Google Map では地図の拡大・縮小表示や、スクロールは非常に簡単に実現できる。他に、数値地図の利用も考えられるが、数値データの入手が困難という問題がある。

一方、Google Map を利用すると次のような欠点が生じる。第1の欠点は、インターネットにつながっていない状態（オフライン）では利用できないということである。つまり、利用者はインターネット環境の整った場所でしか本システムを利用することができない。第2は、Google Map を利用するには Google から提供されているソフトウェア（2.1参照）を利用する必要があるが、そのソフトが更新されると本稿のソフトも変更を余儀なくされる点である。第3の欠点は、Google Map の地図は現在の地図で、国境や県境は現在の状態のものしか表示できないという点である。

こうした欠点にも関わらず、本研究では地図表示という労力を省き、開発時間を短縮するために Google Map を利用することにした。

2章では、本システムの開発に使用したソフトウェ

アを解説・説明し、3章で開発したシステムの説明を行う。また、開発したシステムの中で特に工夫した点を4章にまとめた。

## 第2章 基本事項

本システムを開発するに当たって使用したソフトの説明をしていく。

### 2.1 Google Map API

Google Map は Google 社が提供する地図閲覧サービスで世界地図から街路図まで利用者の必要に応じて表示できる。

Google Map では地図の拡大・縮小を行うコントロール機能があらかじめ用意されている。また、ドラックをすると好きな方向に画面のスクロールができる。

Google Map API は JavaScript の関数群として提供されており、Web ページにそれらの関数を埋め込むことで上記の機能を実現できる。2010年8月現在の最新バージョンは3.0である。

Google Map API の基本機能を以下に示す。<sup>[3]</sup>

(1) 自分の Web ページ上で Google Map と同様の地図を表示できる。現在、多くのサイトがこのサービスを利用している（例えば、ホテルがその所在地を示すためなどに利用している）。

(2) 地図の属性

Google Map API を使えば、利用者の好きなように地図の詳細を設定できる。例えば、地図を開いた時の中心をどこにするのか、地図のズームレベル(0~19で、0は世界全体を表示、19は個々の建物を表示)をいくつにするかの指定ができる。さらに、地図、地形図、航空写真図の切り替えができる。

(3) アイコンと情報ウィンドウ

地図上には、情報ウィンドウを表示できる。このウィンドウは、漫画のふきだしに似ている。情報ウィンドウは最初は表示されず、地図上にあるアイコンをクリックすると初めて表示されるようにもできる。アイコンの先端は細くなっており、その先端は地図上の指定位置を示している。このアイコンは色・大きさを指定することができるので、今回のプログラムでは用途に合わせて使い分けた。

(4) 地図上でマウスをクリックしたりドラッグしたりというイベント情報を取り込み、それに対応した動作記述が可能である。これを利用して地図の対話的な使い方を実現できる。

### 2.2 JSP / サーブレット<sup>[4]</sup>

JSP / サーブレットは、その名の通り、Java をベースとしたサーバサイド処理環境である。

一般的な Web の世界においては、クライアントがサーバに対してあるページの要求を行うと、サーバ側では要求されたページを応答として返す。つまり、Web システムとは、クライアントの要求に対してサーバ

が応答するといったデータのやり取りを繰り返しているにすぎない。このとき、サーバ側に用意された HTML ファイルをクライアントに返すのが、最も原始的で、一般的に静的と呼ばれるサイトの仕組みである。この場合、サーバはただ要求を受けて、要求どおりのページを再び送信するだけのメッセージの役割でしかない。この場合は要求に対してただ1つの決まった応答しかできない。

クライアントから「A」というデータを受けたら「a」、「B」というデータを受けたら「b」というように、クライアント側の要求によって異なる応答を返したいというケースがあるかもしれない。このとき、サーバにはただのメッセージングとして以上の役割が要求される。つまり、サーバが機械的な応答をするだけでなく、クライアントが送ってきたデータの中身をサーバが判断して、返信の種類を変更するという判断が必要になる。クライアントの要求の種類によっては、この「A」「B」などのメッセージをサーバ上に記録しておく必要があるかもしれない。また、「A」「B」などのメッセージを「AAA」「BBB」に加工しなければいけないこともあるかもしれない。このような「判断」「記録」「加工」などの一連の処理を行うのがサーバ側で動作するサーブレットである。つまり、サーブレットの役割は、応答のためのデータを作成するということである。

サーブレットはクライアントからの要求によって呼び出され、起動する。そして、予めプログラムとして指定された一連の処理を行った後、その結果として生成されたデータをクライアントにレスポンスとして返す。この時のデータ形式は一般的に HTML である。またサーブレットは、サーバ側のファイルやデータベースなど豊富なりソースとも密接な連携を取ることができる。それを図1に示す。サーバ側で動作するという意味では、CGI (Common Gateway Interface) や ASP (Active Server Pages)、PHP (Hypertext PreProcessor) などの仕組みと共通している。サーブレットがこれら既存の技術よりも優れているといえる点は次のとおりである。

(1) プラットフォームを選ばない

ASP は、Windows + IIS (Internet Information Server/ Services) や PWS (Personal Web Server) で動作するサーバサイド処理環境である。原則的に Windows 環境以外では実行することができない。しかし、サーブレッ

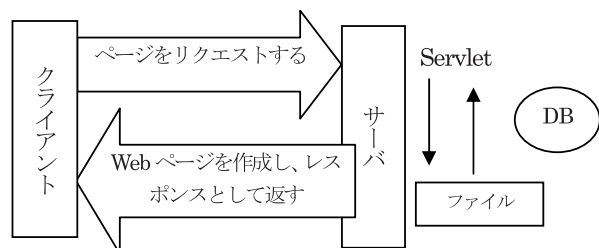


図1：リクエストとレスポンス

トはサーブレット対応の実行エンジン（コンテナ）さえあれば、Windows、UNIX、Mac OS など OS の種類を問わず、動作することが可能である。

## （２）サーブレットはユーザ数の増加に強い

CGI は、Web サーバから外部プログラムを呼び出し、実行する仕組みの総称である。つまり、要求がある度に外部の実行プロセスを起動しなければならないため、それだけリソースの消費は激しい。リソースの圧迫は、特にユーザ数の変動が激しいインターネットの世界では、直接、処理パフォーマンスにも影響する致命的な要因と言える。一方、サーブレットにおいては、サーブレットコンテナが常駐しており、サーブレットが動作するのはあくまでコンテナ内部になるので、新たな実行プロセスが起動するというようなことはない。したがって、リソースの消費も少なく、動作自体が軽く、結果的にユーザ数の増減にも比較的強い耐性が保証されている。

サーブレットは、非常に強力なサーバサイド技術である。しかし、まったく欠点がないというわけではない。Java そのものであるので、静的な HTML を大量に出力したいという場合、どうしても記述が冗長になってしまう。また、プログラムロジックとは関係ないコンテンツを修正したい場合も含め、修正のたびにコンパイル作業を行わなければならない。

一方 JSP（Java Server Pages）<sup>[4]</sup> は HTML 埋め込み型のサーバサイド処理環境で、原則、サーブレットの問題点を解決するために開発された。以下に JSP とサーブレットの違いを簡単にまとめておく。

（１）サーブレットは HTML を出力するが、JSP は HTML に埋め込まれる

## （２）JSP はコンパイルを必要としない

サーブレットはあらかじめソースをコンパイルしておくが必要である。しかし、JSP は事前のコンパイルを必要としない。修正したソースをそのままサーバ上に置きさえすれば、動作させることができる。そのまま動作させることができるということは、実行の度に解析作業が必要となり、サーブレットよりも処理効率が悪いのではないかと思うかもしれない。そのようなことはなく、JSP は、最初に呼び出されたタイミングで一度サーブレットへ変換され、コンパイルされた後にメモリ上に常駐する。つまり、1度目の要求時には若干の遅れを感じるかもしれないが、2回目以降の要求にはサーブレットと同等のパフォーマンスが確約されている。

本稿で開発したプログラムはサーバ側で JSP を使って、データベースを検索したデータを使い、HTML と JavaScript からなるページを生成し、それをクライアントに返すという方法を取っている。JSP / サーブレットを動かすためには、次のソフトウェアを利用している。

## （１）JSP / サーブレットコンテナ

JSP / サーブレットを処理する時は、JSP / サーブレットコンテナと呼ばれる実行エンジンを利用する必要がある。このプログラムでは、フリーウェアでユーザ層も広い Tomcat<sup>[5]</sup> を採用した。

Tomcat は単体でも Web サーバの役割を果たすことが可能であるが、サーバとして開発されたわけではないので、Web サーバとしての処理能力は高くない。通常は Web サーバを別にインストールしておき、Tomcat ではサーブレット / JSP の処理だけを行うという方式が採られている。本システムでは JSP しか利用しないので Tomcat だけをインストールしている。

## （２）データベース

JSP / サーブレットにデータベースが必ず必要というわけではない。しかし、大量のデータを管理する場合、データベースがないとセキュリティ上の問題や処理速度の低下が問題になってくる。本システムではフリーウェアである MYSQL<sup>[6]</sup> を採用したが、領域の形状を記憶するには幾何データ型をもつ Postgres<sup>[7]</sup> のほうが、適当であったかもしれない。

# 第3章 本プログラムの使用方法

この章では、本プログラムの使用方法の手順を説明していく。

## 3.1 データの登録方法

本研究で使用する歴史情報をあらかじめデータベースに登録しておく必要がある。本節ではデータベースの構成とデータ登録を説明する。

### （１）歴史データベースのテーブル

MYSQL で表1のような歴史データベースのテーブルを作成した。

history テーブルは左から、史実の発生年 (year)、その年の何番目の史実かを表す番号 (no)、その史実の重要度 (imp)、その史実が起こった場所 (place)、史実の内容 (history)、史実の内容説明 (content)、史実が起こった場所の緯度 (lngspan)、史実が起こった場所の経度 (latspan) の順番で並んでいる。主キーは (year,no) で、特定の史実の修正や削除に使用される。

緯度と経度の取得方法として、無料でサービスを使用できるサイト「Geocoding」を利用した<sup>[8]</sup>。このサイトの使用法は、まず、トップページで住所やキー

表 1：史実と国境を表現した関係スキーマ

history				
<u>year</u>	<u>no</u>	imp	place	history
content	lngspan	latspan		

nations			
<u>nation</u>	<u>seq</u>	s_year	e_year
lngspan	latspan	color	



ワードを入力する。するとその場所の地図と緯度・経度が表示される。キーワード検索で候補が複数あった場合、ハイパーリンクが候補の数だけ表示され、そのハイパーリンクをクリックすると地図と緯度・経度が表示される。

重要度の決め方は次の通りである。小学校の社会の教科書に記載されている史実は重要度1とした。中学校の社会の教科書に記載されている史実、高校の日本史・世界史の教科書に記載されている史実の重要度をそれぞれ2、3とした。nations テーブルは国境を示す閉曲線を緯度経度の列として表現したものである。その属性は左から nation (国名)、seq (座標点の順序番号)、s\_year (その国境であった始まりの年)、e\_year (その国境であった終わりの年)、緯度経度、color (色) である。color はその領域を塗りつぶす色で、16進6桁で、利用者が任意に決めることができる。この表は国境だけでなく、例えば「織田信長の勢力範囲」というような、意味のある領域を地図上に表示するためにも利用できる。

## (2) 歴史データベースへの登録方法

歴史データベースへの登録方法は DBMS から SQL 文によって行う方法と外部からデータベースへ接続し、登録する方法がある。この歴史データベースに外部からデータを登録するための、登録プログラムを作成した。図2はこのプログラムによる登録画面である。入力項目はほぼ history テーブルの属性順に並んでいる。図2の下に登録・取り消しボタンがあるが、登録ボタンを押すと入力したデータがデータベースに書き込まれる。また、取り消しボタンを押すと、その時に入力した内容がクリアされ、初期化されるだけである。

## データベース登録

年代	<input type="text"/>
重要度	<input type="text"/>
場所	<input type="text"/>
出来事	<input type="text"/>
当事者・国	<input type="text"/>
緯度	<input type="text"/>
経度	<input type="text"/>
<input type="button" value="登録"/> <input type="button" value="取消"/>	

図2 歴史データベース登録画面

## 3.2 システム起動後の画面

本プログラムを起動させると図3の画面が表示される。この画面では検索方法と表示方法の2つを決めることができる。検索方法としては、キーワード検索と年代検索とがある。

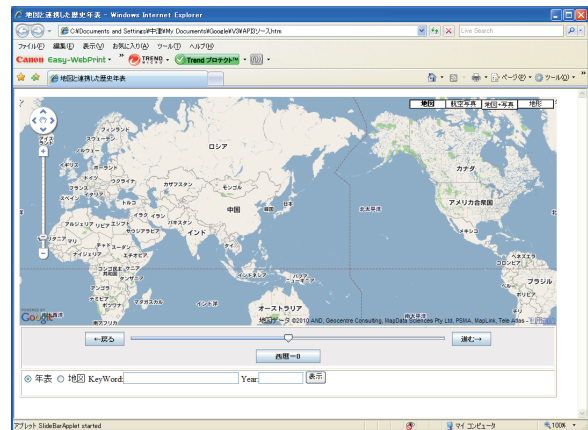


図3 初期画面

キーワード検索ならば検索したいキーワードを入力する。年代検索の場合は、画面下部のスライドバーを操作して大まかな年を決め、「進む→」あるいは「戻る←」ボタンで1年ごとに年を増減させ、最後に西暦年が表示されたボタンを押せば、その値が Year 領域にセットされる。あるいは、Year 領域に直接キー入力してもよい。

キーワード検索は入力したキーワードに関係するデータをすべて検索する。年代検索は入力した年の前後のデータをすべて検索するものである。

表示方法としては、年表表示と地図表示とがあり、表示したいほうの前についているラジオボタンをクリックして選択する。

キーワード又は年を入力し、年表表示又は地図表示を選んだあと「表示」というボタンを押すと、それぞれの入力されたデータに対応したページに進む。キーワードおよび年の両方に入力された場合はキーワード検索が優先される。

## 3.3 年表表示を選んだ場合

キーワードを入力するか、年を入力し、年表のラジオボタンを選択して「表示」のボタンを押すと、図4のような画面が出てくる。図4は年として1603を指定した場合の結果である。

年代	重要度	場所	内容	備考	緯度	経度
1588	2	日本全国	刀狩令の発布	秀吉が農民の武器を没収	34.686	135.526
1590	1	大阪	日本統一達成!	豊臣秀吉	34.686	135.526
1600	2	関ヶ原	関ヶ原の戦い	家康の勝ち	35.34	136.44
1600	2	インド	東インド会社が設立	英国がインドを植民地に	28.55	77
1603	2	江戸	徳川家康が征夷大将軍に任命される	徳川家康が江戸幕府を開く	35.685	139.7522
1603	1	江戸	江戸幕府が始まる	第1代将軍徳川家康	35.685	139.7522
1614	3	大阪城	大阪冬の陣	徳川が豊臣を滅ぼす	34.686	135.526
1615	3	大阪城	大阪夏の陣	徳川が豊臣を滅ぼす	34.686	135.526

図4 1603年で検索した年表表示画面

左から年、その歴史の重要度、史実が起きた場所、史実の内容、史実の内容説明、史実が起きた場所の緯度、経度の順に並んで表示される。

キーワード検索の場合、入力された文字列と史実の内容説明の一部が一致していれば出力する。年代検索の場合、入力された年から前後20年の史実を選んで出力する。データは上から年の古い順に並べられている。画面下の戻るボタンを押すと、初期画面（図3）へと戻る。

### 3.4 地図表示を選んだ場合

キーワードを入力するか、年を入力し、地図のラジオボタンを選択して「表示」のボタンを押すと、図5のような画面が表示される。



図5 キーワード「戦争」で検索した結果

#### 3.4.1 キーワードを入力した画面

キーワード検索を行った場合、入力された文字列と史実の内容説明の一部が一致していれば出力する。地図上に表示されるアイコンの色は、その史実に付けられた重要度を示し、重要度1, 2, 3をそれぞれ赤、青、緑とした。

#### 3.4.2 年を入力した場合

年代検索の場合、入力された年から前後20年の史実を選んで出力する。原則として1つの史実は1つのアイコンで表現されている。アイコンの中のアルファベットは年代順になっている（図6）。さらに指定された年に一致する史実を示すアイコンが他のアイコンよりも大きいものとなっており、これによって、指定した年より前に起きた史実か後に起きた史実かが区別できる。アイコンが狭い範囲に集中している場合には、その部分の地図を拡大して表示すればよい（図7）。

また nations 表で [s\_year, e\_year] の範囲に含まれる年が指定された場合、それに対応した領域が地図上に指定した色で表示される（図6）。領域の上部には黄色のアイコンが表示され、アイコンをクリックすればその領域の説明が表示される。

#### 3.4.3 アイコンをクリックした画面

地図上にあるアイコンをクリックすると、史実の説明部分が表示されるようになっている。表示される内容は、発生年と内容である。また、同じ場所で発生した史実は1つのアイコンで表示され、そのアイコンをクリックした場合はそれらがまとめて表示されるようにしている（図8）。



図6 1603年で検索した地図表示の結果



図7 図6の拡大地図



図8 図6でEのアイコンをクリックした場合

## 第4章 地図上での情報表示の工夫

この章では地図を表示する場合に、利用者の欲しがっている情報をいかに伝わりやすく、またいかに見やすいものにするかを考え工夫した点を説明する。

### 4.1 情報量に応じて表示する年の範囲を限定

10年間で大きく歴史が動いてしまう時代もあれば、10年間まったく事件がおきず何も変わらない時代もある。例をあげると、宗教革命がおきていた時代や戦国時代などは革命や戦が起こる度に勢力図がかわってしまう。それとは反対に江戸時代や平安時代などの安定期に入ってしまった時代では小さな戦や小規模の反乱はあるかもしれないが、国自体に大きな変化はない。このように同じ10年間でも歴史の密度が変わることがたびたびある。この問題を解決するためには、指定された年の前後20年間に何件の歴史データがあるかを調べ、一定件数以上なら検索範囲を前後15年のように5年縮小したり、件数が少なければ検索範囲を前後30年のように5年拡大したりすればよいと思われる。

### 4.2 アイコンの色を使った重要度の区別

地図上のすべてのアイコンを同じ色で表示してしまうと、どれが小学校や中学校の義務教育までで習う基礎知識で、どれが高校以上で習う専門知識なのかわからなくなってしまう。そこで、地図上のアイコンの色を変えることによって、知識レベルがわかりやすくなると考えた。そこで史実の重要度に応じてアイコンの色を変えることにした。また領域は塗り潰して表示されるが、下になった地図が見えなくなることを防止するために、境界線以外は透明色で塗り潰すようにした。

### 4.3 アイコンの大きさと発生順の区別

このシステムでは指定された年と、その年の前後20年の史実も一緒に表示する。表示されたアイコンがどれも同じでは、史実の発生順がわからなくなる。そのため、検索された史実を表すアイコンに年代順にアルファベットをつけることにした。また検索者の指定した年と一致する年に起きた史実には大きなアイコンで、その前後の年に起きた史実は少し小さいアイコンで表示するという工夫を行った。

同じ場所で起きた史実が複数ある場合は、1つのアイコンにまとめることにした。まとめられた複数の史実の中に指定された年のものが含まれている場合は大きなアイコンで表示される。

### 4.4 史実の詳細情報の表示

表示されたアイコンをクリックすると、その史実の簡易な内容を吹き出し形式で表示するようにした。吹き出しを常時表示しなかったのは、地図上に表示され

る情報を少なくし、地図自体を見やすくするためである。

Google Map API バージョン3からは吹き出しの中にHTMLのコードやハイパーリンクを埋め込むことが可能になり、このリンクをクリックすれば任意のWebページを表示できるようになり、より詳しい内容の記述が可能になった。

## 第5章 まとめ

本研究では開発できなかったことや今後の課題を以下に記す。

- ・地図の縮尺によって情報アイコンをまとめたり、ばらしたりすると地図が見やすくなるかもしれない。
- ・年代によって国が変わったり国境が変わったりするのは、歴史を学ぶ上で最も重要な要素であるが、本システムではデータとして予め登録されたものしか表示できない。
- ・事件や出来事のように発生年が決まっており、地点と関係している史実はこのシステムで表現できるが、安土桃山時代といった漠然とした時代や、「浮世絵が流行する」や「ルネサンスが始まる」といった社会現象などは今のままでは表現が難しい。
- ・歴史とは、離散的な事実の集合ではなく、いくつもの事実が時間の流れとともに、相互に関連しあって、次の事実につながってゆく。歴史の流れといった情報の表現が残された研究課題である。

地図と連動して表示したほうがわかりやすい情報は歴史だけとは限らない。位置情報とともにデータベース化された情報なら、どんなものでも表示できるはずで、任意のデータを地図と連動して表示するシステムという一般化も残された課題である。

今後本研究を進めていき、教育の手助けになるシステムにしたいと考えている。

### [ 参考資料 ]

- [1] <http://www.h3.dion.ne.jp/~urutora/sekainepeji.htm>
- [2] <http://castle.jpn.org/map/japan.html>
- [3] <http://code.google.com/intl/ja/apis/maps/documentation/javascript/tutorial.html>
- [4] 山田祥寛、「10日でおぼえる JSP / サーブレット入門」、翔泳社（2002）
- [5] <http://www.jajakarta.org/tomcat/>
- [6] <http://www.mysql.gr.jp/>
- [7] <http://www.postgresql.jp/document/pg74doc/html/index.html>
- [8] <http://www.geocoding.jp/>

（2010年9月9日受理）