Haaga-Helia
University of Applied Sciences

How to determine the most suitable map API?
- web development targeted for Korean market

Taehee Lee

Bachelor's Thesis
Degree Programme in Business
Information Technology
2021

Haaga-Helia
University of Applied Sciences

Abstract

19.04.2021

Author(s)
Taehee Lee

Degree programme
Business Information Technology

Report/thesis title	**Number of pages and appendix pages**
How to determine the most suitable map API? - web development targeted for Korean market	**45 + 3**

Applying web APIs has become a common way of web development since APIs allow developers to access required data in faster and simpler ways. However, developers often face difficulties to determine the correct web APIs which can resolve the development requirements in the most efficient way.

When map data are required in the websites, developers first come up with Google Map API as Google has been the map API category leader and applied to over 1,000,000 websites worldwide. However, web API determination should reflect a lot more considerations besides its popularity to successfully achieve development objectives.

In Korea, many map service providers are providing various features and Korean government applies strict compliances to ap service. Consequently, developers should consider local service providers as well as global service providers when selecting map API for the websites targeted for Korean market. In this regard, however, API directories such as ProgrammableWeb and RapidAPI do not provide up to date information about Korean map API service providers.

Accordingly, this research demonstrates the way to select candidate map APIs according to the development requirements. To be specific, this research selects Google, Naver and Kakao as candidate map APIs based on the popularity, awareness in the market and compliances.

Then, it also explains the way to compare candidates to find the most suitable map API for the given development requirements. This research analyzes the most popular use cases of map APIs to define the development requirements to be applied to the demo web pages in the empirical phase. In the empirical phase, this research collects and analyzes data to reflect both developer's and end users' perspectives about candidate map APIS.

Lastly, this research recommends the most suitable map API according to the different development requirements. The research conclusion can help developers who are making websites including location identification and/or route recommendation in Korea.

Keywords
Web API, Public API, Map API, Google Maps API, Map service, Korean map

Table of contents

1 Introduction 1
2 Research Background 3
2.1 Research Objective 3
2.2 Research Questions 3
2.3 Research Methodology 3
2.4 Scope of the research 3
2.5 Out of Scope 4
3 Web APIs Background 5
3.1 History of web APIs 5
3.1.1 Web APIs for commercial companies 6
3.1.2 Web APIs for social networks 6
3.1.3 Web APIs for cloud technology 7
3.1.4 Web APIs for mobile platform 7
3.1.5 The future of web APIs 7
3.2 Types of APIs 8
3.2.1 Customers of APIs 8
3.2.2 API Architectures 9
3.2.3 JavaScript APIs 13
3.2.4 Type of map APIs 13
3.3 Conditions of good APIs 13
3.3.1 Developers 14
3.3.2 End users 15
4 Map APIs 16
4.1 Basics of map Service 16
4.1.1 Location Identification 16
4.1.2 Route Recommendation 17
4.2 Global service providers 18
4.3 Local service providers in Korea 18
4.4 Compliance regarding Map Services in Korea 19
4.5 Use cases 20
4.6 Map APIs for the project 21
4.6.1 Map API candidates 21
4.6.2 Map API features 21
5 Empirical Section 22
5.1 Project Background 22
5.1.1 Objective 22
5.1.2 Map API Comparison Matrix 23

5.1.3 Assumption 24
5.2 Project Phase 25
5.2.1 Phase 1: Select products 25
5.2.2 Phase 2: Data collection from developer resources 26
5.2.3 Phase 3: Data collection from API code 27
5.2.4 Phase 4: Data collection from route recommendation results 29
5.2.5 Phase 5: Data collection from user reviews 31
5.2.6 Phase 6: Data analysis 32
5.3 Project Result 40
6 Conclusion 44
6.1 Summary 44
6.2 Discussion 45
References 46

Terms and Abbreviation

API	**Application Programming Interface**
CLI	**Command Line Interface**
HTTP	**HyperText Transfer Protocol,**
JSON	**JavaScript Object Notation**
MAU	**Monthly Activity User**
REST	**Representational State Transfer**
RPC	**Remote Procedure Call**
SaaS	**Software as a Service**
SOA	**Service Oriented Architecture**
SOAP	**Simple Object Access Protocol**
W3C	**World Wide Web Consortium**
XML	**Extensible Markup Language**

1 Introduction

Web API allows other web applications to access the specific data or features through the HTTP protocol. Web APIs assist developers to easily achieve their development objectives by only exposing the simple syntax to developers and abstract complex code from them. As a result, embedding APIs has become a very common way of web development process nowadays (Bhuiyan, Begum, Rahman, & Hadid 2018).

Map API is one of the most frequently used web APIs and it provides a handy way to include map services to the websites. That is, map API providers usually have their own map services which is the source of data for map API. The map service and API category were emerged by Google in 2005 and Google has been the global category leader for last 15 years. Consequently, although there are several map services and API providers such as Bing Maps, MapBox, etc., Google Map API comes as the first candidate when developers need to apply map services to the websites based on the popularity of its map services and map API (RapidAPI 2018; Reid 2020).

Selecting map API should reflect not only its popularity but also other factors such as awareness in the market and compliances. That is, developers should consider both international and local service providers to select map API candidates.

However, developers are often facing difficulties in selecting proper web API as there are so many APIs available. Determining the most suitable API is a time-consuming process for many developers. Also, developers do not have sufficient background knowledge about all candidate web APIs which make determination more difficult. In addition, APIs which are manually selected by developers often fail to resolve specific tasks in the most efficient way. In other words, it is important for developers to select the right API to achieve development purpose (Torres & Tapia 2011; Qi et al. 2019).

API directories such as ProgrammableWeb and RapidAPI were initiated to resolve such difficulties developers faced and they provide collaborative information about released web APIs. The objective of these services is to assist developers' searching, comparing and selecting the most suitable API which fits to the development purpose (Doerrfeld 2015).

In this regard, however, aforesaid API directories do not provide the up-to-date information of Korean local map API providers. That is, developers who create websites for Korean market can not understand the information about possible map API candidates

and the lack of information on local Map API providers may lead developers to select API which cannot satisfy development purpose.

As a result, the proposed thesis would provide candidate map APIs for the web development project targeted for Korean market. Then, the research will also introduce the way to determine the most suitable map API out of candidates according to the development requirements.

2 Research Background

2.1 Research Objective

The objective of this research is to find the candidate map APIs when developing web application targeted for Korean market. Then, it will also provide the way to determine the most suitable map API out of the candidate map APIs. The research project will provide a map API comparison matrix to recommend the most suitable map API according to the various development requirements.

2.2 Research Questions

When developing website which requires map API for Korean market:
- How to select candidate map APIs?
- How to compare candidate map APIs?
- Should developers use different map API for different development requirements?
- Which map API is the most suitable for websites targeted for Korean market?

2.3 Research Methodology

During the empirical process, research collects data from three different sources and this research takes a mixed method research in sequential multi-phase. That is, data analysis of this thesis project applies both quantitative and qualitative methods and they are applied in a series.

First source of data is from the candidate Map APIs' documentation and code. Qualitative approach is applied to study these data. Second source of data is from experiment conducted by this thesis project and they are analysed in quantitative method. Lastly, research collects user reviews from Google play store and Apple app store to understand end users' opinions on each map services. These reviews are examined in the qualitative method.

2.4 Scope of the research

This research project will apply map APIs only for websites which require JavaScript code. Also, research limits the use case of websites that it only requires the Map data about Korea.

Map API can provide a wide range of features such as location identification, route recommendation, location information, etc. In this regard, this research will only apply the most frequently applied two features of map APIs based on the use case analysis.

As for the route recommendation which is one of the features provided by map APIs, this thesis only considers the route recommendation by public transportation.

2.5 Out of Scope

Due to the repetitive process, the research project does not separately proceed mobile application development. Also, the research does not consider use cases which are requiring map data about foreign countries for Korean customer because Korean map API service providers do not provide this feature.

Route recommendation by car drivers or walking passengers are excluded from this thesis project because this service is not supported by foreign map API service providers in Korea. Also, additional features of map services such as 3D view, road view and satellite view are excluded as they are only applied for specific industries which have specific business objectives.

Lastly, this study does not examine the security level of each of candidate map API due to the limited resources of this project.

3 Web APIs Background

Chapter 3.1 introduces the history of web APIs in a chronological order. This will provide the opportunity to understand the past, current and future of web APIs. Then, chapter 3.2 explains the several way of classifying web APIs. Lastly, chapter 3.3 summarizes the conditions of good web APIs.

3.1 History of web APIs

History of web APIs is highly related to the desire of sharing data over the network. HTTP enabled sending and requesting data and W3C developed XML. XML is structured as human as well as machine readable language and became the popular way of storing and transporting data in the client-server system because it overcame the complexity of SGML. SGML was used to define the structures of document in the complicated form which XML was based on. XML was soon standardized as SOAP which is a messaging protocol for XML-based message over HTTP. XML Soap was the most common way of exchanging data from the late 1990's until early 2000's as it enabled easier data sharing regardless platforms. However, there were some dissatisfactions arose due to its verbose and redundant syntax and larger file size (Kopecky, Fremantle & Boakes 2014).

In early 2000s, service-oriented architecture (SOA) was introduced as a new type of architecture which aims to encapsulate each software component into an individual service or web API. These individualized services were invoked by HTTP protocols and accelerate the rapid growth of modern web API. The modern web API has no standards but became popular with RESTful architecture and JSON format. Hence, we can say that modern web API is the interface of accessing the machine-readable data by HTTP and transform the data into the JSON or XML format (Espinha, Zaidman, & Gross 2015).

Since modern web API became one typical way of business, many web APIs were released to the market after 2000. According to DuVander (2013), the number of web APIs on the API directories increased from 105 in 2005 to 9,000 in 2013. Figure 1 supports the idea that modern web APIs have been evolved by the companies in various industries who have realized the importance of API for their business after 2000.

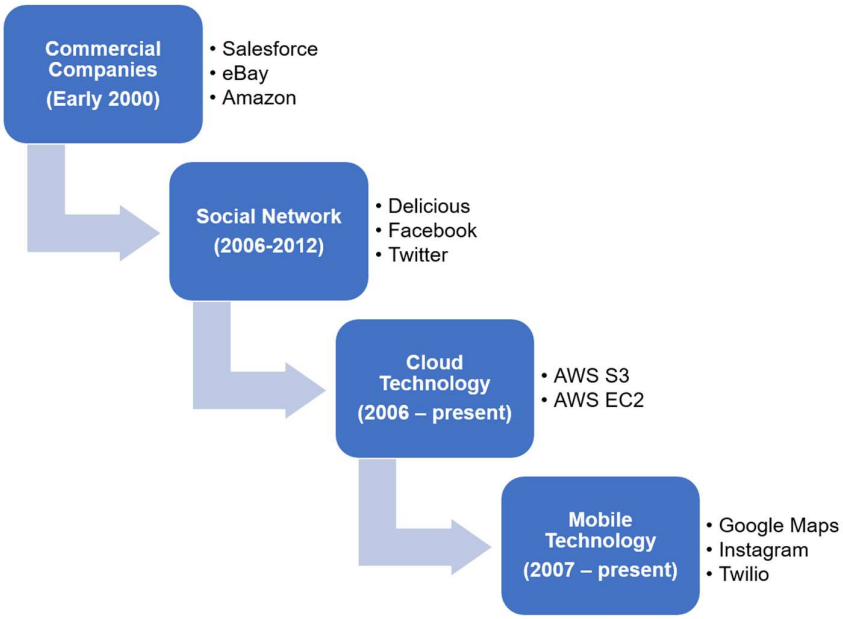


Figure 1. History of modern web APIs (adapted from Lane)

3.1.1 Web APIs for commercial companies

The modern web APIs were actively applied for the commercial startups who wanted to extend their services and products to the larger market and customer groups. This was led by 3 companies which are Salesforce, eBay and Amazon in early 2000. Salesforce started its XML API in 2000 for web-based Salesforce automation and this is considered as the first SaaS. Later in the same year, eBay launched eBay API together with its developers' program. eBay's API was initially limited to the licensed partners only, however, eBay gradually expanded it to the external developers. Now, eBay API became the standard of online shop business which allows partners and third part resellers to access their platform by using web API. Lastly, Amazon started its web services in 2002 which allowed developers and third-party sites to access product contents from Amazon in an XML format and incorporate data into the own websites (Lane 2019).

3.1.2 Web APIs for social networks

The first evolution of web APIs was made by social network services between 2006 and 2012. Delicious, Flickr, Facebook and Twitter changed the paradigms of sharing and exchanging information with other people. Unlike APIs used by commercial industries in the previous phase, these companies did not use APIs to generate the direct financial values for the business. Instead, APIs in the social networks were used as a platform which allows external developers to access to the functionalities and features of social networks.

For example, Flickr launched their API after 6 months of opening its photo sharing services in 2004. This API immediately made Flickr as the popular image platform as users were easily able to upload images on the web pages and social media simply through Flickr's API (Lane 2019).

3.1.3 Web APIs for cloud technology

While social networks were taking the most popularity of using web APIs, cloud technology was started to innovate the way of business together with web APIs. This innovation was mostly carried by Amazon and they fundamentally changed the perspectives on how to do business online. Amazon internally applied API-focused approach through the entire organization. That is, all digital resources were required to be shared by API in the Amazon. This approach drastically transformed the way to view and treat digital resources and they released Amazon Simple Storage (S3) in 2006. Amazon S3 allowed to access data simply via API and CLI. Only 6 months later, Amazon launched Amazon Elastic Compute (EC2) which offers cloud-based servers for developers. These services proved that APIs enable infrastructure deployment, revenue generation and fundamental innovation in business together with cloud technology (Lane 2019).

3.1.4 Web APIs for mobile platform

Based on the foundation made by commerce, social network and cloud, mobile platform made APIs as an inseparable part of development process. Since iPhone and Android were launched in 2007, there have been lots of startups who applied API-first approach to provide emerging resources and applications for mobile platform. For example, Twilio which was launched as an API-as-a-Product in 2007, provided phone call services over the cloud application and resources for the voice enabled applications. The increased usages of web APIs are not limited to the mobile platform. Nowadays, technologies are applying modern web APIs for the effective data management for any objects on the network as well as mobile (Lane 2019).

3.1.5 The future of web APIs

It is difficult to ensure future, however, analyzing the current trends delivers some insights about expected future of web APIs. This chapter introduces ongoing discussions about the future of web APIs.

Better API experiences on the mobile interface

Thanks to HTTP3 which will provide better network coverage and packet transmission under the low network coverage, web APIs which are a big part of mobile development will provide faster performance.

Increasing popularity on Microservices will increase the demand of APIs
As modern web API's history was accelerated after SOA was introduced, the increasing popularity of microservices will also increase the demands on the web APIs because developers will define backend in loosely coupled fragments which are connected by APIs under the microservices architecture.

Besides above, API standard, API driven architecture, API security are frequently nominated topics when it comes to future of web APIs (PURKAYASTHA 2020).

3.2 Types of APIs

There are several ways to classify APIs and this chapter introduces three different ways to categorize APIs.

3.2.1 Customers of APIs

APIs can be classified depending on the customers of APIs. This includes three different types which are Internal, Partner and External APIs. Each type has different level of publicity which eventually have different customers (Figure 2).

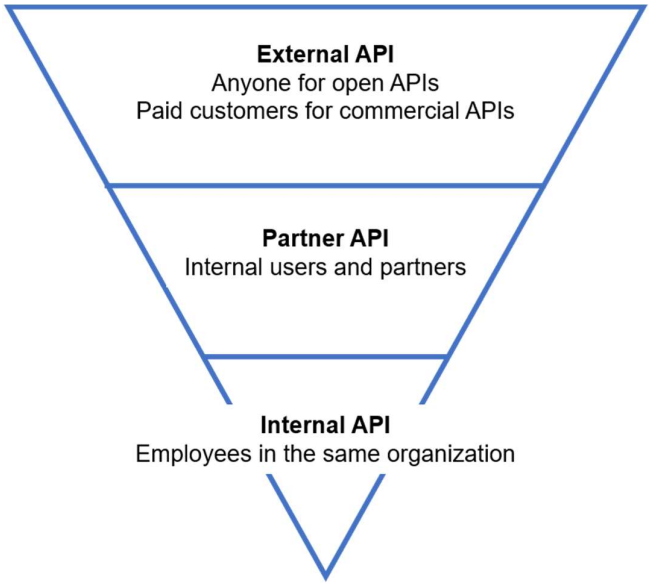


Figure 2. Customers for different API types (adapted from Jin, Sahni & Shevat)

Internal APIs are private and only used with the internal team, department, company, or organization. Hence, the customer for this type is employees within the same organization. Partner APIs are still private, but it is shared with the specific partners as well as employees within the internal organization. External APIs are also called as public APIs and which can be either open APIs or commercial APIs. The open APIs are publicly available for anyone on the Internet while commercial APIs are only available for customers who paid to use the API services (Lane 2020; rapidapi.com 2021).

In this regard, type of one API in this category can be changed from one to another according to the changes of business objectives. This means that the internal APIs can become the external APIs and vice versa. For example, Slack channel was initially developed for its internal developer's platform. Later, they improved this internal platform and expanded its services to the public customer as a message platform (Jin, Sahni & Shevat 2018, 4-6).

3.2.2 API Architectures

Another way of classifying APIs is based on its architectures and protocols. This chapter introduces two types of API architectures which are 'Request-Response APIs' and 'Event-Driven APIs'.

Firstly, Request–response APIs usually provide an interface with HTTP-based web server and endpoints. As a result, client can send HTTP requests to the defined endpoints and the server send responses in either JSON or XML format. Most common paradigms in this category are REST, RPC and GraphQL (table 1).

Table 1. Different types of Request-Response APIs (adapted from Jin, Sahni & Shevat)

Type	REST	RPC	GraphQL
Definition	Provides CRUD operations by HTTP methods on data which are treated as resources	Provides API methods which focus on the action-based requests	Provides a query language for API so that clients decide the structures of response
HTTP methods	GET, POST, PUT, DELETE, PATCH	GET, POST	GET, POST

When to Use?	For APIs need CRUD operations on the resources	For APIs revealing actions than encapsulated resources	For APIs require flexibility in queries

REST is the most popular type of web APIs nowadays. REST APIs treat data as resources and allow CRUD transactions on the data by standard HTTP methods which are GET, POST, PUT, PATCH and DELETE. Figure 3 is the example of requesting the information of student whose student id is 113 to the host server which uses REST API.

```
$ curl -X GET http://{hostserver}/students/113
```

Figure 3. Example of REST request

RPC is one of the simplest API paradigms which lets client to pass both method name and arguments to server. Unlike REST focuses on the resources, RPC cares the action (figure 4).

```
$ curl -X GET http://{hostserver}/students.get?id=13
```

Figure 4. Example of RPC request

GraphQL was created and publicly released by Facebook and it is getting significant attention by developers. As figure 5 illustrates, GraphQL allows clients to define and request a specific structure of data unlike REST and RPC rely on the data structures predefined by server. Then, server sends response in the requested structure to the client (Jin, Sahni & Shevat 2018, 9-12).

```
1  query ($id:String!) {
2   student (login: $id) {
3      name
4      department
5   }
6  }
```

Figure 5. Example of GraphQL request

Secondly, Event-Driven APIs are ideal for developers who need polling to get up to date information about data in a predetermined frequency. WebHooks, WebSockets, HTTP streaming are the most famous mechanisms in this category (table 2).

Table 2. Different types of Event-Driven APIs (adapted from Jin, Sahni & Shevat)

Type	WebHooks	WebSockets	HTTP streaming
Definition	Event notification can be polled by using HTTP call back	Two-way streaming connection by using TCP	Long-lived client-server connection by using HTTP
Use Cases	GitHub sends a POST request to payload URL once there are updates with user selected events	- Trello sends changes made by other users from server to browser - Blockchain pushes real-time notifications about transactions	Twitter pushes data between an app and its streaming API to push data like new tweets with a single connection opened by clients
When to use?	For server-to-server real time events notification	For real time and interactive communication between client and server	For server to client long-lived communication

WebHooks accept one HTTP method, POST to automatically receive messages to the configured URL when event happens. Unlike polling requires clients to continuously request and check updates from API, WebHook allows for clients to automatically receive real time updates from the server by simply creating one more endpoint to receive this event notification (figure6). For example, Slack allows users to track channel and GitHub sends event notification by configuring WebHooks.

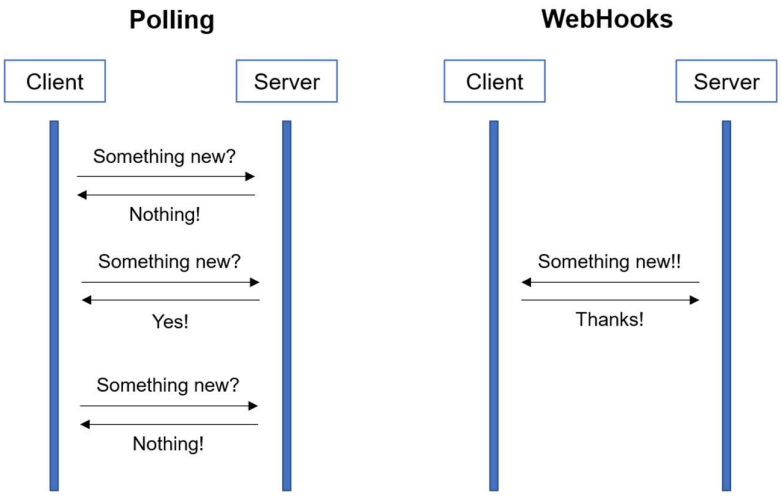


Figure 6. The differences between Polling and WebHooks (adapted from Jin, Sahni & Shevat)

WebSockets, on the other hand, enables a two-way streaming communication session between the client and server. Unlike WebHooks, WebSockets do not require callback URL registration. Also, WebSockets support long-lived open connection, bidirectional connection and error handling which are not provided in WebHooks (figure 7). WebSockets allow clients to send message to server and receive event-driven responses automatically without polling server to reply.

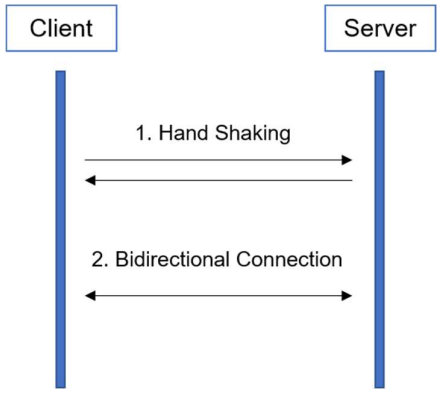


Figure 7. Client-Server communication of WebSockets (adapted from Jin, Sahni & Shevat)

HTTP streaming API allows server to push infinite length of data with only single connection opened by client's request. As figure 8 demonstrates, HTTP streaming is one way connection unlike WebSockets support bidirectional connection (Jin, Sahni & Shevat 2018, 13-16).

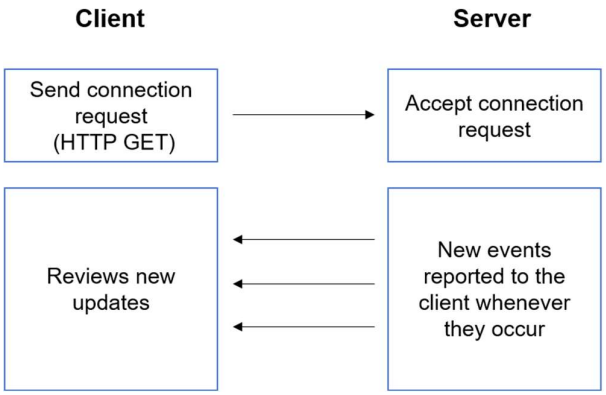


Figure 8. Client-Server communication of HTTP Streaming (adapted from Jin, Sahni & Shevat)

3.2.3 JavaScript APIs

When we talk about web APIs, it is important to understand the client-side JavaScript APIs. According to MDN Web Docs (2021), client-side APIs for JavaScript can be classified in two different types which are browser APIs and third-party APIs.

Browser APIs are built-in with the web browsers by default so that the developers can easily perform complex operations without handling the low-level code. For example, web audio API makes easier audio control on the browser, canvas API allows easier drawing by JavaScript and HTML and fetch API makes it easier to request and handle resources.

Third-party APIs are not installed to the web browsers by default, but developers should retrieve code and information from third-party's resources. Then, developers explicitly write JavaScript code to include data accessed by third-party APIs to their own websites. For example, twitter API can be used to display data from tweets, map APIs can be used to apply map information, YouTube API can be used to embed YouTube videos or searching.

3.2.4 Type of map APIs

Based on the three different approaches of classifying web APIs introduced above, research defined the type of map API as below:

- External API: map API is available for anyone who wants to use service provider's map services and data.
- Request-response API (REST): client requests through HTTP protocols but do not specify the data structure in the request. Server sends response data in the predefined structure over the internet.
- Third-party JavaScript API: map API is not built into the browsers by default. Instead, it should be explicitly embedded by developers to their own JavaScript code.

3.3 Conditions of good APIs

There are no unified standards which can be applied for all types of web APIs' evaluation. Thus, this chapter introduces conditions for good web APIs from literature review. The study results of this chapter will be used as the ground when evaluating candidate map APIs in chapter 5. Empirical section.

3.3.1 Developers

As developers are the main customers of APIs, the most important condition of the API assessment is developers' perspectives. Zibran (2008) underlines the importance of developers' opinions by expressing that good APIs should provide great usability for developers and satisfy the development requirements. Accordingly, this research first explores the three important considerations from developers when they are selecting APIs and they are developer resources, code and data reliability.

Developer resources include many different supports which are offered by web API providers. Developer resources highly influence developers when they select the third-party web API for their development. One of the first developer resources is the API Documentation. API Documentation is the interface where many APIs are first introduced to developers and developers get to know and learn about APIs. Good API documentation requires QuickStart guide, Authentication guide, Tutorials and Code snippets to provide the objective and best practices of using web APIs (Jin, Sahni & Shevat 2018).

Besides documentation, other developer resources such as developer community and debugging and troubleshooting supports are also important resources for developers. To make web APIs successful, developer resources should be rich and properly delivered to the target developers who have development requirements which can be solved by the web API (Jin, Sahni & Shevat 2018; Bush 2019).

Secondly, code is another important consideration for developers who determine the web APIs for their development process. Even though it is difficult to access the logic behind the web API as an external developer, there are still means of evaluating the quality of API code. The developers can evaluate the abstract API code which to be applied to their own JavaScript code. In this regard, the good API code guarantees simplicity, consistency and symmetry and useful abstractions (Monitis 2019).

Lastly, developers value data reliability when selecting web APIs to be embedded to their websites. According to APInf's survey (2017), more than half of the survey participants answered that the data reliability is the most important features when selecting the third-party web APIs. That is, good web APIs should provide data which are correct and accurate so that developers can create their own services based on the reliable data accessed by web API.

3.3.2 End users

Meanwhile, the destination of API value chain is the group of people who is going to use developed websites (Figure 9). In this research, we call those who consume developed web sites as end users.

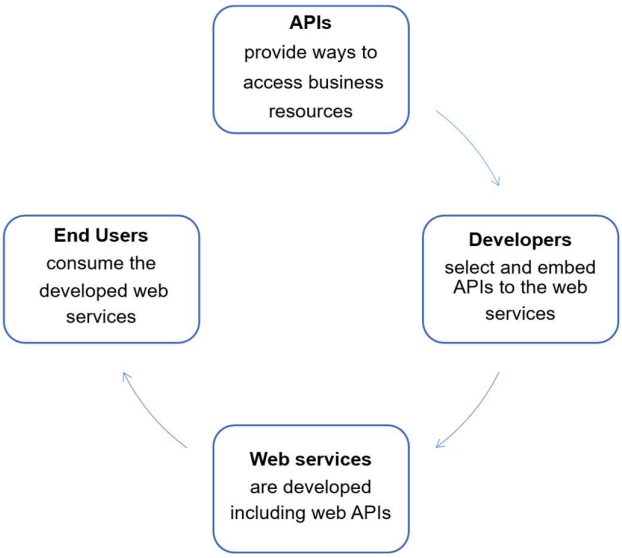


Figure 9. Web API Ecosystem (adapted from Jacobson & Woods)

According to the API value chain, it is important to meet end users' requirements as they are the entities who will bring up the values to the business based on their satisfaction level about website. As a result, good APIs should satisfy not only developers but also end users and developers should consider end users when selecting web API (Jacobson & Woods 2011).

Altogether, good web APIs should provide sufficient developer resources, abstract code and reliable data to satisfy developer's expectation. In addition, developers should take consider of end users' perspective to improve the satisfaction level of developed websites.

4 Map APIs

This chapter clarifies the background of map APIs. Chapter 4.1 outlines how Map services work. Then, Map API providers in the global and Korean markets are introduced in chapter 4.2 and 4.3, respectively. Chapter 4.5 explores the most common use cases of map APIs. Lastly, chapter 4.6 defines the map API candidates and features for the empirical phase.

4.1 Basics of map Service

Most map APIs aim to provide the interface for developers to access their map services. That is, the map API providers tend to have their own map services to identify location, recommend routes, search location information and provide many other features. Map APIs generally abstract the complexity by providing class named maps so that developers can easily handle data from API provider's own map services with map class. This chapter first illustrates the basics of map services which are the data source ofmMap APIs.

4.1.1 Location Identification

The objective of map service is to show the location and this works by combining multiple image files to fit the scale and show them to the users. The basic component of Map service is one image file in vector data so that users can freely change the scale.

In this regard, vector data consist of point, line string and polygon. Each of these components is used to render map. For example, specific location such as school or company which has its own coordinates are using point for the map service. Secondly, line string of the vector data is used to express road, border or contour on the map and they connect the points as figure 10 describes. Lastly, polygon of the vector data is used to show building, park, forest, etc. and they make polygon as can be seen from figure 10. Map services use numerous collections of vector data and these collections are overlapped in many layers to show the map image to the users (Shin, 2011).

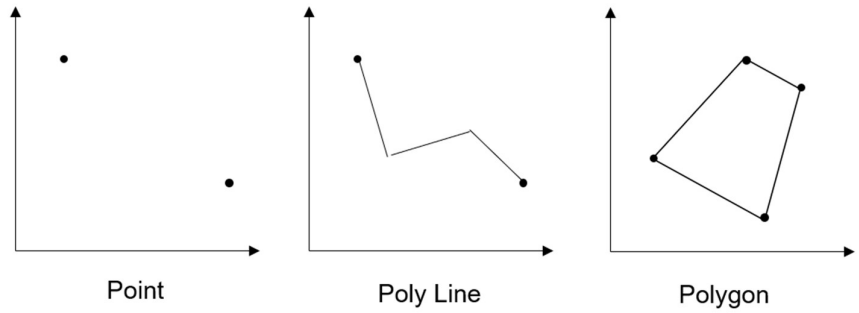
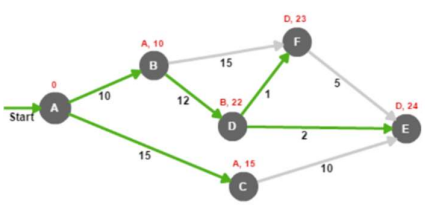


Figure 10. Components of Vector Data (adapted from Shin)

4.1.2 Route Recommendation

One of the popular features of map services is route recommendation. This research is especially interested in how map services recommend the fastest or best route to find answers to the research questions. The most common path-search algorithm applied to the route recommendation feature of map services is Dijkstra algorithm which was invented by Edsger W. Dijkstra in 1956 (Noto & Sato 2000).

This algorithm first calculates time consumed from the starting point to the all connected spots such as building or intersaction. Then it selects the path to the connected spot which has the shortest time and iterates the previous process until they reaches to the destination.



Iteration	Unsettled	Settled	EvaluationNode	A	B	C	D	E	F
1	A	-	A	0	A-10	A-15	X-∞	X-∞	X-∞
2	B, C	A	B	0	A-10	A-15	B-22	X-∞	B-25
3	C, F, D	A, B	C	0	A-10	A-15	B-22	C-25	B-25
4	D, E, F	A, B, C	D	0	A-10	A-15	B-22	D-24	D-23
5	E, F	A, B, C, D	F	0	A-10	A-15	B-22	D-24	D-23
6	E	A, B, C, D, F	E	0	A-10	A-15	B-22	D-24	D-23
Final	**-**	**ALL**	**NONE**	**0**	**A-10**	**A-15**	**B-22**	**D-24**	**D-23**

Figure 11. Dijkstra Algorithm (Baeldung 2021)

For example, figure 11 describes how Dijkstra algorithm finds the shortest route from A by iterating process for 6 times. First, A evaluates time to B and C. Then, nodes B and C are evaluated to find the shorter paths to the next node and the algorithm iterates same process. Finally, algorithm finds that the shortest route from A to each node.

Consequently, the search region of Dijkstra algorithm expands concentrically becase it starts to find the fastest path out of all paths from the departure point in order (Noto & Sato 2000).

Meanwhile, route recommendation feature of map services is not soley created from Dijkstra algorithm. It is known that there are many other algorithms applied to reflect other data such as real time traffic information or cost. As an external developer, it is difficult to know how each map service provider applied algorithms to their route recommendation feature. However, it is believed that algorithms are leveraged in different ways depeniding on the map service provider's business objectives. As a result, different map services tend to return differeutn route recommendation results even though most map servies are based on the similar algorithm advaced from Dijkstra alogrithm.

4.2 Global service providers

The map service and API started with Google's launching Google Maps in 2005. The original solution was for desktop to help people find the way from one point to the other point. In few months later, Google launched Google Earth services which use 3D views of the planet. Later in 2005, Google also started to provide Maps API which immediately became popular for developers who want to apply Google's Map services and data into their own websites. Google Maps API has been used for over 1,000,000 websites around the globe which supports the fact that Google comes as the first candidate of map API for web developers (Google Maps Platform 2013).

As Google started to implement subscription fee on using map API in 2018, there have been some demands of asking alternatives. Under the circumstances, other map API service providers launched their services for developers to access their map data. According to the Top 10 Mapping & Maps APIs (rapidapi.com 2021), Bing Maps, MapBox, FourSquare and Fencer are introduced as an alternative of Google Maps API.

4.3 Local service providers in Korea

There are many IT businesses in Korea who provide map services including location identification, car navigation, route recommendation and location information search. In this regard, this research excludes service proviers for car navigation according to the resarch scope introudced in the chapter 2.4.

It appeared that 79% of Map service users in Korea are using one of Google and 2 local service providers' services. One of the local service provider is Naver which offers the most popular search engine platform in Korea. The other local service provider is Kakao which provides the main online communication platform in Korea.

Figure 12 describes that Monthly Active Users (MAU) for three map services in 2020 are 11.2M for Naver, 5.5M for Google and 5.3M for Kakao. However, when it comes to the monthly average usage time, Kakao and Naver maps users spend 76 minutes and 55 minutes, respectively while Google maps users spend only 15 minutes per month (Mobile Index 2020; Park 2020).

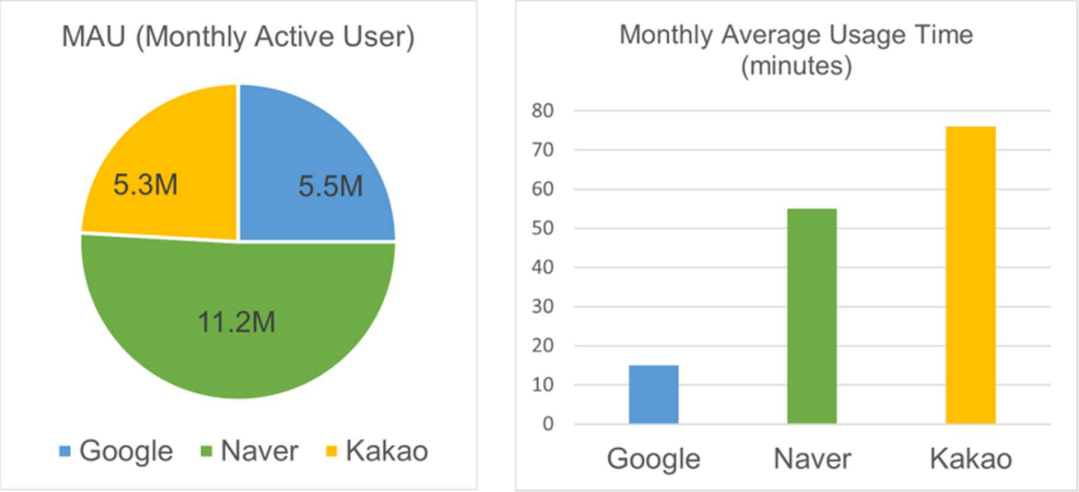


Figure 12. Korean end users behavior in Map services (adapted from Nielsen Korea)

In this reagard, Nielsen Korea (2020) stated that Kakao map takes umbrella branding which separates applications for map, metro, bus and taxi. Subsequently, this strategy caused the low MAU of Kakao Map. This implies that the active users for all maps and route services of Kakao will be more than 5.3M.

4.4 Compliance regarding Map Services in Korea

It is importatn to understand that Korean government has strict restrictions of collecting high quality images for geographic information system or location based service to protect national securities (Kang 2020).

Service providers who use map with 1:5,000 scales should remove certain locations from the map image according to the Korean government's requests. In this regard, local service providers Naver and Kakao get rid of the requested location information from their

map services, however, Google refused to apply this compliance to their map services as it is against to Google's internal policy which is to provide up to data and reliable location data to its users. Also, Korean government does not allow to export map data to abraod without approval of Minister of Land, Infrastructure and Transport (Kang 2020).

Because of thesee regulations, Google does not provide navigation services for drivers nor route recommendation for walking passangers in Korea.

4.5 Use cases

This chapter introduces where map APIs were mostly applied from the use cases in three categories.

Firstly, routing & logicstics industries use map APIs for real time route recommendation. Specifically, businesses take the real-time and precise data about the traffic to optimize their logistics or user experiences. For example, Waymo's automated car uses accurate route data from Google Maps API. Secondly, retail industries use map APIs to provide infomration about store location or recommend direction to the store. Lastly, other industries such as real estate, finance and insurance use map APIs to inform the location of its branch, ATM or property (DitoWeb 2019; Google Cloud Customer 2021).

Some additional features like 3D view or satellite view of Map APIs are also found from existing websites. However, they are often used for industry specific purposes which made it less frequenlty embedded than features introduced in the above cases.

Overall, map APIs are expected to provide basic map services which is a location identification. On top of that, developers can overlap additional features of map APIs according to the development requirements or business objectives. Figure 13 illustrates the layered features of map APIs which are found from the use cases which most frequently applied map APIs to their websites.

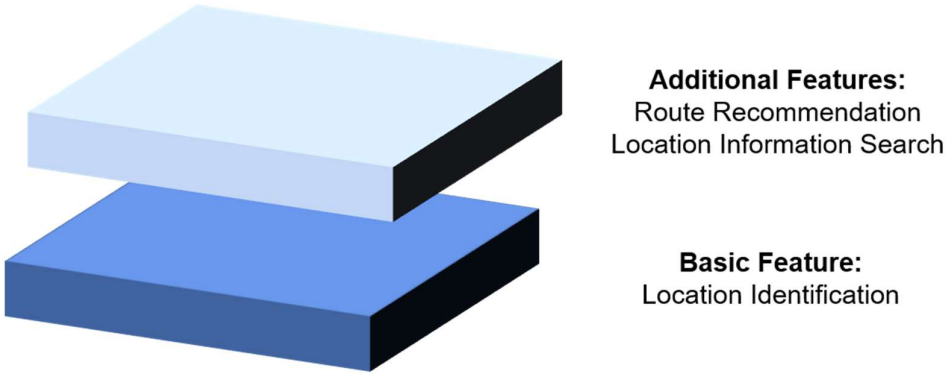


Figure 13. Use cases of Map APIs (created by author)

4.6 Map APIs for the project

4.6.1 Map API candidates

This research explored booth global and Korean service providers to select proper candidate map APIs.

Due to the local compliances, many global service providers except Google are not available in Korean market. Moreover, Google is the category leader and has good awareness by Korean end users. As a result, this research selected Google as the global service provider to be compared in the chapter 5. Empirical section.

Meanwhile, research selected Naver and Kakao as local service providers based on the following reasons. First, they are category leaders in Korean map service market. Second, unlike other local map API providers who are focusing on car navigation, Naver and Kakao have equivalent features to Google such as location identification and route recommendation. Accordingly, research can compare Naver, Kakao and Google Map APIs under the same standard.

4.6.2 Map API features

According to the most popular use cases of map APIs and the layered features created in figure 13, this research will examine basic feature (i.e., location identification) and additional feature (i.e., route recommendation) of map APIs in the emperical phase.

5 Empirical Section

Thesis conducted empirical study based on the selected map API candidates and features. This chapter describes the background, phase and results of empirical research which use three candidate map APIs to answer the research questions.

5.1 Project Background

The project was taken from 25 January 2021 until 22 February 2021. This period incudes project planning, project execution which has 6 phases and project closure (figure 14).

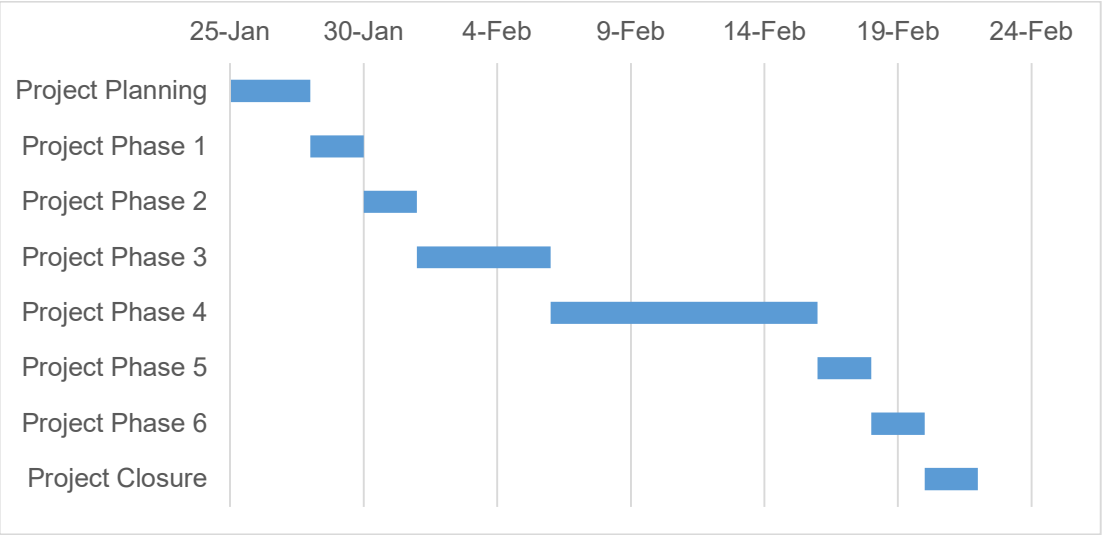


Figure 14. Project Timetable

Project planning includes defining project scope and selecting features based on the background study. Project execution includes product selection, data collection and data analysis. Data were collected from three different sources which are API documentation and code, route recommendation results and user reviews. Lastly, project closure covers creating project conclusion.

5.1.1 Objective

The purpose of this project is to determine the most suitable map API out of three candidates under the given development requirements which reflect the most frequent use cases of Map APIs.

To achieve the project objective, project should compare the candidate map APIs in certain standard. Accordingly, project created a map API evaluation standard as can be seen in the table 3 below.

Table 3. Map API Evaluation Standard

Phase	Evaluation Category	Evaluation Details
Phase 2	Developer resources	Documentation – Quick Start guide
		Documentation – Authentication guide
		Documentation – Tutorials
		Documentation – Code snippets
		Debugging and Troubleshooting
		Developer community
Phase 3	Code	Simplicity
		Consistent and Symmetrical code
		Useful Abstractions
Phase 3 & 4	Data reliability	Location identification by coordinates
		Route recommendation result
Phase 5	End user	User reviews on the map services

Each standard in the above table is based on the background studies about APIs. It means that the evaluation standard is the combination of conditions for good web APIs and they include 4 categories: developer resources, code, data reliability and end users' perspectives.

Both developer resources and code were compared in qualitative approach according to the evaluation details. Data reliability means the ability to return accurate data. For data reliability, this project first evaluated ability to identify the correct location from the given coordinates. Then, project collected data about route recommendation results and analysed them in quantitative approach. Lastly, this project collected and studied the end users review on the map services of each candidate. Project collected data from phase 2 to 5. In phase 6, collected data were analysed according to the evaluation standard.

5.1.2 Map API Comparison Matrix

The comparison matrix is created (table 4) and the project result will introduce the completed matrix.

Table 4. Map API Comparison Matrix

Details	Google	Naver	Kakao
Documentation: Quick Start & Authentication			
Documentation: Tutorials & Code snippets			
Debugging and Troubleshooting & Developer community			
Code: Simplicity, abstractions, consistency and symmetry			
Data Reliability: Location identification			
Data Reliability: Route recommendation			
End user preference			

5.1.3 Assumption

Project set the assumptions which were used as the development requirements for this project. Assumptions were built based on the background studies in chapter 3 and 4 to examine the most frequently applied features and use cases of map APIs. That is, this project includes basic feature (i.e., location identification) and additional feature (i.e., route recommendation) of map APIs.

Specifically, this project created a demo web page which requires data from candidate map APIs. The demo web page is a Location page for one imaginary company located in Lotte World Tower, Seoul, Korea.

The detailed development requirements for this location page are as below:
- Company location is provided as coordinates (Latitude: 37.5125, Longitude: 127.102555) to developer
- Location page should mark company's location from given coordinates
- Location page should allow end users to explore map dynamically
- Location page should provide the best route to the company for the end users
- Route recommendation will allow end users to input their own departure

- Route recommendation will provide the best route to the company by public transportation from the given departure in real time basis

5.2 Project Phase

The project was executed in 6 different phases which to be introduced in this chapter.

5.2.1 Phase 1: Select products

All three candidate map API providers have several products in the map API services. Phase 1 selected the proper products out of them to satisfy development requirements and apply company location identification and route recommendation by public transtportation to the demo websites (table 5).

Table 5. List of products used for the project

Category	Google	Naver	Kakao
Additional Feature	Maps URL	URL Scheme	URL Scheme
Basic Feature	Maps JavaScript API	Web Dynamic Map API	Web Map API

Google provides three products which are maps, routes and places. Maps provide maps in various formats, routes provide the best route and places provide information about the location. Project selected maps product to apply the basic feature which is identifying compny location on the map. Maps product has JavaScript API for the web development process and it allows developers to display maps on the websites in a various ways such as a static image, dynamic maps and street view. This project embedded maps on the demo web page in dynamic maps according to the development requirements. Then, project also used the maps URL product to show the route recommendation by public transportation (Google Maps Platform).

Naver also provides three products named maps, directions and places. Maps provide map services in dynamic, mobile or static ways, directions provide navigation services for drivers and places provide search information about the location. Project selected the web dynamic map from maps product to identify company location. In order to connect our website to the route recommendation services provided by Naver, project used URL schemes for map API (Naver Cloud Platform).

Kakao divides map APIs products into android, web and iOS and this project selected web API services to identify company location on the demo web page. Kako also provides URL

schemes to easily connect web pages using Kakao map API to other Kakao servies such as route, road view and search results. This project selected URL scheme for route to provide real time route recommendation by public transportation (Kakao Map API).

5.2.2 Phase 2: Data collection from developer resources

Before applying APIs to the JavaScript code for the demo web page, project explored the developer resources according to the map API comparison matrix.

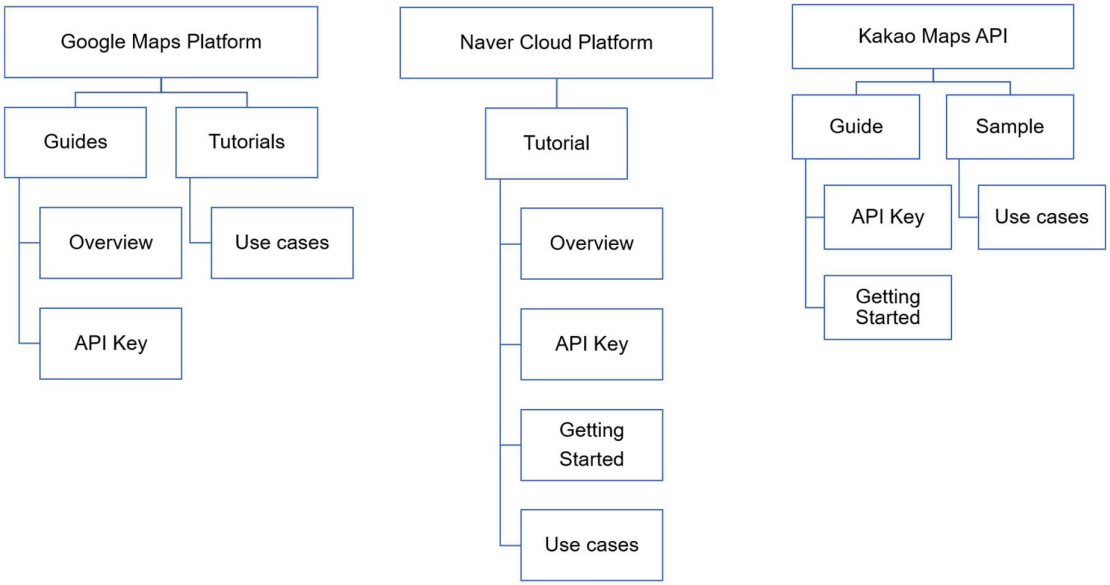


Figure 15. Documentation tree of Map API service provider (created by author)

As figure 15 describes, all three map API providers offer quick start guide, API key and tutorials in the different names and structures. Google has guide as the first category of the documentation which starts with the overview page to deliver the API objective and guide to handle API key and billing errors which developers can face at the starting point of applying map API. Naver, on the other hand, does not have separte quick start category. Instead, it includes API overview, instructions about API key and getting started pages in the tutorial category. Lastly, Kakao provides guide cateogry to explain the information about API key and the Getting Started page.

Meanwhile, Google explictily provides the tutorial category which explains how to use map APIs with detailed instructions and code snippets. Naver's tutorial category continues after the the getting started page and the tutorial includes the detailed explanations of using map API for various purposes together with code snippets. Lastly, kakao does not have

category named tutorial, however, detailed instructions and code snippets can be found in the sample category.

Lastly, project also collected data about troubleshooting and developer's community of three candidate map APIs. Most page of Google documentation is lnked to the relevant troubleshooting. Also, its support tab is very well visible on the top navigation bar of documentation page which provides the links to the developer community and support teams at Google. Meanwhile, Naver and Kakao do not have links to the relevant troubleshooting in each use case of the documentation. Also, they do not have specific developer community for map servicees but integrated Q&A board for all external developers.

5.2.3 Phase 3: Data collection from API code

In phase 3, project applied each of candidate map APIs to create the demo page. Demo web page used JavaScript for client side and Java and Spring Boot for server side. Demo website consists of 4 pages including main, location by Google, location by Naver and location by Kakao (figure 16).

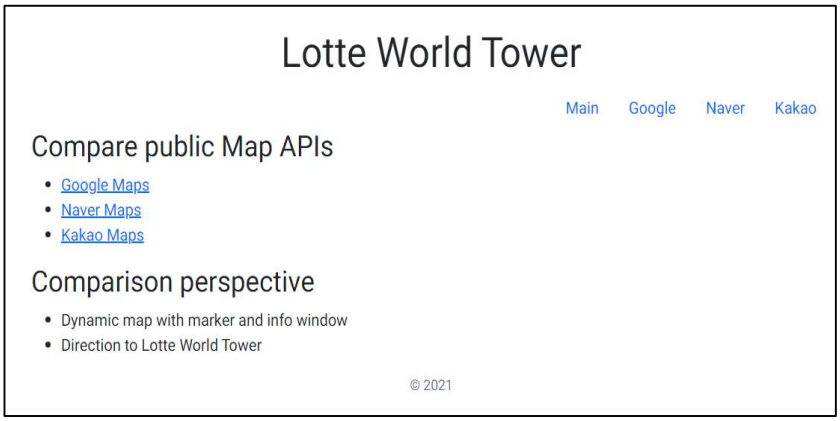


Figure 16. A screenshot of the main page of demo project

Then the demo pages were created by using candidate map APIs. All selected map APIs (i.e., Google Map JavaScript API, Naver Web Dynamic Map API and Kakao Web Map API) were used to show dynamic map on the webpages. In addition, the project applied marker and information windows for all three pages.

Figure 17, 18 and 19 show that how dynamic maps made from each candidate map APIs look like on the browser. In this regard, project used identical code for each page except

the JavaScript code which invokes data from different map API service providers. As can be seen from the figure 17 - 19, embedded maps on the browser show the similar results besides graphical components.

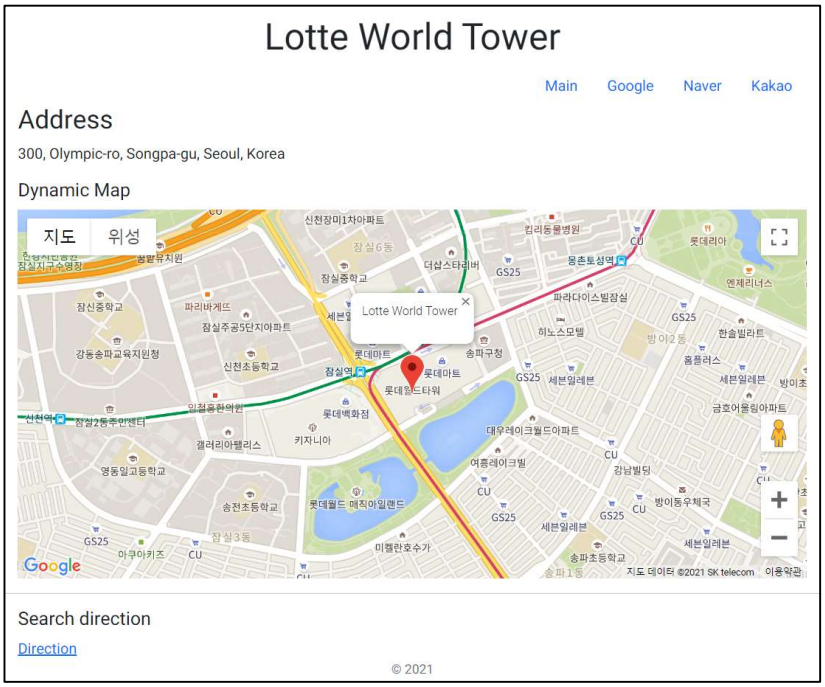


Figure 17. Dynamic Map - Google API

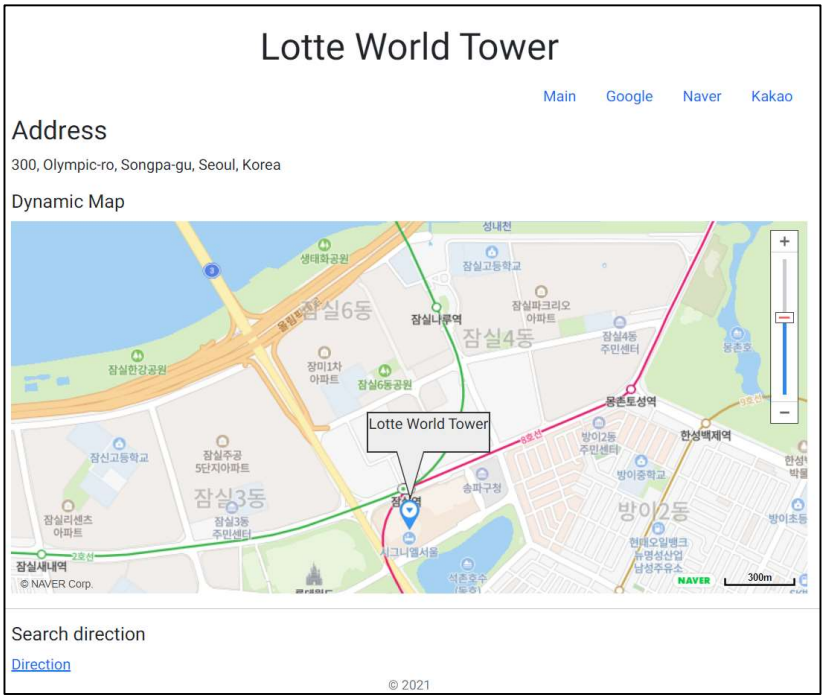


Figure 18. Dynamic Map - Naver API

Figure 19. Dynamic Map - Kakao API

5.2.4 Phase 4: Data collection from route recommendation results

Data resources

According to the development requirments, project also applied the link to route recommendation services from each candidate provider. All three map API documentation introduce how to use their route recommendation services simply adjusting URL. In general, all providers allow develoeprs to implement route services by simply adding URL prameters such as departure and/or destination of the route. Accordingly, each demo page included link to the route recommendation services of each service provider in the webpage. Line 21 in figure 20 shows that developers can easily include a hyperlink to the Kakao route recomendation services by adjusting the destination name (i.e., Lotte World Tower) and coordinates (i.e., 37.5125, 127.102555) of the url scheme.

```
19        <div class="col-8">
20            <h5>Search direction</h5>
21            <a href="https://map.kakao.com/link/to/Lotte World Tower,37.5125, 127.102555" target="_blank"> Direction </a>
22        </div>
```

Figure 20. URL Scheme to link to the route recommendation service - Kakao

Based on the background studies, project found that route recommendation results from different map services can be varied from each other. Project also studied that it is difficult to find that what is the algorithm behind the route recommendation of each map API as an external developer. Accordingly, the project manually collected route recommendation

results from three candidates to analyze and find which map API returns the most reliable route recommendation results.

Data collection

The purpose of phase 4 was to collect data to find the reliable route recommendation service provider. This project selected 20 different departure points to check the route recommendation data.

10 of the deparatures are from Seoul, Korea which is same with destination city. The other 10 departures are from other cities in capital area which have public tansportation connection to Seoul, Korea (table 6).

Table 6. The list of departure

Departures in Seoul	Departures in Captial Area
Seoul Airport	Kintex
Express Bus Terminal	Pankyo Hundai Department Store
Seoul City Hall	Hawesong Fortress
Yeouido Park	Everland
Seoul Worldcup Stadium	Hanam Starfield
Myeong-dong Cathedral	Songdo Central Park
Hanok Village	Suwon Worldcup Stadium
Namdaemum Market	Paju premium outlet
Itaewon	Namyangju
The war memorial of Korea	Naver Headquarter

Each city has its regional public transfortation system so the real time traffic information should be received from the several different databases when route covers more than one cities. As a result, departures from various cities allow project to check if candidates can receive accurate traffic data from various sources in Korea.

Then, the project collected data on five different days including three weekdays and two days on weekend. The population of Korean capital area is 25,900,000 which is more than 50% of population of Korea. Hence, the real time traffic situation is highly influenced by time and day due to the commuters. By collecting route recommendation data from different time and days, project can examine the candidates' ability to get the real time traffic information.

The specific dates and time when data were collecte are listed below:

- 6 February 2021. Saturday. 7:00 a.m. (GMT+9)
- 9 February 2021. Tuesday. 7:00 a.m. (GMT+9)
- 12 February 2021. Friday. 6:00 p.m. (GMT+9)
- 13 February 2021. Saturday. 6:00 p.m. (GMT+9)
- 15 February 2021. Monday. 2:00 p.m. (GMT+9)

5.2.5 Phase 5: Data collection from user reviews

Based on the background studies, research found that the end users are the final entity who contribute the success of web services. Hence, this project lastly collected the data to understand end users' perspectives on three candidates.

As map APIs are directly connected to the API provider's own map services, end users opinions on map services are one good way to forecast end user satisfaction level. The purpose of this phase was to provide developer with better understandigns on the end users.

Data resources

Project retrieved user reviews to understand end users' opinions about three candidates. User reviews are important source of marketing and digital business because they reveal the impressions of customer experiences and influence other customers' decision making process. User reviews also influence production, sales and marketing teams which increase the importance of analyzing use reviews for business (Wu 2017).

As a result, project planned to collect 1,000 user reviews for each map service. To be specific, project planned to collect 500 reviews from Google play store and 500 from Apple app store which were recorded in last 6 months (i.e., September 2020 ~ February 2021).

Data collection

There are two popular ways to scrap large volume of data from web browser which are beautifulsoup library and selenium framewok.

BeautifulSoup is an open-source Python library which is used to extract data from HTML or XML document for web scrapping. Selenium, on the other hand, was originally developed for the automated web application testing. Selenium automates web browsers and developers can let selenium do actions on the browser on behalf of themselves.

This project selected selenium over beautifulsoup because it requires actions on the browser. This project used selenium to open chrome drive, scroll down, click more reviews, collect reviews and repeat the same process until 500 reviews are collected (figure 21).

```python
# Connect to KakaoMaps - Google play store
from selenium import webdriver
from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt
import time


url = "https://play.google.com/store/apps/details?id=net.daum.android.map&hl=en&showAllReviews=true"

driverPath = "./mapApp_reviews/chromedriver.exe" # Chrome Driver path
driver = webdriver.Chrome(driverPath) # Open Chrome
driver.get(url) # Enter the url

# Loop screen scroll to retrieve 500 reviews
SCROLL_PAUSE_TIME = 1.5

for i in range(3):
    # (1) Scroll down 5 times
    for j in range(5):
        driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
        time.sleep(SCROLL_PAUSE_TIME)

    # (2) Click 'more review'
    driver.find_element_by_xpath("//span[@class='RveJvd snByac']").click()
```

Figure 21. Python code to use Selenium framework

5.2.6 Phase 6: Data analysis

Phase 6 analyzed the collected data from this project. Analysis phase includes four parts to evaluate data collected from phase 2 to phase 5.

Developer resources and API code

Phase 2 and 3 collected qualitative data about developer reousrces and map API codes. These data were evaluated based on the map API comparison matrix and the result can be seen in the table 7 below.

Project used 3 points scale rating to evaluate the developer resources and codes of candidate map APIs. Also, project evaluated map APIs in the relative measure. That is, Grade 3 was the highest score and given to the best service provider out of three candidates. On the other hand, grade 1 was the lowest score and given to the provider who offers that feature in the less effective way than other candidates.

Table 7. Evaluation results for developer resources and code

Details	Google	Naver	Kakao
Documentation: Quick Start & Authentication	3	1	2
Documentation: Tutorials & Code snippets	1	2	3
Debugging and Troubleshooting & Developer community	3	2	1
Code: Simplicity, abstractions, consistency and symmetry	3	3	3

Project found that the contents of quick start and authentication guide are very similar to each other. Also, all three candidates provide clear instructions so that developers can easily start using map APIs. In this regard, however, project gave the highest score to Google as it is the only provider who offers troubleshooting for API key and billing in the quick start guide. The reason for grading 1 to Naver is because it is difficult to find the start guide as it is included as a subchapter in the tutorial category.

Project analyzed that all three candidates provide tutorials together with code snippets. In this regard, project gave the highest score to Kakao because it offers tutorials for the most various scenarios. Also, Kakao allows developers to write code on the browser which makes Kakao as the only interactive tutorials out of candidate APIs. Meanwhile, the reason for grading the lowest score to Google is because it only has 5 scenarios under the tutorials.

As for the other developer resources such as troubleshooting and developer community, project graded the highest score to Google. The most examples of Google's documentaion include link to the relevant troubleshooting information. By doing so, developers who use Google Map API can reduce time for troubleshooting while applying Google Map API to their websites. Also, Google has the most active and largest developer community which is another important advantages for developers. Naver does not provide link to the relevant troubleshooting on example or tutoril pages. Instead, Naver has integrated support platform which covers Q&A and FAQ for developers use all Naver services as well as Map API. Naver also has various supporting programs including training. Meanwhile, Kakao has only one Q&A board where develoeprs can communicate to each other but they do not provide any guidances on how to contact Kakao support team.

Meanwhile, project examined the simplicity and abstraction level of code. Project found that codes for three candidates are very similar to each other besides the name of object or methods. Also, codes for three APIs were constructed in the similar ways and they all support great level of simplicity. For example, all candidates use map class to abstract complicated business logic and low level code from the developers who use map APIs as can be seen in the line 16 of figure 22.

```
10  <script>
11      // Initialize and add the map
12      function initMap() {
13        // The location of Uluru
14        const lotteWorldTower = { lat: 37.5125, lng: 127.102555 };
15        // The map, centered at Uluru
16        const map = new google.maps.Map(document.getElementById("map"), {
17          zoom: 15,
18          center: lotteWorldTower,
19        });
20        // Info Window
21        const contentString =
22            '<p>Lotte World Tower  </p>';
23          const infowindow = new google.maps.InfoWindow({
24            content: contentString,
25          });
26          const marker = new google.maps.Marker({
27            position: lotteWorldTower,
28            map,
29            title: "Lotte World Tower",
30          });
31          marker.addListener("click", () => {
32            infowindow.open(map, marker);
33          });
34        }
35    </script>
36
```

Figure 22. JavaScript Code – Google API

Also, all three providers' codes are consistent and symmetric which prevent confusions by developers. They use predefined term for the object and these information can be found from documentation. For example, figure 23 describes that map APIs support symmetrical code such as .close( ) in line 67 and .open( ) in line 69.

```
        // Onclick will make infowindow disapper
        naver.maps.Event.addListener(marker, "click", function(e) {
            if (infowindow.getMap()) {
                infowindow.close();
            } else {
                infowindow.open(map, marker);
            }
        });

        infowindow.open(map, marker);
```

Figure 23. Consistent and Symmetrical code - Naver API

In conclusion, JavaScript code provided by all three candidates are in similar structure with great simplicity, abstraction, consistency and symmetric. Hence, the project gave same score to all candidates.

Location identification results

The demo web pages of this project well identified company location on the map based on the coordinates and all three candidates returned same results. This means that the performane of location identification for three candidates are equally reliable and accurate.

Route recommendation results

The route recommendation results were manually collected from phase 4. Specifically, project collected 100 route recommendation results for each candidate. These 100 results can be interpreted in two different ways. First, 50 route recommendation results were from Seoul to Seoul and the other 50 results were from other capital cities to Seoul. Second, 60 route recommendation data were collected on the weekday and 40 were collected on the weekend.

When collecting route recommendation results, project gave number 1 if the result is different from other candidates and number 0 when the same results are found from two or more candidates. For example, if candidate A's result is different from B and C, the record would be: A-1, B-0 and C-0. As there are only three candidates, each of 100 records will be resulted in one of three possibilities:

- All return different results
- All return the same reuslts
- Only one returns different result from two others

After recording either number 1 or 0 for 100 times for each candidate, project calculated the sum of each candidate. If the sum is higher than others, it means that the provider

returned different results more often than other candidates. The number of showing different results out of 100 results are recorded in the table 8.

Table 8. The number of showing different results out of 100 results (scale: times)

	Google	Naver	Kakao
Different results	56	16	17

In this regard, project found that 14 records from the above table were recorded when all three candidates returned different results. The departures of these 14 records were Seoul Airport, Express Bus Terminal, Yeouido Park and Hawesong Fortress (table 9).

Table 9. Departure information when all candidates returned different results

Departure	Departure city	The number of appearances
Seoul Airport	Seoul	5
Express Bus Terminal	Seoul	5
Yeouido Park	Seoul	3
Hawesong Fortress	Capital Area	1

Project concluded that the reason of repeatedly showing different results on the above routes are not caused by the different abilities of candidates to access the local traffic information. These were repeatedly recorded due to the different ways of leveraging algorithms per candidate providers which cannot be analysed as an external developer. As a result, project excluded these 14 records from the final analysis. That means, 86 records were used for the final analysis.

Project finally calculated the possibility to show different route recommendation results from other candidates in the Table 10. It calcuated the percentage of the possibility out of 86 records. High number in the table means that this candidate tends to return different results from other two candidates more often.

Table 10. Possibility to show different route recommendation results from others (scale: %)

	Google	Naver	Kakao
Average	48.8	2.3	3.5
On weekday	51.9	1.7	1.9
On weekend	44.1	2.9	5.9
Depart from the same city	27.9	0.0	7.0

Depart from the different cities	69.8	4.7	0.0

On average, the possibility to solely return different route recommendation results from others for Google is 48.8% which is significantly higher than Naver (i.e., 2.3%) and Kakao (i.e., 3.5%). Besides the average, Google has the highest possiblity of showing different results in all scenarios while Naver and Kakao's possibilities are lower than 10% in all scenarios.

The possibilites on weekday and weekend are different from the average, however, the trend remains same. That is, Naver shows the lowest possibility and Kakao shows relatively higher possibility than Naver while Google shows extremly higher possibility than other two candidates.

Meanwhile, it is notable that the possibilites of showing different route recommendation results per departure cities are more dynamic. When departure cities are located in the same city with the destination city, Google (i.e., 27.9%) and Naver (i.e., 0.0%) returned better performances than their average possibilites. However, Kakao returned higher (i.e., 7.0%) possibility of returning different results than its average possibility. When departures are located in various cities in the captial area, Google returned much higher possibility (i.e., 69.8%) than its average possibility. Under the circumstances, Naver's result was higher (i.e., 4.7%) than its average and Kakao did not soley return different results.

This phase evaluated the data reliability of route recommendation results as an external developer. Route recommendation results can be influeced by traffic data retrieved from various sources. Traffic data from various data sources include real time traffic information in several sources. Hence, this results can tell the candidate's ability to access and analyze the accurate traffic data from Korea.

According to the route recommendation results studied above, project concluded that Google is applying significanlty different methodologies than Naver and Kako on their route recommendation services in Korea. When both deparutre and destination are in the same city, Seoul, all three candidates returned better results than when departure and destination cities are different from each other. This means that all candidates can return more accurate traffic information when data are only required from Seoul. In this regard, Google's data when route covers several cities are not reliable which means that Google's route recommendation cannot successfully reflect the real time traffic data from several data sources in Korea.

Meanwhile, Naver returned more reliable results than Kakao when the routes are within Seoul and Kakao returned more reliable results than Naver when the routes cover several cities.

End users' reviews

The user reviews are qualitative data and this project analysed them with data visualization. For data visualization, project selected the word cloud and matplotlib libraries.

Word cloud is one popular way of data visualization which delivers a valuable information about a large volume of text data. Word cloud is a cluster of words from the text data source and each of the word is in different sizes. If some word appears more often than others in the text data source, word cloud makes that word looks bigger and bolder.

Matplotlib is a python library used for visualization. It enables easier way to create static, animated or interactive graphs or visualization.

```
41  # Create wordcloud
42  stopwords = set(STOPWORDS)
43  stops = ['I','am', 'is']
44  for i in range (len(stops)):
45      stopwords.add(stops[i])
46
47  wordcloud = WordCloud(max_words=80, stopwords=stopwords,background_color='white').generate(text)
48
49  # Set wordcloud image
50  plt.figure(figsize=(10,10)) #fix image size
51  plt.imshow(wordcloud, interpolation='bilinear')
52  plt.axis('off')
53  plt.show()
54  plt.savefig('./word_cloud')
```

Figure 24. Python code for creating word cloud and visualization

This project announced three stopwords (line 43 of figure 24) and excluded them when creating word cloud. Also, it included only top 80 words from the text data resources when creating word cloud (line 47 of figure 24).

Consequently, project created three word clouds for candidates and they can be seen from figure 25, 26 and 27.

Figure 25. Word Cloud - Google

Figure 26. Word Cloud – Naver

Figure 27. Word Cloud - Kakao

In this reagard, project found that end users commonly comment a lot about route, directions and navigation on all three service provider's reviews. Also, end users mention other service provider's name in the comment whcih explains that end users can easily access, compare and switch from one map service to another map service. Lastly, one most important insights from this analysis is that only Google has the negative words such as issue, problem and uninstall in the most frequently used top 80 words in the user reviews.

5.3 Project Result

Based on the data analysis in the empirical phase, the research questions can be answered as below.

When developing website which requires map API for Korean market:
- How to select candidate map APIs?
Before selecting candidate map APIs, developers should clarify the development requirements. If requirements are about location identification and route recommendation features by map APIs, developers should consider the available global and local API providers. When narrowing down candidates, it is important for developers to be aware of local compliance regarding map services. After narrowing down to the several API candidates which can satisfy development requirements, developers can refer the end user's awareness about candidate's map services as it can influence the end user's satisfaction level about embedded maps in the websites.

- How to compare candidate map APIs?

Even though there are no standard to compare web APIs, developers can evaluate candidate map APIs in a certain way. To be specific, developers should evaluate candidate map APIs in both developer's and end users' perspectives to achieve development objectives. This research has created the map API comparison standard which can be used by developers when comparing candidate map APIs (table 11).

Table 11. Map API comparison standard

Comparison standard
Documentation: Quick Start & Authentication
Documentation: Tutorials & Code snippets
Debugging and Troubleshooting & Developer community
Code: Simplicity, abstractions, consistency and symmetry
Data Reliability: Location identification
Data Reliability: Route recommendation
End user preference

- Should developers use different map API for different development requirements?

The best map API can be different depending on the leverage within the comparison standard. Accordingly, it is important for developers to define development requirements and prioritize requirements so that they can select map API which most efficiently achieve development requirements.

- Which map API is the most suitable for websites targeted for Korean market?

As stated in the above answer, the most suitable map API for websites targeted for Korean market can be varied according to the leverage within the comparison standard. In the empirical phase, research completed the map API comparison matrix. Research applied 3 points scale rating to complete the map API comparison matrix. That is, grade 3 was the highest score and given to the best service provider out of three candidates. On the other hand, grade 1 was the lowest score and given to the provider who offers that feature in the less effective way than other candidates (table 12).

Table 12. Completed Map API Comparison Matrix

Details	Google	Naver	Kakao
Documentation: Quick Start & Authentication	3	1	2
Documentation: Tutorials & Code snippets	1	2	3
Debugging and Troubleshooting & Developer community	3	2	1
Code: Simplicity, abstractions, consistency and symmetry	3	3	3
Data Reliability: Location identification	3	3	3
Data Reliability: Route recommendation (in one city)	1	3	2
Data Reliability: Route recommendation (intercity)	1	2	3
End user preference	1	3	2

Based on the table 12, research can recommend the most suitable map API for the several development scenarios (figure 28).

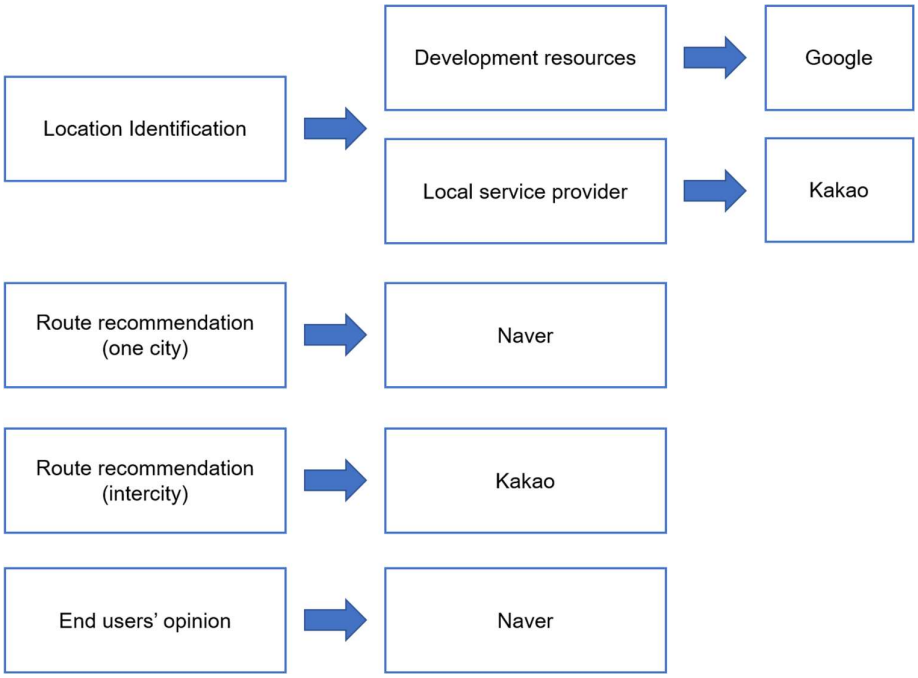


Figure 28. Map API recommendation (created by author)

When map API is only used for location identification, all candidates perform similarly. Hence, another development requirements should also be considered. If the rich resources are important during the development, Google would be the most suitable map API. If local service provider is preferred, Kakao will be the most suiutable map API.

When route recommendation is expected from map API, developers should abvoid Google as its route recommendation data is not reliable in Korea. If route recommendation covers only one city, Naver is the most suitable map API. When it comes to the intercity route recomendation, Kakao is the most sutiable one.

Lastly, if end users' experiences with embedded map on the websites are critical, Naver is the most suitable Map API.

6 Conclusion

6.1 Summary

Applying third party web API has become an ordinary way of web development process thanks to the evolution of web APIs. Developers can easily select and apply web APIs which can simplify their code and shorten the development time. However, it is difficult for developers to select the most suitable web APIs to satisfy development requirements in the most effective way. When developing websites requiring map data about Korea, developers face difficulties due to the lack of information about local map API service providers. On top of that, Korean government has strict regulations regarding map services and Korean end users have specific preferences with map services. These facts made it very difficult for developers who should select map API for websites targeted Korean market.

To achieve development objectives, developers should define and prioritize development requirement before selecting candidate map APIs. This research created and applied the development requirements based on the most common use cases of applying map APIs and they were location identification and route recommendation. In addition, this research also considered the compliances and end users' awareness to select three map API candidates which are Google, Naver and Kakao.

During the empirical phase, this research collected and analysed data to evaluate candidate map APIs in both developer and end users' perspectives. Consequently, research created map API comparison matrix and recommended the most suitable map API according to the development scenarios.

To summarize, candidate map APIs have similar ability at identifying location on the map. Hence, developers should consider other requirements based on the prioritized development requirements to determine the most suitable map APIs. When it comes to route recommendation data, candidate map APIs show different level of data reliability. Naver and Kakao should be considered depending on the departure and destination cities. Lastly, if developers highly value end users' experiences, Naver should be applied to their websites.

6.2 Discussion

It is important to remind that the most suitable map API is varied depending on the development requirements. This means that there are lots of different combination of requirements to be examined to recommend the most suitable map API.

As for the route recommendation, research assumes that Google's not reliable results from route recommendation services may be caused by the ability to access geographic information about Korea as Google is the only foreign company out of candidates. In this regard, Google has opend its data center in Seoul in January 2020. It means that Google may improve the way to return route recommendation based on more accurate geographic information and traffic data as they no longer need to export data to abroad.

References

Baeldung. 2021. Dijkstra Shortest Path Algorithm in Java. URL: https://www.baeldung.com/java-dijkstra. Accessed: 18 February 2021.

Bhuiyan, T., Begum, A., Rahman, S. & Hadid, I. 2018. API vulnerabilities: current status and dependencies. International Journal of Engineering & Technology, 7(2.3), 9-13.

Bush. 2019. 5 Examples of Excellent API Documentation (and Why We Think So). Nordic APIs. URL: https://nordicapis.com/5-examples-of-excellent-api-documentation/. Accessed: 08 February. 2021

Doerrfeld. API Discovery: 15 Ways to Find APIs. 2015. URL: https://nordicapis.com/api-discovery-15-ways-to-find-apis/. Accessed: 10 January 2021

Dito Web. 2019. Top 5 Areas and Use Cases Where Google Maps Excels. URL: https://www.ditoweb.com/2019/09/top-areas-google-maps-excels/. Accessed: 12 February 2021.

DuVander, A. 2013. 9,000 APIs: Mobile Gets Serious. Programmable Web. URL: https://www.programmableweb.com/news/9000-apis-mobile-gets-serious/2013/04/30. Accessed: 20 February 2021.

Espinha, T. Zaidman, A. & Gross, H. G. 2015. Web API growing pains: Loosely coupled yet strongly tied. Journal of Systems and Software, 100, 27-43.

Google Cloud Customer. 2021. URL: https://cloud.google.com/customers/. Accessed: 14 February 2021.

Google Maps Platform. URL: https://cloud.google.com/maps-platform. Accessed: 28 January 2021.

Google Maps Platform. 2013. A fresh new look for the Maps API, for all one million sites. URL: https://mapsplatform.googleblog.com/2013/05/a-fresh-new-look-for-maps-api-for-all.html. Accessed: 11 February 2021

Jacobson, D., Brail, G., & Woods, D. (2011). APIs: A strategy guide. " O'Reilly Media, Inc.". Chapter 3. Understanding the API Value Chain.

Jin, B., Sahni, S. & Shevat, A. 2018. Designing Web APIs: Building APIs That Developers Love. " O'Reilly Media, Inc.". 4-6

Kakao Map API. URL: https://apis.map.kakao.com/. Accessed: 31 January 2021.

Kang, H. Y. 2020. Google Maps in Korea. IT Donga. URL: https://it.donga.com/31076/. Accessed: 13 February 2021.

Kopecky, J., Fremantle, P. & Boakes, R. 2014. A history and future of Web APIs. Information Technology, 56(3), 90-97.

Lane. 2019. Intro to APIs: History of APIs. Postman Blog. URL: https://blog.postman.com/intro-to-apis-history-of-apis/. Accessed: 28 January 2021.

Lane. 2020. Intro to APIs: What Is an API?. Postman Blog. URL: https://blog.postman.com/intro-to-apis-what-is-an-api/. Accessed: 28 January 2021.

MDN Web Docs. 2021. Introduction to web APIs. https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Client-side_web_APIs/Introduction. Accessed: 20 January 2021.

Mobile Index. 2020. MAU 2020: Naver vs Kakao. URL: https://www.mobileindex.com/report/report_view?s=129. Accessed: 13 February 2021.
Moilanen. 2017. Developers say reliability is API's most important feature. URL: https://medium.com/apinf/developers-say-reliability-is-apis-most-important-feature-5726ca05f0d. Accessed: 07 February 2021.

Monitis. 2019. Characteristics of Good APIs. URL: https://www.monitis.com/blog/characteristics-of-good-apis/. Accessed: 05 February 2021.

Naver Cloud Platform. URL: https://www.ncloud.com/. Accessed: 10 February 2021.

Nielsen Korea. Naver vs Kakao : Comparing Maps Services. 2020. URL: http://www.koreanclick.com/insights/newsletter_view.html?code=topic&id=563&page=1&uut_source=board&utm_medium=board&utm_campaign=topic&utm_content=20200220. Accessed: 25 January 2021.

Noto, M. & Sato, H. 2000. A method for the shortest path search by extended Dijkstra algorithm. In Smc 2000 conference proceedings. 2000 ieee international conference on systems, man and cybernetics.'cybernetics evolving to systems, humans, organizations, and their complex interactions'(cat. no. 0 (Vol. 3, pp. 2316-2320). IEEE.

Park, J. 2020. Monthly Activity User: Naver vs Kakao. URL: http://www.busan.com/view/busan/view.php?code=2020070919081316878. Accessed: 16 February 2021.

PURKAYASTHA. 2020. The Evolution of APIs: Past, Present and the Future. Rakuten Rapid API. URL: https://blog.api.rakuten.net/evolution-of-apis/. Accessed: 26 January 2021.

Rapid API. The Top 10 Mapping & Maps APIs (for Developers in 2018). URL: https://rapidapi.com/blog/top-map-apis/. Accessed: 15 February 2021.

Reid. 2020. A look back at 15 years of mapping the world. Google - The Keyword. URL: https://blog.google/products/maps/look-back-15-years-mapping-world/. Accessed: 13 January 2021.

Shin, H. C. 2011. How to develop map services?. Naver Developer. https://d2.naver.com/helloworld/1174. Accessed: 11 February 2021.

Torres, R. & Tapia, B. 2011. Improving web api discovery by leveraging social information. In 2011 IEEE International Conference on Web Services (pp. 744-745). IEEE.

Qi, L., He, Q., Chen, F., Dou, W., Wan, S., Zhang, X. & Xu, X. 2019. Finding all you need: web APIs recommendation in web of things through keywords search. IEEE Transactions on Computational Social Systems, 6(5), 1063-1072.

Wu, J. 2017. Review popularity and review helpfulness: A model for user review effectiveness. Decision Support Systems, 97, 92-103.

Zibran, M. 2008. What makes APIs difficult to use. International Journal of Computer Science and Network Security (IJCSNS), 8(4), 255-261.