모바일 옥내위치기반서비스를 위한 도면 출력 서비스 구현 현황 소개

임재걸*, 레콩탄*
*동국대학교 컴퓨터공학과
e-mail : {yim, lct}@donggul.ac.kr

Current Status of Development of Rendering Drawings Service for Mobile Indoor Location Based Service

Jaegeol Yim*, Thanh C. Le*
*Dept. of Computer Engineering, Dongguk University

요 약

Rendering maps is an essential feature of the user interface component of a location based service (LBS) system. However, a developer may not too much worry about implementing the rendering maps part of his or her system because there are quite a few publicly available libraries that provide all kinds of functions of manipulating maps. Google Maps, Yahoo Map, Naver Map, Daum Map, and so on are example sites that provide those libraries. Rendering drawings is to indoor LBS as rendering maps is to LBS. However, there is no such thing as Google Maps that provides libraries for rendering drawings. This paper introduces a few web services and a library that is useful in developing user interfaces of indoor LBS systems.

1. Introduction

Google, Yahoo, etc. are very famous in providing map services. Using the APIs they provide, we can easily render interactive maps that users can zoom in, zoom out, tagging and so on in our applications.

However, rendering drawing in indoor location based system (ILBS) is totally different since Google, Yahoo and so on do not support this task yet. They cannot do that because they do not have drawings of individual properties.

In this paper, we present our own rendering drawing services. These services were designed and developed suitable for ILBS. As you know, the client working in ILBS are mobile devices, with limited CPU speed, small memory and so on. Our method is a good technology for such cases. That is RESTFul web services, a popular technology which used by many famous company such as Google, Yahoo etc. In RESTFul, the messages are short, simple, and useful. They are pure HTTP messages. RESTFul is used especially for implementing the services to serve for mobile devices' request or other environments where complex messages like SOAP message are not suitable.

The paper's structure will be represented as following: Section 2 discusses related technologies for implementing these RESTFul web services. Section 3 describes design and implementation of our services for rendering drawing for Mobile ILBS. Section 4 shows some experiments which we have performed. Finally, Section 5 is a summary overall the paper and the future research directions.

2. Related works

Choosing a technology and methods to build a system is one of very important steps. Indoor positioning is no exception. The devices used in this area are usually mobiles such as smart phone like Android, iPhone, Windows Phone and so on with characteristics of limited CPU speed, small memory etc. Therefore, we have to implement these services with consideration of the characteristics.

The Web Services is a good choice for developing service systems which communicate over Internet or Intranet. It is well known that Web Services use SOAP protocol to exchange messages between server and client. A message of a web service is quite big and complex for devices which have a small memory and low speed CPU like mobile devices, as Figure 1 describes a SOAP message. So Web Service is not a good choice in developing mobile applications. We have another technology very suitable for our case, and it is also used in popular by many big companies for web development like Google, eBay, Amazon, Facebook. That is RESTFul Web Services. RESTFul use only pure HTTP protocol to exchange messages between server and client. In each of the RESTful web services, they usually define POST, GET, PUT and DELETE methods, POST to add data to server, GET to get data from server, PUT to update data, and DELETE to delete resource data in server. A RESTFul message is simple as shown in Figure 2. This message is used for

request server add more a product as body part. [1] also gives us a clear comparison between web services and RESFul web services.

```
POST /Service1.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://tempuri.org/AddProduct"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <AddProduct xmlns="http://tempuri.org/">
      <p>
        <ID>6</ID>
        <Name>Samsung SII</Name>
        <Price>1000</Price>
      </p>
    </AddProduct>
  </soap:Body>
</soap:Envelope>
```

Figure 1. SOAP Message to add a product to server (With much cover by SOAP)

Request	POST /ProductService/Product HTTP/1.1
Accept	*/*
Content-Type	application/json; charset=utf-8
Content-Length	36

```
{"Id":6,"Name":"Samsung SII","Price":1000}
```

Figure 2. A Message to add a product to server used in RESTFul (pure HTTP message)

To develop a RESTFul Web Service, we could use many other technologies. In .Net, we have ASP.Net Web API (supported in ASP.Net MVC 4.5), other is WCF, or simply as ASP.Net. In Java, we have also Servelet that is also suitable technology for implement of RESTFul. We chose WCF for some advantages as shown in the following.

WCF, stands for Windows Communication Foundation, is a network programming framework of Microsoft. WCF is meant for designing and deploying distributed applications under Service Oriented Architecture implementation. When it was first presented in .Net Framework 3.0, the goal of WCF was integration of all the technologies for Network Programming such as Web Services, .Net Remoting, Socket, other Network APIs into one model programming. The WCF Architecture is given in Figure 3. Before WCF, network programmers have to study each technology to apply for a proper area which this technology the most suitable in there such as Web Services for communication over Internet, .Net Remoting for Intranet etc. With WCF, programmers study only WCF, work only to Data Contract, Message Contract, Services Contract and Policy and Binding. The rest of work such as response to WCF client, other clients then WCF chooses a suitable formatted message as SOAP for WCF client or simple format XML for another. In .Net Framework 4.0, WCF made more powerful with communication by formatted in JSON, ATOM (a popular RSS standard). It can be said that WCF is the most famous technology in Network Programming.

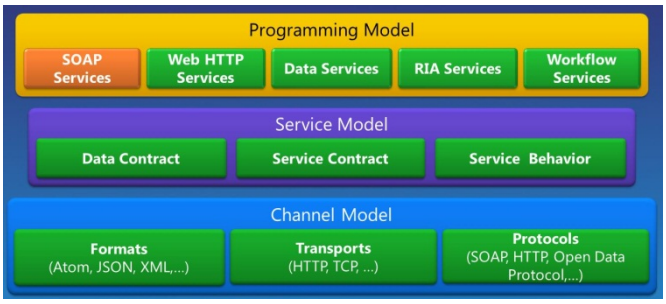

Figure 3. WCF Architecture

The process of RESTFul web service transaction is presented in Figure 4. The exchange messages are pure HTTP messages. (1) The client application creates one or more input parameters, put complex parameters in body of message. Then client makes a HTTP request Message by specific URL and contain body are above complex parameters, simple parameters (string) put in URL, then delivers the message to HTTP Protocol. (2) At this point HTTP protocol channel has a chance to operate on the message before the transport channel uses a message encoder to transmit the final Message as a sequence of bytes to the target service. (3) The raw stream of data is transmitted over the wire. (4) On the service side, the transport channel receives the stream of data and uses a message encoder to interpret the bytes and to produce a WCF Message object that can continue up the channel stack. At this point each protocol channel has a chance to work on the message. (5) The final Message is passed to the Dispatcher. (6) The Dispatcher receives the WCF Message from the underlying channel stack, the target service endpoint using the destination and Action property contained in the Message and configurations in file web.config, deserializes the content of the WCF Message into objects in our case that is Product. (7) When received parameters and specific method which will be invoked, finally work is invoke the target service method. In the similar manner, the response returns to client.

Implementing a WCF service is quite simple under support of Visual Studio. What the programmer has to do is defining DataContract, ServiceContract and configure binding for communication progress between services and client. In this paper, we used .Net Framework 4.5 Developer Preview and Visual Studio 11 Developer Preview to implement these services. Publish WCF, a service endpoint can be part of a continuously available service hosted by IIS, or it can be a service hosted in an application. [2] provides some related documents about this area.

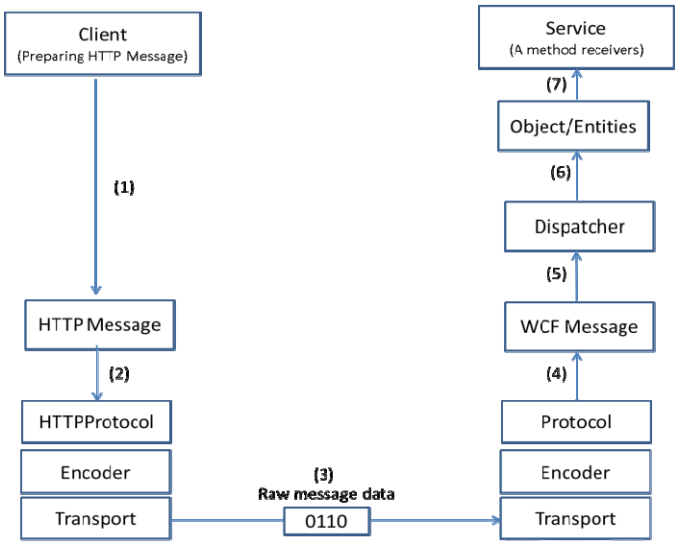

Figure 4. A transaction between client/server by technology RESTFul (Which has the server side implemented by WCF)

3. Implementation of Map Services

In general, map services like Google, Yahoo etc. support for clients download maps, and some functions as choose an area to see, tagging and so on. Our map services support clients to upload a map, list maps which exist in the server, download a map to client, and delete a map. Considering these tasks, we designed our service as it is shown in Figure 5. Where simple parameters will be transfer in URL such as ID, name of the map, and complex parameters like object or content of map will be push in body of HTTP messages. Besides, clients will receive HTTP code in all responses for each request to server such as 200, 201, 204, and 404 are OK, Created, NoContent, NotFound respectively.

Method	URL	HTTP Method
UploadMap	Map?**Name**="name"&**Des**="Des"	POST
ListMaps	Map/List	GET
ListMapsXML	Map/List?Format=XML	GET
DownloadMap	Map/{id}	GET
DeleteMap	Map/{id}	DELETE

Figure 5. A Map Services' design based on RESTFul

As discussed in WCF part, we have to make a Service Contract for each of services. In Visual Studio, we can easily define a Service Contract in WCF services application by adding a new WCF Service. Then, we use declaration as *ServiceContract, OperationContract* for each of methods which this service are supporting, and to define a RESTFul then we need to use more declaration are *WebGet*, *WebInvoke* to set HTTP methods, UriTemplate, Parameters which will invoke an operation and even properties for responses.

```
[ServiceContract]
public interface IMapServices {

  [OperationContract]
  [WebInvoke(Method="POST",
    UriTemplate= "Map?MapName={mapName}&Des={Des}")]
  int UploadMap(string mapName, string description);


  [OperationContract]
  [WebInvoke(Method = "GET",
     UriTemplate = "Map/List",
     ResponseFormat= WebMessageFormat.Json)]
  List<Map> ListMapsJSON();


  [WebInvoke(Method = "GET",
     UriTemplate = "Map/List?Format=XML",
     ResponseFormat = WebMessageFormat.Xml)]
  List<Map> ListMapsXML();

}
```

Figure 6. A part of definition of Map Services

Figure 6 shows our definition of the service with only two methods are UploadMap and ListMaps. And the Figure 7 presents an implementation of the method upload map. In this method, two parameters received from URL are map's name and map's description, and the complex argument is map content which will be received in body of request. In some other cases, these complex objects which are defined with DataContract will automation deserialized by WCF. So we can use these objects as arguments of methods, and put object's serialization to the body of HTTP message.

```
public class MapServices : IMapServices {
  public int UploadMap(string mapName,
                       string description){
      if (mapName == null ||
                mapName.Trim().Equals("")) {
          WebOperationContext.Current.
                OutgoingResponse.
                StatusCode =
                HttpStatusCode.BadRequest;
          return -1;
      }

      FileStream fWrite = new FileStream("c:\maps\"
         + mapName + ".dfx", FileMode.Create);
      Message msRequest =
         OperationContext.Current.RequestContext.
                                  RequestMessage;

      byte[] content = msRequest.GetBody<byte[]>();
      fWrite.Write(content, 0, content.Length);
      fWrite.Close();
       WebOperationContext.Current.OutgoingResponse.
              StatusCode = HttpStatusCode.Created;
      return content.Length;
   }

}
```

Figure 7. The implement of method UploadMap

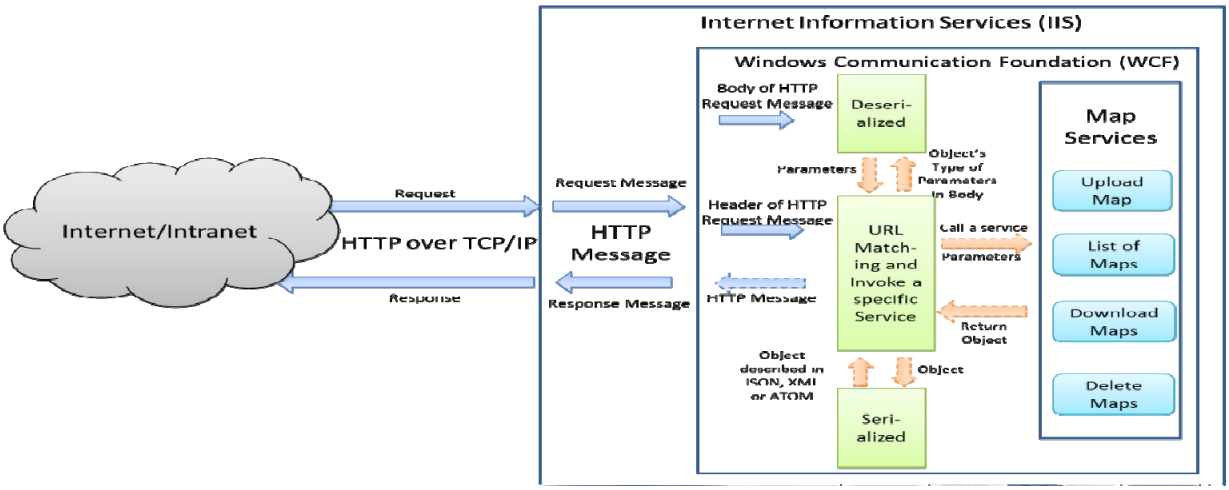

Figure 8. The processing of the system

4. Experiment

The overall of processing of this system is presented in Figure 8. The first, client make a request to server by HTTP message, this message defined by URL in Header part and the Body contains objects as parameters serialized by formatted in XML/JSON/ATOM. The HTTP message transfers over internet/intranet by TCP/IP protocol to Server, Our services hosted in IIS (Internet Information Service), IIS will get HTTP message and transfer to WCF. The header of request message used in URL matching part to find out which service requested and these parameters were need for this service, the body of request message will be transferred Deserialized part to get back objects follow description of services requested. Invoke part execute the services with parameters to get the results like object. The object will be serialized by Serialized part to get object description in one of formats as JSON, XML, ATOM or even your own format. The last Invoke part combines to make a HTTP response message and return it to client.

We also developed a client in Android mobile to use these services. Figure 9 shows an application received a map from our services and drew it on the screen.

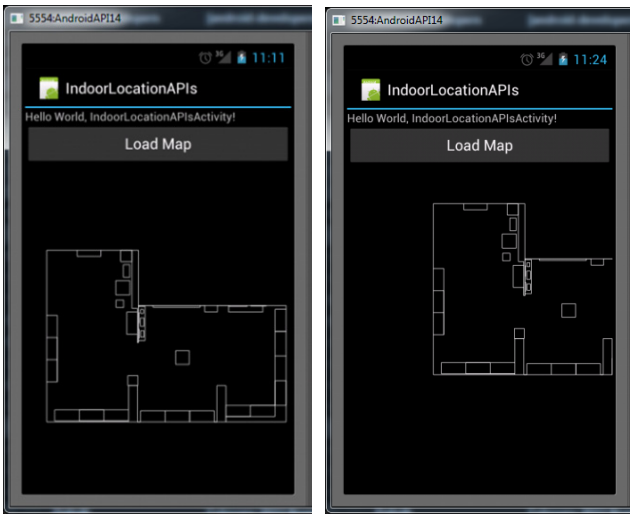

Figure 9. Android client drawing a map based the services

5. Conclusion

This paper describes our services which supports indoor LBS systems. These services are useful for managing maps and providing information and maps to mobile client under suitable format for request and responses, simple formats as JSON, XML, ATOM. We will also develop set function APIs for future research direction. These APIs help mobile client to request, receive and draw maps. Besides, we have planning to develop a big service system with series of services support for almost tasks in indoor positioning system such as building lookup table, various methods for positioning as KNN, Kamal Filter, Particles Filter and so on.

References

[1] Cesare Pautasso, Olaf Zimmermann, Frank Leymann. RESTFul web services vs. “big” web services: making the right architectural decision. WWW, 2008.

[2] http://msdn.microsoft.com/en-us/netframework/cc950529

[3] Vlad Stirbu. A RESTFul architecture for adaptive and multi-device application sharing. ACM, 2010.

[4] Roy Thomas, Architectural style and the design of network-based software architectures. Doctor Thesis, 2000.

[5] Wang Dian-Lai, Huang Xin-Yuan. Research of mobile map service for smart phone based on mobile SVG. IEEE, 2011.

[6] Xiujun Ma, Zhongya Wei, Yanwei Chai, Kunquing Xie. Integrating map service and location-based services for Geo-Referenced individual data collection. IEEE, 2008.