

# INLS 490-154: Introduction to Information Retrieval System Design and Implementation. Fall 2008.

## 11. IR on Web 2.0

Chirag Shah\*

School of Information & Library Science (SILS)

UNC Chapel Hill NC 27514

chirag@unc.edu


### 1 Introduction

In the recent years, IR has expanded from mere information access from the sources to connecting sources and information. This is done by letting different sources share their data and services, often mediated by a user. For instance, an applet running on a mobile device issues a request to a weather server, which in turn sends back the weather data for the location requested. The applet then can render the data and display it that is suitable to the mobile device that it is running on. Separating UI from services and data access allows us the flexibility that is highly desired in multiuser and multi-devices/platforms environments.

Many of these ideas are often collectively referred as Web 2.0, but there is no clear or well-accepted definition of this term. 2.0 or not, the architectures and frameworks of the web allow the developers to design highly configurable applications, content providers to publish their content easily, and users to share and access information effectively. Together, these features have led to much of the success of blogs, wikis, and many social network applications.

In this chapter, we will look at the REST architecture, which proposes to give a unique identifier, often called Universal Resource Identifier (URI), to every resource on the network, so that it can be easily defined and addressed. A resource located on the network with a URI can serve requests and respond with the data that can be interpreted at the client side. Often this data is in XML format, and so we will see how we can parse XML. That will wrap up our discussion on accessing information and making it available to the users using some of the Web 2.0 technologies.

---

\*  This handout for INLS 490-154 Fall 2008 by Chirag Shah (<http://www.unc.edu/~chirags>) is licensed under a Creative Commons Attribution-Noncommercial-No Derivative Works 3.0 United States License.

## 2 REST

REST stands for REpresentational State Transfer, but the name itself is not very indicative of what it stands for. For our purpose, it is important to realize that REST is an architecture style, not a standard. This architecture itself is based on a bunch of protocols. At the core of this architecture is a way to define and address various resources. In case of the web, the protocol is HTTP and the identification and addressing is done with Universal Resource Identifier (URI) or Locator (URL). From this, one can say the web is a REST architecture.

The reason REST architecture has become so popular and important in the recent years is the idea of Web 2.0, which promotes a variety of resources and information being shared and connected to create new applications, interfaces, and information. The example of a mobile device displaying weather information that we saw earlier, requires us to follow the REST architecture.

Let us take another example. Yahoo! allows REST requests.<sup>1</sup> This is done by creating a URL such as `http://search.yahooapis.com`. This is appended by the service name and version number such as `/WebSearchService/V1/`. Next is the method followed by a question mark, such as `webSearch?`. Finally, one can supply additional parameters, such as a query and the filters. Together, we can create a REST request such as

```
http://search.yahooapis.com/WebSearchService/V1/webSearch?appid=YahooDemo
&query=finances&format=pdf
```

Sending this request over the web brings back a response from Yahoo!, which is typically formatted in XML. In the next section we will see how to parse this response.

## 3 Parsing XML

Extensible Markup Language (XML) has become an essential tool in today's web-world to represent and share structured data. Unlike HTML that has predefined/standard markups, XML allows one to create one's own tags/elements. A sample XML file looks like the following.

```
?xml version="1.0" encoding="utf-8"?>
<DOC>
  <DOCNO>1</DOCNO>
  <TEXT>
    <HEADLINE>This is headline</HEADLINE>
    <DATE>10/29/2008</DATE>
    <STORY>This is the story.</STORY>
  </TEXT>
</DOC>
```

To parse such a file, we need three functions:

---

<sup>1</sup><http://developer.yahoo.com/search/rest.html>

1. A function to handle the start tags
2. A function to handle the data between the tags
3. A function to handle the end tags

The parser algorithm is given below.

1. Create an XML parser object.  
E.g., `$xml_parser = xml_parser_create();`
2. Assign handlers for the start and the end of an element.  
E.g., `xml_set_element_handler($xml_parser, "startTag", "endTag");`
3. Assign handler for the data between the tags.  
E.g., `xml_set_character_data_handler($xml_parser, "contents");`
4. Parse XML file and store elements and values in an array.  
E.g., `xml_parse($xml_parser, $data, feof($filePointer))`
5. Free XML parser memory.  
E.g., `xml_parser_free($xml_parser);`

Details of the implementation is left to the reader as the above algorithm can be implemented in many ways depending on the programming environment and the requirements.

## 4 Summary

- Web 2.0 about effectively sharing information, and building communities and facilitating collaboration around information.
- We covered concepts such as URI, REST, and XML.
- REST is an architecture that encompasses several protocols, but not a protocol itself.