UNIVERSITÄT KOBLENZ · LANDAU

Web Engineering
Winter Semester 2015/2016

Homework #2

Prepared by:
Adan Toscana Lopez
Jesus Garcia Perez
Qianhong Ye
Redona Kajmaku

Task 1: *Choose an application (website) from the category "Semantic Web Applications":*
a. Describe why it belongs to the semantic web applications.
b. Analyze this web application (website) based on the 8 quality aspects that are provided on page 18 WE01Introduction.

website: wikipedia
https://en.wikipedia.org/wiki/Main_Page
a. It uses Resource Description Framework (RDF) to make machine readable. For example, DBpedia can easily extract structured content associated with wikipedia resources from different parts in wikipedia.

b. 1) User-friendliness: there is a clear structure. We can easily find what we want and there are many links to further knowledge.
2) Performance: It can reply your request very quickly.
3) Reliability: Most of the acknowledge showed in that webpage will give an origin.
4) Scalability: It allows users to add and edit content, if something is missing or if it is not correct. the content through specific methods.
5) Security: It's a https website, more secure than http websites.
6) Maintainability: There is a specific group to maintain the website
7) Portability: The websites is easy to be accessed from various devices and platforms, providing the same quality of service regardless the technology being used to access it.
8) Interoperability: It is not very good, as the interaction in the website is poor. However, in the logged in section users can communicate with each other, and also modify the content.

Task 2: *What are the differences between Dom and SAX parsers? Please explain.*

DOM parser is a tree-based API which is centered around a tree structure and therefore provides interfaces on components of a tree (which is a DOM document) such as Document interface, Node interface, NodeList interface, Elementinterface, Attr interface and so on.
SAX parser is an event-based API. Usually an event-based API provides interfaces on handlers. There are four handler interfaces, ContentHandler interface, DTDHandler interface, EntityResolver interface and ErrorHandler interface.

A DOM parser creates a tree structure in memory from the input document and then waits for requests from client. A DOM parser always serves the client application with the entire document no matter how much is actually needed by the client. With DOM parser, method calls in client application have to be explicit and forms a kind of chained method calls.
SAX parser does not create any internal structure. Instead, it takes the occurrences of components of a input document as events, and tells the client what it reads as it reads through the input document. SAX parser serves the client application always only with pieces of the

document at any given time. With SAX parser, some custom methods are called [“callback” methods] when some certain events occur during parsing on xml document. These methods do not have to be called explicitly by the client, though we could call them explicitly.

DOM	**SAX**
Tree model parser (Tree of nodes)	Event based parser (Sequence of events)
DOM loads the file into the memory and then parse the file	SAX parses the file at it reads i.e. Parses node by node
Has memory constraints since it loads the whole XML file before parsing	No memory constraints as it does not store the XML content in the memory
DOM is read and write (can insert or delete the node)	SAX is read only i.e. can’t insert or delete the node
If the XML content is small then prefer DOM parser	Use SAX parser when memory content is larg
Backward and forward search is possible for searching the tags and evaluation of the information inside the tags. So this gives the ease of navigation	SAX reads the XML file from top to bottom and backward navigation is not possible
Slower at runtime	Faster at runtime

Task 3: *In this exercise you will create an XML description of a student semester plan. This description should contain*

• the personal information of the student (name, family, birthday(day, month, year), address(street, number, code, city), ...)

• Educational information (semester, field of study,...)

• Current semester plan contains current courses and relevant information on the courses such as name, time, room, subject, professor, ...

We recommend to use Notepad++ as the editor. However you can use any other editor to write the XML description. Please, only submit the XML description as text, no .xml file is needed.

```
<?xml version="1.0"?>
<student StudentID="123456">
        <personalinfo>
                <name>Adan</name>
                <family>Toscano</family>
                <passport>11223344H</passport>
                <birth>
                        <day>09</day>
```

```
				<month>11</month>
				<year>1992</year>
			</birth>
			<address>
				<street>Chile</street>
				<strnum>10</strnum>
				<zipcode>11008</zipcode>
				<city>Cadiz</city>
			</address>
		</personalinfo>
		<eduinfo>
			<field>Computer Science</field>
			<semester>7</semester>
			<ects>180<ects>
			<erasmus>True</erasmus>
		</eduinfo>
		<currentSem>
			<course CourseID="04169">
				<name>Web Engineering</name>
				<prof>Jürjens, Jan, Prof. Dr.</prof>
				<department>Informatik</department>
				<timeLecture>
					<day>Wednesday</day>
					<hourStart>08</hourStart>
					<hourEnd>10</hourEnd>
					<room>K102</room>
				</timeLecture>
				<timeSeminar>
					<day>Friday</day>
					<hourStart>10</hourStart>
					<hourEnd>12</hourEnd>
					<room>B028</room>
				</timeSeminar>
			</course>
		</currentSem>
	</student>
```

Task 4: *What are some of the essential features of Ubiquitous web applications? Why is it challenging to build Ubiquitous web applications in general? To answer these questions you can use the provided references. However, you can use any other scientific source.*

Ubiquitous web applications are distinguished from the personalised service they offer. They are designed to deliver services that are accessible from users at any time and at any place, regardless from the device being used. Basic properties of these applications are the customised services, which are location aware, time sensible and are accessible from various digital platforms.

However, there are a couple of challenges to be faced as designing such applications. The heterogeneous resources that need to be handled raise the need of standardization. W3C has been working and specifying several standardised web formats, such as XHTML, SVG, SMIL or XForms. These standards have an impact in the portability and scalability of the applications.

Based on the study case of Telecom Italia, providing web services which are more user context oriented brings also the need for a new business model for a lot of companies that try to implement such approaches. There are new challenges related to privacy and security that should be highly in consideration, as users share in these applications very sensitive data.

References

A survey on web modeling approaches for ubiquitous web applications, International Journal of web Information Systems Vol.4 No.3, 2006, pp 234-305

Heiko Desruelle, John Lyle, Simon Isenberg, Frank Gielen, On the Challenges of building a Web-based Ubiquitous Application Platform, UbiComp12, USA, 2012

Claudio Venezia, Carlo Alberto Licciardi, Improve ubiquitous Web applications with Context Awareness, Telecom Italia, Torino, Italy