Course Description

# Web Architectures

The course **Web Architectures** is concerned with the Web's information models, the interaction styles and protocols of its software components, and the interference of these aspects with the needs of its users. The course provides some of the architectural and technical foundations and principles necessary to competently discuss and judge questions concerning the web from diverse perspectives.

## Contents

## The Big Idea

Over the past several years, the usage of the Internet has changed from predominantly client/server-based Web server interactions to also involving the use of *more decentralized applications*, where the contributions of the users are essential for the value of the application as a whole: applications evolve to meshes of people, information and services around the Internet.

The simplicity of the *initial architecture of the World Wide Web*, as it was layed down by Tim Berners-Lee and Robert Caillou in 1990 [1], has proven to be a solid base for the development of *architectural patterns*, allowing for a plethora of new forms of communication and interaction among people, among people and applications and informations, and among applications. Conferencing, social networking, and novel forms of E-Commerce applications are examples of categories of such applications having profound implications on the way we live and work.

In this course some *architectural patterns* for this quickly evolving universe of applications are explained, and applications based on these patterns are considered.

The *synchronous interaction* is the basic form of interaction in the WWW. In addition to providing the basis for the access to information coded in HTML pages, it is nowadays the basis for the use of *Web Services* and the access to any kind of online resources. In the domain of business information systems this type of interaction is currently predominant being the building block for the automation of business processes.

The *asynchronous interaction* is well known from forms of interpersonal communication like *instant messaging* and *(micro)blogging*. Recently it has been adopted as a style of interaction for the integration of applications based on the Internet. The term *event-driven architecture* has been coined and corresponding systems are characterized by their ability to instantly react to events in complex environments.

The *peer-to-peer interaction* is likewise motivated by forms of interpersonal communication, in this case *conferencing* and *mass distribution* of digital assets like music of movies. Based on this form of interaction, communication, that traditionally used different networks like the telephone network or broadcast media, nowadays uses the Internet and is increasingly being integrated into other Web applications.

At the beginning of the course, basic architectural components like *identification* and *interaction* are treated, and the aspect of *information architecture* is covered. This serves as a basis for the discussion of the aforementioned patterns. Whilst initially information conveyed in the Web was solely used for display purposes, nowadays the automatic processing of information by applications is equally important. Information architectures for this purpose are often based on the *Extensible Markup Language* (XML). In order to cope with the decentralized and open nature

of the web, in addition, concepts like the ones from the *Ressource Description Framework* (RDF) are needed.

## Intended Learning Outcomes

After participation in the course, the students are able to analyze and discuss web applications from a architectural perspective. They

- perceive *openness* and *decentralization* as key properties of the web,
- understand the impact of fundamental principles of *resources*, *interaction*, and *identification*,
- understand the role of *architecture* and *architectural patterns* for the analysis and evolution of web applications, and
- are able to identify and discuss role of *information modeling* in the web in contrast to more traditional data(base system) centric approaches.

Application classes like *peer-to-peer applications*, *synchronously or asynchronously interacting applications and services* can be distinguished and discussed based on their architctural patterns, protocols and typical uses.

## Structure of the Course

### Evolution of the Web during the past Decades

The evolution of the Web since its emergence in 1990 is examined from multiple perspectives in this introductory part of the course.

- Firstly the constituent idea of the web as a *network of pieces of information* is considered. This perspective is explicated in Part IV of the networks book authored by Easley and Kleinberg [2].
- Secondly the Web is considered as a means to *access dynamic content*, normally generated as a result of queries to databases. This aspect of the web is present since its early years. It has been analyzed systematically in recent years and the notion of *deep web* has been created in this context. This perspective is explicated in the 2007 article by He, Patel and others [3].
- Thirdly the Web is considered as *means to collaborate* - the term *Web 2.0* has been coined by Tim O'Reilly in 2007 for this perspective. The basic ideas can best be perceived through the initial article by Tim O'Reilly[4].
- Fourthly the Web is considered as a *network of linked data*, that can adequately be processed by applications without relying on human intervention. The difference to the deep web approach is, that the data is distributed potentially over the whole web, and there is no central system or instance managing the data. This approach is being forwarded by a group of researchers around Tim Berners-Lee. A recent overview article by Bizer, Heath, and Berners-Lee serves as a base for a first examination of the topic[5].

## Fundamental Concepts

The fundamental concepts of the Web are few, and that is to a large extent the power of the initial Web architecture. In a early discussion of design issues[6] this approach has been characterized by the rule:

```
When expressing something, use the least powerful language you
can.
```

Therefore the fundamental architectural definitions[7] deal with just three topics: *identification*, *interaction*, and *data formats*. A rather accessible introduction to the concepts may be found in chapter 5 of the book authored by Taylor and Harrison[8].

### Identification

The Web is an information space in which the items of interest, referred to as *resources*, are identified by global identifiers called *Uniform Resource Identifiers* (URI). The types of URIs and considerations and design choices are explained in the official document of the W3C[7].

### Interaction

The *Hypertext Transfer Protocol*(HTTP) is a key element of any software architecture developed for the Web. Its basic design can be well appreciated based on the explanation in chapter 8 of the textbook by Møller and Schwartzbach[9]. Beyond the use of HTTP for the access to more or less static information resources, it is increasingly being used also as architectural foundation of distributed systems in the Web. Roy Fielding has elaborated this approach with the notion of *REspresentational State Transfer (REST)* in his doctoral dissertation[10].

### Data Formats

The data format first used in the Web is the *Hypertext Markup Language (HTML)*. Over the years data formats have grown in number because content providers are not constrained by the Web architecture in the use of data formats. This flexibility is important because there is a continuous evolution of the applications, resulting in new data formats and refinements of existing ones. As the use of the Web advanced to the interconnection of information sources and applications, an extensible approach catering to convey documents, data, and binary content becomes necessary. It has been developed since the end of the 1990ies with the family of standards for the *Extensible Markup Language*(XML). An introduction to the relevant concepts is given in chapter 3 of the book authored by Taylor and Harrison[8].

## The Architects's View of the Web

### Web 2.0

In this section a view is taken beyond the creation of more or less "classical" Web sites into what often is referred to as Web 2.0. The term generally refers to a set of social, architectural, and design patterns resulting in the migration of major parts of organizations to the Web as a

platform. These patterns focus on the interaction models between communities, people, computers, and software. Thus for the consideration of Web 2.0 applications the perspective analogous to the one of an architect of buildings is important: it has the focus on the support of life, work, and social interaction by technology. For the classical architect the supportive technology is the building, for the Web architect the supportive technology is the Web application.

As a introductory text the chapters 1 and 2 of the book by Governor et al.[11] is recommended.

### Architectural Patterns

With the advent of Web 2.0, the need arises to architect Web applications on top of the fundamental Web technologies. In order to promote uniform solutions to recurring problems in such applications, the concept of *architectural pattern* is common. Patterns are abstract designs that may be applied to multiple and divers manifestations of a common problem. Specific patterns of Web 2.0 include "Service-oriented Architecture Pattern", "Mashup Pattern", "Collaborative Tagging Pattern" and many others. A rather comprehensive overview can be obtained in chapter 7 of the book by Governor et al.[11].

## Information Architectures

In the early years of the Web, information has been transferred between servers and browsers mainly for the purpose of displaying it to users. Standards like HTML, JPEG, pdf and a plugin architecture for extensions have established the information model for this purpose. In the last decade the transfer of information among applications in the Web for the purpose of automatic processing is gaining importance. As a consequence the need for more rigorous information models with explicit semantics arises.
The XML has been established as a unanimously accepted standard and it is therefore treated in some depth in the first part of this chapter. One of the key strengths of the XML approach is its alleged universality. In fact there is probably hardly any information modeling task, a XML enthusiast wouldn't find a solution for. The price however often is complexity, and the aim to use the least complex information model for a given problem often cannot be reached with an XML approach.
Alternate approaches to information modeling in the Web have therefore gained momentum in recent years. The *Java Script Object Notation* (JSON), the *Resource Description Framework* (RDF), and the so called *Microformat* approach are dealt with in the second part of the chapter.

### Extensible Markup Language (XML)

An explanation of key XML concepts also with motivating examples and exercises can be found in chapter 2 of the textbook of Møller and Schwartzbach[9]. In order to see the contrast to the display oriented approach of HTML it may be a good idea to have a look into chapter 1 of the textbook.

### XML Schema Languages

Schema languages are necessary to define XML vocabularies and document structures for concrete information exchange scenarios. They provide type systems in oder to be able to map local type systems as they are provided by programming languages or database management systems to a globally accepted system. As one widely accepted example for such a schema language *XML Schema* should be studied in some depth based on the explanation in chapter 4 of Møller and Schwartzbach[9]. The discussion of the alternate approach of *Document Type Definition* (DTD) may be considered as a first easy-to-comprehend approach to the issue.

### Java Script Object Notation (JSON)

Originating from the scripting language supported by browsers, Java Script, the *Java Script Object Notation* (JSON) has gained importance in recent years. It is a less complex however also less expressive alternative to XML. Douglas Crockford, its main promotor, has coined the notion of "fat-free alternative to XML" [12]. The information available at the respective Web site [13] as well as the more programming oriented introduction by Microsoft [14] provide a good access to the approach.

### Resource Description Framework

XML documents represent basically hierarchical structures, which for many cases is not an adequate solution. Therefore an approach to model information as general graphs has been promoted since the mid 1990ies for the purpose of embedding *metadata* into the representation of Web resources. It is called *Resource Description Framework* (RDF) and forms also the basis for the provision of *linked data* in the Web as well as for the vision of the *Semantic Web*. A rather accessible introduction is provided by Joshua Tauberer's contribution[15].

### Micro Formats

A pragmatic approach to exchanging data, mainly in the context of personal information management, is provided by so called *Microformats* (µF). It is a web-based approach to semantic markup seeking to convey metadata and other attributes in web pages and other Web resources. This approach allows software to automatically process information such as contact information, geographic coordinates, and calendar events. An introduction is given in chapter 1 and 2 of the book by John Allsop[16].

## Synchronous Interaction

When it comes to designing the interaction of software components in the Web, the *request/response* type of interaction as it is suggested by the concept of the HTTP is a obvious choice. Looking at it as a general principle it is called *synchronous interaction*, because the components synchronize their processes at the time of the request/response interaction. It is the typical interaction between browser and server: when requesting a resource with HTTP GET, the browser waits until the server delivers the HTML representation of the resource. And it is the standard choice for designing the interaction between applications, when the exchange of

information between them is necessary. Therefore we start with looking at different architectural styles to design such synchronous interaction.
In the following chapters alternate types of interaction like asynchronous interaction and peer-tio-peer interaction will be looked at. These types are not as obvious as the synchronous interaction, however they are important to adequately fulfill the needs of users of novel types of applications like *social networking*.

### Web Services

The notion of *Web Service* is being used for the provision of access to a computational facility in the Web at a certain URI. This may mean access to certain functions of business information systems that in this way are made available for for the invocation by systems of customers or partners, or it may may mean access to data like weather information, sensor data, maps and many other types of data. In many cases, the computational facility behind a Web Service invokes other Web Services to perform its task. In this case the notions *Web Service composition* or *Mesh-up* are often used.
In fact there are different definitions for Web Service. A rather general and stable one has been provided by the *World Wide Web Consortium* in 2004[17]:

```
A Web Service is a software system identified by a URI,
whose public interfaces and bindings are defined, described using
XML.
Its definition can be discovered by other software systems.
These systems may then interact with the Web service in a manner
prescribed by its definition,
using XML based messages conveyed by Internet protocols.
```

A explanation of the most relevant concepts of Web Services can be found in chapter 7 of the book authored by Taylor and Harrison [8].

### REST Architectural Style

In the *Representational State Transfer* (REST) style it is attempted to describe architectures which use HTTP or similar protocols by constraining the interface to a set of well-known, standard operations like - in case of HTTP - GET, POST, PUT, DELETE. Here, the focus is on interacting with stateful resources, rather than messages or operations.
Recently the REST STyle has gained a lot of attention since emerging public services in the Web like Amazon's S3[18] and various services of Google have strengthened their support. Frequently RESTful Web Services form the base of the so called Mesh-ups in Web 2.0.
A introduction can be found in chapter 5.6 of the book authored by Taylor and Harrison[8].

### SOAP Architectural Style

Web Services in the SOAP style, often denoted "Big Web Services", use XML messages that follow the SOAP standard to invoke operations at endpoints. It is a popular style within traditional enterprises. In such systems, there is often a machine-readable description of the operations offered by the service written in the *Web Services Description Language* (WSDL). The latter is not a

requirement of a SOAP endpoint, but it is a prerequisite for automated client-side code generation in many Java and .NET SOAP frameworks. An explanation is found in chapter 16 of the book authored by Taylor and Harrison [8].

### Business Process Modeling

As mentioned earlier, the Web Services standards define a unified way to access functions in *business information systems*. This makes it possible to combine such functions across system platforms into larger application flows. In case these flows correspond to the business processes of the organization, the notion of *business process modeling* is used. The vision is, to design tools for this task that are usable and in fact used by the business process owners directly (as opposed to having to engage IT Experts). This would allow organizations to adapt their business processes instantaniously to the needs of the market.
The topic is dealt with in some depth in chapter 9 of the textbook by Papazoglou[19].

## Asynchronous Interaction

In the previous chapter, a architectural style has been introduced, that has *service* as its central concept for modeling the interaction of software components. Services are offered by *service providers* and used by *service requestors*. This architectural style has a number of consequences. The communication between provider and requestor usually is a synchronous interaction. That means, that the requestor has to "know'" the name and address of the service and the syntax and semantics of the service invocation. After invocation the requestor is waiting for the response of the provider before continuing its computation. The requirement for a priori knowledge and the synchronicity of invocation represent a *tight coupeling* of the software components.

Such tight coupling is undesirable, at least when the number of interacting software components is high. The need to wait for responses leads to performance degradations, especially in the case of nested invocations and the dependency on detailed syntax and semantic regulations increases the effort induced by changes in the system.

Therefore a strong movement to a *more loosely coupled architecture* can be observed. With the *Event driven Architecture* a architectural style has emerged, that loosens the dependencies of the service oriented architectural style in terms of both, the timing and the binding of the communicating components.

### Event Driven Architectures

Event driven architecture is a architecture style, that has *event* as its central concept for modeling the interaction of software components. The interaction deals then with production, detection, consumption of, and reaction to events. An *event* can e defined as a significant change of state of the system: for example in an auction platform, the end of a specific auction would be considered as an event. The state of that specific auction changes from 'open' to 'closed'. The design of the platform would deal with the detection of that event, the communication of the event to other software components like the billing system, the database subsystem and others.

The consumption of an event in such components would deal with the required actions within the components according to the business logic.

An overview of the concepts of event driven architectures can be obtained in chapter 1 of the textbook of Hugh Taylor et al.[20]. A motivation of the event driven approach as opposed to the service oriented one is contained in a post in Jack van Hoft's blog[21].

### Instant Messaging

Instant Messaging (IM) is well known as a medium for interpersonal communication. Services like *ICQ*, *AIM*, *Google Talk* are being used for instant communication, in the beginning primarily by teenagers, but nowadays increasingly also in business.

As the need for event driven architectures in Web applications is arising, a means for asynchronously communicating events in the Web among applications is necessary. There is a number of commercial services fulfilling this need, like Amazon's *Simple Queue Service*(SQS)[22] and Google's *Android Cloud to Device Messaging Framework* (C2DM)[23]. These are however firstly proprietary in terms of their interfaces and protocols. Secondly using such services in an application means that all messages come into the hands of the particular provider. This may or may not be considered as a problem for a Web application.

An alternative, that will be pursued in some depth in this chapter, is the use of standardized Instant Messaging protocols and the corresponding open source frameworks for the conveyance of messages in the Web. This approach allows to provide an application in the Web without dependencies on third parties like the aforementioned ones.

The *Extensible Messaging and Presence Protocol* (XMPP, formerly named *Jabber*) is an open, XML-based protocol originally aimed at near-real-time, extensible instant messaging, that can also be used for application-to-application communication. The protocol has been standardized by the *Internet Engineering Task Force* (IETF) and can therefore be considered as a open standard.

In order to find a first access to the XMPP idea a publication by IBM[24] can be used. Deeper insights can by gained in chapters 1, 2, and 8 of the textbook by Peter Saint-André[25].

### (Micro)blogging

Somewhat similar to the approach of using IM frameworks for the conveyance of messages is the idea of using blogging services or frameworks for this purpose. The additional potental of this approach comes from the way of addressing: using IM, the sender of a message has to *address* one or more recipients explicitly. When using blogging, the sender *publishes* the message without explicitly addressing recipients. The messages are rather conveyed to the recipients based on *subjects* and their *subscription* to subjects.

This approach is rather new. It is described to some extent in a paper by Böhringer(in German) and another one by Wallach[26][27].

Besides the consideration of this novel usage of the blogging approach, the base protocols should be looked at. *Atom* is perhaps the most comprehensive approach. It is described in chapter 23 the textbook authored by Taylor and Harrison [8].

### Peer-to-peer Interaction

*Peer-to-peer* (P2P) denotes any distributed network architecture composed of participants that make a portion of their resources such as processing power, disk storage or network bandwidth directly available to other network participants. In many variants there is no need for central coordination instances such as servers or stable hosts. Peers are both suppliers and consumers of resources, in contrast to the traditional client–server model where only servers supply, and clients consume.
Peer-to-peer was popularized by file sharing systems like Napster. Peer-to-peer file sharing networks have inspired new structures and philosophies in other areas of human interaction. In such social contexts, peer-to-peer refers to Internet enabled egalitarian social networking that is currently emerging throughout society.
An introduction to the background and concepts of peer-to-peer interaction is available in chapter 6 of the textbook authored by Taylor and Harrison [8].
Although issues of security and privacy are still not sufficiently addressed in many P2P approaches, the concept is promising.

#### File Sharing: Gnutella

One of the first application classes of the Web, that promoted the user from the role of a mere recipient to the role of a contributor was the so called *file sharing*, which became popular towards the end of the 1990ies. Although this technology is best known for doubtable usages like the mostly illegal sharing of music or movie files, it is also the basis for the sharing of user generated content in collaborative situations without necessarily involving a central agent.
The concepts of file sharing can be studied considering *Gnutella*, which is well described in chapter 10 of the textbook authored by Taylor and Harrison [8].

#### Mass Distribution: Bittorrent

Following the idea of employing the resources of the multitude of systems in a network, *Bittorrent* has been established as a way to distribute large amounts of information to many recipients. A first insight into the Bittorret concepts can be gained in chapter 13 of the textbook authored by Taylor and Harrison[8].

#### Conferencing: SIP

Classical applications of the telephone network like *telephony* and *conferencing* move more and more to the Internet. *Voice over IP* (VoIP) is the notion for a set of protocols on top of the IP, that allow for attaching telephones or telephony applications to the Internet, which is also the base for the Web. This may be considered as a mere technical direction, however using one uniform network offers significant potential beyond the current applications:

- The usage of the same underlying protocols in telephony and Web applications, eases the creation of novel integrating applications. Scenarios that are being discussed include online shopping applications, where sales persons can be contacted by telephone in the context of the current "location" of the customer's Web visit.
- The architecture of the VoIP standards separates the *conference management*, i.e. the the establishment and release of the conference, from the transfer of audiovisual data. This allows for using the conferencing part also for other applications like online games or joint editing.

A introduction to the concepts of the *Session Initiation Protocol* (SIP), currently the predominant set of standards in the VoIP arena, has been supplied by the SIP Center[28]. Different usage scenarios for SIP are also discussed there[29].

### Where are we Going from Here

The Web is on its way from the publication medium, which it has been since its inception in the beginning of the 1990ies, to the general networking medium, that connects people to other people, to information, and to services. The architectural challenges that this transition poses, have partly been addressed in this course. Additional challenges are on the horizon, like ubiquity induced by the network access of mobile phones, cars, ..., or like the need to increase the connectivity not only on the syntactic level but also on the semantic level.

A major challenge will be trust: in how far will and can people trust the information that they obtain, in how far can they trust the privacy of the communication that they have across the Web. This issue does have architectural aspects, however it will not be possible to come to a solution without addressing the societal and political aspects of the issue.

## Didactic Concept, Schedule and Assignments

The course is mainly built upon online workshops. They take place on three evenings as synchronous events with a duration of three hours each and with equal emphasis continously as asynchronous cooperation in the form of discussions and clarifications through E-mail, discussion forums, and other tools in the learning platform.

The learning objectives emphasize the ability to competently take an active part in discourses on issues of Web architecture. Therefore the didactic concept emphasize interactions between the student and the lecturer as well as among the students. Of course the interactions require a solid body of knowledge which s to be established based on the referenced resources.

The learning process is organized into four phases, each emphasizing a certain topic of the course. The phase starts with a brief introduction by the lecturer of around 30 minutes. Then learning teams are established of around 4 to 7 students. The teams have the task to acquire the relevant concepts, clarify questions, formulate glossary entries, questions, assumptions and hypotheses during the phase. This work is supposed to mainly take place asynchronously. The

phase concludes with discussing questions, assumptions and hypotheses and working on assignments based on the acquired knowledge.

### Introductory Lesson on Site

The introductory lesson is used to lay out the objectives and the approach of the course, and to introduce the first phase.
In the first phase the chapters *Evolution of the Web*, *Fundamental Concepts*, and the *Architect's View of the Web* are treated. For the upcoming asynchronous phase, the groups are requested to produce glossary entries for the - in their view - 10 most important concepts or issues and to produce a five minute online presentation about the architects view of the Web. The target audience is assumed to be the management of a development group, which is about to start a Web project.

### 1st Online Workshop

The first workshop concludes the first phase in a plenary session, where selected groups present their results. Selected questions, assumptions and hypotheses will be discussed and clarified. The presentations and glossary entries will be critically discussed in order to clarify the requirements for such artifacts in terms of substance and style.
The second phase is then introduced with a lesson of about 30 minutes. In the second phase the chapter *Information Architectures* is treated. The groups are allowed to reform for the work in the second phase. The groups then work in virtual breakout rooms on a first assignment dealing with the design of a XML vocabulary for a given domain. Selected results are then presented in the plenary. Assignments for the upcoming asynchronous groupwork phase are explained. The assignments deal with the design of information structures in XML Schema, JSON, and RDF employing diverse state-of-the-art tools. In addition questions, assumptions and hypotheses should be formulated.

### 2nd Online Workshop

The second workshop concludes the second phase in a plenary session, where selected groups present their results. Selected questions, assumptions and hypotheses will be discussed and clarified. The solutions to the assignments will be critically discussed focussing on the identification of design alternatives.
The third phase is then introduced with a lesson of about 30 minutes. In the third phase the chapter *Synchronous Interaction* is treated. The groups are allowed to reform for the work in the third phase.The groups then work in virtual breakout rooms on a first assignment dealing with the REST architectural style. The groups are asked to define resources and interactions for E-Commerce transactions in a given scenario. Assignments for the upcoming asynchronous groupwork phase are explained. The groups firstly are asked to produce glossary entries for the - in their view - 10 most important concepts or issues. Secondly they are asked to step deeper into selected areas of synchronous interaction in the Web. This may happen in the form of the development of a 10 Minute presentation on a specific area or in the form of a prototype implementation using publicly available Web services. In addition questions, assumptions and hypotheses should be formulated.

### 3rd Online Workshop

The third workshop concludes the third phase in a plenary session, where selected groups present their results. Selected questions, assumptions and hypotheses will be discussed and clarified. The solutions to the assignments will be critically discussed focussing on the identification of differences of the architectural styles and discussing the consequences of theses differences.
The fourth and last phase is then introduced with a lesson of about 30 minutes. In the fourth phase the chapters *Asynchronous Interaction* and *Peer-to-Peer Interaction* are treated. The groups are allowed to reform for the work in the phase.The groups then work in virtual breakout rooms on a first assignment dealing with the use of Twitter for publish/subscribe messaging . Assignments for the upcoming asynchronous groupwork phase are explained. The groups firstly are asked to produce glossary entries for the - in their view - 10 most important concepts or issues. Secondly they are asked to review their learning process during the course as a whole and formulate questions, assumptions and hypotheses.

### Concluding Workshop on Site

This on site workshop concludes the fourth phase and the course as a whole in a plenary session, where selected groups present their results. Selected questions, assumptions and hypotheses will be discussed and clarified.

## Examination

During the concluding on site appointment a written examination for the module is to be passed. This course contributes tasks corresponding to 45 minutes working time.

## References


↑ "Berners-Lee, Tim; Cailliau, Robert (November 12, 1990). "WorldWideWeb: Proposal for a hypertexts Project". http://w3.org/Proposal.html. Retrieved July 20, 2010.

↑ Easley, David and Kleinberg, Jon. Networks, Crowds, and Markets: Reasoning About a Highly Connected World. Cambridge University Press.

↑ He, Bin and Patel, Mitesh and Zhang, Zhen and Chang, Kevin Chen-Chuan, (2007). "Accessing the deep web". *Commun. ACM* (ACM) **50** (5): 94--101. ISSN 0001-0782. http://doi.acm.org/10.1145/1230819.1241670.

↑ Tim O'Reilly (2005-09-30). "What Is Web 2.0". O'Reilly Network. http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html. Retrieved 2010-08-06.

↑ Christian Bizer, Tom Heath, Tim Berners-Lee (2009). Linked Data - The Story So Far. International Journal on Semantic Web and Information Systems (IJSWIS), Vol. 5(3), Pages 1-22. DOI: 10.4018/jswis.2009081901

↑ Tim Berners Lee (2009-08-27). "Evolvability". World Wide Web Consortium. http://www.w3.org/DesignIssues/Evolution.html. Retrieved 2010-10-20.

↑ 7.0 7.1 Berners-Lee, Tim; Bray, Tim; Connolly, Dan; Cotton, Paul; Fielding, Roy; Jeckle, Mario; Lilley, Chris; Mendelsohn, Noah; Orchard, David; Walsh, Norman; Williams, Stuart (December 15, 2004). "Architecture of the World Wide Web, Volume One". W3C. http://www.w3.org/TR/webarch/.

↑ 8.0 8.1 8.2 8.3 8.4 8.5 8.6 8.7 8.8 Taylor, Ian J. and Harrison, Andrew (2009). *From P2P and Grids to Services on the Web*. Springer.

↑ 9.0 9.1 9.2 Møller, Anders and Schwartzbach, Michael (2006). *An Introduction to XML and Web Technologies*. Addison Wesley.

↑ Fielding, Roy Thomas (2000), *Architectural Styles and the Design of Network-based Software Architectures*, Doctoral dissertation, University of California, Irvine, http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm

↑ 11.0 11.1 Governor, James and Hinchcliffe, Dion and Nickull, Duane (2009). *Web 2.0 Architectures*. O'Reilly.

↑ Crockford, Douglas (December 6, 2006). "JSON: The Fat-Free Alternative to XML". http://www.json.org/fatfree.html. Retrieved July 3, 2009.

↑ json.org (July 30, 2010). "Introducing JSON". http://www.json.org. Retrieved July 30, 2010.

↑ Aziz, Atif (February, 2007). "An Introduction to JavaScript Object Notation (JSON) in JavaScript and .NET". http://msdn.microsoft.com/en-us/library/bb299886.aspx. Retrieved July 20, 2010.

↑ Tauberer, Joshua (January, 2008). "What is RDF and what is it good for?". http://rdfabout.com/. Retrieved July 21, 2010.

↑ Allsop, John (2007). Microformats: Empowering Your Markup for Web 2.0. Friends of Ed.

↑ Austin, Daniel (February 11, 2004). "Web Services Architecture Requirements". http://www.w3.org/TR/wsa-reqs/. Retrieved July 22, 2010.

↑ Amazon (July 22, 2010). "Amazon Simple Storage Service (Amazon S3)". http://aws.amazon.com/de/s3/. Retrieved July 22, 2010.

↑ Papazoglou, Michael (2008). Web Services: Principles and Technology. Pearson Education.

↑ Taylor, Hugh (2009). Event-Driven Architecture: How SOA Enables the Real-Time Enterprise. Addison Wesley.

↑ van Hoof, Jack (December, 2006). "How EDA extends SOA and why it is important". http://soa-eda.blogspot.com/2006/11/how-eda-extends-soa-and-why-it-is.html. Retrieved July 23, 2010.

↑ Google (July 27, 2010). "Amazon Simple Queue Service (Amazon SQS)". http://aws.amazon.com/de/sqs/. Retrieved July 27, 2010.

↑ Amazon (July 27, 2010). "Android Cloud to Device Messaging Framework". http://code.google.com/intl/de/android/c2dm/index.html. Retrieved July 27, 2010.

↑ Jones, Tim (September 18, 2009). "Meet the Extensible Messaging and Presence Protocol (XMPP)". http://www.ibm.com/developerworks/webservices/library/x-xmppintro/index.html. Retrieved July 23, 2010.

↑ Saint-André, Peter (2009). XMPP: The Definitive Guide: Building Real-Time Applications with Jabber Technologies. O'Reilly.

↑ Böhringer, Martin (December 16, 2009). "Microblogging". http://www.gi-ev.de/service/informatiklexikon/informatiklexikon-detailansicht/meldung/microblogging-264.html. Retrieved July 23, 2010.

↑ Sandler, Daniel; Wallach, Jan (April 2009). *Birds of a FETHR: Open, decentralized micropublishing.*. Proceedings of the 8th International Workshop on Peer-to- Peer Systems (IPTPS '09). http://www.usenix.org/events/iptps09/tech/full_papers/sandler/sandler.pdf.

↑ SIP Center (July 29, 2010). "What Is SIP Introduction". http://www.sipcenter.com/sip.nsf/html/What+Is+SIP+Introduction. Retrieved July 29, 2010.

↑ SIP Center (July 29, 2010). "Personal/Selective Presence with SIP Presence and XML directives". http://www.sipcenter.com/sip.nsf/html/Personal+Selective+Presence. Retrieved July 29, 2010.
Fielding, R.; Gettys, J.; Mogul, J.; Frystyk, H.; Masinter, L.; Leach, P.; Berners-Lee, T. (June 1999). *Hypertext Transfer Protocol — HTTP/1.1*. Request For Comments 2616. Information Sciences Institute. ftp://ftp.isi.edu/in-notes/rfc2616.txt.
Niels Brügger, ed. *Web History* (2010) 362 pages; Historical perspective on the World Wide Web, including issues of culture, content, and preservation.
Fielding, R.; Gettys, J.; Mogul, J.; Frystyk, H.; Masinter, L.; Leach, P.; Berners-Lee, T. (June 1999). *Hypertext Transfer Protocol — HTTP/1.1*. Request For Comments 2616. Information Sciences Institute. ftp://ftp.isi.edu/in-notes/rfc2616.txt.
Berners-Lee, Tim; Bray, Tim; Connolly, Dan; Cotton, Paul; Fielding, Roy; Jeckle, Mario; Lilley, Chris; Mendelsohn, Noah; Orchard, David; Walsh, Norman; Williams, Stuart (December 15, 2004). *Architecture of the World Wide Web, Volume One*. Version 20041215. W3C. http://www.w3.org/TR/webarch/.

*27.02.2019*