# AN OPTIMUM ARCHITECTURE FOR SOA

**Sajjad Waheed**

Mawlana Bhashani Science and Technology University, Tangail

Email: swaheed.iu@gmail.com

***Abstract:*** *Service-oriented architecture (SOA) has emerged as a new architecture in the system software development. Web 2.0 would not be able to work if SOA is not structured well. SOA is structured and has better ability of reuse of legacy software and other service components than Web 2.0. This study points out the differences and similarities between the SOA and Web 2.0, looked if a convergence of these two is possible and finally discussed a reference model after studying similar models for SOA.*

***Key words:*** *Service-oriented architecture, Web 2.0, reference model, AJAX, OSI model*

## 1. Introduction to SOA and Web 2.0

The Internet is quickly becoming a mean for sharing information and a way of communication. Internet has evolved into a modern day necessity. The current tendency is that users rely on the Internet for commerce, email, banking, instant messaging, and various other daily tasks. This evolution is often referred to as Web 2.0. Web 2.0 denotes personalization of web in terms of streaming videos, social networking, collaboration, and updated client technology. One may ask if Web 2.0 is the end of the way?

The answer to that question may look strange at first, but obviously the future is Service-Oriented Architecture (SOA). One of the reasons is that SOA provides increasing software efficiencies. SOA has ability to connect dissimilar applications and provides with an unprecedented degree of integration. In the enterprise world, Microsoft® SharePoint™ focuses on services and integrates with other products seamlessly. Thus writing complex middleware to integrate corporate systems may end within few years.

In some recent works, SOA and Web 2.0 were termed as "notions of complexities-hiding and reuse" capabilities along with "concepts of loosely coupled services" [1]. In some other works, however, they were not seen as parallel philosophies, whereas others view both as complementary – regarding SOA as global Web 2.0 [2].

Tim O'Reilly coined the term "Web 2.0" as 'a quick growing set of Web-based applications.' From an end-user perspective, Web 2.0 is a set of SOA technologies for consumers. Traditional SOA is the answer for addressing integration challenges for enterprise web-users. The web contents aggregation capability of Web 2.0 can be integrated into an SOA solution.

SOA is the philosophy of encapsulating application logic in services with a uniformly defined interface [3] and makes these publicly available via discovery mechanisms [4]. In another introduction, SOA was described as "an emerging architectural style for crafting next generation enterprise applications. While the SOA approach strongly reinforces well-established, general software architecture principles such as information hiding, modularization, and separation of concerns, it also adds additional themes such as service choreography, service repositories, and the service bus middleware pattern" [5]. Web 2.0 is defined as the philosophy of mutually maximizing collective intelligence and added value for each participant by formalized and dynamic information sharing and creation [6].

### 1.1 Organization of the paper

This write up is organized in a manner that in the following chapters we shall gradually examine the properties of SOA and Web 2.0. We have put emphasis on the similarities and dissimilarities of SOA and Web 2.0 and their applications. Thus a picture of SOA and Web 2.0 becomes clear to the reader. Few previously obtained results are discussed.

Some researchers tried to converge on the SOA and the Web 2.0. Their formulations were also considered and studied. Since there are

Date of submission : 02.01.2011 Date of acceptance : 23.11.2013

differences in basic philosophy, a mixture of SOA and Web 2.0 seems quite impossible. For this reason importance of SOA was discussed in brief.

Since any basic architecture should have a reference model for it future developments, we also looked into previous studies, compared them and then go ahead with out proposition. Our proposition was noted as the conclusion of the paper where we concluded that the SOA architecture should not have more than seven layers, conforming with the standard of OSI model.

## 2. Comparing Dissimilaties

A comparative analysis was performed on 40 real-world use cases of both the Web 2.0 and the SOA. In the case of the Web 2.0, forty popular applications [6] were studied, while case studies provided by Systeme Anwendungen und Produkte in der Datenverarbeitung (SAP), IBM, and Gartner were leveraged to investigate SOA solutions. The Institute for Media and Communications Management (MCM) at the University of St. Gallen's business model framework was leveraged to structure the analysis along seven major components as depicted in the figure 1.

From an end user perspective, Web 2.0 is a set of SOA technologies for consumers; traditional SOA is the answer for addressing integration challenges for enterprise Web users. On the other hand, Web 2.0 embodies the next generation of the Web that supports collective intelligence, community-based collaboration, social networking, content publishing, and content integration over the Internet.

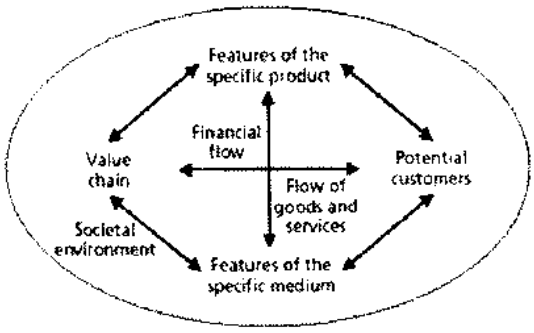


**Figure 1 :** Major components of SOA [6]

The Web content aggregation capability of Web 2.0 can be integrated into an SOA solution. Mobile communications (like 3G), knowledge sharing, desktop applications, services mashups, podcasts, and the 3D virtual world and many more have used Web 2.0 technologies. From a solution architecture perspective, a SOA is the foundation of Web 2.0 and beyond [7].

### 2.1 Web 2.0 Applications

According to [6], Web 2.0 has got the following applications:

- Communities can unify users through a common ideal such as social networking or knowledge sharing.
- Platforms or tools that help users create and share content with Web logs and online directories. Mashup platforms let users retrieve content or functionality from arbitrary sources, mix it with other resources, and expose it for further reuse by other applications.
- Online collaboration tools support users in collaboratively performing certain tasks, such as maintaining time schedules or processing text online.

### 2.2 SOA Applications

In comparison, the following ways can identify the SOA applications:

- SOA allow for a cross-organizational integration of services.
- SOAs facilitate the intra-organizational integration of disparate services.
- Heterogeneous applications can be encapsulated and composed to a seamlessly integrated IT environment [8].
- SOA-based applications development significantly reduces development time.
- SOA helps better standardizations between applications, thus strengthening security of the system.

### 2.3 The Similarities and differences between SOA and Web 2.0 [2]

Major analogies between SOA and Web 2.0 are in the following grounds:

1. Notion of reusing and composing existing resources.
2. Affinity to collaboration and coupling of remote resources of services.
3. Shared principle of agility and the support of permanent structural changes.
4. Both Web 2.0 and SOA support loose coupling of remote applications via uniform interfaces.
5. They both hide complexities and reduce programming efforts.

### 2.4 The differences between SOA and Web 2.0

1. Most of the Web 2.0 applications are to facilitate human interactions and human-readable contents; whereas SOA interconnects dispersed business functionality and facilities.
2. Web 2.0 are clearly about presentation and user interface, whereas SOA deployments are more abstract and less visible.
3. SOA needs to abide more by rules and regulations of the corporate and government level which does not apply for the Web 2.0.
4. Web 2.0 uses many techniques that focus on static syndication of contents and services, whereas SOA incorporates service coordination protocols.
5. Web 2.0 involves error-prone elements (humans) or have semantic differences; while SOA is typically limited to the machines which do not have formal errors.

## 3. Importance Of Soa

Many people see SOA as a product - often missing the point that SOA is not a product. Rather SOA can work as a bridge between business and IT through using a set of business aligned IT services [9].

The basic concept behind SOA is to provide a basic set of services that each application can access to provide this common functionality. SOA provides the baseline or enterprise services. The idea is to write the code once and then use it everywhere. Conceptually, the result is less code, lower cost, and increased standardization. The idea of less code does not instantly appeal. From an enterprise level, the benefits are obvious. The programmer needs only to write code to implement the unique functionality for a given project. If less code is needed for a project, the result is that fewer programming hours must be absorbed. The total Software Quality Assurance (SQA) costs should be decreased since there is less code to document. Fewer lines of code mean less documentation and less debugging [10], [11].

When new applications start using the SOA to implement base functionality, a higher degree of consistency will exist between applications [12]. Less coding raises assurance for safety-related systems and quality management systems (ISO 9001). Finally, standardization helps businesses to simplify their enterprise architecture and forces more up-front planning [13]. Lack of standardization is perhaps the largest cyber-security vulnerability in most organizations [14]. The multitude of software packages, operating systems, configuration management nightmare.

## 4. Convergence Of Web 2.0 And Soa Concepts

The philosophies of Web 2.0 and SOA are different. Their differences are found in the design and used technologies from the real-world applications. However, very recently few use cases showed opportunities for merging both the SOA and Web 2.0 in technical way and also in principle. One major example of convergence is referred to as the global SOA, which often refer to as *Internet of Services* (IoS) [2]. The basic idea is that a combination of principles from both user self-service and collective end-user intelligence (Web 2.0) and a composition of reusable building blocks (SOA) can facilitate the wide dissemination of many resources.

Until now, Internet users are unable to easily retrieve and use certain services. Those services mostly reside within company boundaries and are not readily accessible for professional use. The current web interfaces are designed to be processed by machines and the web-based

services are less used by the web surfers.

The basic of *Internet of Services* arises from the combination of common principle and technologies of the SOA and the Web 2.0. Resources that are accessible via the Web are registered in platforms and can thus be discovered, tagged and also *mashed up* [= *composed* and *interlinked* with the goal of new resources design].

These platforms create enterprise mashups to provide businesses with speed, flexibility, and agility in creating and changing cross-enterprise applications. The result of this approach that combines the Web 2.0 and the SOA technologies resulted in development of professional business applications, value-added services including location-based services, and interoperability services.

Using this platform, arbitrary stakeholders can provide and host services. The actual users can access these platforms for discovery purposes through intuitive interfaces without requiring any coding effort. Thus different channels can enter and use the platform [shown in the figure 2]. If the architecture of the platform is open, it shall facilitate human interactions through Web-based resources. There could be a quick development of global mesh of services.

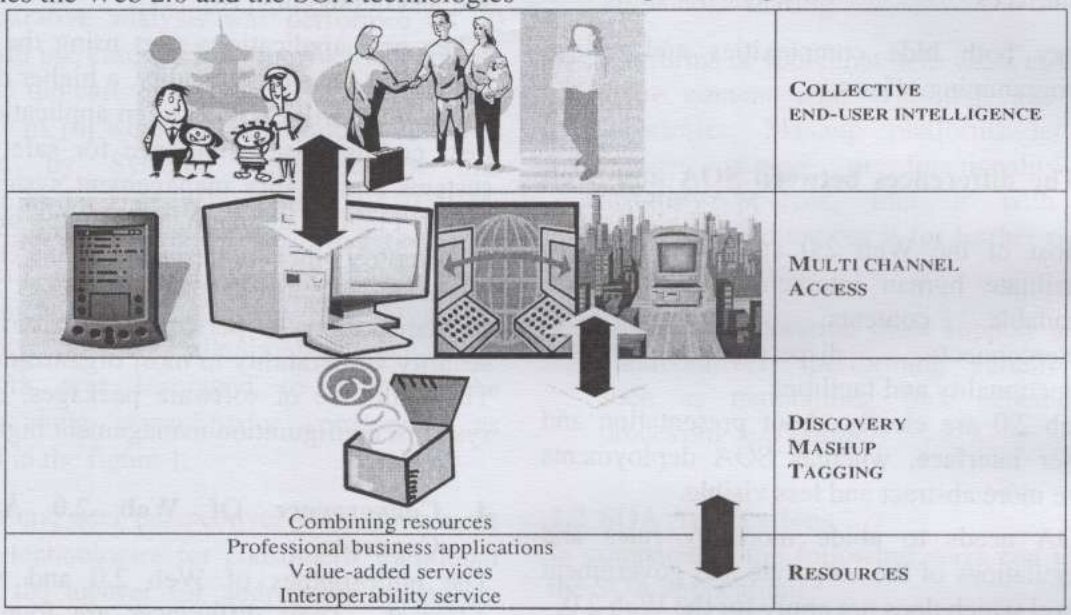


**Figure 2 :** Capabilities of SOA [2]

Technically unsophisticated business entrepreneurs would bring out business models in quickly, and efficiently. This will enable enterprises to communicate with their business partners and customers outside their web-domain [8]. In the late 2006, a firm Kapow Technologies [16] announced it break through for enterprise mashups. According to their findings, it needs to empower users of their Web integration platform to integrate resources available via Web on several different levels.

This empowering helps in two ways: Firstly, it seamlessly integrate and expose use-interface content and functionality. Secondly, it establishes application mashups. These applications can be accessed through Representational State Transfer (REST) or Web Service Description Language (WSDL) interfaces into new services.

The United Nations Centre for Trade Facilitation and Electronic Business (UN/CEFACT) provides another excellent example of the combination of Web 2.0 and SOA [2]. It proposes a novel approach for the standardization of business processes. Instead of prescribing yet another fixed standard for the establishment of cross-organizational SOAs, the UN/CEFACT envisions establishing a publicly accessible.

## 5. Soa Logical Layers

In his first ever proposition over SOA's

architecture, [15] proposed a seven layer composition for the SOA architecture. According to this study, SOA was shown as a partially layered architecture that composes of services aligned with business process. It could be shown in the following figure:

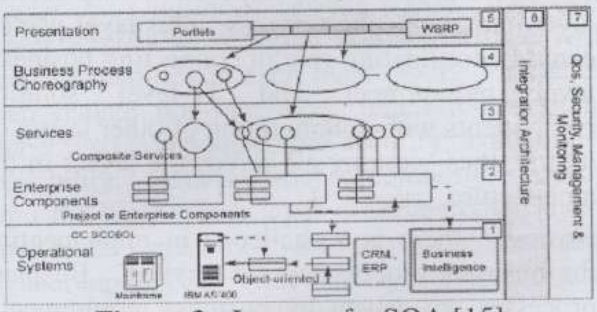


**Figure 3 :** Layers of a SOA [15]

The services and components in an enterprise are related as such that the large-grained enterprise or business line components realize the services and are responsible for providing their functionality and maintaining their quality of service. Business process flow supports exposed services composition into composite applications. An integrated architecture supports the routing, mediation, and translation of these services, components, and flows using an Enterprise Service Bus (ESB). The deployed services must be monitored and managed for quality of service and adherence to non-functional requirements.

### 5.1 Seven Layers of SOA

As mentioned above, [15] mentioned about seven layers. Those layers are discussed below with brief elaborations.

*Layer 1: Operational System.* This layer is the basic layer, consisting of legacy systems, meaning that the programs which have been developed with an outdated technology make-up for the vast majority of programs in many user application environments [8]. This includes existing customer relationship management (CRM) and enterprise resource planning (ERP) applications, *older* object-oriented applications, and business intelligence applications. Composite layered architecture of SOA can leverage existing systems and integrate those using service-oriented integration techniques.

*Layer 2: Enterprise components layer.* This layer is responsible for realizing functionality and maintaining the QoS of the exposed services. This layer typically uses application servers to implement the component, workload management, high availability and load balancing.

*Layer 3: Services layer.* The services need for business applications are found in this layer. They are composed as dynamic or static formats, or a composition of both. Therefore the enterprise components can provide services at runtime using the functionality provided by their interfaces. The interfaces get exported out as service descriptions in this layer.

*Layer 4: Business process composition layer.* The services offered in the layer 3 are defined in this layer. Services are bundled into a flow to act together as a single application. These applications support specific use cases and business process. Examples are, IBM® WebSphare ® Business Integration Model, used for designing application flow.

*Layer 5: Access or presentation layer.* [18] explained this layer, with a mention that this layer is not much relevant to the architecture of SOA. However in [12/19], no such layer is mentioned directly of indirectly. Still gradual convergence of standards that seeks services to leverage Web services at the application interface or presentation level is getting more attention very lately.

*Layer 6: Integration (Enterprise Service Bus - ESB).* This layer provides a location independent mechanism for integration. The layer integrates services through introducing a reliable set of capabilities like intelligence routing, protocol mediation and other transformation mechanisms. One example is the Web Services Description Language (WSDL), which specifies the location where the service is provided.

*Layer 7: Quality of Services (QoS).* This is mainly a background process. This layer mainly provides the capabilities for monitoring and maintaining QoS for security, performance and availability. It uses sense-and-response mechanisms, and monitors SOA during their runtime and the important standards implementation of WS-Management.

### 5.2 Nine Layers of SOA

After a gap of about four year, in [19], the authors came up with nine layers proposal for the SOA. The authors assumed there are several functional and non-functional service requirements those collectively establish SOA objectives. Another assumption was that every layer, or combination of few layers, can perform one service which in turn influences that layer. The critical part of SOA development was to identify the service requirements and map them to appropriate layer. These layers are briefly discussed below:

*Layer 1: Operational systems.* This layer includes all application assets running in an IT operating environment that supports business activities, whether custom, semi-custom, or off the shelf; the layer consists of existing application software systems. This in turn can lower the overall implementation cost and free up some of the overall budget for newer initiatives and the development of business critical services. Few software are commonly found in the operational system layer: Java 2 Enterprise Edition, .NET application, existing database, *older* applications and systems, existing customer relationship management (CRM) and enterprise resource planning (ERP) and so on.

*Layer 2: Service component.* This layer contains software components to realize an operation on a service. Service components reflect both the functionality and QoS for each service they represent. The services are found in many ways. Any replacement does not affect any of the customers since required modifications are encapsulated in the service component. Given SOA's recursive nature, service components can also be service consumers.

*Layer 3: Services.* This layer consists of all services defined in the SOA. Service specifications are strictly defined and should include (*i*) operation signatures, (*ii*) service endpoint information, (*iii*) invocation protocol details, (*iv*) service semantics, (*v*) service dependency information, (*vi*) SOA management description, (*vii*) attachments showing service dependencies and (*viii*) classification information.

Some services of this layer may be a newer or older version of other services in the set. These relationships help to create a new composite service. Moreover, service descriptions are independent of implementation and transport, thus an enterprise can expose a service consistently across multiple customer-facing channels.

*Layer 4: Business process.* The organization assembles the services exposed in the service layer into composite services. This layer plays a central coordinating role in connecting business-level requirement and IT-level solution components with collaboration of other layers on its top, thus working all across the layers in the architecture.

However, there is a challenge in implementing the business process layer's services. Business processes are driven by business requirements, which typically tend to be informal, subjective, and difficult to quantify. Therefore, translating descriptive and subjective requirements into quantifiable, objective, and machine-readable formats is challenging.

Another challenge is in service collaboration. A typical business process generally requires the collaboration of multiple Web services. This strongly implies the need to optimize the entire business process before its execution.

*Layer 5: Consumer.* This layer deals interactions with the user or with other programs in the SOA systems. With the help of this layer, an organization can quickly create the front-end business process with composite applications to respond to changes in the marketplace through rich client-user interfaces. Technologies allowing Web services to be invoked from the customer layer decrease development and implementation cycle times and enables the reuse of pre-built access mechanisms, resulting in reduced complexity and maintenance works. This can be rendered through Web Service for Remote Portlets (WSRP).

Another technology is gaining popularity in the consumer layer is Asynchronous JavaScript and XML (AJAX) [19]. AJAX enables the exchange of XML contents over HTTP without the need for Web browser refreshing and provides a richer, more responsive user interface. Both WSRP and AJAX are two examples of SOA solutions that decouple user interface from the customer layer.

*Layer 6: Integration.* This layer primarily integrates layers 2 through 4 and let an organization to mediate, route and transport

service requests from the service request to the correct service provider. The components of enterprise service bus, included but not limited, could be (*i*) message transformation with point-to-point tightly coupled endpoint integration, (*ii*) intelligent routing and (*iii*) protocol transformation.

*Layer 7: Quality of Service (QoS).* The QoS layer lets an SOA signal non-compliance with service qualities in each SOA layer. This layer can realize non-functional requirements [non-functional components are security, availability, reliability, manageability, scalability, latency and the like]. This layer observes other layers and sends signals or events to rectify or prevent non-compliances. Thus this layer helps SOA to meet its non-functional requirements.

*Layer 8: Information architecture.* This layer captures the reference points for all the cross-industry and industry specific data structure, XML-based metadata architectures, and protocols of exchanging business data. It also covers the enabling frameworks of discovery, data mining, and analytic data modeling.

*Layer 9: Governance and policies.* This layer includes all policies from manual governance to WS-Policy. It provides guidance and policies for managing service-level agreements that includes capacity, performance, security and monitoring. This layer is tightly connected with the QoS layer. In fact, this layer is vital for accelerating SOA solutions planning and designing and to increase sales opportunities for managing customer's existing solutions.

### 5.3 The Abstract Model

In [11], an abstract architecture for SOA was described: *Reference Architecture Foundation for Service Oriented Architecture*. According to this, SOA was defined as 'a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. It provides a uniform means to offer, discover, interact with and use capabilities to produce desired effects consistent with measurable preconditions and expectations.'

This architecture did not explicitly describe the layers; rather put emphasis on the components of the SOA. There are four major principles for this reference architecture: (*i*) it is technologically neutral, (*ii*) it has economy of design, avoids complexity, uses minimum number of components and relationships, also known as parsimony, (*iii*) separates concerns and (*iv*) possesses applicability. SOA was viewed as a Service Ecosystem that focuses on people's participate in it to conduct their business.

The reference architecture is composed of many models. These models are social structure model, acting in an ecosystem model, service description model, service visibility model, interacting with services model, policies and contract model, governance model, security model, management model, and testing model. For realization of the SOA ecosystem, four more models were proposed: service description model, service visibility model, interacting with services model, and policies and contract model.

## 6. The Proposed Model

From our discussion above, it was clear that the SOA works as a basic building block of the Web 2.0 services. Even when tried to merge these two services, it did not bring the expected output. Therefore it is wise to devote more in developing service-oriented architecture for the Web 2.0.

Also in the above section, we also discussed about the layers of SOA in two different contexts and one abstract model. Though they were developed for a better service-oriented architecture, in some part they looked little clumsy. Some of those layers and abstract models mentioned already may not be in use even if someone wants to follow that particular architecture or have got almost similar working principles. However, we considered the assumptions of [19] for a newer construction for SOA layers.

Many layers make one architecture very complex, and less number of layers may not construct a better architecture. In this perspective, a mixture of [15] and [16] could be considered as a new implementation of the SOA. Since the SOA provides reusability of resources, the layers also should be defined to help implementing the reuse.

In [16], business process and governance policies cut across all the layers. Unlike [15], the authors did not put *Business process and composition layer* as a separate layer in [16]. Since *Business process* and *governance and policies* layers intersect in defining the rules and

policies for the business process, and the input and output transformations from and to the consumer layer must abide by some business rules. Also abstract layers like policies and contract model, management model and governance model are covered under this heading. Thus *business and government policies* could be put as a single layer encapsulating all other layers.

**Table 1** Comparing different layers

| Seven layers | Nine layers | Abstract model | Proposed Model |
|---|---|---|---|
| Operational systems | Operational systems | Social structure model | Operational systems |
| Enterprise component layer | Service components | Acting in an ecosystem model | Services and components |
| Services | Services | Service description model | Consumers |
| Business process composition | Business process | Service visibility model | Integration |
| Access or presentation | Consumers | Interacting with services model | Information architecture |
| Integration | Integration | Policies and contract model | Quality of Service |
| QoS | QoS | Governance model | Business and government policies |
| | Information Architecture | Security model | |
| | Governance and Policies | Management model | |
| | | Testing model. | |

*Enterprise components* layer, according to [15], and the layers of *service components* and *services*, according to [16], could be merged in a single layer just above the *operational layer*. Since various components are considered to render SOA services, the new layer could define both services and manage components that would be required to provide those services. Thus it covers the abstractions of few models of [18], including security and realization methods. The new layer could be named as *services and components*.

Since *consumer*, *integration* and *QoS* are important for providing a better SOA model, those layers are needed in the architecture, along with *operational systems* and *information architecture* layers. Thus the stack for our proposed architecture would look like what is shown in the table 2.

**Table 2** The proposed layers of SOA

| Layer numbers | Name of the layers |
|---|---|
| 1 | Operational systems |
| 2 | Services and components |
| 3 | Consumers |
| 4 | Integration |
| 5 | Information architecture |
| 6 | Quality of Service |
| 7 | Business and government policies |

## 7. Conclusions

Any architecture must have an unambiguous and transparent structure. The OSI model is one such structured and transparent model. In this paper, we have seen many different layers of SOA found in different models. The discussion in the section 5 reveals that there are many layers in the SOA architecture. As Web 2.0 will use SOA as its basic architecture, a generalized model is important for proper functioning of the Web 2.0.

Therefore, proposing OSI-model type architecture still remains a very challenging field for the researchers for a proper development of the Web 2.0.

The proposed layers of SOA in the section 6 meet the goal of generalization and transparency of the SOA architecture. Since SOA would not converge into Web 2.0, rather SOA will perform as the basic architecture of the Web 2.0, the proposed seven layered architecture, mentioned in the table 2, will remain as very good reference architecture for the Web 2.0.

## References


1. Andrew McAfee, "The Impact of Information Technology (IT) on Businesses and their Leaders", *HBS Faculty Blog*, http://archive.is/HWRUt, published May 20, 2006
2. Christoph Schroth and Till Janner, "Web 2.0 and SOA: Converging Concepts Enabling the Internet of Services", *IT PRO*, May-June 2007, pp 36-41.
3. http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.pdf
4. McAfee, "Will Web Services Really Transform Collaboration", *MIT Sloan Management Review*, vol. 46, no. 2, 2005, pp. 78-84
5. Mark Colan, "Service-Oriented Architecture expands the vision of Web services, Part 1", IBM, ibm.com/developerworks/webservices/library/wssoaintro.html
6. R. Högg et al., "Overview of Business Models for Web 2.0 Communities", *Proc. Gemeinschaften in Neuen Medien*,Technische Universität Dresden, 2006, pp. 23-37
7. Harry M. Sneed, "Integrating legacy Software into a Service oriented Architecture", *Proceedings of the Conference on Software Maintenance and Reengineering*, 2006 (CSMR'06)
8. Mulholland et al., "Mashup Corporations: The End of Business as Usual", *Evolved Technologist Press*, 2006
9. http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html
10. Liang-Jie Zhang, Sally Ericksen, and Jaideep Roy, "A Web 2.0 Tune-up"; *IT PRO*; May-June 2007; pp 9
11. R W Schulte, "Predicts 2003: Enterprise Service Buses Emerges", *Gartner Research Note*, 2002; (http://www.gartner.com/DisplayDocument?doc_cd=111977).
12. Kapow Technologies, www.kapowtech.com.
13. T. Janner et al., "From EDI to UN/CEFACT: An Evolutionary Path Towards a Next Generation e-Business Framework", *Proceedings 5th International Coneference on e-Business*, 2006, King Mongkut's Univ. of Technology
14. Ali Arsanjani, "Service-oriented modelling and architecture", *IBM*, 2004.
15. Ali Arsanjani, Liang-Jie Zhang, Michael Ellis, Abdul Allam, and Kishore Channabasavaiah, "S3: A Service-Oriented Reference Architecture", *IT PRO*, May-June 2007, pp 10-17.
16. http://en.wikipedia.org/wiki/ajax, visited October 2013
17. "Reference Architecture Foundation for Service Oriented Architecture Version 1.0" (Committee Draft 02), http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/soa-ra-cd-02.pdf, visited October 2013
18. Brian Travis, "Section 3: Developing Service-Oriented Architectures", http://msdn.microsoft.com/en-us/library/aa302164.aspx, visited October 2013
19. Sudhir Batra, "AJAX - Asynchronous Java Script and XML", web: www.sbatra.at/seminararbeit_sbatra2.pdf, visited October 2013.