Brave New Web:
Emerging Design Principles and Technologies as Enablers of a Global SOA

Christoph Schroth¹,², Oliver Christ²

¹ University of St. Gallen, =mcm Institute, St.Gallen, Switzerland
² SAP Research CEC, St.Gallen, Switzerland
{christoph.schroth, oliver.christ} @ {sap.com}

Abstract

Web Services have experienced great interest during the last years as they were expected to play a key role as enablers of seamless application-to-application integration both within company boundaries and on a global, cross-organizational scale. As a technical foundation for the realization of Service-Oriented Architectures (SOAs), Web Services encapsulate complexity inherent to individual applications and allow for their loose coupling. However, a truly global mesh of such services has not yet become reality due to various reasons. Novel technologies and design principles are currently about to emerge which allow human users to use, customize, combine, interconnect and finally expose Web-based content or functionality as new resources which are often referred to as Mash-ups. In this article, we provide an overview of existing Mash-ups as well as tools and platforms that empower users to build them in a highly efficient and intuitive fashion. Statistical data and case studies are leveraged to examine new ways of resource provision and consumption and also the relevance of upcoming intermediaries. Finally, we investigate remaining research challenges on the path to a truly global SOA.

1. Introduction

The number of enterprises that automate cross-organizational business transactions is constantly growing. Renowned scientists such as Malone cite the relentless march of improvements in the cost-performance ratio of information technology as main driver of this development [1]. Benefits of performing transactions electronically include extending market reach, saving time, cutting costs and responding to customer queries more agilely [2].

Web Services [3] have experienced great interest as they were expected to play a key role as enablers of this seamless application-to-application integration both within company boundaries and on a global, cross-organizational scale. They allow for encapsulating proprietary application functionality with the help of uniformly defined interfaces and thus facilitate the loose and quick coupling of potentially heterogeneous applications.

As argued in [4], Service-Oriented Architectures (SOAs) [5] on the basis of Web Services mostly only exist within company boundaries and have not yet taken over a significant role in realizing cross-organizational collaboration. Various reasons such as high technical complexity, implementation and maintenance costs, inflexibility and the lack of widely accepted standards for defining service choreographies as well as message semantics have prevented the emergence of a global mesh of interoperable Web services.

Novel technologies and design principles are now about to experience increasing acceptance as they allow for using, customizing, interconnecting and finally exposing Web-based content or functionality again as new resources. Web 2.0 is frequently used as an umbrella term to summarize the collection of programming languages and general principles that can be considered not as substitute, but as enrichment of SOA concepts and technologies.

Numerous various definitions of Web 2.0 exist that mostly only describe certain aspects of the overall concept. Tim O’Reilly, who originally coined the term, identifies seven major characteristics inherent to the Web 2.0 concept [6]. First, the Web is considered as a platform for building systems that do not necessarily have a specific owner and are “tied together by a set of protocols, open standards and agreements for cooperation”. Harnessing collective intelligence of Web users represents the second major paradigm. The ownership of mission-critical data is regarded a further cornerstone of numerous Web 2.0 applications. Fourth, O’Reilly propagates the end of the software release cycle as another central paradigm. The use of lightweight programming models that allow for loosely coupled systems and applications, the provision of software above the level of a single device and the realization of rich user experience represent the last major paradigms inherent to the concept of Web 2.0. Besides such analyses that properly describe parts of the super-ordinate concept, there are only very few comprehensive scientific definitions available. An in-depth investigation of numerous different, successful Web
2.0 applications performed by Hoegg et al. [7] condensed the respective characteristics in the following statement which works as underlying definition for this work: “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”.

Besides social communities (e.g., LinkedIn.com, GiveMeaning.com) or online collaboration tools, **Mash-ups** are considered as central to the Web 2.0 era. The Economist defines Mash-ups as follows [8]: “The term Mash-up is borrowed from the world of music, where it refers to the unauthorised combination of the vocal from one song with the musical backing of another, usually from a completely different genre. Web Mash-ups do the same sort of thing, combining websites to produce useful hybrid sites and illustrating the internet’s underlying philosophy: that open standards allow and promote unexpected forms of innovation.” The BusinessWeek [9] explains this novel form of using Web-based resources similarly: “Mash-ups portend big changes for software companies, Web sites, and everyone online. No longer just a collection of pages, the Web is morphing into a sort of global operating system, à la Microsoft […] Windows. And now, people are learning to program Web 2.0 with much of the same innovative energy of the personal computer’s early days.”

We basically adhere to these explanations and take the following summary as foundation for this work: A Mash-up is a Web-based resource, be it content or application functionality, which has been created through reuse and composition of two or more different resources.

Mash-ups can be setup by using resources such as the OpenStreetMap¹ service, which provides free geographical data such as street maps. By leveraging lightweight protocols and a simple interface, it can be seamlessly integrated into other Web-based applications and thus be enriched with additional, arbitrary features such as pictures of local buildings or tourist attractions. Platforms and tools are provided in the Web that empower users to loosely couple readily available applications, to enrich and compose them into a novel service, which can again be made publicly available. As a result, meshes of different Web-based resources emerge that can be considered rather ungoverned since there is no central control entity existing that imposes formal guidelines for reuse and syndication. The possibility to actively contribute and gain reputation by publishing resources on the one side and to harness the aggregated knowledge and intelligence of all other platform participants on the other side represent core benefits for users.

Recently, the relationship between Mash-ups and traditional SOAs has experienced considerable interest [10, 11]. In this article, we investigate this interrelation and propose Mash-ups as key enablers of a truly global SOA that is freely developed by human users who are fascinated and empowered by a set of Web 2.0 technologies and design principles.

In the following sections, we first of all provide an exact definition of SOA and also elaborate on the shortcoming of conventional, Web Services-based implementations with regard to the emergence of a global mesh of interoperable services (Section two). We then use the key three elements of a SOA, namely the service provider, the service consumer and a registry [5] to systematically analyze success factors of the above mentioned Mash-ups on the basis of statistical data and case studies (Section three). As argued below, new ways of resource provision and consumption as well as certain intermediaries are characteristic for the emergence of Mash-ups which can be considered new and human-driven forms of SOAs.

### 2. Vision and shortcomings of the traditional Web Service-stack to enable a global SOA

#### 2.1. SOA and the vision of loosely coupling heterogeneous applications worldwide

Since the late 1990s, many definitions of SOA have been published [3, 12]. The normative OASIS Reference Model for SOA [5] defines SOA 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”.

![Figure 1: Basic SOA Architecture](http://example.com/sOA.png)

According to this model, the major components of a basic SOA and their possible interactions are (see Figure 1): a service provider publishes his service interface via a service registry where a service requester/ consumer can find it and subsequently may bind to the service provider. The central concept of the SOA Reference Model is the existence of services which provide access to capabilities

---

by well-defined interfaces to be exercised following a service contract with constraints and policies. This enables a loose coupling of services (thereby minimizing mutual dependencies) and complies with some of the probably most-known principles in software-engineering, information-hiding and modularization [13]. Services are provided by entities, the service provider, and are to be used by others, the service consumers. Services may be composed on the basis of other, existing services, thereby adhering to the principle of reuse. They are autonomous (solely control the logic they encapsulate), uniformly described and publicly retrievable via certain discovery mechanisms.

The concept of supporting loosely coupled, business-aligned and networked services as introduced above can be realized with the help of numerous different technologies. As WSDL\(^2\) and SOAP-based Web Services are the most widely spread standards used to setup SOAs, these are in the focus of this work [3]. The Web Service Description Language (WSDL) defines a uniform, machine-readable XML format for service interfaces. The Universal Description, Discovery, and Integration (UDDI\(^3\)) standard specifies publicly available service registries (see Figure 1) that are needed for service search, identification and invocation. SOAP specifies the data format and an exchange protocol for the messages to be sent between service providers and requesters. While SOAP only defines formal aspects of data structure, the interpretation of information semantics has not yet been defined by a common standard. Machines may describe the same piece of information with different terms, thereby preventing from seamless system interoperability. The Business Process Execution Language (BPEL\(^4\)) is widely accepted as a standard for imposing a choreography on different services. Also based on XML, it focuses on the description of a business process from one participant’s point of view.

Many have generated great enthusiasm about Web Services, which are considered as “open, Internet-era standards for exchanging data between applications” [4]. With a SOA, tight couplings between the applications of different companies are expected to be replaced by loose couplings. As a consequence of common standards for data description and connection protocols, applications can be seamlessly connected to others without costly reprogramming efforts. By loosely coupling application functionality via commonly defined interfaces, companies are enabled to modify their operations more agilely in response to changing business requirements. Summing up, especially companies expected the dawn of Web Services to turn traditional supply chains inside out and to replace static cross-organizational business relations by a flexible mesh of loosely coupled services which can be identified via publicly accessible registries.

2.1. Shortcomings of the conventional Web Services stack with regard to realizing a global SOA

Right now, Web services have not lived up to this promise as mostly people are still involved in loose couplings all over the world. Rather than automatically inter-connecting applications across company boundaries with the help of Web Services technologies, most loose couplings are based on “e-mail and instant-messaging applications, groupware such as Lotus Notes/ Domino and nascent “social software” such as wikis, Friendster.com and Linkedin.com” [4]. As argued in [14], the “corporate household” problem prevents Web Services-based SOAs from being setup quickly and easily. Companies and their respective systems are subject to rigid governance mechanisms [15] and adhere to different conventions with respect to business documents and also internal business processes: “The applications used in different departments or different locations of a single company, for example, are virtually guaranteed to have inconsistency in customer lists, part-numbering schemes, approved suppliers, addresses for these suppliers and so on” [4].

Besides the heterogeneity of conventions with respect to data structures, semantics and processes, the lack of comprehensive, trustworthy and widely accepted service registries prevents global SOAs from quickly establishing. The few globally available UDDI-based registries are mostly only usable for technical experts, do not offer any additional functionality such as service performance monitoring and only feature a small number of references to different Web Services.

Apart from that, the traditional Web Services stack aims at supporting the setup of loosely coupled application interconnections especially in a professional context and assumes users to be technically sophisticated. This prevents normal Internet users from easily discovering and interacting with services available via the Web. WSDL-based service interfaces, but also the UDDI-compliant service registries are not designed to be interpreted and used by humans but rather by machines.

2. Web 2.0 design principles and technologies as enablers of a global SOA

\(^2\) http://www.w3.org/tr
\(^3\) http://www.oasis-open.org/
\(^4\) http://www-128.ibm.com/developerworks/
After elaborating on the actual vision and the promise of traditional Web Services to allow for the efficient setup of loosely coupled application interconnections, we present emerging design principles and technologies which enable a global Service-Oriented Architecture which is driven by individual Web users and their requirements. By introducing new ways for resource provision and consumption, but also as a consequence of the emergence of new intermediaries, the major obstacles which traditional Web Services struggled with can be overcome. In this way, Mash-ups can be freely designed by individuals on the basis of lightweight programming models and interfaces. The following sections will thoroughly investigate the prerequisites for the success of these Mash-ups and thereby leverage statistical data as well as case studies.

3.1. Resource provision

As mentioned above, traditional Web Services are provided as functionality described by uniform interfaces (mostly WSDL compliant) that define input and output messages as well as the supported functions of the services. These interfaces are not human-readable and do not facilitate the interaction of users with the underlying services. Resources in the Mash-up context do not aim at technical experts in the corporate context any more, but at the huge number of individuals, the “long tail” [16] of Internet users who require intuitive visual means for retrieving resources in the Web and for capturing their respective functionality.

The Website ProgrammableWeb.com acts as a major aggregator of numerous (currently more than 1600) references to diverse existing Mash-ups all over the world. It provides statistics about used resources, a classification of Mash-ups into categories as well as user statistics to evaluate their popularity. Therefore it serves as one central source of empirical data used in this work.

To investigate the most frequently used resources and their respective Application Programming Interfaces (APIs), the Web Site exposes an “API-Scorecard” (Figure 2) which has six major API-providers on the X-axis and the most relevant categories of APIs offered on the Y-axis. In this context, the term API refers to the interface type, data formats and the protocol to be used for interconnecting to an application. The last line of the figure lists the number of APIs referenced on the Website and the number of Mash-ups using one of these APIs for each of the 6 major providers. For example, 20 Google-based APIs and 1046 Mash-ups that leverage one of these APIs are listed on ProgrammableWeb.Com.

Several facts can be inferred from this graphic: First, large companies such as Google, Amazon and Microsoft start to successfully provide Web-based resources which are leveraged by numerous users for creating Mash-ups on their basis. As can be seen in Figure 2, at least 836 Web-based applications have already integrated the “Google Maps”- resource that offers geographical data. The Website “Vuelta a California” can be cited as an excellent example for a Google Maps-based Mash-up. It uses information from a Web log that deals with road profiles for bikers in California and mashes it with road map data from Google Maps to achieve a rich visual representation.

The second important fact that can be inferred from the API scorecard is the shift away from the professional corporate context towards a private, end-user driven field.

---

5 http://www.berndstraehle.com/map.html
of applications. As opposed to the focus of traditional Web Services, the resources used for building Mash-Ups aim at the long tail of Internet users and deal with media management, shopping functionality, entertainment or desktop applications. Heavy-weight enterprise applications for automating business transactions can only rarely be found in this context.

Third, the Website also provides information about the protocols and interface standards used by the different listed resources. The analysis of the provided data leads to the following results: Besides the SOAP protocol which is also used in the context of traditional Web Services, particularly “light-weight” protocols such as REST [17] and RSS are widely leveraged to allow for fast and seamless mashing of different resources.

The Representational State Transfer (REST) is an architectural style that has been proposed by Fielding in his doctoral dissertation. With the help of REST, Web clients can interact with arbitrary Web resources (sources of specific information) in a uniform way and thereby exchange XML-based messages. Resources are uniquely addressable by means of a global identifier (URI) and support a well defined and also constrained set of supported operations (so-called HTTP verbs such as GET, PUT). By using resource identifiers and HTTP verbs together, clients can invoke REST-based services and request the delivery of, for example, an XML file containing the desired specific information. The Simple Storage Service (S3) provided by Amazon is one of many exemplary applications of REST technology: On the basis of a simple service interface, users are enabled to store any amount of data on Amazon servers and also retrieve it from arbitrary locations on the Web. REST incorporates the principles of complexity-hiding and uniformity since it significantly reduces programming efforts and forces both clients and recourses to adhere to a common set of supported operations and interface descriptions.

The Really Simple Syndication (RSS) format represents a further technology that is affiliated with numerous resources in the Web 2.0 context. An XML based file format, RSS supports the easy aggregation of content from arbitrary sources in the Web. With the help of this standard, the information offered by frequently changing Web sites such as Web logs can be exported into XML files of a uniform structure and is thus made publicly available. The files’ structures are determined by the RSS specification which intentionally only allows for a very limited set of content categories such as title, Web link and textual description of a piece of information. Web applications that aim at distributing content via RSS technology must ensure that the respective data files always contain the latest information. Applications that present the information obtained from different RSS-based sources in a combined manner are referred to as content aggregators. Users are thus enabled to gather information that has been created by a number of authors and is easily accessible via uniform interfaces, redesign it and also publish it.

Summing up, the Web increasingly can be considered as a comprehensive and global development platform containing numerous easily usable and mashable resources that are provided by large firms as well as by individual end-users. As argued above, the provision of resources that use lightweight protocols such as REST or RSS and also the focus on end-user requirements rather than Business-to-Business relationships represent core success factors for this new global SOA consisting of numerous Mash-ups. Figure 3 finally shows the currently most popular providers of resources leveraged for the creation of Mash-ups.

![Figure 3: List of most popular Mash-up resource providers (according to ProgrammableWeb.com)](http://www.amazon.com/gp/browse.html?node=16427261)

**3.2. Emerging intermediaries**

As presented above, a publicly available registry constitutes one of the three major elements of a SOA. UDDI is widely accepted as standard and offers a platform-independent, XML-based registry specification for stakeholders worldwide to list the services they provide via the Web. The key information published by such a registry contains white pages (address, contact, and known identifiers of the service providers), yellow pages (industrial categorizations based on standard taxonomies) and finally green pages (technical information about services exposed by the business).

Similar to the technologies used for describing services in traditional SOA implementations (e.g. WSDL), the UDDI standard aims at technical experts and only specifies very limited functionality. Merely UDDI-based registries are not adequate to serve the large number of normal Internet user who require intuitive, human-readable interfaces to identify and to evaluate existing services and to finally integrate and mash them into their applications.

---

6 http://www.amazon.com/gp/browse.html?node=16427261
7 http://www.rssboard.org/rss-specification
Novel forms of intermediaries are currently about to emerge which offer resource registry functionality and extend the role of the traditional UDDI-based implementations. As depicted in Figure 4, they do not only contain references to services, but also offer statistics about resource performance or information about evaluations conducted by previous users.

Figure 4: Advanced intermediaries facilitate retrieval and use of Web-based resources

The typical resource discovery and utilization process is therefore different from the one depicted in Figure 1. A resource provider first of all publishes a reference to his resource at the intermediary which stores it and subsequently starts to monitor parameters such as general resource availability and response latency. Potential consumers may then access the intermediary’s Website and use the performance metrics and other evaluation results while searching for adequate resources. Some intermediary platforms even allow for building customized code for integrating selected resources into the users’ Mash-ups in order to relief them from any programming efforts. The intermediary WebRPC\(^8\), for example, offers this functionality and uses the advertising slogan “Find, customize, monitor, and call SOAP, REST, or HTML services in minutes”. After performing these four steps, the user may finally call the customized resource.

Besides such one-stop-shop resource registries, other kinds of intermediaries are about to emerge which all follow the goal to simplify user/service interaction and thus facilitate the development of a global SOA. The platform FeedBurner\(^9\) represents an excellent example for such intermediaries: It allows users to register RSS-compliant pieces of information (“feeds”) and provides users such as bloggers, podcast publishers and other Web-based content creators with several value-added services. After registering the URL of an RSS feed from an arbitrary source on the platform, FeedBurner offers free readership statistics, promotion services that make sure feed search engines are notified about updates as well as a human-readable and harmonized feed address as a substitute for the mostly heterogeneously designed original addresses. In this way, users benefit from both flexibility and persistence: In case the actual source of a user’s feed changes, he can edit the respective entry at the central Feedburner platform and thus ensures that the potentially high number of users who read or even mash the feed with other resources are not affected by this change. For them, the feed’s address remains the same. According to the information provided by the platform, as of August 2006, FeedBurner hosts feeds for 250,000 publishers, and delivers feeds to 19 million subscribers.

The emergence of such intermediaries that act as central resource publishing and management platforms can be motivated with help of institutional economics. According to North [18], the reason for institutions to exist is the uncertainties involved in human interaction; institutions act as constraints and framework to structure that interaction and make it more reliable. The proper establishment and application of informal constraints, formal rules and enforcement are adequate to lower transaction costs and boost economic performance. Both the WebRPC and the Feedburner platform are recognized as trustworthy central points of access to different kinds of Web-based resources. By monitoring performance and by gathering user-based evaluations, they lower the risk for new users to select a bad performing resource for their Mash-up. They also act in a normative fashion as they provide common standards for naming and format and thus reduce uncertainty of resources to be unavailable as a consequence of a URL change, for example.

3.3. Use of resources

After investigating new principles of offering resources and after presenting novel forms of intermediaries which provide institutional frameworks, we deal with the changes regarding user-side consumption of resources in this section. Before presenting three major cases of tools and platforms that empower users to mash up almost arbitrary kinds of web-based content or application functionality, we first present a Top Ten list of Mash-up categories (Figure 5) as of February 2007 and according to the data gathered by the ProgrammableWeb.com platform which references more than 1600 different Mash-up use-cases. As can be seen in this figure, end-user centric applications which are related to geographical maps, media management or online shopping prevail, while enterprise class Mash-ups do not yet occur very frequently.

Many of the Mash-ups cited and examined above have been created by individuals who are still technically experienced to some degree. Light-weight programming models and protocols as well as intermediaries which support the retrieval and evaluation of resources facilitate

\(^8\) http://www.webrpc.com/
\(^9\) http://www.feedburner.com/
the Mash-up setup procedure. However, some programming effort is still required in those cases. The following three major cases of Mash-up-platforms highlight a rather recent way of even empowering users without any technical sophistication to create and change Mash-ups and to thus foster the emergence of a user-driven, global SOA.

The **Google Website**\(^{10}\) represents a first intuitive example for a Mash-up platform as it allows users to aggregate and arrange different “gadgets” which represent either mere content (e.g., pictures, text, weather forecasts, Webcams) or certain application functionality (e.g., calculators). An intuitively usable gadget repository can be accessed for free and enables users to select and subsequently drag and drop some of the gadgets on their individual user interfaces. Finally, individuals are also empowered to create and register gadgets on their own by leveraging freely available templates and by complying with the Google de-facto standard. This kind of Mash-up-platform facilitates the mere aggregation of resources (which is also often referred to as syndication) but does not foresee any kind of interaction between them. Gadget interoperability issues are thus completely avoided since the resulting Mash-ups solely reside on the presentation layer. The gadgets offered by Spring Widgets\(^ {11}\) also belong to this category of presentation-layer syndication Mash-ups.

**Yahoo pipes**\(^ {12}\) represent an example for a second category of Mash-up platforms which are not limited to mere resource syndication. Figure 6 depicts an exemplary use-case for this platform: The “pipe” (which has been designed by a user) reads in an RSS-based feed provided by the New York Times homepage, passes it through a customizable content analysis (marked orange) and uses filtered keywords to find related photos at the widely known media repository Flickr\(^ {13}\). The pipe’s output issues links to the photos in a RSS-compliant way. Mash-up platforms like this are different from the ones assigned to the first category as they involve the processing of resources rather than their mere syndication. Content from the New York Times website is filtered and parts of another feed are selected according to the result of the filtering at the time the user triggers the Mash-up application. This advanced category assumes the resources to comply with certain standards to allow for their seamless connection.

![Figure 6: Yahoo Pipes facilitate mashing of content from arbitrary sources](http://www.kapowtech.com/)

Still, resources are only loaded and processed at one point of time and do not interact in a stateful fashion similar to Web Services that are called according to a previously defined choreography. Only very few examples exist that try to automate the mashing of application functionality rather than their simple aggregation. The firm **Kapow Technologies**\(^ {14}\) has recently published its vision of so-called Enterprise Mashups. It focuses on empowering users of their Web Integration Platform to integrate Web-based resources on several different levels and thus efficiently realize rapid portal creation, content migration, synchronization and syndication, composite application integration and SOA enablement. According to the firm’s publications, logic-based Mash-ups can be setup on the basis of a merely visual, technology- and complexity hiding modeling interface. Enterprise applications which are encapsulated as, for example, Web Services or by REST-based interfaces can mashed up to build new functionality.

### 3.3.2. Results

---

\(^{10}\) [http://www.google.com/](http://www.google.com/)

\(^{11}\) [http://www.springwidgets.com/](http://www.springwidgets.com/)

\(^{12}\) [http://pipes.yahoo.com/](http://pipes.yahoo.com/)

\(^{13}\) [http://flickr.com/](http://flickr.com/)

\(^{14}\) [http://www.kapowtech.com/](http://www.kapowtech.com/)
The analysis and categorization presented above shows that mainly three types of platforms have emerged which all aim at empowering individuals to mash up resources in an as quick and intuitive manner as possible. First, platforms such as Google facilitate the mere syndication of gadgets which are not interlinked at all. Second, environments such as Yahoo Pipes allow for processing and interconnecting resources that adhere to a certain standard (in this case RSS) and expose the result of this processing as a new resource. The mashing of resources on a application logic-level represents the third and last category investigated in the course of this work. Only very few examples exist which are capable to partly realize this sort of Mash-up (e.g., Kapow technologies).

Projects are under way and start-up firms are created that tackle this remaining challenge on the path to a global SOA that is driven by end-users and facilitated by lightweight programming models and intuitive Mash-up platforms and tools.

3. Conclusion

In this article, we elaborated on the Web Services stack as traditional enabler for coupling distant and potentially heterogeneous Web-based resources. As Web Services have not lived up to their promise to facilitate a global network of loosely interconnected services, new approaches are required to cope with this challenge. As a key component of the Web 2.0 era, novel technologies and design principles are now about to emerge which allow humans to use, customize, combine, interconnect and finally expose Web-based content or functionality as new resources on the Web. The results of reusing and composing existing resources are referred to as Mash-ups.

In this work, we provided an overview of already existing Mash-ups as well as different tools and platforms that empower users to build them in a highly efficient and intuitive fashion. Statistical data and case studies were leveraged to examine new ways of resource provision and consumption and also the relevance of upcoming intermediaries. Finally, we investigated remaining research challenges on the path to a truly global SOA: The development of platforms and tools for the automated mashing of application functionality rather than the mere syndication of resources will represent a key milestone which has to be achieved in future work.

Acknowledgments

The term “Brave New Web” has been brought up in the Web 2.0 context by Kenneth P. Morse, Managing Director, MIT Entrepreneurship Center

5. References

[8] Mashing the Web, The Economist 376 (8444), Special section, p. 4, 2005