GET READY FOR MASHABILITY!
Concepts for Web 2.0 Service Integration

Pavel Drášil, Tomáš Pitner
Masaryk University, Faculty of Informatics, Botanická 68a, 602 00 Brno, Czech Republic

Thorsten Hampel, Marc Steinbring
Department of Knowledge and Business Engineering, University of Vienna, Rathausstr. 19/9, A-1010 Vienna, Austria

Keywords: Web 2.0, Service Classification, Mashup, Business Models, License Agreements.

Abstract: Mashups represent – beside the ease-of-use, high interactivity and social networking factors – another significant phenomenon of Web 2.0. However, this "mashing process" is mostly based upon ad-hoc approaches and techniques, rather than upon an in-depth analysis of the "mashing potential" of the services. Therefore, the first goal of this paper is to provide a conceptual foundation for the mashups – to identify services being typically integrated into mashups, and propose classification criteria forming a "Mash-Tree" that were subsequently applied to selected services – representatives of each classification category. We concentrate mainly on mashing of Web 2.0 services; integration in an enterprise environment is behind the scope of this paper. Secondly, the mashing potential is studied from three perspectives – technical aspects of mashability, business models for mashups, and potential legal issues concerning services themselves as well as the data in them. We hope that the proposed mashup conceptualization together with our analysis of mashability can help to develop future Web 2.0 mashups that not only better meet stakeholders' expectations but also respect legal terms and are technologically sound.

1 INTRODUCTION

Although these days the buzzwords “web 2.0” and “mashup” have become an important topic of conversation – backed up by many successful applications, it is hard to find a clear and concise definition of the latter term. While the term “web 2.0” a.k.a “new net” *per se* has already been defined and discussed since the groundbreaking article by tim o’reilly was published in 2005 (o’reilly, 2005), the question of what a “mashup” service actually is remains. What are the underlying services being “mashed-up”? What are the main factors for a successful development of mashups? How do they differ from conventional solutions? And above all, do they keep their promise to be a flexible service by communicating via open and usable application interfaces? Based on a closer research of particular mashups and services, this article analyses key elements for a progressive deployment of services in the new web.

Web services are probably the most widespread application of the Service Oriented Architectures (SOA) concept (Alonso et al, 2004). On one hand they concentrate on building the service-oriented application architecture using a set of individual services. At the same time they take advantage of standardized protocols, formats and interfaces for communication and information exchange.

There are two substantial new developments which form an important basis for the idea of service oriented infrastructures, namely the ability of services to be self-described using semantic techniques, and a new culture of offering flexible service and customized service usage by mashups.

A *mashup service* or simply *mashup* can be defined as a Web Service or application built upon the integration of other services or data usually providing a certain added value, contextualization, or innovation above the integrated services (Merrill, 2006). Due to their variety, we will identify typical services being mashed-up and provide a classification of them, creating a “Mash-Tree” (see

Drášil P., Pitner T., Hampel T. and Steinbring M. (2008).
GET READY FOR MASHABILITY! - Concepts for Web 2.0 Service Integration.
In *Proceedings of the Tenth International Conference on Enterprise Information Systems - SAIC*, pages 160-167
DOI: 10.5220/0001709501600167

Figure 1) – a view of different components for the successful development of mashups. This classification is based upon our recent research and analysis of established mashups. Respecting the research notes *Service Oriented Architectures* from Gartner Group (Gartner, 1996), where SOA was mentioned first, this article takes both aspects – business and technology – into consideration to find concepts for mashups.

2 SERVICE CLASSIFICATION

Out of the market research we differentiate six major types of Web 2.0 services which can be divided by their general interest and focus of the service they offer. We have selected popular and representative services from each group and apply the following classification to them. A distinction is drawn between creating, sharing, storing and presenting content as well as between their coordination/communication and community orientation.

Content Creation

Typical services in this area are *wikis*[1], *blogs*[2] or even *online-databases* (such as Google Base[3]) are enriched with semantic patterns for self-organized structuring (e.g. tagging) of peer created content. Contemporary Web 2.0 services also enable *text processing* (e.g. Google Docs[4]) or *spreadsheet calculations* (e.g. Zoho[5]).

As an orthogonal technology, various services can be combined with syndication feeds like *RSS* (Really Simple Syndication) or Atom that bring hypertext characteristics to effectively any kind of resource.

Content Sharing

Along with the fundamental ideas of Web 2.0 (O'Reilly, 2005), the added value of user-generated content is further amplified by its distribution and online availability. Web 2.0 services offer many opportunities to share a wide range of media and documents. We will mention the most prominent categories as represented by *Flickr*[6] or *Google Picasa*[7] for photo sharing, *Adobe Share*[8] for documents, *MediaMax*[9] *and DivShare*[10] for multimedia sharing services, while *Slideshare*[11] or *V3 Solution and Resource Center*[12] are suited to sharing slide presentations and screencasts.

Each service provides a certain kind of access management. Open and cooperative group policies may limit access to online information depending on the level of publicity or confidentiality. *OpenID*[13] and *Enterprise Sign On Engine*[14] are examples of *Authentication and Authorization Infrastructures* (AAI) enabling service access via a light-weight identity service.

Content Storage

In addition to this end-user content sharing, we also come across services providing generic storage and data management functionality for other services, thus contributing to the "web as the platform". One of the most successful ones is *Amazon S3*[15] acting as the underlying data storage for many famous data-heavy services. There are also infrastructural services analyzing web traffic for maintenance and marketing purposes, see *Monitor.us*[16] or *Google Analytics*[17].

Content Contextualization, Presentation and Management

These services provide content created by interacting with other web interfaces for a final enriched presentation to the user. Thus, communication via APIs (Application Programming Interfaces) and customization possibilities form the added value of these services. Services for presenting *web-based desktops*[18] or *geo-referenced map presentations* (like *Google Maps*[19]) can be noted. Individual information retrieval and bundling are gaining more and more importance these days which technically means interconnecting services via their application interfaces. Services like *Yahoo!*

[1] http://www.wikidot.com or http://www.wikitravel.com
[2] http://www.wordpress.com
[3] http://base.google.com
[4] http://docs.google.com
[5] http://www.zoho.com
[6] http://www.flickr.com
[7] http://www.picasa.com
[8] http://share.adobe.com
[9] http://www.mediamax.com
[10] http://www.divshare.com
[11] http://www.slideshare.net
[12] http://leonardo-v3.eu
[13] http://www.openid.net
[14] http://www.esoeproject.org
[15] http://aws.amazon.com/s3
[16] http://mon.itor.us
[17] http://analytics.google.com
[18] http://www.netvibes.com or http://www.pageflakes.com
[19] http://maps.google.com

Pipes[20] or *Dapper*[21] fit into this class. In future these services will consequently also be provided under different license models. At present, there are service providers, e.g. *Postini, Inc.*[22], utilizing Google Apps API directly in their customers' infrastructure. Contextualizing data services like verifying postal codes or ordering numbers from third party services is a promising next step into flexible web service integration.

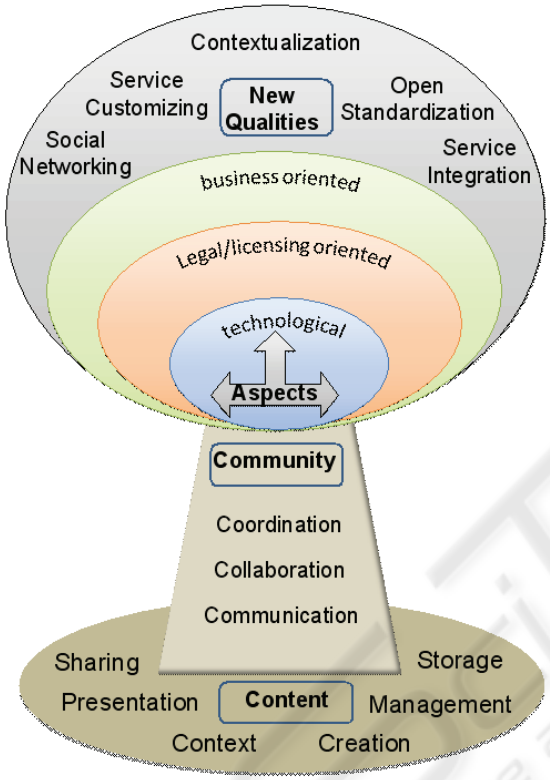

Figure 1: The Mash-Tree shows qualities of Web 2.0 services.

Coordination and Communication

Having started with web-based e-mail services at the beginning of the Internet era, most e-mail providers now supply complex communication and collaboration possibilities. As an example, the Google service suite, i.e. *Google Calendar* together with *GMail* enhanced by *Google Talk* as well as the integration of instant messaging features into *Yahoo! Mail* provide an open API-based access. Unlike complex e-mail infrastructures like *Microsoft Exchange*, lightweight web-based collaboration services (like *Zimbra*[23]) are approaching flexible service interoperability. Typically, they utilize tags for personal e-mail content structuring. Zimbra's special features are its encapsulated modules (Zimlets) for integrating APIs provided by distinct client applications.

Community

Communities are playing a decisive role in the success of the development of Web 2.0 services, as most of them exhibit a strong network effect and may even depend on reaching a critical mass (Hendler & Golbeck, 2007). People with similar interests and/or common activities use social networking services such as *MySpace*[24], *XING*[25], *Facebook*[26], or even more specialized community services.

Table 1 depicts services selected from each classification category.

Table 1: Service Classification Overview.

Content	Creation	Blogger Bubbl.us Google Docs Wikidot.com Zoho Online Office
	Sharing	Adobe Share Clipmarks Diigo DivShare Flickr MediaMax Picasa Web Albums SlideShare
	Storage	Amazon S3
	Contextualization, Presentation and Management	Pageflakes Google Maps
Community	Communication	GMail + GTalk
	Coordination	Backpack Calendarhub Google Calendar Google Groups Remember The Milk
	Community	MySpace

[20] http://pipes.yahoo.com
[21] http://dapper.com
[22] http://www.postini.com
[23] http://www.zimbra.com
[24] http://www.myspace.com
[25] http://www.xing.com
[26] http://www.facebook.com

3 ASPECTS OF MASHING

We will now concentrate on technological, business, and legal aspects of mashing the Web 2.0 services. We will show which techniques, protocols, and formats are predominantly used as well as what quality of support developers can expect from the service providers. Finally, we will discuss what business and legal issues are related to the mashing development and operating.

3.1 Technological Aspects

Mashing Techniques

Nowadays, many web services are loudly supporting the principles of Web 2.0. Unfortunately, some of them are using this fashionable label for marketing purposes only and are somewhat remote from implementing its ideas. One of the basic principles of Web 2.0 – "web as a platform" – is hard to implement without services offering their functionality for programmatic access. Our analysis correlates with another result (Novak & Voigt, 2007) and shows an alarming lack of realization of Web 2.0 principles, since one third of the services reviewed have been created for direct usage by humans only.

Services without an application programming interface can be (from strictly technological point of view) accessed using web-scraping techniques, i.e. analyzing the HTML code of their web interface and then simulating user actions normally performed in a web browser. This is, however, unreliable, error-prone and disputable from a legal point of view.

Mashing on Server or Client

The basic decision developers have to make when designing a mashup service is whether the code accessing particular services and combining their functionality will be run on a server or on client. Both variants have their advantages and disadvantages and further elaboration of this problem is beyond the scope of this article (see however Ort et al., 2007). What is important here is that apart from the traditional server-side model, there are also client-side models in which developers have very limited possibilities in terms of operations allowed and available technologies.

Protocols and Formats

The technical realizations of APIs in services that offer such a possibility differ a lot (cf. Table 2). However, all APIs except those of instant messaging and emailing services use HTTP as a communication protocol. The reasons for this are clear – its simplicity and the fact that it can easily pass through firewalls in the Internet.

Table 2: Services, API, and Protocols.

Service	Official API
Adobe Share	REST
Amazon S3	REST, SOAP
Backpack	XML over HTTP
Blogger	GData
bubbl.us	no
Calendarhub	no
Clipmarks	no
Diigo	no
DivShare	REST
Flickr	REST, XML-RPC, SOAP
GMail + GTalk	POP3/SMTP/IMAP, XMPP
Google Base	GData
Google Calendar	GData
Google Docs	GData
Google Groups	no
MediaMax	REST
MySpace	no
Picasa Web Albums	GData
Remember The Milk	REST
SlideShare	REST
wikidot.com	no
Zoho Online Office	REST, XML-RPC

Although some services utilize the traditional RPC (Remote Procedure Call) messaging model in the form of either SOAP (Simple Object Access Protocol) or XML-RPC (eXtensible Markup Language with Remote Procedure Call), the most popular messaging model today is REST (Representational State Transfer). The term is, however, often used in a much looser sense than it was defined in Fielding (2000), meaning that the service offers a simple interface transmitting domain-specific data over HTTP without an additional messaging layer. The data is transferred in HTTP responses in various formats including service-specific XML, JSON (JavaScript Object Notation), binary data and plain text. For data retrieval, RSS and ATOM formats are very popular. In addition to pre-defined RSS/ATOM channels, services usually provide their users with the possibility of defining their own channels with data selected by user-specified criteria.

Google, one of the major Web 2.0 companies, developed its own generic protocol for reading and writing data on the Internet, called GData. It is based on RSS and ATOM, and allows data updates, too. Google uses this home-made protocol in all its APIs.

It is not an exception when a service offers more API variants or even more semantically equivalent APIs. Service client developers can choose not only which format they want to receive data in (XML, JSON etc.), but sometimes also which messaging model and protocol they want to use (REST, SOAP or XML-RPC). The two main reasons for this are the freedom of choice for developers and – especially in newly established services which are trying to attract users away from competitive services – copying the API of some rival service in order to make the transition between services easier (e.g. *ma.gnolia.com* offers del.icio.us-like API).

The relationship between technology standardization and service „mashability" is depicted in Figure 2.

Developer Support

The basic support for service client developers is API documentation. This can be found in a variety of forms and extents with every service offering a public API. Some services go even further and provide libraries, which facilitate the use of their API in specific programming environments (Java, JavaScript, PHP, Ruby, or .NET). Where no library is officially provided, there are third-party libraries which can do the job. "Third party" usually represents some enthusiastic service user(s).

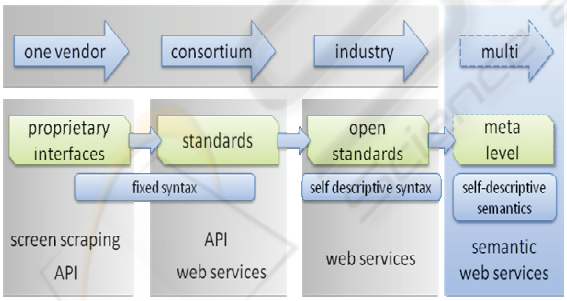

Figure 2: Technological Standards and Integration.

Authentication and Authorization

Services often require client applications developers to request an "API key" that uniquely identifies every client application when accessing the service through its API. This allows services to set limits to the usage of their API (e.g. to one query per second or 1,000 queries per day for a registered application, as SlideShare does). Such limits are not an issue for single-user applications but this is quite a rare case in the "web as a platform" environment. For multi-user applications such limitations can be a serious problem.

3.2 Business Aspects

When the understanding of the "Web Service" concept in general is only of a technical nature (as described in the last section), the counterpart in an economic sense is service supply and demand. Mashups can be seen as a specific combination of existing services that bring a particular added value. This combination is more than just technical services and their interaction with each other. Repercussions also occur for the non-technical service in general. For instance, the popular service "Google Maps" can be found in other solutions integrating (geo)spatial information such as *Trulia*[27]. It presents information which is enriched with governmental and private data collections. But what are the essential elements of successful acceptance?

Out of our survey on well-established mashups, their success can be seen in their implementing of the following basic conditions. It is a combination of access to information, semantic enrichment and individuality of the service (see Figure 3).

Access to Available Information

A necessary precondition for flexibly usable services is the access to primary sources of information. The newly created service has access to the primary source's data, e.g. purchasing power in relation to the geographical area based on public surveys. Besides free access to data, which is in most cases owned by the government, some service providers have become specialists in private information collections, like *Navteq*[28] or *Claritus*[29], where access costs money in the majority of cases.

Value-added Semantic Combination

The success of mashed-up services is based on an easy way of combining and consolidating existing information in order to create a new value. The popularity of services like *Yahoo Pipes* (addressing different APIs for a customized search) or *Microsoft Popfly* (customizing an API-controlled Web Desktop), or even a general-purpose *IBM Mashup*

[27] http://www.trulia.com
[28] http://www.navteq.com
[29] http://www.claritus.com

Starter Kit[30] is based on that advantage. A good example showing the power of new forms of mashed-up services is given by *Placebase, Inc.*, a company presenting detailed information based on geo-referenced statistical combination, like a census or neighbourhood data.

Generating an Individualized Service

Mashing data or information into a meaningful service requires a focus on customers. Firstly, the presentation of emerging information out of the user's demand is a necessary condition for success. Mashups must aim at an individual composition of services to present this information. Secondly, influenced by a specific "culture" of both, offering and using services, the attitude of a long-term balance between taken and given is reached by allocating web access to the public. In fact, the success of Google Maps is a result of this attitude. This diversity of so-called "open web services" will lead to a customized use of services and a more accurate settlement of demand again. Deriving from this fact, a third step can be seen in the flexibility of the underlying business model. It contains adaptable license agreements and legal terms of use depending on the contextualized usage.

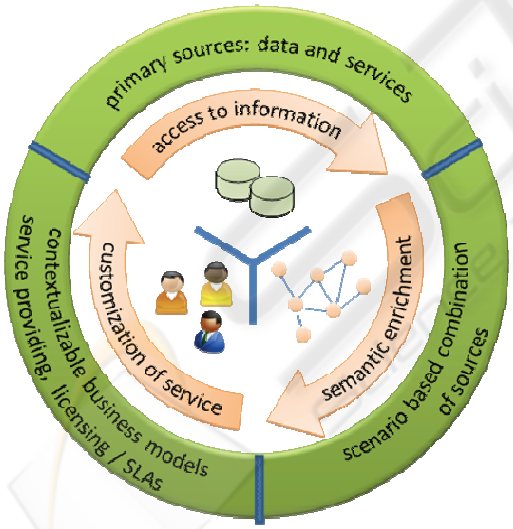

Figure 3: Success Factors for a Mashup Service.

In combination with innovative techniques these business-related principles of the new web are promising advantages for a Web 2.0 era, see Figure 3.

[30] http://www.alphaworks.ibm.com/tech/ibmmsk

3.3 Legal Issues

Each of the services reviewed here, except one (*Bubbl.us*), provides its users with a document named "terms of use" or "terms of service" (ToS), even if it is not obviously visible for the user in every case. The style of licensing documents ranges from a list of clear and simple explanations to complex documents spanning 10 or more printed pages. The latter is typical especially for services provided by larger companies (Google, Yahoo etc.). Some services are aware of this difficulty and provide users with a simplified version of their ToS, explaining selected aspects in a more comprehensible way. Google is the only service provider offering some ToSs (namely for Blogger and Google Groups) in Czech.

User-supplied Data Licensing

Many Web 2.0 services are data-centred. The data can be provided by the service itself or, much more frequently, by its users. Many services just manage data provided or created by their users. Therefore the rules for handling this data are very important.

As a general rule, services do not claim ownership of data provided by users. On the contrary, they disavow the data in order to prevent themselves from accusations of publishing law infringing data. On the other hand services need to have special rights to the content due to their functionality – e.g. image sharing services would not be able to create thumbnails without the right to modify supplied pictures. At this point, services' conditions differ significantly – some services claim solely the rights necessary for providing their functionality, while others claim much wider rights, e.g. using supplied non-private data for advertising or promotion.

Data privacy is another important criterion. All services are treating personal data and the data labelled as "private" as confidential and reveal them without the submitter's approval only under very serious conditions, explicitly named in their ToS.

Almost all services allow their users to restrict access of other users to supplied data using sophisticated AAI implementation, but a cross-service infrastructure is rarely adopted. It usually includes private access (only owners can access the data), public access (everybody on the Internet can see or even modify the data) and also some specification in between. Approaches to data sharing vary a lot.

Licensing user-supplied published data for other service users is not considered by many services and

is rather left up to users, see Table 3. Only a few services deal with this type of licensing, systematically allowing users to assign one of the pre-defined licenses based on *Creative Commons*[31] to every piece of data. Flexible determined license structures are representing an effective approach to multidimensional agile usage.

Table 3: Licensing User-generated Content.

Service	Licensing for service provider	Licensing for other users
Ad. Share	Necessary only	
Amazon S3		All rights reserved
Backpack	Service promotion	
Blogger	Service promotion?	
Bubbl.us		
Calendhub	Data ownership	All rights reserved
Clipmarks	Complete rights	
Diigo	Use in their services	All rights reserved
DivShare		All rights reserved
Flickr		7 possibilities (CC*)
Gmail/Talk	Necessary only	
G. Base	Use in their services	
G. Calendar	Necessary only	
G. Docs	Necessary only	All rights reserved
G. Groups	Use in their services, promotion	
MediaMax		
MySpace	Use in their services	All rights reserved
Picasa	Use in their services, promotion	
Remember The Milk	Necessary only	
SlideShare	Usage for their business	7 possibilities (CC*)
Wikidot	Usage in their services	14 possibilities (CC*)
Zoho		All rights reserved

Terms of Service

Although each service has its specific ToS, some aspects are common for all reviewed services. Firstly, services are, not surprisingly, sensitive about user accounts – they require accounts not to be created in any automated way, allowing creating accounts for humans only, and forbid account sharing among multiple people. Secondly, services are anxious about the data and therefore explicitly prohibit any kind of systematic harvesting or indexing of its content. The third common point is software copyright. It is not allowed to copy, reproduce, alter, modify, reverse engineer or create derivative works from any of the services.

Table 4: Terms of Service and GUI.

Service	Incorporating GUI	Automated access to GUI
Ad. Share	forbidden	unspecified
Amazon S3	no GUI	no GUI
Backpack	with permission	unspecified
Blogger	forbidden	unspecified
Bubbl.us	unspecified	unspecified
Calendhub	forbidden	unspecified
Clipmarks	forbidden	forbidden
Diigo	forbidden	forbidden
DivShare	with permission	no robot w/o permission
Flickr	with permission	via provided intf. only
Gmail/Talk	forbidden	forbidden
G. Base	forbidden	via provided intf. only
G. Calendar	forbidden	forbidden
G. Docs	with permission	forbidden
G. Groups	with permission	unspecified
MediaMax	unspecified	"Reasonable Usage"
MySpace	with permission	forbidden
Picasa	forbidden	via provided intf. only
Rememb. TM	forbidden	forbidden
SlideShare	forbidden	no "burdening" robots
Wikidot	limited	forbidden
Zoho	allowed	unspecified

Some services prohibit incorporating their graphical user interface into other applications or accessing it via any automated processes (cf. Table 4). But even in cases where these activities are not explicitly forbidden, they are probably violations of the previously mentioned rule. The only service directly supporting incorporating its GUI into another application is Zoho, which allows using *Writer*, *Sheet* and *Show* from its office suite as editors for appropriate document formats.

All services restrict user supplied content not to violate copyrights and not to contain vulgarity, nudity, racism and other improper or illegal information. For some services, this is the only limitation. Other services go further and restrict the data type in accordance with the service's intention – to personally taken photos (Flickr), for example. Another example of content limitation is in Google

[31] * http://creativecommons.org

Base which (at the time of writing) allows textual data to be in English and German only. The Figure 4. shows a scale of "openess" and mashability with respect to service and content licensing.

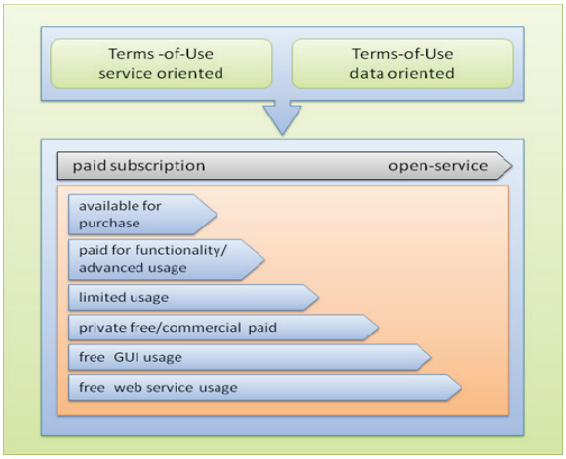

Figure 4: Mashability Determined by ToS Factors.

4 CONCLUSIONS

The main goal of this research was to provide a conceptual, technical, business, and legal foundation for Web 2.0 mashup development and deployment.

We proposed a basic classification of Web 2.0 services from functional points of view and applied this classification to a selected set of popular services. Then, the "Mash-Tree" was proposed.

Further, we have identified "mashing" potential and issues of the services in three dimensions: technical, business, and legal. The technical analysis showed that – despite the overall popularity of SOAP in enterprise computing – in Web 2.0 the HTTP and REST are predominantly used.

We have revealed that many often-alluded to practices like screen scraping can rarely be implemented legally. Our investigations also brought interesting results about licensing the user-generated content stored in the Web 2.0 services.

By highlighting technical success factors as well as distinguishing different types of business and legal aspects onto the market, this article contributes to the improvement of further mashup development.

Our research of well-established Web 2.0 services shows that a full mashing potential has not yet been achieved in the market. A ubiquitous practicability and usage is closely linked with self-description and (machine) readability of a service, which is not achieved in all cases yet. In a vision of the New Web, services with different focus can be flexibly mixed into a new context.

The potential of service mashability for flexibility of software usage and tailored software customization is bringing new business models to bear, replacing previous models produced under monolithic, fixed infrastructures. Scenario based licensing as well as cross service charging (Micropayment 2.0) can be a successful achievement for users and producers.

We hope that our conceptual framework, classification and aspect evaluation can serve mashups developers as well as future research.

ACKNOWLEDGEMENTS

The research has been partially supported by the Czech National Program Information Society, project 'E-learning in the Semantic Web Context', No. 1ET208050401.

REFERENCES

Alonso, G., Casati, F., Kuno, H. and Machiraju, V. (2004). Web Services – Concepts, Architectures and Applications. Springer-Verlag, Berlin, 2004

Creative Commons (2008). http://creativecommons.org, cited 06/03/08

Fielding, R. (2000). Representational State Transfer, http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm, cited 06/03/08

Gartner Group (1996). Service Oriented Architectures, Part 1 and 2, SSA Research Note SPA-401-068 and SPA-401-069, Gartner Press, 1996

Hendler, J., Golbeck, J. (2007). Metcalfe's law, web 2.0, and the semantic Web, Journal of Web Semantics, Vol. 10, Elsevier B.V., 2007

Merrill, D. (2006). Mashups: The new breed of Web app, IBM Developerworks, 2006, http://www.ibm.com/developerworks/xml/library/x-mashups.html

Novak, J., Voigt, B. (2007). Mashups: Structural Characteristics and Challenges of End-User Development in Web 2.0, in: i-com 02/2007, Zeitschrift für interaktive und cooperative Medien, Volume 5, p. 19-25, Oldenbourg Verlag, Munich, 2007 (in German)

O'Reilly, T. (2005). What is web 2.0? – design patterns and business models for the next generation of software.

Ort, E., Brydon S., Basler M. (2007). Mashup styles, Part1 and Part2: Server-Side Mashups, 2007, http://java.sun.com/developer/technicalArticles/J2EE, cited 06/03/08