

# Web Service Discovery - Reality Check 2.0

Stephan Hagemann, Carolin Letz and Gottfried Vossen

**Abstract—** In practice the ability to find the appropriate Web service decides between an evaluation and use of that service on the one hand and the functionality being implemented anew on the other. This paper evaluates existing public portals for Web service discovery with respect to their characteristics and their acceptance by developers. For this, we distinguish different possible settings and use cases, and we evaluate how these are supported in practice. It turns out that only few of the publicly available Web service registries are still growing in size and importance, with the use case best supported lying in the pre-programming phase.

**Index Terms—** Web services, Web service discovery, UDDI, public service directories, Web 2.0

## I. INTRODUCTION

WEB services (WS) and their standards have matured over the last couple of years. Their importance as realizing components for service-oriented architectures (SOA) has increased as well [9, 30]. A key driver for SOA implementations is the hope to save development time and costs through a higher degree of reuse of components in the form of readily implemented services [16, 4]. To achieve this aim it is necessary, among other things, to make Web services discoverable.

Different theoretical approaches, ranging from artificial intelligence to automata theory, have focused on certain aspects of reasoning about Web service discovery, comparing the expressiveness of Web service models and examining the complexity of service matching, e. g., [7, 17, 19, 25, 27]. These theoretical approaches are in contrast to the current practical state of Web service discovery on the public Internet, where much less sophisticated techniques are employed, an apparent clash that has been observed before [10].

At the end of 2005 three groups of approaches to Web service discovery have been identified and analyzed in [2]. The comparison includes the Web service standard approach, the

UDDI business registry, specialized Web service search engines and portals, and general-purpose search engines. Features taken into account are the search functionality, the number of available services, the availability of discovered services, the choice of interfaces, the availability of a service rating, of test and demo functionality, of a WSDL parser, and of information about service costs.

Years 2006 and 2007 have seen a dramatic rise of public interest in Web 2.0 applications. A common interpretation of this buzz-word is the shift from the Web of information towards the Web as a platform [22]. This trend was not noticed in [2]. Therefore we have redone this type of study. However, we have not merely taken a snapshot of the current state of Web service discovery on the public Internet, but have also uncovered trends and tendencies in the events of the past two years. Consequently, the criteria for comparison have been extended to put a special emphasis on the opportunities Web 2.0 developments have to offer.

The paper is organized as follows: In Section II, the foundations of Web services and their discovery are summarized, where the focus is on practical techniques for discovery as relevant for this study. Furthermore, the foundations of Web 2.0 and the support for information retrieval in Web 2.0 are presented. In Section III, different practical approaches towards Web service discovery on the public Internet are examined and updates compared to the findings in [2] are given. The different approaches are compared and opportunities for Web 2.0 techniques are pointed out that help to enhance Web service search engines. In Section IV, the paper concludes with an outlook on future research opportunities that combine Web service discovery and Web 2.0.

## II. WEB SERVICE DISCOVERY

Different scenarios for Web service registries have been described by the standardization committee OASIS [26], by software vendors such as IBM [12] or Microsoft [21], and by software designers and researchers [9]. The following summarizes and consolidates these scenarios.

### A. Web Service Registry Scenarios

The first and most general use case is to operate a registry as a global domain-independent business registry, the *UBR*. In this scenario it is used as public registry service, open to every service provider and consumer, and free of charge. It functions both as a registry and a search engine for establishing B2B contacts in an automated way [9, 12, 26].

Instead of offering a global registry for every domain it is

Manuscript received June 3, 2008. A previous version of this paper with survey data from May 2007 has been published at the 3rd International Conference on Next Generation Web Services Practices, October 29-31, 2007.

Stephan Hagemann is a Ph.D. student at the Department of Information Systems at the University of Muenster, Germany (e-mail: stephan.hagemann@gmx.de).

Carolin Letz was with the Department of Information Systems at the University of Muenster, Germany and now works in industry (e-mail: Carolin.Letz@gmx.de).

Gottfried Vossen heads the database group within Department of Information Systems at the University of Muenster, Germany (phone: +49 251 8338150, fax: +49 251 8338 159, e-mail: vossen@uni-muenster.de).

also possible to establish *domain specific business registries* that contain entries related to only one industry branch [12]. Such registries need to establish usage rules that must be adhered to by all users to ensure the quality of the entries and to avoid unwanted entries that are not domain related.

Registries are also set up as *registries within the Intranet* of enterprises [9, 12]. Services of one department can be offered to all organizational units. Application designers use the registry to search for available services and as reference list that does not only contain the services but that also helps finding the related technical documentation [21]. Moreover, external services can be approved that the application developers are allowed to use.

The Intranet usage scenario can be extended such that the company does not only use its own and external services, but for some services also acts as an external service provider towards trading partners and customers (*B2B and EAI Service Registry*) [12, 21, 26].

#### B. Web Service Search Use Cases

At design time, a Web service registry can be used in a combination of browse and drill-down search patterns. A search result list can be inspected manually or refined through a more specific search. In order to obtain more information on a service its details are inspected. Finally, a designer can invoke the service allowing for static binding. This sequence of browse, drill-down and invocation patterns is supported by the UDDI standard.

Besides static binding, the search functionality of a registry can also be used for dynamic binding, where certain aspects of the service are determined at runtime. We will speak of *dynamic binding* if the client application calls a service interface as defined by a template (called *tModel* in the UDDI standard). The service template can be implemented by more than one Web service. Thus, the interface and the binding details are known to the requestor a priori. Only the exact service and its provider are chosen at runtime. Other variants of dynamism are irrelevant in practice: dynamism where provider and service are unknown at design time is considered unrealistic [1], and the concept of the service bus is mainly employed within institutions [18, 30].

#### C. Implementations for WS Registries

UDDI is an XML-based Web service standard that describes how to implement and interact with a service registry. It is a framework that describes which data structures and APIs a Web service registry must offer to support Web service publication and search. *WS-Inspection* [3] defines how providers should publish their services, allowing for a decentralized search where consumers to search the services of providers they trust. *WS-Dynamic Discovery* [5] can complement this in ad-hoc networks where Domain Name Services or directory services do not exist. It defines a multi-cast discovery protocol to find services and as such not applicable for Internet wide use.

*Electronic business XML* (ebXML, [www.ebxml.org](http://www.ebxml.org)) is a

standardization initiative that promotes an open XML based infrastructure for world-wide, interoperable, secure and consistent exchange of information about electronic business. Their standards are especially helpful for Web service detection in two ways: First, ebXML registries are Web services and can therefore be registered in UDDIs, becoming available as any other Web service [23]. Second, ebXML provides a framework to define core components for e-business that can be reused in any context, such as the naming of data types in a WSDL document [31].

#### D. Web 2.0 Information Sharing

The interest in Web 2.0 [29] has risen ever since the term got wide attention after the publication of [24]. Its core has been defined as the read/write Web [14], also subsumed under "Harnessing Collective Intelligence" [22]. A strong convergence is currently seen between principles from SOA and Web 2.0 [15]. The differences between the two are important to understand the different attitudes these approaches to service-orientation have.

SOA places much emphasis on a controlled environment, employing heavy-weight standards such as UDDI, which has resulted in what is called the *Web services stack* [28]. Web 2.0, on the contrary, seems to neglect most of these standards and instead employs light-weight models based on HTTP and the REST architecture style [11]. It is inherently user-oriented, its values and patterns are said to be "the basis for the next generation of the Internet" [22]. Ajax is one example of a commonly applied pattern in the Web 2.0 context [6]. It leverages existing Web technologies, namely XML, JavaScript, and XMLHttpRequest, by combining them to make Web pages more responsive and intuitive.

At another end, many Web sites have started to add community features to their service portfolio. The most prominent examples include Amazon, Delicious, Flickr, and MySpace. A common way to exploit the community on a Web site is through *tagging*. Different characteristics of tagging systems have been identified in the literature [20]. They share the fact that they allow users to classify resources by freely chosen keywords known as *tags*. The important difference when compared to a hierarchical classification scheme is that no a priori consensus is necessary; at best consensus on certain tags emerges over time. Then tags form a flat name space called *folksonomy*. A folksonomy can be searched, thereby supporting user-created access-patterns to the resources of a site.

Similar in spirit are Web sites providing access to their data or services (the prominent examples are the same as above). This also allows reuse by others, which is precisely the original SOA idea. In the context of Web 2.0 this has seen a growing interest under the name *mash-up*. Mash-ups combine (typically freely available) services to create added value. Of course, mash-ups are similar to composed applications, which is the term used in the SOA community.

From our discussion above, it is fair to say that Web 2.0 and SOA are essentially concerned with the same topic, namely

information and software reuse. As we will see, they are, however, working at different ends of the spectrum.

### III. SERVICE DISCOVERY ON THE PUBLIC INTERNET STATE OF THE ART

Two years after the survey in [2] has been published, the respective sites are revisited in this section to compare them anew. In the following we will only mention URLs if they deviate from the pattern “www.name-of-service.com”. The comparison includes: the UBR, Xmethods (www.xmethods.net), Bindingpoint, WebserviceX (www.webservicex.net), WebserviceList, StrikeIron, Woogole [8], RemoteMethods, WSDLicious (wsdlcio.us), ProgrammableWeb, as well as general purpose search engines.

All services have been visited between April 30 and May 6, 2008. If they were unavailable during a first attempt they were revisited again later. If they were still unavailable they were considered to be obsolete.

#### A. Criteria for Comparison

In this section categories are introduced that are used to compare the different Web service discovery approaches.

*Basic features of Web service discovery* include the search functionality itself, the number of services available through a specific search approach, the availability of status information, and the possibility to discover different kinds of technical interfaces to one service.

*Quality of service features of Web service discovery* encompass additional functionality, such as test or demo functions and a WSDL parser as well as information about service costs and terms of usage. These additional features help the user in making a choice.

Under *Web 2.0 features for Web service discovery* we subsume those aspects that have of late been attributed to Web 2.0 approaches and techniques. As a first item in this category we consider all forms of community support, as a second one information on service usage.

#### B. Comparison Results

In this section the results of our analysis are summarized and the different Web service search approaches are compared according to the criteria developed earlier. A full version of the analysis with detailed results for each service can be found in [13].

Table I presents a summary of our analysis. For every service two rows are included: The first row represents the results of [2] (plus historical information for the new criteria where these were accessible), the second row summarizes the current results. We have not listed SalCentral and eSynaps, since these proved to be unreachable or irrelevant in [2] already and we consider them obsolete. WSDLicious was operating when the last version of this paper was written, but is offline now. Further, Bindingpoint has shut down as well.

The columns represent the evaluation criteria from the three groups listed in Section III.A. Column “Search” takes one or more of the following entries: *search* (keyword search), *list* (listing of all services), *categories* (browsable categories), and *template* (template-based search). Column “Number of services” contains the absolute numbers and also shows the change that has taken place in the number of services since [2]. The numbers in gray are the data from the previous version of this paper. The numbers should be taken with a grain of salt, as the number of services is not accurately countable for pages that do not have a full list. “Alternative interfaces” contains additional possibilities for accessing the registry. Note however, that interfaces may have different capabilities although the protocol is the same. “Community support” lists which mechanisms the site provides that can foster the building of a community. It does not give information on if and how these possibilities are actually used. All other columns should be self-explanatory.

Overall, the number of publicly available Web service registries as well as the number of available services has decreased. Except for StrikeIron and ProgrammableWeb, all registries make a neglected impression. The interest in this topic seems to remain nevertheless, as the increased number of results listed at WebserviceList, StrikeIron, and Remote-Methods compared to the evaluation in the previous version of this paper point to the same direction.

The UBR registry is the only approach that has tried to implement the UBR usage scenario as described in Section II. The other registries are intended to be open to a large community but they are not synchronized and connected with each other. Therefore, they support the UBR usage scenario in a restricted way. A business specific marketplace as described in Section II is not among the public search catalogues. Only StrikeIron might be regarded as an attempt to realize this scenario domain independently because it handles quality control and administration for their providers and consumers. For this reason, StrikeIron can be considered a service broker.

Most service registries contain Web services that are only poorly described, links to provider home pages are broken, the services are not available and links to WSDL files are broken, too. This indicates that the quality and availability of registered services is hard to guarantee. As long as the registry provider does not care about these quality aspects the registry does not inspire trust. Also test and demo functionality as well as user feed-back mechanisms do not seem to ameliorate the quality problems. Public quality control only works in those places where a great public is attracted, as projects such as Wikipedia show.

Furthermore, many services offer functionality that is not of interest to a large community or that is too trivial, such as most conversion services. If the services are not of interest, the Web service registry degenerates to a playground. Obviously, services of interest do not need a central registry to become known (e.g., the Amazon E-Commerce services). Such a service provides added value to a large community. It can be regarded as an example for the B2B usage scenario as

described in Section II.

Support for consumers who seek information on how to solve a design problem at hand or who are not sure if they want to use a Web service at all do not find help in most registries. ProgrammableWeb is an exception that reveals the typical Web 2.0 features of social interaction. Working programming examples, the mash-ups) and the possibility to share experiences in discussion forums are important additional features that are not included in standards such as UDDI.

The number of services found by Google when searching for "filetype:wSDL" seems to be fluctuating greatly over time. The exact number of services is, however, not of great interest. As we have pointed out in [13] the percentage of usable services that such a search yields is low due to missing pages and invalid WSDL documents.

Overall, Web service registries on the public Internet suffers from a lack of acceptance. It can be seen that technical standards are not as important as providing additional functionality around such registries. Such services could include quality control, administration and payment transactions as offered by StrikeIron or community specific knowledge and experience sharing as offered by ProgrammableWeb.

#### IV. CONCLUSIONS

The results presented in this paper show that three aspects must be taken into consideration for an evaluation of public Web service registries: the type of service offered, the quality of the offer, and the way in which the services are presented to support customer decisions.

*Type of service.* A central Web service registry is not a success factor per se. Central reachability alone is not decisive. Moreover, it is important that the Web services themselves provide added value for businesses.

Information is an asset in itself and access to crucial business information is a service in itself. A convenient way of delivery is via the Internet, as it gets rid of the problem of updating local databases. We therefore see information providing services as good candidates for successful Web services.

*Quality of the offer.* For a registry to be attractive to developers, the services offered must be available, the registry provider must ensure their technical quality and the services must be described in such a way that they can easily be integrated into other applications. Test functionality must be available in such a way that testing is possible without reading the technical interface description.

The attractiveness of a registry can be increased by offering additional services, as was seen with StrikeIron and ProgrammableWeb. However, even with high quality services and comfortable additional services for service providers and customers, a registry must first of all be able to offer services that are of interest. If this is not fulfilled even the best designed registry will fail.

*Presentation.* As the analysis above has shown, programmers searching for Web services do not go straight to a

Web service registry or to a service provider they know during the first stages of searching. In an early stage of a software development project, a make-or-buy decision must be made. On a smaller scale, this is the same with the decision for or against a Web service or a Web 2.0 component. Web service registries do not support this early decision stage. As ProgrammableWeb shows the early decision stage needs to answer such questions as "Has anybody before successfully used a Web service for the problem I need to solve?", "Where do I find successfully running examples?", "Which services exist?", or "Which providers are there?". Only the last two questions can be answered through a Web services registry approach. The first questions seek experiences and advice from members of a peer group of programmers. Sharing knowledge and experiences is the strong feature of Web 2.0. In this aspect, Web sites such as ProgrammableWeb are complementary to the Web service registry approach.

Our analysis shows that there are currently no successful Web service registries that base their strategy on being an access point for service search alone. StrikeIron acts as a broker. However, the small number of services offered limits the attractiveness of the site. ProgrammableWeb adds a community to the registry. It remains to be seen whether this site can remain attractive and continue growing beyond the current hype of Web 2.0.

#### REFERENCES

- [1] G. Alonso, F. Casati, H. Kuno, and V. Machiraju. Web Services. Concepts, Architectures and Applications. Springer-Verlag Berlin, Germany, 2004.
- [2] D. Bachlechner, K. Siorpaes, D. Fensel, and I. Toma. Web Service Discovery - A Reality Check. Technical report, DERI, 1 2006.
- [3] K. Ballinger, P. Brittenham, A. Malhotra, W. A. Nagy, and S. Pharies. Web Services Inspection Language 1.0 (WS-Inspection). IBM, Microsoft, 2001. URL: <http://download.boulder.ibm.com/ibmdl/pub/software/dw/specs/ws-wsilspec/ws-wsilspec.pdf> (2008/04/30).
- [4] C. Baroudi and F. Halper. Executive Survey: SOA Implementation Satisfaction. Technical report, Hurwitz and Associates, 2006.
- [5] J. Beatty, G. Kakivaya, D. Kemp, T. Kuehnel, and B. Lovering et al. Web Services Dynamic Discovery (WS-Discovery). Microsoft, 2005. URL: <http://msdn.microsoft.com/en-us/library/bb706924.aspx> (2008/04/30).
- [6] D. Crane, E. Pascarello, and D. James. Ajax in Action. Manning, Greenwich, CT, 2006.
- [7] A. Deutsch, L. Sui, and V. Vianu. Specification and Verification of Data-driven Web Services. In Proc. of the 23rd ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS), pages 71–82, 2004.
- [8] X. Dong, A. Y. Halevy, J. Madhavan, E. Nemes, and J. Zhang. Similarity Search for Web Services. In (e)Proc. of the 30th Int. Conference on Very Large Data Bases (VLDB), pages 372–383, 2004.
- [9] W. Dostal, M. Jeckle, I. Melzer, and B. Zengler. Serviceorientierte Architekturen mit Web Services. Konzepte-Standards-Praxis. Spektrum Akademischer Verlag, München, Germany, 2005.
- [10] J. Fan and S. Kambhampati. A Snapshot of Public Web Services. SIGMOD Record, 34(1):24–32, 2005.
- [11] R. T. Fielding. Architectural Styles and the Design of Network-based Software Architectures. PhD thesis, University of California, Irvine, 2000.
- [12] S. Graham. The Role of Private UDDI Nodes in Web Services, Part 1: Six Species of UDDI. IBM, 2001. URL: <http://www.ibm.com/developerworks/webservices/library/ws-rpu1.html> (2008/04/30).

- [13] S. Hagemann, C. Letz, and G. Vossen. Web Service Discovery: Reality Check 2.0 - A Comparison. Technical Report 5, ERCIS, Münster, Germany, July 2007.
- [14] D. Hinchcliffe. Continuing an Industry Discussion: The Co-Evolution of SOA and Web 2.0. Dion Hinchcliffe's Web 2.0 Blog, 8 June 2006. URL: [http://web2.wsj2.com/continuing\\_an\\_industry\\_discussion\\_the\\_coevoluti\\_on\\_of\\_soa\\_and.htm](http://web2.wsj2.com/continuing_an_industry_discussion_the_coevoluti_on_of_soa_and.htm) (2008/04/30).
- [15] D. Hinchcliffe. The story of Web 2.0 and SOA continues - Part 1. Enterprise Web 2.0 Blog, 17 May 2007. <http://blogs.zdnet.com/Hinchcliffe/?p=107> (2008/04/30).
- [16] N. S. Latimer-Livingston, C. Graham, J. M. Correia, and N. Schroder. Survey Shows Why Firms Undertake Web Services Projects. Technical report, Gartner Group, 2003. URL: [http://www.gartner.com/DisplayDocument?doc\\_cd=116069](http://www.gartner.com/DisplayDocument?doc_cd=116069) (2008/04/30).
- [17] C. Letz. Web service detection in service-oriented software development. PhD thesis, University of Muenster, Germany, 2007.
- [18] F. Leymann. Web Services: Distributed Applications Without Limits. In Proc. of BTW 2003, pages 2–23, 2003.
- [19] L. Li and I. Horrocks. A software framework for matchmaking based on semantic web technology. In WWW '03: Proceedings of the 12th international conference on World Wide Web, pages 331–339, 2003.
- [20] C. Marlow, M. Naaman, D. Boyd, and M. Davis. HT06, tagging paper, taxonomy, Flickr, academic article, to read. In Hypertext, pages 31–40. ACM, 2006.
- [21] Microsoft Corporation. Enterprise UDDI Services: Three Usage Scenarios, 2003. URL: <http://www.microsoft.com/windowsserver2003/techinfo/overview/uddisynop.msp> (2008/04/30).
- [22] J. Musser and T. O'Reilly. Web 2.0 - Principles and Best Practices. O'Reilly Media, Sebastopol, 2007.
- [23] U. OASIS. ebXML Registry Project Team: Using UDDI to Find ebXML Reg/Reps, 2001. URL: <http://www.ebxml.org/specs/rrUDDL.pdf> (2008/04/30).
- [24] T. O'Reilly. What Is Web 2.0 - Design Patterns and Business Models for the Next Generation of Software. Tim O'Reilly Blog, 30 September 2005. URL: <http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html> (2008/04/30).
- [25] M. Paolucci, T. Kawamura, T. R. Payne, and K. P. Sycara. Semantic Matching of Web Services Capabilities. In Proc. of the 1st Int. Semantic Web Conference, pages 333–347, 2002.
- [26] The Stencil Group, Inc. The Evolution of UDDI. UDDI.org White Paper, 2002. URL: [http://www.uddi.org/pubs/the\\_evolution\\_of\\_uddi\\_20020719.pdf](http://www.uddi.org/pubs/the_evolution_of_uddi_20020719.pdf) (2008/04/30).
- [27] D. Trastour, C. Bartolini, and J. Gonzalez-Castillo. A Semantic Web Approach to Service Description for Matchmaking of Services. In Proc. of the 1st Semantic Web Working Symposium (SWWS), pages 447–461, 2001.
- [28] G. Vossen. Have Service-Oriented Architectures Taken a Wrong Turn Already? In Proc. IFIP TC 8 CONFENIS 2006, pages xxiii–xxix, Vienna, Austria, 2006.
- [29] G. Vossen and S. Hagemann. Unleashing Web 2.0 - From Concepts to Creativity. Morgan Kaufmann, Burlington, July 2007.
- [30] S. Weerawarana, F. Curbera, F. Leymann, T. Storey, and D. F. Ferguson. Web Services Platform Architecture. SOAP, WSDL, WS-Policy, WS-Addressing, WS-BPEL, WS-Reliable Messaging, and More. Prentice Hall Upper Saddle River, NJ, 2005.
- [31] J. Whittle, S. Probert, M. Adcock, G. Boxman, and T. Becker. Core Component Overview. Version 1.05 ebXML Core Components. Technical report, OASIS, UN/CEFACT, 2001. URL: <http://www.ebxml.org/specs/ccOVER.pdf> (2008/04/30).

Germany. She worked as data warehouse consultant for the Financial Industry from 2001 to 2005 and as research assistant at the chair of Computer Science at the Department of Information System at WWU from 2005 to 2007. Currently, she is working as business analyst for the Financial Industry.

**Gottfried Vossen** is a Professor of Computer Science in the Department of Information Systems at the University of Muenster in Germany. He is the European Editor-in-Chief of Elsevier's *Information Systems - An International Journal*, and a Director of the European Research Center for Information Systems (ERCIS) in Muenster. His research interests include conceptual as well as application-oriented problems concerning databases, information systems, electronic learning, the Semantic Web and Web 2.0. Dr. Vossen has been member in numerous program committees of international conferences and workshops. He is an author or co-author of more than 150 publications, and an author, co-author, or co-editor of 21 books on databases, business process modeling, the Web, e-commerce, and computer architecture.

**Stephan Hagemann** received his MSc degree in Information Systems from the University of Muenster (WWU), Germany, in 2005. He worked as a research assistant at the chair of Computer Science at the Department of Information System at WWU from 2005 to 2008, where he is currently a PhD candidate. His current research interests include Web 2.0, Folksonomies, and Web Services.

**Carolin Letz** received her Diploma Degree in Mathematics in 2001 and her PhD in Computer Science in 2008 from the University of Muenster (WWU),

TABLE 1  
OVERVIEW OF WEB SERVICE DISCOVERY RESOURCES (AS OF MAY 8, 2007).

Name	Search	Number of services/ change	Availability information	Alternative interfaces	Test/ demo	WSDL parser	Terms of use/ price	Community support	Service usage (mash-ups)
UDDI Business Registry	search, categories	-	no	SOAP	no	No	no	-	no
	Offline								
XMethods	list	3 124	no	SOAP, UDDI, WS-Inspection, DISCO, RSS	yes	Yes	no	-	yes
	~	483 500 (-85%) (-84%)	~	~	~	~	~	~	~
Bindingpoint	search, categories	4 056	imprecise	-	yes	Yes	yes	ratings, reviews	?
	offline								
WebserviceX	search, categories	70	no	RSS	example SOAP docs	yes	no	-	no
	~	71 71 (+1%) (+1%)	~	~	~	~	~	~	~
Web Service List	search, categories	1 000	no	RSS	no	no	no	-	no
	~	484 280 (-52%) (-68%)	~	~	partial	~	~	~	~
StrikeIron	search, categories	1 000	yes	SOAP	yes	yes	yes	-	no
	~	198 70 (-80%) (-93%)	~	-	~	~	~	~	~
Woogle	search, categories, templates	885	yes	-	partial	imprecise	no	-	?
	offline								
Remote-Methods	list, categories	-	no	-	no	no	no	ratings, reviews	no
	~	396 372 (-) (-)	~	~	~	~	~	~	~
WSDLicious	list, categories	45	no	-	no	no	no	-	no
	~	offline 63 (+40%)	~	~	~	~	~	~	~
ProgrammableWeb	search, list, categories	86	no	RSS	no	no	yes	blog, ratings, comments, forum, developer list, examples	yes
	~	744 429 (+865%)(+500%)	~	~	~	~	~	~	~
Google	search	25 600	imprecise	SOAP	no	no	no	-	no
	~	15.400 34 100 (-40%) (+25%)	~	Ajax	~	~	~	~	~