IDEA GROUP PUBLISHING
701 E. Chocolate Avenue, Suite 200, Hershey PA 17033-1240, USA **ITB12880**
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com

This paper appears in the book, *Emerging Trends and Challenges in Information Technology Management, Volume 1 and Volume 2* edited by Mehdi Khosrow-Pour © 2006, Idea Group Inc.

Software Risk Mitigation through Web Services

Daniel J. Hinz, J. W. Goethe University, E-Finance Lab, Mertonstrasse 17, 60325 Frankfurt/Main, Germany,
P: +49 69 7162-5354 (-5355), dhinz@wiwi.uni-frankfurt.de

ABSTRACT
Web services are praised by research and practitioners as a powerful device to integrate business processes even across firm boundaries. However, sometimes even apparently simple operations are not ideally suited to be implemented by web services. This paper explains potential business factors that can restrain web services from unfolding their full potential. Based on a simple causal model for software risk this paper then analyzes why web services may be employed as broadly as possible even in untypical usage scenarios, as they may capture additional benefits from better software risk mitigation.

1. INTRODUCTION
During the last years web services have received much attention in the research community. They are best suited in scenarios with many using applications in number and diversity and when providing widely useful services. In contrast they lose a lot of their attractiveness in usage scenarios, where widespread and easy access to a service is not the deciding factor, e.g. for highly specialized tasks or for low transaction volumes. Also in traffic-intense scenarios web services often cannot approximate the communication effectiveness of a proprietary interface [8]. Is this a reason not to employ web services in these scenarios? This paper shows, how and why web services should be preferably employed even in untypical usage scenarios to capture additional benefits from improved software risk mitigation.

2. EXPLAINING THE SUCCESS
A common definition for web services is "a software application on a network, that has an interface through which other programs can access it" [6]. Cornerstone of this approach is the interface, which has to be highly standardized to enable this ubiquitous accessibility needed for the decoupling of business logic from applications and of servers from clients. In practice, this standardization has to be threefold in data, processes, and communication [4].

Therefore, web services take their advantage of usage scenarios with a broad user spectrum, either in quantity or in diversity. The more applications are accessing a web service, or the more diverse the application group is, the more a web service is the model of choice compared to a local application or a proprietary interface. A frequently cited example is a flight booking system in a travel agency [3, 4]. Without web services, a travel agent would have to query databases from multiple airlines, while web services allow the use of one interface that simultaneously submits standardized queries to the airline systems.

Public examples of web services usually fulfill this condition. Amazon offers a set of web services to allow thousands of third-party sellers and vendors to "easily manage large quantities of inventory on our platform and download the latest product information to make sure that their products are competitively priced" (http://www.amazon.com/gp/browse.html/ref=smm_sn_aws/002-5192368-2746463?%5Fencoding=UTF8&node=3435361). Another popular example is the web service of Google. After obtaining a license key, "software developers can query more than 8 billion web pages directly from their own computer programs" (http://www.google.com/apis/).

These examples share the characteristic that they are valuable to a wide range of users, sometimes across thousands of enterprises and even for private use. The promise seems to be fulfilled that "traditional IT infrastructures in which infrastructure and applications were managed and owned by one enterprise are giving way to networks of applications owned and managed by many business partners" [5]. And also within the boundaries of single enterprises service-oriented architectures (SOA) are evolving as state of the art.

3. RUNNING INTO LIMITATIONS
However, obviously not all tasks within an enterprise or especially not across firm boundaries are sufficiently standardized so that web services can live up to their full power.

The challenge to successfully deploy web services or a whole service-oriented architecture is primarily not a technical one, but grounded in business factors. A recent empirical study in 2005 by Anderson et al. showed that business factors and even methodological factors drive a successful web services strategy more than technological factors do [1]. Or, as Hammer (2001) put it: "one should not assume that the architectural elegance of the proposed solution to inter-enterprise collaboration reduces the complexity of implementing and maintaining that functionality" [7].

4. IMPLEMENTING WEB SERVICES AS RISK MITIGATION MEASURES
Are these limitations a reason not to implement web services in these cases? This section introduces a simple causal model for the effects of software standardization, complexity, and maturity on the functioning of software. For each of the major influence factors (parent nodes) the effects of the employment of web services are described.

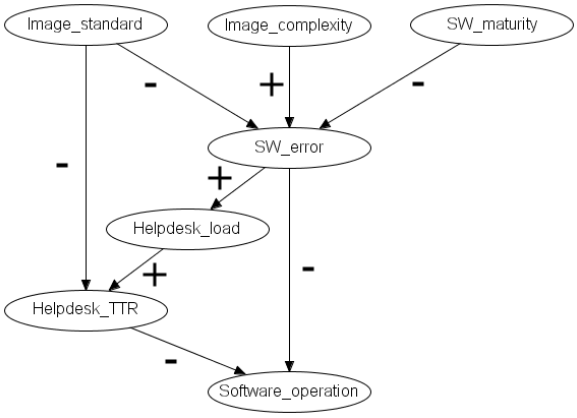
4.1. Causal dependencies for software risk
The measuring objective of this model (see Figure 1) is software uptime of an end user computer, i.e. the time the software of a computer can be used normally (represented by the node "Software_operation").

The software of a system becomes inoperative, when a software error occurs ("SW_error"), and it remains in that state, until the helpdesk can resolve the problem. The downtime depends on the actual helpdesk time to resolution ("Helpdesk_TTR"), a longer time to resolution therefore has a negative effect on software uptime, indicated by the "-" in Figure 1. In the same way an increased number of software errors reduces software uptime, both directly and indirectly by increasing the helpdesk load ("Helpdesk_load") and therefore increasing the time to resolution. This is positive dependency is indicated with "+".

The three main influence factors are the adherence to software standards, the complexity of a software image, and the maturity of the installed software (nodes "Image_standard", "Image_complexity", and "SW_maturity").

Usually the software in enterprises is grouped into profiles or images, to allow for automated set-up of new workstations and to ensure mutual compatibility of the applications in one image. Any deviation from that

Figure 1. Causal model of software stability



standard, e.g. be installing additional software, significantly increases the chance of software errors on the system. This can be due to compatibility issues, but also due to execution of malicious code. In addition, debugging attempts of a non-standard system is much more time-consuming, as standard debugging procedures may not help. While it is technically easy to enforce the adherence to image standards, e.g. by depriving end users of local administrator rights, staff productivity can decline if justifiably needed applications cannot be installed.

Another influence factor is the complexity within an image. The more applications have to run on one computer, the higher is the chance that they will not harmonize. This can be mitigated by thorough testing of the image, but for example the installation of patches for individual applications always poses a risk for the whole system.

The last major influence factor is software maturity. Applications tend to approach perfection, the longer they are in service. In contrast, new applications tend to have teething troubles [2].

4.2. Effects of web services on software risk

Although the influence factors described above are of a general type and will be always an issue, the employment of web services can have a positive effect on all of them thus mitigating software risk.

Web services can help to solve the quandary that users should not be allowed to alter the configuration of their computer systems, but at the same time have to get access to any application they need for their work. If an application is available via web service, it can be easily integrated by the end user in his application environment without the need to install additional software. Furthermore, the threat of malicious code can be decreased, as web services only return XML information.

In the same way, web services can help to reduce the complexity of an image. Without web services, each application has to include the capability to perform its tasks either locally, or it needs a more or less proprietary interface to applications or data sources on the network. Both burdens a system with additional complexity and increases the chance for mutual incompatibilities. To query a customer database in an international logistics company for example, one application needed a locally installed Oracle Client Version 9, while an older application still relied on Version 8. Although theoretically compatible, this setting caused major difficulties. Similar problems can arise with other database query interfaces like ODBC, Jet, etc. and for example Java Runtime Environment versions. By employing web services, one common standard is used which can widely replace local database interfaces and application logic.

In addition, even on software maturity there is a positive effect by web services. Naturally, web services also tend to be immature on initial release and only improve over time. Moreover, newly developed or adapted local applications employing web services may be unstable as well. But by relieving applications from implementing basic functionalities over and over again, there is simply less room for bugs. It is much easier to ensure correct and high quality execution of one central service than of dozens of local applications.

All three influence factors can benefit from the use of web services. It is not necessary to transfer all applications at once to the service architecture, as this would be hardly possible in a larger enterprise. Each additional web service contributes to the improvement of the three influence factors, thereby generating business value in relationship to the effort put into the new architecture.

5. CONCLUSIONS

While web services are a theoretically well-established methodology, in real world scenarios web services may not unfold their full potential. Often business factors are a reason, why major benefits of web services cannot be realized. Although this is not at all a weakness of the web service concept itself, it may be a reason why web services are not introduced. However, it is shown that even in adverse scenarios, which would not call for the use of web services, they might be employed as broadly as possible to capture additional benefits from better software risk mitigation, as they enforce the use of standards, reduce software complexity, and increase application maturity.

6. REFERENCES

[1] Anderson, D., Howell-Barber, H., Hill, J., Javed, N., Lawler, J., and Li, Z.; A Study of Web Services Projects in the Financial Services Industry, Information Systems Management, (22:1), 2005, pp. 66-76.

[2] Barlow, R.E. and Singpurwalla, N.D.; Assessing the Reliability of Computer Software and Computer Networks: An Opportunity for Partnership with Computer Science, The American Statistician, (39:2), 1985, pp. 88-94.

[3] Benatallah, B., Sheng, Q.Z., and Dumas, M.; The Self-Serv Environment for Web Services Composition, IEEE Internet Computing, (7:1), 2003, pp. 40-48.

[4] Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., and Weerawarana, S.; Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI, IEEE Internet Computing, (6:2), 2002, pp. 86-93.

[5] Curbera, F., Khalaf, R., Mukhi, N., Tai, S., and Weerawarana, S.; The Next Step in Web Services, Communications of the ACM, (46:10), 2003, pp. 29-34.

[6] Geerts, G.L., Paretta, R.L., and White, J., Clinton E.; An Introduction to Web Services, The CPA Journal, (LXXIV:8), 2004, pp. 70-71.

[7] Hammer, K.; Web Services and Enterprise Integration, Business Integration Journal, (November), 2001, pp. 12-15.

[8] Menascé, D.A.; QoS Issues in Web Services, IEEE Internet Computing, (6:6), 2002, pp. 72-75.