Journal of Universal Computer Science, vol. 18, no. 11 (2012), 1405-1409
submitted: 1/5/12, accepted: 15/5/12, appeared: 1/6/12

Some Reflections about Service Oriented Architectures, Cloud Computing Applications, Services and Interoperability

J.UCS Special Issue

Francisco J. García-Peñalvo
(Computer Science Depart. / Science Education Research Institute / GRIAL Research Group
University of Salamanca, Salamanca, Spain
fgarcia@usal.es)

Marc Alier Forment
(UPC Universitat Politècnica de Catalunya Barcelona-Tech, Barcelona, Spain
marc.alier@upc.edu)

Miltiadis D. Lytras
(The American College of Greece, Gerakas, Greece
miltiadis.lytras@gmail.com)

Abstract: Web information systems are at a point where certain technologies and practices have reached a mature level of development and adoption by the industry. These information systems are widespread in institutions with different contexts and scopes, but in a global world, these systems have interoperation needs through standards that define ways to distribute high quality contents or services. On the other hand, there is an increasing number of new web technologies, tools and kinds of contents, devices and services that thrive outside the world of institutions. These have to be combined with new models of distributing, delivering and accessing services: Open Resources, Open Source Software and Creative Commons Licenses, and even the "freemium" business models, which companies like Google, Layers, Prezzi or Twitter have adopted, presenting a deep impact on the panorama of Internet services. This special issue is devoted to the problem of how to integrate service based tools and contents, mobile applications and cloud computing services into the legacy systems. Systems that institutions have been implementing and putting to use with loss of effort and internal transformations; now realizing that their systems are limited, closed and far away from the crest of the wave of systems innovation. This is a problem whose solution seems to lie in the domain of Service Sciences, both from a technical point of view of SOA and interoperability standards, and from a point of view of business models, ways of collaboration inside software development teams and fair licensing strategies. This special issue includes experiences and perspectives organized in 9 papers.

Keywords: Interoperability; Software Services; Cloud Computing; SOA; Information Systems
Categories: D.0, D.2, H.0, H.4, K.4, K.6

1 Introduction

Software has been in constant evolution since its very beginnings in the 1940's, when software was a mere configuration of the wiring in the ENIAC Computer – which was not even a Von Newman machine with the program stored in memory – and the wiring duty was left for the women while male electrical engineers worked on the hardware design and building [McCartney 99]. Software has been evolving in many dimensions: in the tools and programming languages we use to create software; in the ways we approach the creation process; in the models and metaphors that describe the software elements and their relationships with their counterparts in the real (or virtual) world; in its complexity and boundaries, from little programs of hundreds of lines running in one computer to huge distributed systems interoperating with other software, organizations and individuals; in the role names we give to its creators from programmers to analysts, architects, developers and other names like "ninjas" or even "jedis" [McMillan 12] and, last but not least, in the ways software is daily becoming present in our lives, improving and transforming the business processes too.

This evolution in the short history of software has not been steady. In the first 30 years of the history of software – and modern computer science for that matter – there was a rapid pace of innovation where most of modern operating systems and software development techniques were invented and disseminated among the whole Information Technology (IT) community. In those days, the size – in bytes – of software was relatively small. A software developer could make sense of a program binary code, analyse it and learn from it. Also, file data formats could be easily reverse-engineered [Ceruzzi 03].

The invention of the microchip by Intel prompted that software become more complex, it begun to be regarded as a product in itself (the first software companies were founded) and started to be regarded as pieces of intellectual property protected by copyrights and end user license agreements. While these software protection policies facilitated the birth of the software industry, it came at high costs.

The evolution process of hardware, powered by Moore's Law, made computers powerful and more accessible [Schaller 97]. Software needed to cover increasing new features and applications. Software houses kept their code and innovations as company secrets; while the software industry was immersed in what Pressman [Pressman 97] called an Endemic Crisis, where high development costs and poor quality may be something that belongs to the complex nature of software itself.

A well-known example of this holding back of innovation is the graphic user interface and the object oriented programming, invented in the 1970's and kept secret at the Xerox Palo Alto Research Center (XPARC) until Steve Jobs "stole" it and made it mainstream with the Mac and Next computers [Isaacson 11].

But somehow, today software industry is thriving and the impact of software on economic and social life is pervasive and ubiquitous. What broke this holding back on innovation? Let's remember that it took Microsoft 11 years to make the features of the Mac function with Windows 95. The reasons for this had to do with piracy or were maybe due to the fact that Microsoft used an innovation model based on keeping everything secret and protected by licenses instead.

It looks as if it is not accidently that today's main Internet server infrastructure is built on the Linux operating system. Created by a teenager Linux Torvalds using a textbook on operating systems as starting point, Linux source code was made public from day zero, and a community of worldwide developers started working together and sharing ideas and code. The consequence was that in very few years a solid server platform was implemented. In the late 1990's IBM dropped the development of OS/2, the operating system developed in alliance with Microsoft, to embrace Linux as a centre piece of its new strategy. This is a good example of open innovation [García-Peñalvo *et al*. 10].

Tim Berners Lee made a choice of publishing the World Wide Web in an open and standardized RFC (Request For Comments). It allowed for an easy and rapid adoption of the web. Companies and open source projects built servers, extensions, browsers and scripting languages. Projects like Apache and products like Netscape navigator (later open sourced as Mozilla) allowed for a fast adoption of the web. A demand for better network access arose and a critical mass of users calling for innovations like search engines, e-commerce, e-learning, etc., was created [Berners Lee 93].

The success of the World Wide Web lies in its reliance on simple, open, standardized protocols and languages that enable interoperability at many levels. Web apps can be created in almost any platform and accessed from any browser and any operating system. Only when browser interoperability has been compromised by commercial interest (browser wars), it has meant trouble for developers and software industry in general.

The universal adoption of HTTP protocol and the introduction of XML triggered a widespread adoption of web services (sometimes under a defined Service Oriented Architecture – SOA – strategy, sometimes as hacks or workaround to get things done) providing a common ground for distributed software interoperability [García-Peñalvo *et al*. 11].

Thanks to the open web standards and interoperability, today's web information systems are at a point where certain technologies and practices have reached a mature level of development and adoption by the industry. These information systems are widespread in institutions with different contexts and scopes, but in a global world these systems have interoperability needs that must be supported by standards that define ways to distribute high quality contents or services.

However, there is also an increasing number of new web technologies, tools and kinds of contents, devices and services that thrive outside of the world of institutions. These should be combined with new models of distributing, delivering and accessing services which companies such as Google, Layers, Prezzi or Twitter have adopted, presenting a deep impact in the panorama of Internet services.

This special issue is devoted to the problem of how to integrate service based tools and contents, mobile applications and cloud computing services into the legacy systems. Systems that institutions have been implementing and putting to use with loss of effort and internal transformations; and that now realize that their systems are limited, closed and far away from the crest of the wave on systems innovation. This is a problem that seems to have a solution in the domain of Service Sciences, both from

a technical point of view of SOA and interoperability standards, and from a point of view of business models, ways of collaboration and fair licensing strategies.

A new wave of innovation in software is taking place, based on open simple standards that we can understand and innovate upon.

2 Content of the special issue

Pascual *et al*. open this special issue presenting a lightweight approach to use context information in conventional web applications. Also, the authors propose a web browser that is responsible for processing the XML tags and sending the context information to the web application using web services.

Griffiths *et al.* propose a case of study of the Apache Wookie (incubating) technology. They want understand the user from a sociological point of view. The complexity of the relationship between the context of use and user needs, and the feedback loops between them is discussed, and the role of technological interventions as an element in a discourse is considered.

The orchestration of learning services is tackled by Quierós and Leal in their contribution. They propose a standard based approach with a pivot component to orchestrate the interaction among all learning components involved in an automatic evaluation of programming exercises system. Also in the learning context Alier *et al.* present an integration proposal that uses the IMS Basic Learning Tools Interoperability (IMS BLTI) standard in order to turn Google Docs into an engine that powers collaborative learning activities inside the LMS Moodle.

Joha and Janssen deal with Software as a Service (SaaS) approach but use a user organization perspective instead of the more traditional service provider one. Based on the case studies presented, they have identified eight discriminating design choices that are important when designing SaaS business models. These include the (1) SaaS service characteristics, (2) SaaS value source, (3) SaaS user target group, (4) data architecture configuration and tenancy model, (5) SaaS governance and demand/supply management core competencies, (6) cloud deployment model, (7) SaaS integration and provider strategy and, finally, the (8) SaaS pricing structure.

Cloud computing is a paradigm shift in computing with the potential of changing the whole perspective with which we look at computing today. Kim's paper is devoted to a study of quality of service in a cloud social networking paper. The author makes real-time estimation and analysis of the parameters in a web server for service-based content delivery, the utilization level and serviceability as key elements for cloud computing services in its different forms, i.e., Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). On the other hand, Colomo-Palacio *et al.* provide an overview of the lessons learned and the issues that emerged from a project aimed at adapting a real software solution to adopt a cloud computing approach. The technology evolution and adaptation processes to cloud computing are very common in different kinds of companies, including software vendors, but they are complex issues full of problems and for this reason this paper presents a highly valuable experience.

Milovanović *et al.* propose an interoperability framework for biometric solutions that accelerate future development of biometric solutions. The aim of their

contribution is to allow using a multimodal approach that combines several biometrics modalities in order to improve preciseness and performances of the final systems, independently of the interoperability issues among existing biometric solutions, acquisition devices and databases.

Finally, the data grounding issue is tackled by García *et al.*. Grounding is the process in charge of linking requests and responses of web services with the semantic web services execution platform, and it is the key activity to automate their execution in a real business environment. The authors introduce a practical solution for data grounding based on a mapping language to relate data structures from services definition in WSDL documents to concepts, properties and instances of a business domain and a set of functions that perform the lowering and lifting processes using the mapped specifications.

Acknowledgements

This work has been partially supported by the project "oiPLE: Open Integrated Personal Learning Environment" (reference number TIN2010-21695-C02).

References

[Berners-Lee 93] Berners-Lee, T. J., Cailliau, R., Pellow, N.: "The World Wide Web initiative"; Proc. of INET'93, Internet Society, San Francisco, USA (1993).

[Ceruzzi 03] Ceruzzi, P. E.: "A history of modern computing"; 2nd edition, MIT Press, Cambridge, Massachussets (2003).

[García-Peñalvo *et al.* 10] García-Peñalvo, F. J., García de Figuerola, C., Merlo, J. A.: "Open knowledge: Challenges and facts"; Online Information Review, 34, 4 (2010), 520-539.

[García-Peñalvo *et al.* 11] García-Peñalvo, F. J., Conde, M. Á., Alier, M., Casany, Mª J.: "Opening Learning Management Systems to Personal Learning Environments"; Journal of Universal Computer Science, 17, 9, (2011), 1222-1240.

[Isaacson 11] Isaacson, W.: "Steve Jobs"; Simon & Schuster, New York (2011).

[McMillan 12] McMillan, R.: "Looking for a coding job? Better a ninja than a brogrammer"; Wired Enterprise (2012) http://www.wired.com/wiredenterprise/2012/05/ninja_job/.

[McCartney 99] McCartney, S.: "ENIAC: The triumphs and tragedies of the world's first computer"; Walker, New York (1999).

[Pressman 97] Pressman, R. S.: "Software engineering: A practitioner's approach"; 4th edition, McGraw Hill, New York (1997).

[Schaller 97] Schaller, R. R.: "Moore's law: Past, present and future"; IEEE Spectrum, 34, 6 (1997), 52-59.