ComposableWeb 2009
Lightweight integration on the Web

Proceedings of the

First International Workshop on Lightweight Integration on the Web (ComposableWeb'09)

Florian Daniel[1], Sven Casteleyn[2], Geert-Jan Houben[3]

[1]University of Trento, Italy
daniel@disi.unitn.it

[2] Vrije Universiteit Brussel, Belgium
sven.casteleyn@vub.ac.be

[3] TU Delft, Netherlands
g.j.p.m.houben@tudelft.nl

Preface

While the word mashup is widely used today, to some of us it is still not really clear what a mashup is and what it is not. Some mashups focus on integrating RSS feeds, others on integrating RESTful services, SOAP services, Atom feeds, or user interfaces. Yet, everybody recognizes that mashups represent a new way of expressing innovation, sometimes even *user innovation*, i.e., innovation in the form of simple web applications "implemented" or "mashed up" by web users.

Typically, implementing a mashup means *integrating* resources available on the Web into a new, value-adding application. The integration may occur at the user interface level (most mashups do integrate presentation content, not just data), at the application logic level (web service are one of the cornerstones of mashups), or at the data level (RSS/Atoms feeds or XML files are common practices today), or at a combination of them. Therefore, we say a *mashup* is a web application that is developed by composing data, application logic, and/or user interfaces originating from disparate sources available on the Web.

In general, developing mashups is a hard and tricky task. For instance, a mashup composer can of course use any conventional *programming language* to integrate, i.e., mash up, the resources and components of his choice. For instance, among the most used languages today we find PHP on the server side and JavaScript on the client side; many other languages are used as well. Given the heterogeneity of technologies, programming languages, interaction protocols, the complexity of the necessary integration logic, and similar, *manual development* of mashups is only an option for highly skilled programmers. And even they could experience a hard time in mastering all the development challenges. Service composition approaches "a la BPEL" are not able to cope with the above mentioned heterogeneity of technologies.

With the advent of *mashup tools*, the mashup phenomenon has however become more popular even among web users, as they offer users the unique chance of getting involved into the development process of web applications in an intuitive and *assisted* fashion. Usually, mashup tools aim at simplicity more than at completeness of features, and they support fairly sophisticated development tasks in the browser.

Despite the current emphasis on mashups and lightweight integration on the Web, we still register a lack of agreed-upon reference models (e.g., for component and composition languages), development processes and methodologies, architectures, execution platforms, analysis techniques, and so on, that effectively aid mashup development. In particular, the involvement of web users into the development of composite online applications demands intelligent concepts and a high degree of assistance – especially to the most inexperienced users. It even results in new (social) development practices, which in turn may require new software support.

In this context, several challenging research issues are emerging, among which the following seem of particular interest:

1. *Reusable components*: Expressive component models for data, application logic, and user interface components, as well as suitable description languages, and discovery and selection facilities (e.g., registries and protocols) are needed.

2. Simple, *lightweight composition languages*: Easy-to-learn yet expressive execution languages are required, which enable the plug-in style development of composite applications.
3. *Graphical composition tools*: Composition languages should be equipped with suitable graphical modeling formalisms that are able to hide the actual composition complexity and allow for computer-aided development environments.
4. *Suitable execution platforms*: Ready compositions require proper execution support (e.g., an interpreter or parser). We expect such support to be provided through online hosting and execution platforms.
5. Design aimed at *interoperability*: Components and mashups should be interoperable, meaning that they have cross-platform reusability. Mashup-specific standards might be necessary.

In light of these considerations, the goal of ComposableWeb is to challenge the Web Engineering community with these new research issues and to stimulate the discussion of key issues, approaches, open problems, innovative applications, and trends in these and related research areas, so as to identify technologies, solutions, instruments and methodologies that effectively support the lightweight composition of web applications.

This volume collects the proceedings of the first edition of ComposableWeb, held in conjunction with the International Conference on Web Engineering (ICWE) on June 23, 2009, in San Sebastian, Spain. Out of all submissions, eight papers (six full papers, two position papers) attained sufficient quality and were selected for presentation during the workshop; unfortunately, other submissions could not be accepted.

We would like to thank all authors, of both accepted and rejected papers, for their contributions, the presenters and the people participating in the workshop for their engagement and contributions to the discussions during the workshop, and the chairs of ICWE'09 for their support in the organization of the event.

Trento, Brussels, Delft
June 2009

Florian Daniel
Sven Casteleyn
Geert-Jan Houben

Organization

Organizing Committee

– Florian Daniel, University of Trento, Italy
– Sven Casteleyn, Vrije Universiteit Brussel, Belgium
– Geert-Jan Houben, TU Delft, the Netherlands

Steering Committee

– Sven Casteleyn, Vrije Universiteit Brussel, Belgium
– Florian Daniel, University of Trento, Italy
– Maristella Matera, Politecnico di Milano, Italy
– Geert-Jan Houben, TU Delft, the Netherlands
– Olga De Troyer, Vrije Universiteit Brussel, Belgium

Programme Committee

– Sören Auer, University of Leipzig, Germany
– Boualem Benatallah, University of New South Wales, Australia
– Fabio Casati, University of Trento, Italy
– Peter Dolog, Aalborg University, Denmark
– Marlon Dumas, Tartu Univesity, Estonia
– Schahram Dustdar, Technical University of Vienna, Austria
– Rama Gurram, SAP Labs Palo Alto, USA
– Frank Leymann, University Stuttgart, Germany
– Michael Mrissa, University of Lyon, France
– Moira C. Norrie, ETH Zurich, Switzerland
– Cesare Pautasso, University of Lugano, Switzerland
– Gustavo Rossi, Universidad Nacional de La Plata, Argentina
– Takehiro Tokuda, Tokyo Institute of Technology, Japan

Table of Contents