Dagstuhl Seminar 07051
Programming Paradigms for the Web: Web Programming and Web Services
28.01.2007–02.02.2007
Executive Summary

Rick Hull[1], Peter Thiemann[2], and Philip Wadler[3] (editors)

[1] Bell Labs, Lucent Technologies
600 Mountain Ave., 2D-510, Murray Hill, NJ 07974, USA
`hull@alcatel-lucent.com`

[2] Albert-Ludwigs-Universität Freiburg, Institut für Informatik
Georges-Köhler-Allee 079, 79110 Freiburg, Germany
`thiemann@informatik.uni-freiburg.de`

[3] University of Edinburgh, Department of Computer Science
James Clerk Maxwell Building, The King's Buildings, Mayfield Road, Edinburgh
EH9 3JZ, Scotland
`wadler@inf.ed.ac.uk`

Abstract. The world-wide web raises a variety of new programming challenges. To name a few: programming at the level of the web browser, data-centric approaches, and attempts to automatically discover and compose web services. This seminar brought together researchers from the web programming and web services communities and strove to engage them in communication with each other. The seminar was held in an unusual style, in a mixture of short presentations and in-depth discussion sessions in small groups. This style enabled the participants to identify and discuss burning questions in small birds-of-a-feather sessions as well as in large plenary sessions. It required active participation of all attendees.

Keywords. Web programming, web services, programming paradigms, analysis and verification, implementation techniques and optimizations

1 Introduction

The web raises a variety of new programming challenges. To name a few: programming user interfaces at the level of the web browser, data-centric approaches, and attempts to automatically discover and compose web services. This seminar brought together researchers from the web programming and web services communities. Both groups had much to learn from each other, and the focus on programming paradigms was a useful perspective on the diverse web community.

"Web (application) programming" describes writing software ("web applications") that relies on a web browser as its user interface. Typical tasks for web programming include the generation of dynamic web pages, accessing databases, querying web services, and dealing with concurrency. Web applications often involve many "tiers", each of which is a homogenous level of software which interacts over a network with other tiers; a typical application might involve, for example, a client (a web browser programmed with JavaScript, VBScript, or Flash), a server (programmed with Java, Ruby, PHP, Python, or Perl), and a database (programmed with SQL or XQuery); connecting these tiers can be a key challenge for web programmers. As another challenge, web applications do not presently support modes of user interaction as rich as their desktop counterparts, and providing anything other than the simplest form-based interaction can be a great difficulty for programmers—thus, we asked, how can web programming paradigms support coding rich user interfaces?

"Web services", by contrast, are programs that interact primarily with other software systems using web technologies. The web-services paradigm, which might be viewed as an instance of the Service-Oriented Architecture (SOA), provides both a framework and specific interfaces (e.g. SOAP, WSDL) for a new generation of distributed software. The paradigm provides for rich flexibility in creating services that use other web services. To date, programming of web services has focused largely on adaptations of workflow approaches to a peer-to-peer framework, and the Business Process Execution Language (BPEL) has emerged as the industrial programming language of choice. There has also been significant research on "semantic web services", which provide explicit mechanisms to represent and reason about the impact of services on the world, as well as their messaging and internal behavior. Frameworks such as WSDL-S, OWL-S, SWSO, and WSMO may provide the basis for programming paradigms to work effectively in this context.

Prime discussion topics were: the application of these techniques to web applications, browser-based programs, and web services, programming languages for the web, scripting, authoring, type checking, databases, web service semantics, service composition, process and data flow, XML and other data manipulation, concurrency, sessions and transactions, performance, and scalability.

To maintain a focus on programming, speakers were asked to center their talk on actual code that illustrates their research. Here 'code' was broadly interpreted to include a program in a programming language, a formula of logic, a specification, or a query.

As an outcome of the seminar we expected to understand better the interplay between the various styles of programming for the web, along with proposals towards a more unified approach to such programming. Elsewhere in this volume, we have started to compose a list of the key scientific challenges (or at least discussion items leading in that direction) in this domain.

2 Participation and Program

The seminar brought together 37 researchers: 23 from Europe, 9 from America, 2 from Israel, and 2 from Japan. The seminar was held in an unusual style, in a mixture of short presentations and in-depth discussion sessions in small groups. This style enabled the participants to identify and discuss burning questions in birds-of-a-feather sessions as well as in plenary sessions. It required active participation of all attendees.

All participants were asked before the seminar to prepare a couple of slides with what they regarded as the burning questions of the field. On the first day, everyone presented their burning questions in a plenary session. From this, we extracted five broad topics for working groups:

- patterns and paradigms
- web services
- data on the web
- software engineering
- security

Participants freely chose amongst the working groups, which then broke out to discuss the burning questions by topic and to brainstorm for further ideas and connections. Through plenary sessions and additional theme-specific breakout groups, we further deepened the topics. Interspersed with these group work sessions, some participants contributed lecture-style talks.

At the end of the seminar, each working group presented its findings.

During the seminar, a wiki[1] was used to keep track of the daily schedule, to manage burning questions, and to collect the notes from the different working groups. It enabled easy collaboration between the different working groups; it served as a basis for presenting the outcomes of the breakout sessions in the plenary sessions; and it was extremly helpful in compiling the results of the seminar. In theory, all attendees of the seminar could have participated in working on their presentation page, though only a few chose to do so.

The editors would like to thank all of the participants for their thoughtful input and careful discussion: Serge Abiteboul, Nick Benton, Ezra Cooper, Daniel Deutch, Susan Eisenbach, John H. Field, Christophe Fouqueré, Alain Frisch, Martin Gasbichler, Michael Gruninger, Haruo Hosoya, Richard Hull, Dean Jacobs, Trevor Jim, Shriram Krishnamurthi, Niels Lohmann, Florian Loitsch, Bogdan-Eugen Marinoiu, Florian Matthes, Jay McCarthy, Tova Milo, Yasuhiko Minamide, Tom Murphy, Anders Møller, Matthias Neubauer, Barry Norton, Peter Patel-Schneider, Matthias Radestock, Mukund Raghavachari, Helmut Seidl, Manuel Serrano, Bertrand Souville, Jianwen Su, Peter Thiemann, Philip Wadler, Stefan Wehr and Jeremy Yallop.

[1] see https://proglang.informatik.uni-freiburg.de/cgi/wiki/proglang/extern/wiki.py/Dagstuhl07051

3 Conclusions

The meeting was very productive; it provoked many new ideas and provided new perspectives on the topic. The participants learned a lot from each other—in particular, there was a lively exchange of knowledge between the programming-languages and the database communities. We hope to further consolidate the results in an article that provides research directions for web programming and related areas.