FORWARD: Design Specification Techniques for Do-It-Yourself Application Platforms

Gaurav Bhatia
UC San Diego
gbhatia@cs.ucsd.edu

Yupeng Fu
UC San Diego
yupeng@cs.ucsd.edu

Keith Kowalczykowski
app2you, Inc.
keith@app2you.com

Kian Win Ong
UC San Diego
kianwin@cs.ucsd.edu

Kevin Keliang Zhao
UC San Diego
kezhao@cs.ucsd.edu

Alin Deutsch
UC San Diego
deutsch@cs.ucsd.edu

Yannis Papakonstantinou
UC San Diego
yannis@cs.ucsd.edu

ABSTRACT

The FORWARD project, as well as its app2you predecessor, is a Do-It-Yourself web application platform that enables non-programmers to create custom forms-driven workflow applications, where users with potentially different roles and rights interact. Our experience with the commercialized version of app2you (app2you.com) shows that forms-driven workflow applications exhibit a viable balance between application scope and ease of specification. The proposed demo uses a real-world application, the TechCrunch50 2008 reviewing application, to illustrate key specification techniques the Do-It-Yourself design facility provides to non-programmer application creators. The demo audience will also have hands-on experience building applications without any programming.

1. DIY APPLICATION PLATFORMS

The app2you and FORWARD projects [2, 3] provide Do-It-Yourself (DIY) application platforms that enable non-programmers to rapidly create and evolve forms-driven workflow applications customized to their data and process needs. A DIY platform provides an *application design facility* (also called *application specification mechanism*) where a non-programmer application creator can specify the application by manipulating visible aspects of the application's pages or by setting configuration options. A simple, back-in-the-80s example of DIY application creation is form builders, where the application creator introduces form elements into a form page. The platform, in response, creates a corresponding database schema. Actions on the form lead to SQL INSERT, UPDATE and DELETE statements.

A DIY platform must maximize two metrics:

- *Application scope*, which is characterized by the computation, process collaboration, and presentation (pages) that can be achieved by applications specified using the platform's design facility.
- *Ease of specification* of applications using the platform's design facility. When ease increases, a larger number of less technically sophisticated creators are empowered to create applications.

There is an inherent tension between scope and ease of specification; one can be traded against the other. At the one extreme, building applications using Java, Ajax and SQL provides unlimited scope, but is not easy since the application creator must know programming, database design and spend significant time coding. Platforms such as Ruby on Rails and WebML [1] make specification easier and faster, but still not easy enough to enable non-programmers. At the other extreme, creating an application by copying an application template, as done for example in Ning, is very easy but the scope of the platform is limited to Ning's finite number of templates. DIY application platforms fall between these two extremes.

app2you and FORWARD focus on *database-powered* web applications, where the applications' entire state is captured by the database as opposed to also having out-of-database state that is stored in memory or external systems. Depending on the state of the database, each user has rights to access certain pages, read certain records in them and execute certain actions. Database-powered web applications present an excellent scope vs. ease tradeoff point [2, 3]: the demonstrated DIY design facility enables such applications to be built by non-technical business process owners without involving any programming or database design.

The purpose of the demo is to show in action the ease of specification by which fairly complex forms-driven workflow applications can be built. A number of techniques employed by the design facility for the easy specification of applications are demonstrated.

2. DEMO USE CASE

To illustrate the techniques employed by the design facility, we build a simplified version of a real-world app2you.com application, the TechCrunch50 (TC50) 2008 conference reviewing process. Figure 1 visualizes the workflow of the application, where about 1000 startups submitted business plans along with information packages, in order to present themselves and their products. The app2you.com application was used to collect the submitted business plans, have them commented and graded by the TC50 committee,

Twelfth International Workshop on the Web and Databases (WebDB 2009), June 28, 2009, Providence, Rhode Island, USA. .

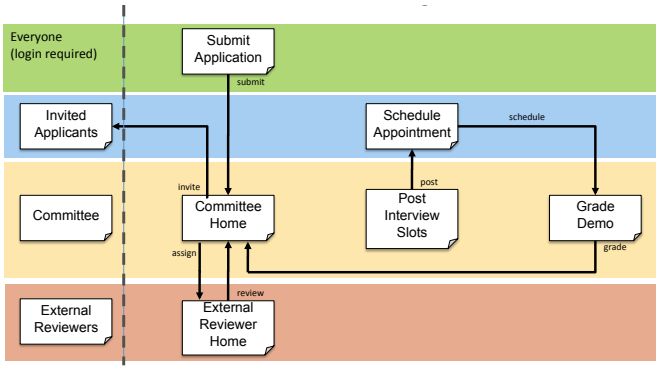

Figure 1: TechCrunch50 2008 Reviewing Process

schedule multiple rounds of appointments where selected candidates met the committee online to demo their products, occasionally route business plans to external reviewers, and select the 50 candidates to present at the conference.

The demo is a showcase of the key design specification techniques that enable the easy and rapid creation of such fully-functioning web applications. It also makes the case on how the scope of forms-driven workflow applications goes beyond the scope of plain forms and report builders. Due to the highly interactive, WYSIWYG nature of the demo, a high-resolution video is available at:
`http://www.vimeo.com/4402986`, password `FORWARD`

3. DESIGN SPECIFICATION TECHNIQUES

3.1 Page-centric Design

The first design specification technique is *page-centric design*, which allows the process owner to design her application through the WYSIWYG model of pages, as opposed to engaging in low-level web and database programming. The design facility also contains wizards that prompt for properties using questions expressed in easy-to-understand language. The design facility, in response, automatically creates the pages' structure and the underlying schemas and queries. While page-centric design is typically utilized by form builders for WYSIWYG design of forms, app2you and FORWARD expanded this to also encompass WYSIWYG specification of *contextual forms*, i.e. forms and actions that operate within the context of reported records.

Page-centric design also facilitates immediate feedback on whether a design satisfies the owner's requirements, since the owner can both inspect and experience the page directly. An inherent difficulty in producing a WYSIWYG model for a collaborative application is that the pages typically behave differently depending on which user accesses them and the application state. *Simulation scenarios* allows the owner to submit sample data and assume the role of particular sample users, in order to verify the application behavior.

3.2 Workflow-centric Design

Page-centric design by itself turns out to be insufficient for allowing the owner to translate a non-trivial multi-step process she has in mind into a working application. Using Figure 1 as an example, it is intuitive for the owner to think of the `assign` action as a single specification operation, i.e. a committee member assigning an application to an external reviewer. Instead of expecting owners to decompose a workflow transition into coordinated activity across two pages, Figure 2 shows a *workflow-centric* wizard that systemmatically prompts the owner for: 1. the action on the source page (`assign`) 2. the access rights of the action (the committee) 3. the context of the action (the business plan application) 4. the contextual form and additional data it collects (the external reviewer to assign to) 5. how the action affects the target page (report business plan applications that external reviewers can access) 6. additional side-effects (send email notification).

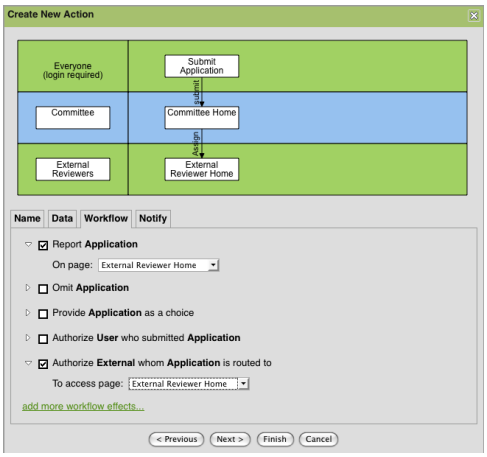

Figure 2: Designing the `assign` Action with a Workflow-centric Wizard

3.3 Semi-Automated Report Creation

In keeping with the high level specification of page-centric design and workflow design, owners can also design complex reports without specifying query details such as projections, selection conditions and join conditions. Rather, the design facility provides *semi-automated report creation* that 1. suggests semantically meaningful joins of various data sets; 2. avoids joins that lead to provably redundant information; 3. presents the suggestions as multiple choices, explained using easy-to-understand language; 4. discovers the best placement of information on the report to illustrate as many associations and constraints between the reported data sets. In effect the interface must compensate for the minimality of the owner-provided information with algorithms that offer semantically meaningful options and automate implementation details.

Acknowledgements

We thank Abhijith Kashyap, Yannis Katsis, Michalis Petropoulos for their contributions in prior versions of app2you. We also thank the members of UCSD's database group for their many insightful comments and discussions.

4. REFERENCES

[1] S. Ceri, P. Fraternali, and A. Bongio. Web modeling language (webml): a modeling language for designing web sites. *Computer Networks*, 33(1-6):137–157, 2000.

[2] K. Kowalczykowski, K. W. Ong, K. K. Zhao, A. Deutsch, Y. Papakonstantinou, and M. Petropoulos. Do-it-yourself database-driven web applications. In *CIDR*, 2009.

[3] K. Kowalczykowski, K. W. Ong, K. K. Zhao, A. Deutsch, Y. Papakonstantinou, and M. Petropoulos. Do-it-yourself database-driven web applications (extended version). `http://db.ucsd.edu/pubsFileFolder/319.pdf`, 2009.