TOWARDS DO-IT-YOURSELF DEVELOPMENT OF COMPOSITE WEB APPLICATIONS

Andreas Rümpel, Carsten Radeck, Gregor Blichmann, Alexander Lorz, and Klaus Meißner
Technische Universität Dresden, Germany
{andreas.ruempel,carsten.radeck,gregor.blichmann,alexander.lorz,klaus.meissner}@tu-dresden.de

ABSTRACT

Different tools and platforms support mashing up web-based resources to build applications by utilising development techniques suitable for end users. Those approaches are often based on simple predefined building parts and either constrain application complexity or require detailed knowledge of data types and communication paradigms. This paper proposes a novel approach of semantically connecting mashup components at an end user abstraction level. To provide immediate feedback and instant payoff from development effort, mashup development, usage, and reconfiguration tasks are seamlessly interwoven.

KEYWORDS

Mashup platforms, End user development, Situational composition, Web-based services, Semantic recommendation

1. INTRODUCTION AND MOTIVATION

Mashups empower users to meet their situational needs by composing a steadily increasing number of open, reusable, and distributed web services and APIs. While initially focused on aggregation of data and logic, recent approaches adhere to *universal composition* (Daniel et al. 2009, Pietschmann 2010). Thereby, components of all application layers, including the UI, are uniformly described and integrated. We apply this idea, resulting in component-based and presentation-oriented mashups, called composite web applications.

Current mashup platforms ease the composition process through graphical modelling and different abstraction levels. However, there are still shortcomings especially with regards to end user development (EUD). Advanced understanding of technical concepts, like details about data structures, is necessary. Whilst there are numerous reasons that cause current EUD for mashups to fall short of expectations, two prominent ones should be emphasised: First, there is insufficient immediate payoff from the development effort in terms of time or cost savings. Second, the complexity of real-world problem domains exceeds the problem analysis and modelling capabilities of average end users.

Based on the identified shortcomings of current mashup platforms supporting EUD, this paper proposes key features of a situational mashup development process suitable for domain experts and experienced web users without requiring deep technological knowledge. A central issue of the EUD process in our research project *Engineering of Do-it-Yourself Rich Internet Applications (EDYRA)* is the semantic description of mashup components and their communication interfaces, facilitating a recommendation process at a user-adequate level.

The remaining paper is structured as follows. Section 2 outlines the state of the art concerning EUD in the area of web service composition and in the context of mashups. Next, in Section 3 we propose the EDYRA vision for a semantically guided development of composite web applications. Finally, Section 4 concludes this paper and outlines work in progress.

2. END USER DEVELOPMENT FOR SERVICE-BASED APPLICATIONS

Service composition and mashup environments ease the development of composite web applications in comparison to traditional methods. Factors for enabling the end user building his own powerful applications and the technological prerequisites have been outlined in (Lorz et al. 2011). This section provides a brief overview of related EUD approaches in web service composition and its support in existing mashup development platforms.

2.1 End User Web Service Composition

In the area of web services, several approaches aim to simplify the composition task in comparison to classical languages like BPEL and, thus, enable non-programmers to create composite applications. In *ServFace* (Feldmann et al. 2009), composition takes place at the presentation layer utilising generated *Service Frontends*. While EDYRA focuses on rich internet applications, ServFace is restricted to simple form-based user interfaces. Major drawbacks are the limitation to SOAP web services and the lack of adaptivity.

A template-based guidance approach is proposed in *SOA4All* (Mehandjiev et al. 2010). Predefined templates for different tasks are instantiated with concrete web services at runtime, based on semantic technologies. Compatible web services for each slot are automatically determined and presented to the end user. Since this procedure is rather restrictive, we strive for a more flexible solution.

2.2 Mashup Platforms with EUD Support

Several scientific approaches focus on EUD of composite web applications and are discussed in this section. Similar to EDYRA, *DashMash* (Cappiello et al. 2011) strives for an interwoven design and execution environment. Based on a categorisation, e.g., in viewer, filter and data sources, recommended components can be added via drag and drop to the current composition. Recommendations rely on different quality aspects regarding, e.g., syntactic and semantic compatibility of inputs and outputs. Thus, it is possible to wire components automatically. For customising the wiring between components, the user has to manually combine inputs and outputs. Recommending components does not regard information extracted from performed tasks or previously created compositions by the user or the community. Intelligent layout generation is not sufficiently addressed at the moment, but in our opinion, it is crucial for providing a convenient development experience for end users.

With *EzWeb* (Lizcano et al. 2008), mashups can be built by end users at runtime through a combination of *gadgets* via wiring in a dedicated connector view. Gadgets created by gadget developers provide a screen-flow-based presentation layer for different encapsulated third-party web resources. One major drawback is the lack of recommendations for suitable components. Additionally, inserted components have to be wired manually by the end user through linking outputs and inputs via channels. In general, wiring is only based on data types, corresponding functionalities are not taken into account. Collaborative development and adaptivity are currently not addressed, neither by DashMash, nor by EzWeb.

WIRE applies the idea of wisdom-aware computing, i.e., providing recommendations by leveraging existing compositions and thus the wisdom of the crowd (Chowdhury et al. 2010). Composition knowledge is provided as *Advices* including *Patterns* and *Triggers* stating the condition under which the advice triggers, and depends on the considered composition metamodel. In contrast to the aforementioned approaches, WIRE does not rely on explicit semantic annotations but rather on implicit semantics gathered by different mining techniques and statistical data analysis. We will investigate the applicability in our (more complex) meta model, since the basic idea of reusing composition knowledge of experienced users is desirable. However, we argue that there is a need for semantics especially when it comes to context-aware composition, since concrete components may not be suitable in the end user's context, for instance, with respect to the device.

With an increasing number of available web-based resources and web users, EUD gains momentum especially in composing web mashups. Although the presented approaches already abstract from programming tasks, there is still technical knowledge required to successfully create composite web applications.

3. EDYRA: SEMANTICALLY GUIDED MASHUP DEVELOPMENT

In this section, key ingredients and basic principles of our envisioned mashup development process and platform are introduced. They incorporate essential requirements for EUD tools as pointed out in the literature, e.g., (Namoun et al. 2010, Cappiello et al. 2011).

EDYRA is based on the results of the CRUISe project (Pietschmann 2010), which enables universal composition of heterogeneous web resources encapsulated as black-box components in a service-oriented fashion. Component interfaces are specified in a WSDL-like way using the *Mashup Component Description Language (MCDL)* comprising operations, events, and properties. A *composition model* acts as a platform independent description of all mashup aspects, like the layout and the publish-subscribe communication of components. Based upon those models, EDYRA extends the CRUISe infrastructure and development process by means required for EUD (cf. Figure 1). In general, our vision of building applications at runtime emphasises the selection and reuse of prefabricated components or entire compositions instead of specifying from zero.

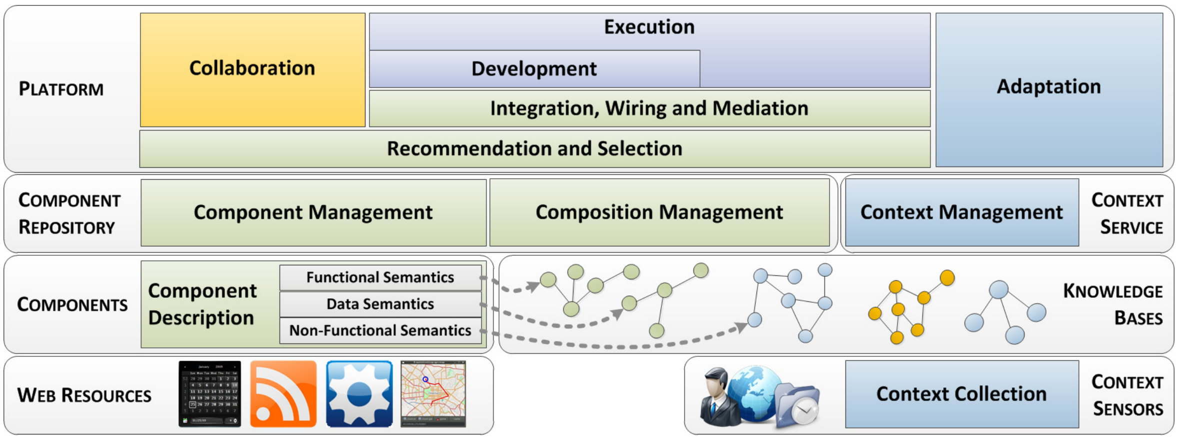

Figure 1: Overview of the envisioned architecture

In contrast to professional software engineers, end users implicitly and iteratively conduct design in the small (Cao et al. 2010). Thereby, they usually reflect on the consequences of their actions which is impeded by a sharp distinction between design and run time. Additionally, as shown by (Namoun et al. 2010), end users have difficulties to distinguish between these two perspectives. This underpins the necessity to stronger interweave both. Following the WYSIWYG principle and the approaches presented by DashMash and EzWeb, we thus strive for the **development of the application during runtime**, where automatically recommended, integrated, and coupled components can be used immediately (cf. lower middle part of the *platform* in Figure 1). We call this process **live sophistication** of composite web applications. Automating development steps results in a reduced learning effort, which is an important requirement for successful EUD. To support varying end user skills, we suggest the provision of adequate views, **differently abstracting** the necessary end user's knowledge of composition logic and component interface details: non-technically skilled end users are supported by high automation, requiring user intervention only in case of ambiguities, while an advanced user can manipulate, for instance, the wiring similar to EzWeb. Since this choice is explicitly made by the user at the moment, we aim at a higher degree of automatisation.

To enable recommendation and composition at runtime, the **semantic description** of components including their communication interface is absolutely necessary. Therefore, as shown in the left part of Figure 1, we leverage light-weight semantic annotations as proposed in (Pietschmann et al. 2011). In contrast to EzWeb, besides data semantics in terms of concepts in domain ontologies, we additionally focus on non-functional semantics and functional semantics. This includes functionality of the overall component, operations, and events by means of *capabilities* as well as pricing and quality of service. Capabilities comprise actions performed by users or components themselves on certain domain objects. Actions and affected object are formalized as ontology concepts, too. For instance, `<action=Search, domain object=Route, provider=Component>` represents a component's capability to deliver public transportation routes. The necessary input parameters are part of the corresponding operation's data semantics, and thus may differ from the domain object. Building up on this, we propose **automated wiring of components and mediation** of different data structures in line with a composition model kept synchronised with the application. As an example, if a user is creating an appointment via a calendar component, the platform recommends suitable activities, e.g., to integrate an additional component for inviting people from his address book.

To retrieve and **recommend** components best fitting the user's requirements and context, matchmaking approaches, e.g., based on subsumption of data, functional, and non-functional semantic annotations, will be applied. Therefore, we currently utilise a context-aware, template-based technique (Pietschmann et al. 2011) to query alternative components from the *component repository* and to determine possible follow up components by deriving templates from unbound operations or events of components being already present in the mashup. Unlike DashMash, one of our next steps is to study a combination with recommendations taking previously defined composition models from similar users into consideration.

A major challenge during live sophistication is intelligent **automatic layout generation**. In this context, several issues need to be resolved. It is necessary to figure out, which components shall be displayed in parallel, which components should be displayed close to each other, displaced, relocated, or minimised, when adding new components. Further, layout generation will **consider context data** managed by the *context service* like a user's preferences, abilities, tasks, and usage characteristics, as well as device properties. Such context information will also be employed to provide context-aware mashups. We therefore investigate the transfer of means for context and adaptation management as proposed earlier (Pietschmann et al. 2011b) to our platform as indicated on the right hand side of Figure 1.

Particular challenges arise from the universal composition approach, since, besides the rather trivial visualisation of UI components, also components without a UI have to be represented adequately. Therefore, we propose to offer a view, where all currently integrated non-UI components are displayed. Adequate user guidance in case of presenting recommendations will be provided in a structured and visually optimized way. Besides utilising text-based search, the user can either choose between commonly used components or components matching the current composition and context. These are ranked by personal preferences and community-based suggestions.

4. CONCLUSION AND WORK IN PROGRESS

The hurdles for low-skilled users to create or reconfigure their own composite web applications are still high. Early approaches tried to hide composition complexity by providing simple pre-manufactured building blocks selectable out of a very small pool. The resulting insufficient overall complexity of final composite applications provides an excellent foundation for the involvement of semantic description of mashup components and their communication interfaces. In this early phase of our research project EDYRA, we propose a mashup development process and platform by breaking those semantic associations down to a level suitable for the defined target group.

To figure out which results the EDYRA platform and development process can offer within different application domains, we are implementing several evaluation scenarios. To this end, we have developed suitable mashup components and annotated their purpose and communication interfaces semantically. We plan to formalise the task hierarchy, corresponding to those mashup components, via an implicitly generated semantic task model, cf. (Tietz et al. 2011). Example domains include social travel planning and smart office. In addition to proof of concept prototypes, we plan to conduct user studies in order to show our approach's feasibility and suitability. Furthermore, future work includes the support of asynchronous and synchronous multi user collaboration based on a uniform semantic data layer.

ACKNOWLEDGMENT

Funding for the EDYRA project is provided by the Free State of Saxony and the European Union within the European Social Funds program (ESF-080951805).

REFERENCES

Cao, J. et al., 2010. End-user mashup programming: through the design lens. *Proceedings of the 28th International Conference on Human Factors in Computing Systems*, Atlanta, USA, pp. 1009–1018.

Cappiello, C. et al., 2011. Enabling End User Development through Mashups: Requirements, Abstractions and Innovation Toolkits. *Proceedings of the Third International Symposium on End-User Development (IS-EUD)*, Torre Canne, Italy, pp. 9–24.

Chowdhury S. R. et al., 2010. Wisdom-Aware Computing: On the Interactive Recommendation of Composition Knowledge. *Proceedings of the 6th Workshop on Engineering Service-Oriented Applications (WESOA).*

Daniel, F. et al., 2009. Hosted Universal Composition: Models, Languages and Infrastructure in mashArt. *Proceedings of the 28th International Conference on Conceptual Modeling*, Gramado, Brazil, pp. 428–443.

Feldmann, M. et al., 2009. Overview of an End User enabled Model-driven Development Approach for Interactive Applications based on Annotated Services. *Proceedings of the 4th Workshop on Emerging Web Services Technology*, Eindhoven, The Netherlands, pp. 19–28.

Lizcano, D. et al., 2008. EzWeb/FAST: reporting on a successful mashup-based solution for developing and deploying composite applications in the upcoming web of services. *Proceedings of the 10th International Conference on Information Integration and Web-based Applications & Services*, Linz, Austria, pp. 15–24.

Lorz et al., 2011. Introducing the EDYRA Vision: Engineering of Do-It-Yourself Rich Internet Applications. *Proceedings of the International Conference on Internet Technologies & Society (ITS 2011)*, Shanghai, China.

Mehandjiev, N. et al., 2010. Assisted Service Composition for End Users. *Proceedings of the 8th European Conference on Web Services*, Ayia Napa, Cyprus, pp. 131–138.

Namoun et al., 2010. Conceptual and Usability Issues in the Composable Web of Software Services. *Proceedings of ICWE Workshops, 2nd International Workshop on Lightweight Integration on the Web*, Vienna, Austria, pp. 396–407.

Namoun, A. et al., 2010. Service Composition for Non-programmers: Prospects, Problems, and Design Recommendations. *Proceedings of the 8th European Conference on Web Services*, Ayia Napa, Cyprus, pp. 123–130.

Pietschmann, S., 2010. A Model-Driven Development Process and Runtime Platform for Adaptive Composite Web Applications. *International Journal on Advances in Internet Technology*, Vol. 4, No. 2, pp. 277–288.

Pietschmann, S., Radeck, C., Meißner, K., 2011. Semantics-Based Discovery, Selection and Mediation for Presentation-Oriented Mashups. *Proceedings of the 5th International Workshop on Web APIs and Service Mashups (Mashups 2011)*, Lugano, Switzerland.

Pietschmann, S., Radeck, C., Meißner, K., 2011. Facilitating Context-Awareness in Composite Mashup Applications, *Proceedings of the 3rd International Conference on Adaptive and Self-Adaptive Systems and Applications (ADAPTIVE 2011)*, Rome, Italy.

Tietz, V., 2011. Towards Task-Based Development of Enterprise Mashups. *Proceedings of the 13th International Conference on Information Integration and Web-based Applications & Services (iiWAS2011)*, Ho Chi Minh City, Vietnam.