Model-driven Development of Interactive Web User Interfaces with HTML5

Michael Huber and Philipp Brune
University of Applied Sciences Neu-Ulm, Wileystraße 1, D-89231 Neu-Ulm, Germany

Keywords: Model-driven Web Application Development, Model Transformation, Code Generation, Metamodeling, Rich Web UI.

Abstract: Graphical user interfaces (GUI) of modern web applications offer a look-and-feel comparable to desktop applications, mainly by using JavaScript or other Rich internet Applications (RIA) technologies. With W3C's upcoming HTML5 standard, even more powerful concepts for browser-side GUI programming are introduced, e.g. the <canvas>-tag. Model-driven development of web applications and RIA has been studied for many years. However, all existing approaches for model-driven web and RIA development focus on entire web-pages. With the increasing complexity of modern web GUIs, the model-driven development of GUI components itself comes into focus. Therefore, in this paper a method is proposed for the model-driven development of interactive, JavaScript-based GUI components based on the <canvas>-tag. Using a metamodel based on an UML 2.0 profile, the approach is usable together with existing UML-based methods or standalone. The implementation is described and its feasibility and implications are examined by means of a proof-of-concept example.

1 INTRODUCTION

In recent years, the graphical user interfaces (GUI) of web applications have changed dramatically with respect to their interactivity, functionality and visual design. By means of technologies like JavaScript or AJAX (Asynchronous JavaScript and XML) (Mesbah and van Deursen, 2007) or Rich Internet Application (RIA) frameworks (Toffetti et al., 2011), web applications today provide richer and more interactive user interfaces with many of the characteristics of desktop application software. In addition, for the increasing number of mobile devices used as web client platforms, a higher interactivity of the GUI and new types of user interaction (i.e. gestures) are required (Ortiz and Garcia De Prado, 2010).

Therefore, the world-wide web consortium (W3C) is currently working on the specification of a new version of the web markup language known as HTML5 (Fulton, 2011). Among many other new features, HTML5 introduces the `<canvas>`-tag to create and display individual and interactive graphics within a web page (Fulton, 2011). Even though the HTML5 specification is not finally released at the moment, the `<canvas>`-tag is already supported by many modern web browsers.

Model-driven development (MDD) (Sabbah, 2006) has received considerable attention for many years as a promising paradigm in software engineering. Regarding MDD of web applications also various approaches have been proposed and discussed, like WebSA (Melia and Gomez, 2006) or UWE (UML-based Web Engineering) (Kraus et al., 2007). Most methods use a graphical notation like UML for describing the models. Other solutions are based on specific notation languages (Wolfgang, 2011) or textual notation (Buchwalder and Petitpierre, 2006; Melia et al., 2008).

In recent years, model-driven development of RIA applications has attracted increasing attention and various authors have proposed respective approaches (Bozzon et al., 2006; Preciado et al., 2008; Linaje et al., 2007; Melia et al., 2008; Valverde and Pastor, 2009; Ortiz and Garcia De Prado, 2010; Toffetti et al., 2011). However, all these approaches focus mainly on two aspects of the model-driven development of RIA (Valverde and Pastor, 2009), namely 1) the definition of the GUI by combining components (widgets) from a selected RIA technology, and 2) the specification of the handling of interaction events generated by the user (Koch et al., 2009). Regarding the first aspect, all approaches consider the GUI components itself to

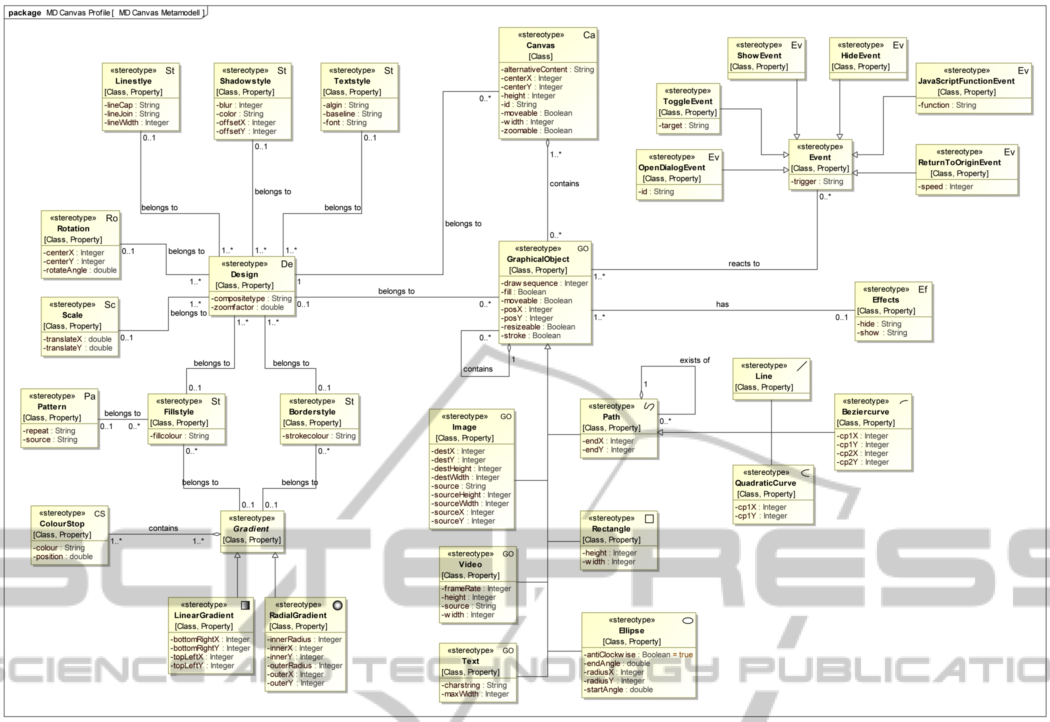

Figure 1: Overview of the UML2 metamodel proposed to describe an interactive HTML5 canvas-tag based GUI component within a web page.

be *atomic*. The possibilities for a model-driven development of these RIA GUI components itself and their embedding with the described MDD approaches have not been examined so far. This holds especially true for GUI component using upcoming technologies like the HTML5 `<canvas>`-tag.

Therefore, in this paper an MDD approach for developing interactive GUI components using HTML5 and the `<canvas>`-tag is proposed and examined by studying a proof-of-concept example. To demonstrate the integration with a general MDD web development framework, UWE (Kraus et al., 2007) was chosen, since it is UML-based, offers a good tool support, is well documented and supports also recent standards like Java Server Faces (JSF).

The structure of this paper is as follows: Section 2 presents the proposed UML metamodel and in section 3 the model transformation and code generation is described. In section 4 the proof-of-concept example is presented and examined. We conclude with a summary of our findings.

2 DESIGN OF THE METAMODEL

To ensure compatibility with UWE, the metamodel was designed using an UML 2.0 profile. Obviously, it needs to map all native functions of the `<canvas>`-tag, in order not to be restricted compared to manual development. Moreover, it should be possible to add interactivity actions like click-events or drag-and-drop to the whole canvas as well as to all its graphical elements. In addition, a good usability experience during the modeling process is required, like early error detection or the ability to predefine complex graphical objects like a three-dimensional cylinder to use it as one element. Therefore, a modular design is needed, especially for displayed items on the canvas and for attachable events triggering different actions. The model also needs to ensure the seamless integration with UWE.

Fig. 1 shows an overview of the resulting UML metamodel. It was designed around five main stereotypes, namely `Canvas`, representing an interactive GUI component based on the `<canvas>`-tag, `GraphicalObject`, representing any graphical object displayed inside the canvas, `Design` to assign visual styles to these graphical objects, `Effects`, modeling visual effects like fading in or out, and `Event`, describing user-triggered interaction events. Standard events used for navigating in a GUI are modeled as specializations of the stereotype `Event`.

A set of multiple Object Constraint Language (OCL) constraints defining class invariants ensures an

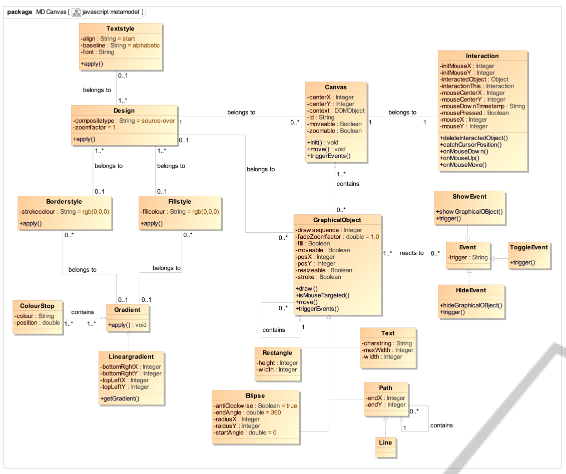

Figure 2: Class diagram of the JavaScript library that mimics the metamodel stereotypes and is used to implement the generated GUI component within a web page.

error detection during modeling to avoid minor errors, i.e. an undefined value for the id-attribute of the canvas, leading to an incorrect display in the browser.

To improve the practical usability of the proposed approach, the metamodel stereotypes were represented by a specific diagram type in MagicDraw (NoMagic, 2012). This simplifies keeping track of the elements in a complex model.

3 MODEL TRANSFORMATION AND CODE GENERATION

Since interactive GUI components usually represent only parts of a web page, it is necessary to integrate the present approach into general model-driven web development methods. The UWE method was selected to evaluate this integration. Therefore, its presentation model containing all displayed elements on one page needed to be extended by a new stereotype representing the canvas element.

For generating code the existing UWE code generator written in Java by Sebastian Stiegler in 2002 (Stiegler, 2002) was used and extended. By introducing new configuration parameters it is now possible to select whether the models provided to the generator should be interpreted as pure UWE models, canvas models or both.

In order to efficiently transform the model into executable HTML5 and JavaScript code, a JavaScript runtime library was designed for drawing of the canvas elements and processing events. This library uses the class model shown in Fig. 2 which mimics the relevant stereotypes of the UML metamodel

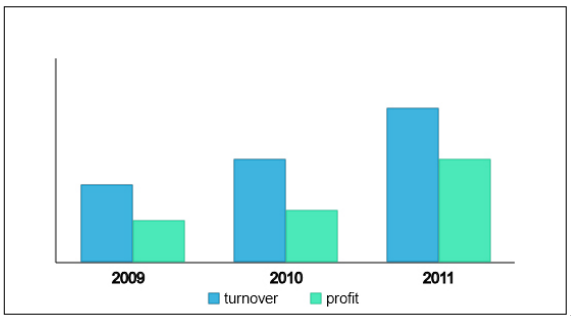

Figure 3: Screenshot (from Mozilla Firefox 10.0) of the generated interactive turnover and profit bar chart within a web browser window. By clicking on the legend entries, the visibility of the corresponding bars in the chart is toggled.

with JavaScript. While transforming UWE presentation models containing canvas elements, the generator creates the `<canvas>`-tag in the website and then checks for the corresponding canvas model and generates the respective JavaScript code using the library.

4 PROOF-OF-CONCEPT EXAMPLE AND RESULTS

To evaluate the presented approach, an interactive GUI component was developed showing the last three years turnover and profit data in an interactive bar chart diagram like used frequently for business analysis. The result as displayed by a web browser is shown in Fig. 3. To demonstrate the modeling of user interactions, it is possible to show and hide each data set by clicking on the corresponding legend entries.

The time needed for modeling and code generation using the presented approach proved to be significantly lower then the time needed for a comparable manual implementation. The time reduction originates mainly from the use of the JavaScript library and the overall higher level of abstraction. Additionally, a domain-specific toolbar was developed which strongly simplifies routine tasks like the assignment of stereotypes. The biggest advantage is the simplified implementation of the user interactions. By means of the specified OCL constraints, also errors related to the tagged values could be identified and solved in an early stage during the modeling process.

5 CONCLUSIONS

In conclusion, this paper presented the design and implementation of a MDD approach for interactive

GUI components based on the HTML5 `<canvas>`-tag. The designed metamodel maps the functionality of the `<canvas>`-tag appropriately. Based on the Unified Modeling Language (UML), this approach could be in principle integrated as an extension into most existing UML-based model-driven web development methods. This has been described and verified for the UWE method.

Future extensions may add further, more complex pre-defined graphical elements or frequently used special events. Also a binding to other HTML elements for controlling the canvas or the inclusion of physical behaviour in animations is possible.

So far, the approach mainly operates on the level of graphics primitives. For an obvious future application of the described approach in the development of interactive online games or virtual online worlds, higher level (or more domain specific) graphical abstractions (like support for 3D objects) are required. Further research is needed to investigate the design and possibilities of such an extension to the presented approach.

ACKNOWLEDGEMENTS

The authors would like to thank Prof. Dr. Hennicker from Ludwig-Maximilians-Universität Munich for kindly providing the source code of the UWE code generator created by Sebastian Stiegler.

REFERENCES

Bozzon, A., Comai, S., Fraternali, P., and Tofetti Carughi, G. (2006). Capturing ria concepts in a web modeling language. In *Proceedings of the 15th international conference on World Wide Web*, WWW '06, pages 907–908, New York, NY, USA. ACM.

Buchwalder, O. and Petitpierre, C. (2006). Weblang: A language for modeling and implementing web applications. In Kang Zhang, G. S. and Visaggio, G., editors, *Proceedings of the Eighteenth International Conference on Software Engineering and Knowledge Engineering, ICSE 2006*, pages 584–590, San Francisco, California. Skokie, Ill: Knowledge Systems Institute Graduate School.

Fulton, S. und Fulton, J. (2011). *HTML5 canvas*. OReilly, Farnham.

Koch, N., Pigerl, M., Zhang, G., and Morozova, T. (2009). Patterns for the model-based development of rias. In *Proceedings of the 9th International Conference on Web Engineering*, ICWE '9, pages 283–291, Berlin, Heidelberg. Springer-Verlag.

Kraus, A., Knapp, A., and Koch, N. (2007). Model-driven generation of web applications in uwe. In Nora Koch, A. V. and Houben, G.-J., editors, *Proceedings of the 3rd International Workshop on Model-Driven Web Engineering MDWE 2007*, Como, Italy. CEUR-WS.org.

Linaje, M., Preciado, J., and Sanchez-Figueroa, F. (2007). A method for model based design of rich internet application interactive user interfaces. In Baresi, L., Fraternali, P., and Houben, G.-J., editors, *Web Engineering*, volume 4607 of *Lecture Notes in Computer Science*, pages 226–241. Springer Berlin / Heidelberg.

Melia, S. and Gomez, J. (2006). The websa approach: Applying model driven engineering to web applications. *Journal of Web Engineering*, 5(2):121–149.

Melia, S., Gomez, J., Perez, S., and Dijaz, O. (2008). A model-driven development for gwt-based rich internet applications with ooh4ria. In Daniel Schwabe, F. C. and Dantzig, P., editors, *Proceedings of the Eighth International Conference on Web Engineering, ICWE 2008*, pages 13–23, Yorktown Heights, New York, USA. IEEE.

Mesbah, A. and van Deursen, A. (2007). Migrating multi-page web applications to single-page ajax interfaces. In *Proceedings of the 11th European Conference on Software Maintenance and Reengineering*, CSMR '07, pages 181–190, Washington, DC, USA. IEEE Computer Society.

NoMagic (2012). Magicdraw. http://www.magicdraw.com.

Ortiz, G. and Garcia De Prado, A. (2010). Improving device-aware web services and their mobile clients through an aspect-oriented, model-driven approach. *Information and Software Technology*, 52(10):1080–1093.

Preciado, J. C., Linaje, M., Morales-Chaparro, R., Sanchez-Figueroa, F., Zhang, G., Kroiß, C., and Koch, N. (2008). Designing rich internet applications combining uwe and rux-method. In *Proceedings of the 2008 Eighth International Conference on Web Engineering*, ICWE '08, pages 148–154, Washington, DC, USA. IEEE Computer Society.

Sabbah, D. (2006). Model-driven software development - introduction. *IBM SYSTEMS JOURNAL*, 45(3).

Stiegler, S. (2002). Diploma thesis. http://www.pst.informatik.uni-muenchen.de/DA_Fopra/web-eng-uwe-generator.pdf.

Toffetti, G., Comai, S., Preciado, J. C., and Linaje, M. (2011). State-of-the art and trends in the systematic development of rich internet applications. *J. Web Eng.*, 10(1):70–86.

Valverde, F. and Pastor, O. (2009). Facing the technological challenges of web 2.0: A ria model-driven engineering approach. In Vossen, G., Long, D., and Yu, J., editors, *Web Information Systems Engineering - WISE 2009*, volume 5802 of *Lecture Notes in Computer Science*, pages 131–144. Springer Berlin / Heidelberg.

Wolfgang, U. (2011). Multi-platform model-driven software development of web applications. In *Proceedings of the 6th International Conference on Software and Data Technologies (ICSOFT 2011)*, pages 162–171. INSTICC Press.