Business Process Models as a Foundation for High-Fidelity-Prototypes of Mobile Enterprise Applications

Tobias Braumann, David Broll, Matthias Jurisch, Bodo Igler
Faculty of Design – Computer Science – Media
RheinMain University of Applied Sciences
Wiesbaden, Germany
Email: {tobiasbraumann,lustra87}@gmail.com, {matthias.jurisch,bodo.igler}@hs-rm.de

***Abstract*—Mobile Enterprise Applications are becoming more and more relevant to enterprises as the dissemination of smartphones has risen over the last decade. However, developing these applications is a very challenging and resource-intensive task. In this context, prototyping can lead to several benefits, including a better app usability. While Mobile Enterprise Applications are often used to support or carry out business processes, no prototyping approach exists that is based on business process models. In this paper, we present a tool that fills this gap. The tool uses a Business Process Model and Notation (BPMN) model annotated with screen designs as a source for generating a prototype. The prototype is integrated with a business process execution engine that runs the business process.**

Keywords–Mobile Enterprise Application; BPMN; Process Model; Prototyping.

I. INTRODUCTION

Since the beginning of the decade, mobile applications have become more and more ubiquitous. This trend also reached enterprises [1], where employees expect to use smartphone apps for their daily work with the high usability they are accustomed to from using consumer apps [2]. These expectations and the continuously and fast changing ecosystem of mobile app development pose a significant difficulty for the development of Mobile Enterprise Applications (MEA).

MEAs differ from regular consumer apps in several ways. E.g., they are often used to support some kind of business process, have only few potential users in comparison to consumer apps and need to adhere to enterprise specific guidelines [2]. Integrating business processes into mobile applications requires implementing new interfaces to process engines and adhering to process guidelines. These are some of the challenges for MEA development that are caused by MEA-specific aspects. Since other aspects of mobile application development also need to be taken care of, these factors contribute to developing MEAs being a time consuming and expensive process.

To reduce the effort required to develop mobile applications in general, prototyping can be used. A good prototyping process can prevent misunderstandings and make the conceptual phase of the development process significantly easier to handle and therefore reduce costs [3][4]. More importantly, this can also lead to a better usability of the final product. This will improve the willingness of employees to use the final MEA. However, no prototyping tool that supports all of the aforementioned aspects of MEAs exists. To our knowledge, there is no prototyping approach that caters to the business process aspect of this type of application.

In this paper, we present a prototyping approach that focuses on using business process models [5] as the primary source for MEA prototypes. This is accomplished by annotating process models with screen designs and creating a prototype using code generation and business process execution engines that can interpret and automatically execute business processes. This work is embedded into the scope of the Prototyping Framework for Mobile App Design in Large Enterprises (PROFRAME) [6]. The presented work will lay the foundation for the implementation of PROFRAME.

The remainder of this paper is structured as follows: Section II gives a brief overview on related work and identifies the research gap. The general approach of this paper is presented in Section III. Details on the behavioural modeling of the prototypes are given in Section III-A, designing screens is discussed in Section III-B and code generation and prototype execution are presented in Section III-C. The implementation is described in Section III-D. Section IV discusses advantages and disadvantages of our approach. A conclusion is given in Section V.

II. RELATED WORK

According to [2], a huge gap between the development of MEAs and standard non-mobile enterprise applications can be observed. However, the demand for MEA development in the next few years will be much higher than the supply [1]. Hence, it is important to support a very efficient way of implementing MEAs.

One way to improve the development of MEAs is improving the prototyping process. Several models for classifying prototypes have been proposed in the literature. Nielsen [4] proposed a distinction between vertical and horizontal prototyping fidelity. A horizontal prototype supports most functionalities of a product, whereas a vertical prototype supports only a few functionalities but is technically more similar to the final product. The filter fidelity model [7] adds more dimensions to this view, e.g., regarding interactivity, data model, weight and many other dimensions. Breadth and depth of functionality are also included in this model.

For prototyping mobile applications in general, many products and approaches can be found in the literature. However, regarding prototyping for MEAs, only a few tools can be found (e.g., Kony, Verivo Akula and SAP Mobile) [8]. These tools are often focussed on a specific use case or bound to a specific platform. None of them take business process modeling into

 ISBN: 978-1-61208-592-0

account, so the depth of functionality according to the filter fidelity model is low.

Integrating process models into application development has been discussed in the area of process-driven development. AgilePDD [9] proposes an agile approach to implementing business processes. In the prototyping phase of this process, business process models are used to define the behaviour of the prototype. While this approach seems promising, it does not define how a prototype should be generated from the process model or how process steps should be represented as screens. The approach is in general focussed on modeling use cases with business process models, whereas generating code from these models is only mentioned as an option to be considered [10].

From the presented literature, we can conclude that no prototyping approach or tool for MEAs exists that facilitate a high fidelity regarding the representation and integration of business process models into the prototype. This issue is at the core of our research, since a prototype that better resembles the final product improves its usability.

III. Approach

We consider three major requirements for our work: the tool needs to support (1) modeling a business process, (2) designing a user interface and (3) generating a platform-independent prototype that can be executed on mobile devices. The basic idea of our approach is to use business process models as the primary source for the prototype. The process model defines the behaviour of the app. To add a graphical user interface, the process model is annotated with screen designs for specific parts of the business process. With this information, a prototype for the app is generated.

The Business Process Model and Notation (BPMN) [11] is the most popular language for modeling business processes [12]. In practice, several implementations of process engines exist. They are able to interpret and execute BPMN models and integrate business processes with several backend systems. Therefore, BPMN is used as the process model language in the presented prototyping tool.

An overview of the prototyping process is given in Figure 1. A Business Analyst/UX-Designer uses the *BPMN/Form Modeler* to create a model of the process that shall be implemented with an app including a user interface design. To model the process itself, one can simply use an existing process modeling tool that supports BPMN. This can be done in close coordination with the customer, e.g., at a kick-off-meeting for a project. The result of this process is an *extended BPMN file*. This file can then be used to generate a *Web App* that cooperates with a business *process engine*. The customer can then use this app as a prototype, which allows a clear separation of the code generation and the prototype modeling. For customization, a developer that modifies the code generation can be included in the process.

To support the described process, answers to the following questions are required: What aspects of a process should be represented as screens (Section III-A)? How can screens be designed and how can data be reused over several screens (Section III-B)? How are prototypes generated (Section III-C)?

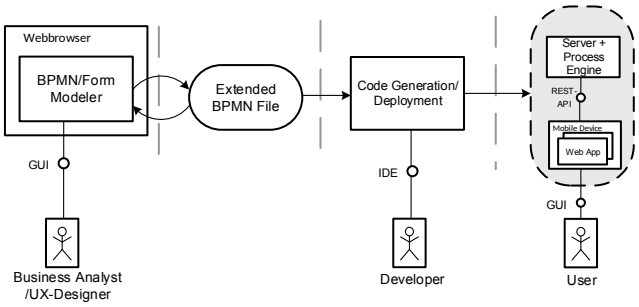

Figure 1. Architecture of the approach

A. Process Model

BPMN in general is well-known for its graphical representation of business processes. An example model is shown in Figure 2. The most important element of BPMN is the *task* (e.g., *Apply for Vacation*). Tasks represent any kind of activity. Several kinds of tasks exist, the kind of task is represented by an icon at the top of a task. *Apply for vacation* is a user task and *Check Vacation request* is a script task. User tasks require user interaction whereas script tasks are automatically executed by the business process engine.

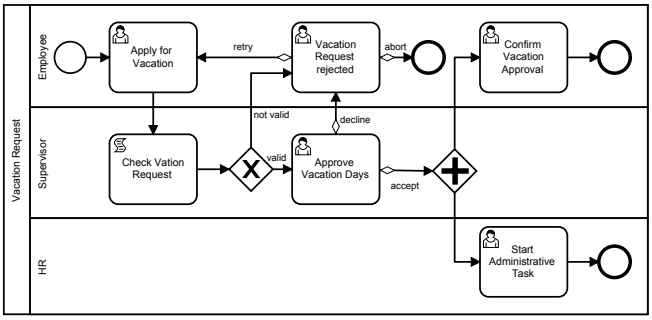

Figure 2. Example process

To connect tasks, so called *Sequence Flows* that are represented by arrows are used. Gateways (represented by rhombuses) are used to model situations where the flow is split, either because of decisions (x) or parallel execution (+). The swimming pool element (*Vacation Request*) is used to structure the control flow. A swimming pool can contain multiple swimlanes (e.g., *Employee*) that distinguish different domains of activity.

Our approach proposes a representation of tasks as screens: when the model is executed, each user task corresponds to one screen on the mobile device. A swimming pool corresponds to an app and a swimlane corresponds to a user role. For the example shown in Figure 2, users with role *Employee* would be shown at most three different screens (*Apply for Vacation, Vacation Request Rejected,* and *Confirm Vacation Approval*) and users with role *Supervisor* or *HR* one (*Approve Vacation Days* and *Start administrative Task*). Sequence flows determine the control flow of the business process.

B. Form Modeler

To design the forms that correspond to user tasks, a form modeler is used that is able to store screen designs as

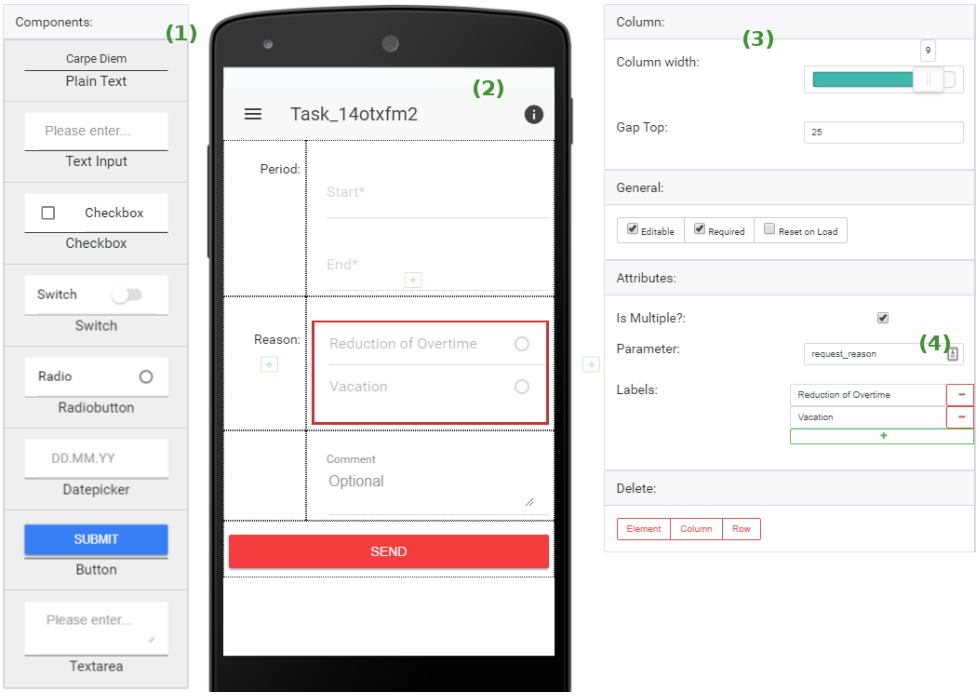

Figure 3. Form Modeler Screenshot

annotations in BPMN files. The form modeler needs to add a screen design to each user task and store the design as an annotation in the BPMN model. A screenshot of the form modeler that implements this idea is shown in Figure 3. The user of the form modeler can drag and drop user interface components (1), e.g., *Plain Text*, *Text Inputs* and *Radio Buttons*, into the screen layout (2). Properties of components can be modified using controls on the right (3).

Our approach uses a grid layout to model the screen design. By using a grid layout, the prototype is not bound to a specific screen size or orientation. The grid is shown as dashed lines in the screenshot. Users can add and remove rows and columns. Each cell in this grid can only hold one widget. To improve the design, the user can modify row height and column width.

By modifying a component's properties using the box on the right (3), the user can edit several aspects regarding its behaviour and appearance. E.g., inputs can be set as editable and required and their label can be defined. The property *parameter* (4) is used to specify parameter ids that are used to identify data throughout the business process. The parameter ids are identifiers in a global data space bound to a workflow. When a screen is used to input data into a field with a certain parameter id and another screen shown later in the process has a component with a matching parameter id, the second screen will show the data entered in the first screen.

The screen shown in the example corresponds to the task *Apply for Vacation* from Figure 2. To view the data entered into this screen, e.g., in the task *Approve Vacation Days*, it is only required to add a UI component to that task and set its parameter to `request_reason`, similar to the example shown in Figure 3 (4).

C. Code Generation and Process Execution

The previously described steps allow the creation of an annotated BPMN model that contains information about the behaviour of the application, as well as the UI design. Based on this information, code generation can be used to create a prototype.

Besides allowing generating app prototypes, using BPMN as a foundation for the prototype allows execution of the process model on a business process execution engine. To exploit this circumstance, the generated prototype is separated into two parts: (1) A business process engine that is given the business process model and executes it and (2) a prototype that interacts with the business process execution engine. The engine controls the process and data related to it. This allows the synchronization between prototypes for different user roles involved in the process, which are all created in the generation process. Also, the business process engine can be integrated with other enterprise systems, which allows accessing real-world data by the prototype.

D. Implementation

As a component for modeling business processes, Camunda Modeler [13] is used. Camunda Modeler needed to be extended to provide an interface to the form modeler. Angular [14] is used to implement the form modeler from scratch to allow a seamless integration with the process modeler. These components write their data into the shared extended BPMN file.

Prototype generation form the shared BPMN documend is implemented using XSLT [15]. To support multiple mobile platforms, the generated code uses the Ionic framework [16], a HTML and Javascript-based Framework, as SDK. With Ionic, the visual design of the app can be easily changed using CSS. This supports the integration of enterprise corporate design

 ISBN: 978-1-61208-592-0

guidelines into the product. To execute the business process, the Camunda Core Engine [17] is used. To interact with the engine, the prototype uses the Camunda REST API.

IV. Discussion

Designing a prototype for the business process shown in Figure 2 using the presented method takes about 25 minutes. After presenting the application example to experts from the MEA development field, they concluded that it would take one or two days for a developer to create a similar app. Since a developer needs to be involved in this process in addition to a business analyst, some communication overhead is also very likely. This indicates that the approach is able to create prototypes with a reasonable effort and is therefore able to compete with other prototyping approaches.

In comparison to prototyping approaches mentioned in Section II, we see several benefits. One important advantage of using BPMN as the foundation for prototyping is that it supports reusing existing process models to create a prototype. Even an automated transformation from existing files is possible. This is not possible for prototyping approaches based on other models. Another important aspect of this approach is the possibility to build applications using more than one user role easily. Supporting a business process execution engine allows the integration of existing enterprise systems into the prototype, since these systems can be integrated into the process engine. This allows the prototype to access real-world data, which give the user of the prototype a better understanding of the functionality.

A drawback of our approach is the limitation regarding visual design choices of the form modeler caused by the grid layout and the limited set of standard components. While inexperienced users might see the simplicity as an advantage, especially designers might need more freedom in positioning components and a broader collection of usable widgets.

Regarding the prototype's fidelity according to the filter fidelity model, our approach allows building prototypes that have a high breadth and depth of functionality and a close relation to data and appearance of the final product. This makes it easier to demonstrate to customers how an app can support their business processes and help manage expectations. This can lead to reduced costs for reworking requirements and app concepts during the MEA development process and improve the usability of the final product.

V. Conclusion and Future Work

In this paper, we have presented a prototyping approach for MEAs that is based on the usage of business process models written in BPMN. Prototypes are created using a business process annotated with screen designs. The annotated process is then used to generate a prototype that consists of an app and a business process execution engine that executes the process.

This approach facilitates fast prototyping of MEAs, since it is possible to reuse existing BPMN models for prototyping and integration with other enterprise applications through the business process execution engine. The generated prototypes can achieve a high level of fidelity regarding several aspects. Especially the depth of functionality and the visual quality of the prototype is high.

As future work, we plan to quantitatively evaluate the benefits of this prototyping approach regarding the ability to develop MEAs. Another next step could be the integration of existing screens from a standard screen library into the prototyping tool.

Acknowledgment

This work was funded by the German Federal Ministry of Education and Research, grant no. 03FH032PX5; the PROFRAME project at RheinMain University of Applied Sciences. All responsibility for the content of this paper lies with the authors.

References

[1] Gartner, "Gartner says demand for enterprise mobile apps will outstrip available development capacity five to one," https://www.gartner.com/newsroom/id/3076817, website [retrieved: September, 2017], 2015.

[2] A. Giessmann, K. Stanoevska-Slabeva, and B. de Visser, "Mobile enterprise applications–current state and future directions," in 2012 45th Hawaii International Conference on System Science (HICSS), Jan 2012, pp. 1363–1372.

[3] F. Kordon and Luqi, "An introduction to rapid system prototyping," IEEE Transactions on Software Engineering, vol. 28, 2002, pp. 817–821.

[4] J. Nielsen, Usability Engineering. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993.

[5] R. S. Aguilar-Saven, "Business process modelling: Review and framework," International Journal of Production Economics, vol. 90, no. 2, 2004, pp. 129–149.

[6] M. Jurisch, B. Igler, and S. Böhm, "PROFRAME: A Prototyping Framework for Mobile Enterprise Applications," in CENTRIC 2016, The Ninth International Conference on Advances in Human-oriented and Personalized Mechanisms, Technologies, and Services, 2016, pp. 7–10.

[7] T. Hochreuter, K. Kohler, and M. Mareen, "Towards a more conscious use of prototypes in mobile user experience design," in Prototyping to Support the Interaction Designing in Mobile Application Development (PID-MAD), 2013.

[8] J. Wong, V. L. Baker, A. Leow, and M. Resnick, "Magic quadrant for mobile app development platforms," https://www.gartner.com/doc/3744122/magic-quadrant-mobile-app-development, website. [retrieved: September, 2017], 2015.

[9] A. Herden, P. Farias, P. de Andrade, and A. Albuquerque, "Agile PDD: One approach to software development using BPMN," in 11th International Conference Applied Computing, Porto, Portugal, 2014.

[10] A. Herden, P. Farias, and A. Albuquerque, "An approach based on BPMN to detail use cases," in New Trends in Networking, Computing, E-learning, Systems Sciences, and Engineering. Springer, 2015, pp. 537–544.

[11] Object Management Group, "Business process model and notation, version 2.0," http://www.omg.org/spec/BPMN/2.0/PDF/, website. [retrieved: September, 2017], 2011.

[12] P. Harmon, "The state of business process management 2016," http://www.bptrends.com/bpt/wp-content/uploads/2015-BPT-Survey-Report.pdf, technical report. [retrieved: September, 2017], 2016.

[13] Camunda, "Modeler - Camunda BPM," https://camunda.org/features/modeler/, website. [retrieved: September, 2017], 2017.

[14] Google, "Angular," https://angular.io, website. [retrieved: September, 2017], 2017.

[15] W3C, "XSL Transformations (XSLT) Version 1.0," https://www.w3.org/TR/xslt, website. [retrieved: September, 2017], 1999.

[16] Drifty Co., "Build Amazing Native Apps and Progressive Web Apps with Ionic Framework and Angular," https://ionicframework.com/, website. [retrieved: September, 2017], 2017.

[17] Camunda, "Engine - Camunda BPM," https://camunda.org/features/engine/, website. [retrieved: September, 2017], 2017.

 ISBN: 978-1-61208-592-0