Hybrid Mockup-driven Development: An Agile Model-driven Approach for Web Applications

Gürkan Alpaslan and Oya Kalıpsız
Department of Computer Engineering
Yıldız Technical University
İstanbul, Turkey
e-mail: gurkan89@hotmail.de, oya@ce.yildiz.edu.tr

***Abstract*— Model-driven development is an effective way thanks to the automated code generation, documentation and creating a prototype of the project. Agile principles target to improve software developing pace and customer satisfaction with iterative approach and agile principles. By considering these two approaches, their inefficiencies are avoided and their advantages are combined. In this paper, we combine these two approaches for web applications; for this purpose, we choose the *hybrid MDD* method as skeleton, which is an agile model-driven approach. We target with this approach to create a more efficient software method for web applications.**

Keywords-Agile model-driven development; Hybrid model; Mockup-driven development model

I. INTRODUCTION

Model-driven software development (MDD) is based on the approach that "everything is model" [1]. Models are the abstract representation of the system elements [2]. MDD's basic goal is to develop software on a higher abstraction level than programming languages. Models are produced before writing source codes and the software is developed using these models. This way also provides software documentation.

An agile development method [3] is an approach which has values, principles and practices. Agile methods propose the iterative software development. With every iteration, a piece of software is produced, and, by the end of the last iteration, total software emerges. In agile methods, customers are the part of the software life cycle and all stakeholders are working together.

The main goal of combining agile methods and MDD is to benefit from both approaches. The method of that is to implement MDD principles to agile methods by using agile architecture as the skeleton. For this purpose, there are different approaches in the literature [4]. The first method is the Agile Model-Driven Development (AMDD) high level life cycle [5][6][7]. This structure divides the software life cycle into two parts, namely, the initial part and development part. In initial part, the system scope is determined and initial models are produced. The development part is the section where the iterations are implemented. The more improved structure is the Hybrid model [8]. Hybrid model has also two parts, but in this approach, different teams are defined. This approach is defined for general projects, not for the customized platforms, with limitations lightweight project constraint.

This paper's goal is to customize the Hybrid model for web applications by adding new suggestions for improving productivity and software quality. For this purpose, Mockup-driven development approach [9] and Hybrid model are combined. Mockup-driven development is a model-driven development for web applications. Mockups are the structures that produce models for web applications that are used as the prototypes of web applications [10]. Mockup-driven development propose to develop web applications from mockups which are more visual than diagrams. By automated code generation, produced mockups can convert to Hyper Text Markup Language (HTML), Cascading Style Sheets (CSS) and JavaScript codes.

In the following section, we describe the agile model-driven basics. Then, we describe the mockup-driven development approach in Section 3. After that, we describe our approach, hybrid mockup-driven development, in Section 4. An ongoing case study is described in Section 5. Finally, we conclude our experience and future work are described.

II. AGILE MODEL-DRIVEN DEVELOPMENT

Agile model-driven development proposes an iterative and incremental approach that supports all model-driven benefits like documentation and automated code generation [11][12]. In this approach, all models are produced by the agile models. Agile models [13] are the structures which are developed by implementing agile principles. Some of the agile core principles [14] are:

- Assume Simplicity
- Embrace Change
- Incremental Change
- Maximize Stakeholder Investment
- Rapid Feedback
- Working Software is Your Primary Goal
- Travel Light

To develop models properly with these principles is an important part of the approach. In AMDD, the total requirement list is divided into portions and the portions are implemented consecutively.

Generally, AMDD proposes to develop the software by test-driven development method. Test-driven development

 ISBN: 978-1-61208-449-7

[15][16] proposes to produce the software test units before writing the source codes. After the source code is produced, the code is tested and the results are evaluated. Three results are possible: passed, failed and refactor. Passed means the code has passed the test succesfully, failed means that code cannot work or fulfill the requirements and refactor means code works but should be improved.

III. MOCKUP-DRIVEN DEVELOPMENT

In web application development, HTML, CSS and JavaScript codes are used for designing both visual interface and system logic run. Mockups provide these elements by the automated code generation with supporting mockup tools. For this purpose, they are proper and simple to use in web applications. It provides to decrease error possibility and development time.

Mockups can define as the models of web applications [9]. Mockup tools provide an design interface which has items using in web systems. Developer can design the attributes of web items and make to connection with other items.

In our projects, we plan to use Axure [17] mockup tool. An example of mockup is shown in Figure 1.

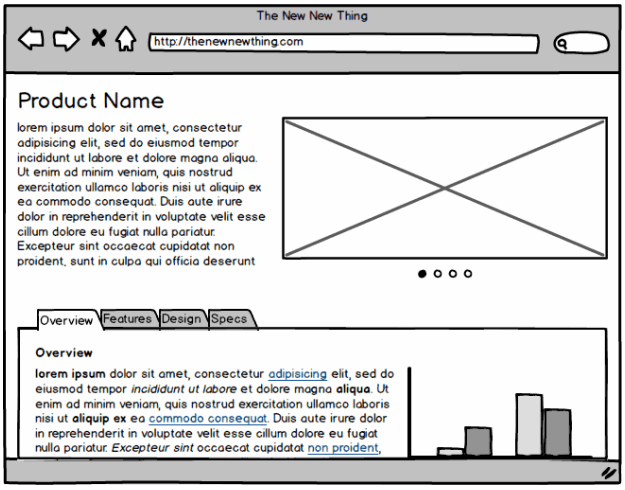

Figure 1. An example mockup design [18]

In Figure 1, an example website is designed by mockup items. Information writings and graphics are used in example. The graphic on the right up of the page in Figure 1 can change with the clicking little four circles. Bottom design of the page in example can also change with the clicking buttons that named as overview, feature, design and spaces. This level is the model part of the process. After the model developing is completed, this model transforms to real websites.

IV. HYBRID MOCKUP-DRIVEN DEVELOPMENT APPROACH

In our approach, we combine the hybrid method and mockup-driven development approach by adding some new parts and customize the hybrid method for web applications by using mockups. The reason of customizing for web application is that the mockup-driven development method is suitable for web applications. Mockups are more visual than diagrams and easier to understand for all stakeholders who have no knowledge about software engineering. This advantage makes it easier to analyze the requirements and working with mockups.

The approach can be used for small or medium size projects. The reason of that is the challenges of remodeling when lots of iteration occur.

A. *The Steps of the Approach*

The hybrid mockup-driven development is an iterative approach where three interconnected teams work in parallel. Figure 2 shows the processes of the approach. In the figure, the numbers near the processes represent the teams that work on the process. These teams are the business analyst team, the model-driven development team and the agile development team. The business analyst team is responsible for communication with customers and prepare the requirements. The MDD team is responsible for preparing the mockup environment and the generated codes, which are produced by mockup tools with automated code generation. The agile development team is responsible for hand-crafted codes and integrating them with generated codes. The team is also responsible for preparing the test units and testing the iteration software.

In the agile method, all the teams work together with high interaction. In the approach, these teams work together with support each other in every step. Teams use a common sharing area and all team member has information what the other teams have done. Also, before the iteration begin, all teams meet together and discuss that what and how they will do next iteration.

The steps in the approach are listed below.

- The business analyst team gets the requirements from customers.
- The business analyst team and MDD team produce the mockups. In this part, using pair development and a customer representative are working together for mockups.
- The agile team prepare the test environment and test code units for mockups.
- The MDD team prepare the mockup tools and mdd environment.
- Produced mockups transform to HTML, CSS, JavaScript codes with automated code generation by MDD team.
- The automated produced codes are shared with the agile team and by the agile team, this codes are completed with hand-crafted codes. Final code parts are implemented to test units.
- Finally, developers have a software part and they present it to customers.
- They get the feedbacks, document the reviews and pass to the next iteration.
- With every iteration, this period repeats until the total software has been produced.

 ISBN: 978-1-61208-449-7

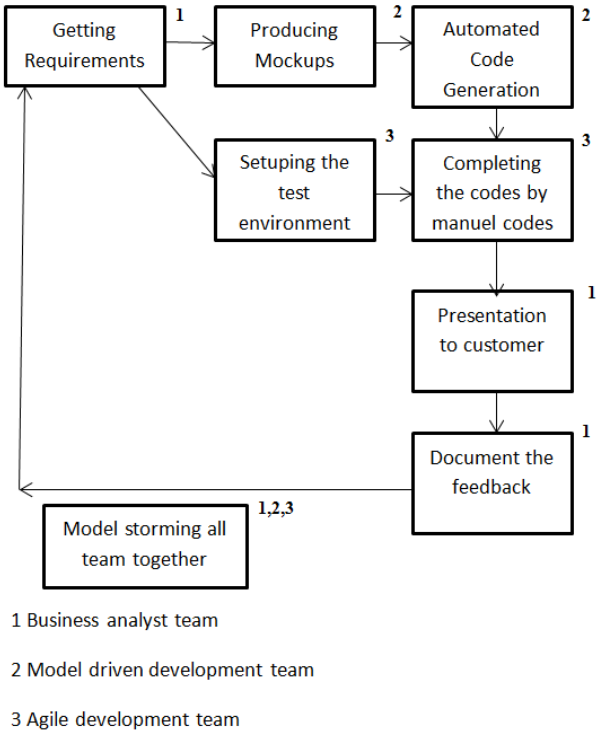

Figure 2. Hybrid mockup-driven development

The model storming part is a transition part for all teams to be prepared for next iteration. In this part, all teams meet together and firstly evaluate the completed iterations reviews. After that, they prepare the next iteration activity plan.

System test is done with the method of test-driven development approach, which described in Section 2. Test plans designed by agile development team. Black box test can be used for this type of small size web projects and the systems which has supported with automated code generation. A benefit of MDD is decreasing the error possibility. Test cycle is a sequential activity. When every iteration is done, the iteration's test is implemented. Test results are documentated like the method described in Section 2 and proper updates are implemented.

B. *Evaluation of the Approach*

Hybrid mockup-driven development propose an iterative approach with paralel working different teams. So, it divides the workload into small portions. With paralel working teams, waiting time of the other teams is decreased. Also, it provides to intermediate changing on any problem or fixing part while the development process.

The basis of the approach is the agile princibles which are described in Section 2. All parts of process run with supporting agile princibles, so it takes the advantage of agile architecture. Mockups are used, so more visual development is provided.

Generally, the basic benefits of the approach are software development pace, flexibility, error detection easiness, and simplicity.

C. *The comparison of the processes*

The approach is compared to hybrid model and AMDD high level life cycle. The comparison is done on these criteria: main objective, main contribution, and usage areas. In main objective part, the basis attributes of processes are described. In main contribution part, the basis contribution of software industry is described. In usage area part, customized platform of processes are described. Table 1 shows the basic differences.

TABLE I. THE COMPARISON OF THE PROCESSES

Process name	Main objective	Main contribution	Usage areas
AMDD High level life cycle	An iterative and test-driven approach with MDD princibles usage	First method that combining agile and MDD princibles	General projects
Hybrid MDD method	Development on models with difference teams iterative and incremental	The difference teams and work sharing describing on AMDD life cycle	General projects
Hybrid mockup-driven method	Development on mockups with difference teams iterative and incremental	Mockup usage with Hybrid approach	Web app.

AMDD high level life cycle is the first approach that shows that is possible to combine agile methods with model-driven methods. A hybrid process improves that model with adding different teams approach and new life cycle diagram. In our approach, we target to improve the hybrid method by using mockup and to demonstrate that mockup-driven development approach can use with hybrid method.

V. CASE STUDY

We gave this approach to a few student project teams in 'Software Engineering' class of our university and requested them to develop their projects with this approach. Also, we requested another teams to develop their project with the traditional method. So, we target to compare our approach and the traditional method.

This is an ongoing project; currently, the student project teams are in the development phase. We selected two teams developing the different projects for better analyzing the approach. One of the teams includes five students developing an online "Cinema Ticket System". Another team includes four students developing an online "Language Course Registration System". We described the approach to the teams and decided to use Axure mockup tool [17]. They

 ISBN: 978-1-61208-449-7

firstly splitted the three specialized team as MDD team, agile development team and business analyst team. After that, we requested them to split their project into iteration modules. Currently, they are working on first iteration; we plan to complete the entire project in three months.

VI. CONCLUSION AND FUTURE WORK

Developing software with mockups by producing prototype is an efficient way thanks to their visual interface. By this visual interface, customer requirements are realized more properly and the time for analyst part can be reduced. Paralel working teams can provide faster development and more communication. Thanks to the automated code generation, reducing the development time and amount of software errors are targeted.

Howewer, integrating automated generated codes and hand-crafted codes can cause some problems in large scale projects. In the forward iterations, rebuilding of previous iteration model is an challenge for the projects have lots of iterations. For this purpose, limited the approach for small or medium size projects can be more proper.

In future work, we get the analyst results of the case study work described in Section 5. We report the development challenges of the approach by the feedback of teams. We identified the evaluation criteria. They are flexibility, rapidity, learning, cost effectiveness, metadata management, tool support, problem domain analyst, simplicity and maintainence. Also, we compare the approach with the traditional method and other agile model-driven methods by the analyst results.

REFERENCES

[1] S. Vale and S. Hammoudi, "Context-aware model driven development by parameterized transformation", Proceedings of MDISIS, pp. 167-180, 2008.

[2] T. Stahl and M. Volter, Model-Driven Software Development, John Wiley & Sons, 2006.

[3] T. Dyba and T. Dingsory, "What Do We Know about Agile Software Development?", IEEE Software, pp. 6-9, 2009.

[4] R. Matinnejad, "Agile Model Driven Development: An Intelligent Compromise", Software Engineering Research, Management and Applications (SERA), 2011 9th International Conference on, pp. 197-202.

[5] S.W. Ambler, "Agile Model Driven Development" , XOOTIC Magazine, 2007.

[6] Agile Model Driven Development, http://agilemodeling.com/essays/amdd.htm, retrieved: 03.2015.

[7] S.W. Ambler, The Object Primer 3rd Edition: Agile Model Driven Development with UML 2.0 New York: Cambridge University Press, 2004.

[8] G. Guta, W. Schreiner, and D. Draheim "A Lightweight MDSD Process Applied in Small Projects", In Proceedings of the 35th Euromicro Conference on Software Engineering and Advanced Applications, pp. 255-258, 2009.

[9] E. Benson, "Mockup Driven Web Development", 22nd Intenational World Wide Web Conference, pp. 337-342, 2013.

[10] F. Basso, M. Pillat, F.R. Frantz, and R.Z. Frantz, "Study on Combining Model-Driven Engineering and Scrum to Produce Web Information Systems", 16th International Conference Enterprise Information Systems, pp. 137-144, 2014.

[11] G. Alpaslan and O. Kalıpsız, "Model driven development with agile process", Elektrik - Elektronik, Bilgisayar ve Biyomedikal Mühendisliği Sempozyumu , 2014.

[12] G. Alpaslan and O. Kalıpsız, "Researching of the agile model driven processes", Ulusal Yazılım Mimarisi Sempozyumu, 2014.

[13] A. Cockburn, Agile Software Development, Addison-Wesley, December 2001.

[14] V.C. Nguyen and X. Qafmolla, "Agile Development of Platform Independent Model in Model driven Architecture", 3th International Conference on Information and Computing, pp. 344-347, 2010.

[15] D. Astels, "Test Driven Development: A Practical Guide", Prentice Hall, 2003.

[16] S. Tilley, "Test-Driven Development and Software Maintenance", Proceedings of the 20th IEEE International Conference on Software Maintenance , pp. 488-491, 2004.

[17] Interactive Wireframe Software & Mockup Tool, http:// http://www.axure.com/, retrieved: 03.2015.

[18] Creating Your First Mockup | Balsamiq, http:// http://support.balsamiq.com/customer/portal/articles/871902-creating-your-first-mockup, retrieved 03.2015.

 ISBN: 978-1-61208-449-7