Agile Approach and MDA in Software Development Process

JaroslavaKniežová, Ing. PhD.
Associate Professor
Comenius University
Faculty of Management
Department of Information Systems
Bratislava, Slovakia

Abstract

There are several approaches defined in software development process. Each of them usually gives the rules and steps of developing the software for the customer in a good quality and also in as quickly as possible way. Achieving this brings the satisfaction to the customer as he gets effective software solution and also to the solution provider as he can lower the costs of software development by shortening the development time and this way increases his income. At present the agile approach is very common in software development companies. Using MDA in developing process should optimize the coding process as the analytical and design models should be transformed into the code. There can be seen some differences between these two approaches. In agile approach only minimum of modeling is used in development process. On the other side for optimal using of MDA the detailed models have to be created. This article contains description and comparison of these two approaches. The possibility of using both of these approaches in one project as well as the advantages or disadvantages of them are described too.

Modeling in Development Process

Several approaches for information system development process have been defined. The used approach influences the methodology, which has to be used in development process. It can be said that the first significant moment in information system modeling was the formation of traditional methodologies. These methodologies contain phases of the development process and usually several phases at the process beginning consist of the modeling works. According to these methodologies the models of information system should be always created before the coding works start. This should assure that no additional costs appear during the development process when the development team comes to the more detailed information. Detailed analysis and design should be done before coding, so all details are defined in the models and the coding can be done quickly then. Although the methodologies were changing through the history with the aim of achieving lowering costs for the solution provider and also the price for the customer with keeping the result quality, modeling of information systems has stayed in the phases of the development process until the agile approach came (described later in this paper).

According to RUP methodology the phase, which is focused on the modeling, i.e. Analysis and Design phase, is most important phase of the process, because if this phase is done in good details, the later coding is very quickly and of low costs. This methodology has its

evaluation cycle as after each phase of the project the finished works are evaluated according the previously defined criteria. And the evaluation after Analysis and Design phase is very important too, because on this evaluation and the decision based on it the whole project success depends.

It can be said that only UML language is used for modeling at present. The language was defined so as to unify the language of the analysts and their models. UML should be used together with the chosen methodology, i.e. the development team should follow the phases of the methodology and use UML for the models creating within them.

Model Driven Architecture in Development process

Model driven architecture (MDA) is the approach in development process, which is based on the UML language and is said to be the future of fully using UML in development process [1]. The main idea of MDA is that the software product is the result of models transformation process. According to this approach several models should be created with the defined sequence of their transformation. At the end of this process, the last model should be transformed to the source code. The suitable CASE (Computer Aided Software Engineering) system is needed and supposed to be used.

In this approach the main works are done when modeling in the development process and therefore models creation is supposed to require majority of the time and finance project calculation. The model which is the basis for the transformation to the source code has to contain all details and all defined solutions.

The models created in MDA (Figure 1) approach are:

A. CIM – Computer Independent Model

In this model the basic system requirements are modeled, project dictionary and the main system use cases are defined. There is a minimum of computer processing details in this model and the information should be modeled on use case (conceptual) level.

B.PIM – Platform Independent Model

This model is created when more details are added. The being built system is modeled in more details including the way of tasks processing – the use cases are modeled in details of the way how they will be done by the system. The details are captured at logical view, i.e. all data and algorithm details abstracted from platform specific processing.

C.PSM–Platform Specific Model

This model is worked out for the platform which will be used. The code – level details are included in the models. This model is the last one in the transformation and is used for code generation.

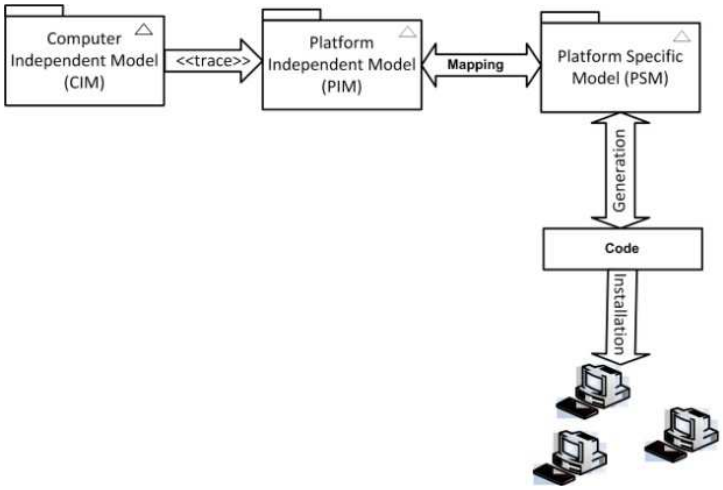

Figure 1Models in MDA

As the MDA architecture is supposed to be used very often in the future for model based application developing and the majority of the project works will cover modeling, there are many publications with attention to this problem (for example [4], [5]). Developing an application fully MDA way means using CASE tool to generate source code from the detailed models and this means minimum coding and maximum modeling.

Agile Approach in Development Process

It can be said that the 2001 year and the definition of the Manifesto for Agile Software Development is the beginning of the agile approach in software development. The Manifesto for Agile Software Development contains principles for agile developing which can help to achieve the project aims given by the development team. The aims in agile development are based mainly on one factor and this is the time. Agile development means developing the application in as short time as possible. Although the result is not complete and/or is not of required quality the time is the main (and often the only) factor of the project success. As the time has to be shortened as much as possible, some of the project works are skipped. This approach was formed based on the idea of a group of programmers, according to whom the system modeling spends a lot of time and produces no result for the customer (only for the development team members) and therefore it should be minimized. Although the programming works will be reworked because of not having solved the details in models, models are not created in this approach or are created very minimally and without details, only on the conceptual level.

It can be said that developing of an application using the agile approach brings more programming and less modeling through the project life cycle. Agile approach is very common at present and there are lots of companies using it. There are methodologies, which support this approach, for example SCRUM. Development life cycle in agile methodology is based on very often usually daily organized meetings on which the work details needed on that day are discussed. It is very important that the customer and future system users are present at these meetings and every team member who cooperates on the discussed project

part should be present too. It can be said that these meetings replace the detailed modeling the actual particular requirements are defined and the way of implementation of the particular tasks are on the programmer who implements them.

Figure2Principle of Agile and TraditionalDevelopment

Understanding the agile approach means knowing and understanding its main principles (Figure 2). In traditional development the first step was discussing the required functionality of the future system and then the needed time and money were derived. In the agile approach the time and money are given at the beginning and then the system functionality, which is to be delivered, is derived from these constants.

MDA and Agile development - comparison

According to the characteristics about these two approaches, given above, it can be defined one main difference between them. As the MDA approach represent creating detailed models, this kind of project works are the main part of the project life cycle and can spend most time and money as well. Most team members should be analysts, who create these models. On the other site, the agile approach means very minimum of the modeling and maximum of the programming, including refactoring instead. This is really a big difference between these two approaches which influences other important facts. To find out and describe these facts as the advantages or disadvantages it is needed to define the need and the position of the modeling in the project lifecycle. The reasons (the aim) to model (and take the advantages of modeling) can be defined as:

1. Discuss and solve problems

Models are created so as to catch the information about the future system. The analysts design and discuss the solution with the customer and the accepted issues are described in the models. These models are the starting point for the programmers. The advantages are:

- There are more people cooperating on one problem. The customer as future system user plays very important role in this process and his opinion is the key factor for the project success. When creating models, the customer's opinion can be captured in these models. Also the other team members can share these models and this way they can cooperate on the solution. The model, which was created this way, keeps the solution as the result of customer's and anyone's team member discussion.
- The models keep accepted solution. In this case the result of the analytic works is 'on the paper' and it is not possible to ignore once agreed solutions for other team members and for the customer as well.

- Keeping the historical information. Sometimes solving some problems can be postponed and the development team members continue in working on these issues after some (sometimes longer) time. As the previously discussed details are in the models, the team members can look through the models instead of discussing these issues once more.

2. Change management

It can be said that no project without any change in its requirements exists. It is very usual that the customer changes the requirements during the project works. The development team has to accept these changes but it is possible (and it is good because of cost control) to discuss the rules of such changes. Any rule is agreed between the development team and the customer, it will be always much easier to manage them when the changes are defined in the models.

There can be also the disadvantages of modeling defined as:

1. Time spending

Creating models requires time. The analysts have to discuss all problems, design the solutions and discuss them again so as to be able to define all needed details in the models. Because of this, the time increases and together with it the costs increase. Usually there is no possibility or it is extremely unprofitable to stop or postpone the project in this phase because although the models are created, there is usually nothing to deliver to the customer and therefore no payment from the customer.

2. 'Nothing' to show to the customer

Although the models are discussed between the analysts and the customer and are shown to the customer, the customer still cannot 'click' on any button and everything is only 'on the paper'. The customer may get feeling that the development team has done nothing and may not deliver the product on time.

As the conclusion from this comparison it can be claimed that the big difference between MDA and Agile approach is in the amount of modeling. Whether to model or no to model – this question can be answered according to the priority, which is defined for the particular project. As the modeling brings lower risk in the project (everything is in on the paper and the costs are controlled), decision for modeling can be made in cases when the risk is higher or risk managing is set as the priority. On the other side, modeling can spend a lot of time and therefore if the time is the priority, minimizing of modeling is good decision then.

Other difference can be defined between MDA and Agile approach except of the amount of modeling. That is the phase in which the changes of the requirements are solved (Figure 3).

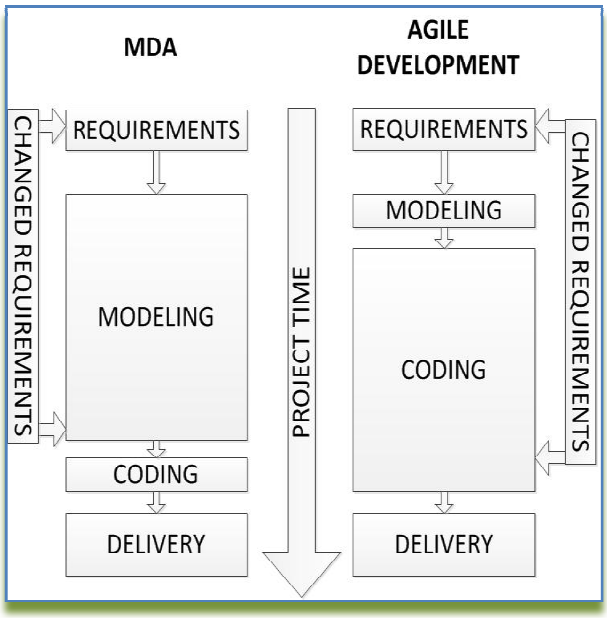

Figure3Requirements Changes in MDA and Agile Approach

MDA modeling covers more project time than in agile development, as the code is generated. All problems have to be solved, so all requirements changes have to be solved till this phase too. On the other side in agile approach coding covers the most project time, so the majority (or all) of the requirements changes, is solved in this phase. This is the difference and the advantages and disadvantages could be defined so as to say what is better.

The advantages of solving changes in models are:

1. Time spending in case of bigger changes. If the changes are elementary, the programmer can work out them very quickly. But if the changes are more significant or influences other parts, it is always better to solve them in sooner phases of the project life cycle, when there is not so much work done.
2. Changes with influence to other project parts. Some of the changes may require changing also other part of the project and in this case it is better to solve them in the model, where these dependences can be modeled and better recognized.
3. Documenting the changes. If the changes are described and solved in the models, the history of changes can be kept. If the changes are solved only in the code, it is very rare to keep the detailed history about every change. The programmers usually quickly continue to solve the following task so as to save the time instead of documenting the change in details.

The disadvantages of solving changes in the models:

1. Changes can be seen for the customer. If the change is done directly in the code and can be seen directly when working with the system, the customer can better specify if this change is complete or not. The customer can better discuss what he needs and if the done works are good.

2. Time spending in case of elementary changes. If the changes are not so big, it is always more quickly if only the programmer solves it instead of communicating the change between more people in the development team and describing the change again and again to the other member.

As the conclusion from this comparison it can be said that in case of small, elementary changes it is better to solve them in coding phase with the attention to document them. There are suitable software tools for making notes about done works and for small changes this way is sufficient. On the other side when the changes are bigger and dependent on other parts (or other parts are dependent on the changes), in this case it is better to model them.

The previous described views on the compared approaches were based on the difference: how much to model and where in the life cycle to solve the changes of the requirements. There is one more important fact in the MDA approach. This approach requires the compatible CASE tool, which could generate the code of a good quality from the model. As the code is generated at the end of the life cycle, the risk can be defined, when a lot of project works are done (models in very details) and if anything is wrong in generating and the code has to be recreated, the costs increase very fast. This risk can be lower very rapidly in case the development team uses proved CASE system and knows its functionality well. This risk depends on the experiences of the developer team and not on the chosen approach and therefore in case of comparison the two approaches, is not relevant. It is very important though therefore it has to be mentioned.

Using both approaches in one project

There are big differences between these approaches and usually for one project only one methodology and with it one approach is used. Although the question about the possibility of using both approaches in one project can be given. The aim of this way of combination is to take the advantages of both approaches and avoid the disadvantages on the other side. The beginning of the project lifecycle can be the same in both approaches, i.e. the requirements definition. After this point the development team has to decide which approach will be used.

According the previous described advantages and disadvantages it can be said, that there is the way how to combine the approaches. After the requirements definition the development team has to characterize the problems which are to be solved and the dependences between the particular tasks to be solved. If there are some project parts which are more complex and dependent between each other, it is better to model them (and use MDA). And if it is possible to divide the project parts with more and less complexity, then it is the situation when the approaches can be combined. If the more complex problems will be modeled and for developing of the less complex problems the agile approach will be used, in this case the advantages of both can be taken and the disadvantages can be avoided.

Conclusion

The two approaches were described and compared in this article. MDA approach is supposed to be the future of using of UML language and developing the software using the suitable

CASE system. The agile approach is very often used in praxis at present with aim to lower project time and with it also the project costs. There are big differences between these two approaches, which can be summarized as:

- The amount of the modeling in the project. While in the MDA approach, there is much more modeling in the project lifecycle, in agile approach, there is only minimum of creating system models.
- The place of requirements changes solved in the project lifecycle. While in the MDA approach the changes are solved when modeling, in agile approach the changes are solved in coding. Agile approach accepts recreating the code because of the requirement change.

The differences bring the disadvantages and advantages of these approaches resulted mainly from the decision whether to model or not. In some cases created models can be big advantage for examplewhen more complex problems which have to be solved or in case of requirements changes in this kind of problems. On the other side, a lot of not so complex problems give the reason to minimize modeling as creating models in these cases and communicating the models between the development team members could spend more time unnecessarily.

The described advantages and disadvantages give the question about combining these two approaches so as to take only the advantages and avoid the disadvantages. This is possible if the project tasks can be divided into two (or more) parts, one with more complex tasks with more dependences between each other, and the other, less complex with less dependences. Of course there should be minimum dependences between these two parts. The dependence appears when a change in one task requires changes in other tasks being dependent on this task. The MDA approach could be used for the project part with more complex problems to be solved and the agile approach for the other part. The project part with the less complex problems can be much bigger, because there can be many problems of this kind to be solved. Using the agile approach for this part can shorten the project time very significantly.

References

[1] J. Arlow and I. Neustadt, “UML 2 and the Unified Process: Practical Object-Oriented Analysis and Design,” 2nd ed., Pearson Education, Inc, Addison Wesley Professional, 2005

[2] F. Truyen, “The Fast Guide to Model Driven Architecture – The Basis of Model Driven Architecture”, Cephas Consulting Corp, Whitepaper, 2006

[3] Composite authors, “Manifesto for Agile Software Development,” available at: http://agilemanifesto.org/

[4] M. Tavač, V. Tavač, “DBRE and MDA integration,” In: Objekty 2011 proceedings of the 16th international conference on object oriented Technologies, Žilina, Slovakia, November 2011, ISBN 978-80-554-0432-5, pp. 52-65

[5] M. Tavač and V. Tavač, “The General Algorithm for the Design of the MDA Transformation Models,” In: CICSyN2013, Fifth International Conference on Computational Intelligence, Communication Systems and Networks, Madrid, Spain, 5-7 June 2013, ISBN 978-0-7685-5042-8, pp. 171-179

[6] M. Meško, E. Kršák and P. Hrkút, “The recursive segment 3D reconstruction algorithm,” In: In: CICSyN2013, Fifth International Conference on Computational Intelligence, Communication Systems and Networks, Madrid, Spain, 5-7 June 2013, ISBN 978-0-7685-5042-8, pp. 261 – 264

[7] Scrum Org., ScrumInc., SCRUM Guides, Retrieved from: http://www.scrumguides.org/