Suggested Framework for Agile MDA and Agile Methodologies

Dr. Asim El-Sheikh
Arab Academy for Banking and Financial Science
www.aabfs.org
A.Elsheikh@aabfs.org

Dr. Ahmed Omran
Arab International University
www.aiu.edu.sy
a-omran@aiu.ac.sy

ABSTRACT

Modeling is an essential part in software development projects. This shifts the focus of software development from writing code to building models. Consequently Object Management Group (OMG) introduced Model Driven Architecture (MDA). MDA is the open approach to control both business and technology changes. MDA is created to help UML designers to cover all specification of the software system with all levels of modeling to move systematically from requirement phase towards design and coding phases. This paper focuses on the Business-Technology gap in software projects as a research problem. A new framework for Agile Methodologies will be introduced as a general guideline for the Agile Software Projects. This framework will integrate MDA capabilities and enhances the development process in Agile Methodologies and Agile Project Management. The empirical result is Agile Methodologies (with MDA capabilities) can decrease the negative effects of the research problem during the development of Information System (IS) projects, and consequently increases potential capabilities in Agile Methodologies.

General Terms

Management, Design, Experimentation, Standardization, Languages, Verification

Keywords

Agile Methodologies, UML, MDA, Agile MDA, Agile Methodology Framework, IT-Business gap

1. INTRODUCTION

Model driven development (MDD) is a major trend in software development during the next decade [4]. MDA provides a set of guidelines for the structuring and specifications of the software systems to be expressed as models [6]. MDA defines software system functionality using multiple levels of modeling. Computing Independent model (CIM) where business models are expressed at high level of abstraction. Platform Independent Model (PIM) is generated with details about abstracted technical solutions. Finally, PIM is translated into one or more platform-specific models (PSMs) that computer can run. See Figure1[7].

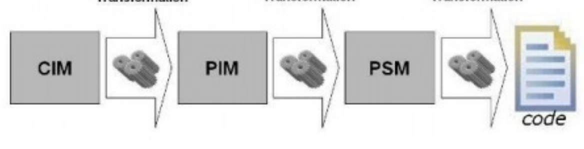

Figure 1: Modeling concepts in MDA

Agile Methodologies trend will meet MDA trend. Consequently Agile Software will gain MDA capabilities during Projects life cycles [1].

2. From UML and MDA to Agile MDA

2.1 UML and MDA

Under the technical leadership of the Three Amigos, an international consortium called the UML Partners was organized in 1996 to complete the Unified Modeling Language (UML) specification. The UML 1.0 specification draft was proposed to the OMG in January 1997 [11][17]. Concepts in software engineering from many other OO methods were modified to be integrated with UML. As a result, the Unified Modeling Language is the international standard.

When OMG began in 1989, the focus was on object-oriented middleware standards and CORBA was developed. OMG decided that their goal could only be achieved by supporting an open system of models and interfaces that were independent of any specific programming language, operating system or network protocol. The Model Driven Architecture (MDA) is one of their open systems as shown in Figure 2.

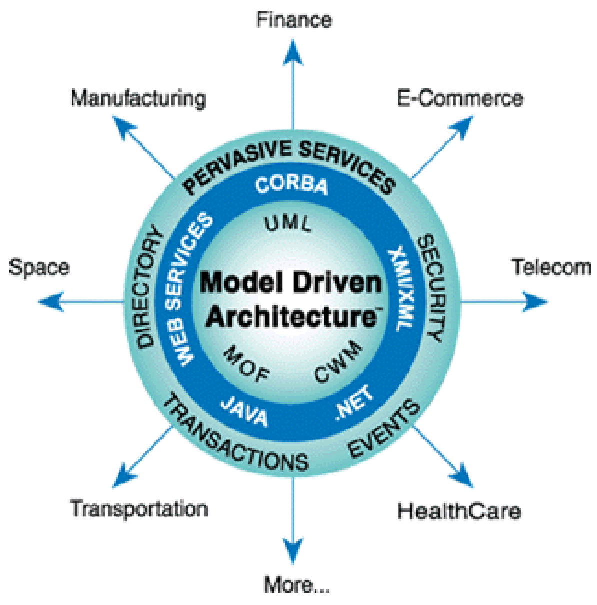

Figure 2: Model Driven Architecture (MDA)

The idea behind the OMG's new Model Driven Architecture is simplicity itself. What the OMG proposes is that companies create high-level UML models of how applications will be structured and integrated. These descriptions will be independent of any actual implementation details [2]. From such a high-level UML model, which the MDA architecture terms a Platform-Independent Model or PIM, a more constrained UML design, termed a Platform-Specific Model or PSM, can be generated. A PSM design can then be converted into language code designed for a specific platform [12]. See Figure 3.

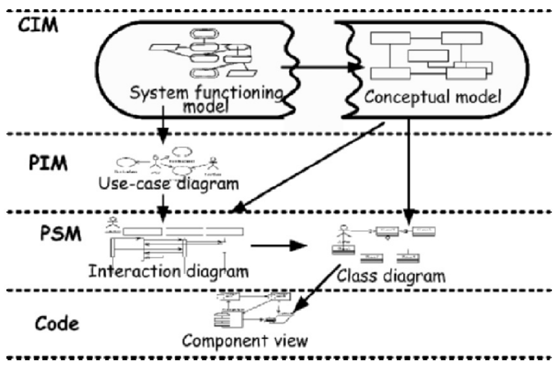

Figure 3: MDA and levels of system modeling

2.2 Agile MDA

People think of models as blueprints that are filled in with code, and model-driven development supporting automation of transformations between these several models. Agile MDA has small different perspectives. Agile MDA' models must be complete enough that they can be executed standing alone. There are no "analysis" or "design" models, because all models are equal. Models are linked together, rather than transformed, and they are then all mapped to a single combined model that is then translated into code according to single system architecture. This approach to MDA is called Agile MDA [10].

This perspective allows delivering a running system in small increments in direct communication with customers. In this sense, executable models act just like code, though they provide the ability to interact better with the customer's domain [3].

3. Research Problem

MDA and Agile Methodologies will be integrated in order to generate new capabilities that can help and affect positively software projects. By doing such integration companies will gain new potential capabilities in Agile Methodologies that will lead to overcome old obstacles in the Agile projects. Literature review will be discussed to explain the research problem (IT-Business gap). This includes literature of what is Agile MDA? Why to Agile MDA? Finally a set of factors for Agile projects is implemented and tuned at IBM in order to align business and technical disciplines. Ambler discussed new capabilities in Agile Methodologies for big and complex software projects [8].

In this paper, a framework that maps Agile MDA with Agile Methodologies will be introduced as a complete framework of process model for project development. Number of small software projects will be used as case studies to test this framework.

4. Expected Contribution

This study addresses an important problem in Software Engineering (IT-Business gap). The introduced framework provides a step towards Agile Methodologies and MDA to enhance the development process of software projects, and consequently decrease the research problem.

5. Literature Review

OMG introduced MDA but many literatures discussed the need to Agile MDA in order to be applied in Agile Projects. Mellor discussed Agile and MDA as two concepts and they are not in contrast. These two concepts can provide capabilities in certain contexts. Executable UML models are the Agile view of MDA. Executable models (Agile MDA) should be integrated (not transformed) in order to build the whole functionality of the system [5]. Lazar et al. introduced general outlines how these principles can be mapped and applied to MDA concepts. They discuss how Agile development processes tend to minimize the modeling phase and the usage of UML models, constructing executable models using the existing tools. Agile MDA processes try to apply Agile principles in the context of executable models [3]. Mellor and Balcer published their book: "Executable UML a Foundation for Model-Driven Architecture and explained how most people think of models as blueprints that are filled in with code, so MDA is commonly viewed as supporting "heavyweight" process-heavy modeling techniques, but MDA can do better than this [9].

The following Literature integrated MDA and Agile Methodologies. Montogna et al. introduced an Agile form of MDA and introduced their framework that allows the development of executable model based on Agile principles and service oriented methodologies. Their framework depends on three layer of architecture (Services, Structure, and Deployment) and testing concepts (Add a test, Run the test, Add production code, and Run the test) [13].

Ambler introduced four interrelated papers. He discussed number of important questions and ideas for organizations in order to implement it. MDA as is may be questionable issue to be applied. He explained number organizational and environmental factors will affect MDA in software projects. First one is (1- Are You Ready For the MDA?) In this paper Ambler asked how to make MDA work in the firm? What are important roles of project's stakeholders. Second one is (2- Examining the Model Driven Architecture (MDA). In this paper Ambler argues that MDA should be checked in the context of the enterprise environment before start embracing it. How this MDA will be represented? how to be tested? Finally (3- A Roadmap for Agile MDA) in this paper Ambler argues that any Agile organization can use MDA integrated into their methodologies. He explained and discussed all processes, tools, and techniques to generate the required deliverables. Ambler in his fourth paper explains how to enable Agile Methodologies in the complex systems. Actually this paper is a case study at IBM and it introduces best practices and guidelines in order to keep the development team Agile in such environment: The Agile Scaling Model (ASM): Adapting Agile Methods for Complex Environments.

Ambler introduced "The Agile Scaling Model (ASM)" framework to defines a set of factors that can draw the roadmap for effective adoption and tailoring of Agile strategies to meet unique challenges faced by a system delivery team. At IBM Agile techniques held such promise to adopt Agile processes on a wide-scale basis throughout IBM Software Group, an organization with over 25,000 developers." [8].

6. Suggested Framework for Integration of Agile MDA and Agile Methodologies using IEEE1074 and IEEE12207

Projects' case studies in this research should follow the introduced framework during the Software Project Life Cycle. This framework will guide developers to apply Agile MDA in their Agile SDLC. There will be no "analysis" or "design" models, because all models are equal in this framework. Models are linked together, rather than transformed, and they are then all mapped to a single final combined model. Finally Models are translated into code according to single system architecture. This framework will be developed as a general guidance for the most common Agile Methodologies: Extreme Programming (XP), Scrum, Dynamic Systems Development Method (DSDM), Adaptive Software Development (ASD), and Crystal. Two important IEEE standards are used during the creation of this framework:

IEEE 1074 is used as a guideline for creating a common framework for developing life cycle models. It describes the activities and processes needed for the development and maintenance of software, and provides examples for typical situations.

IEEE12207 standard is used for evaluating Software Life Cycle Processes. This standard defines a set of processes that cover the entire life-cycle of a software system, which are in turn defined in terms of activities. The activities are broken down into a set of tasks.

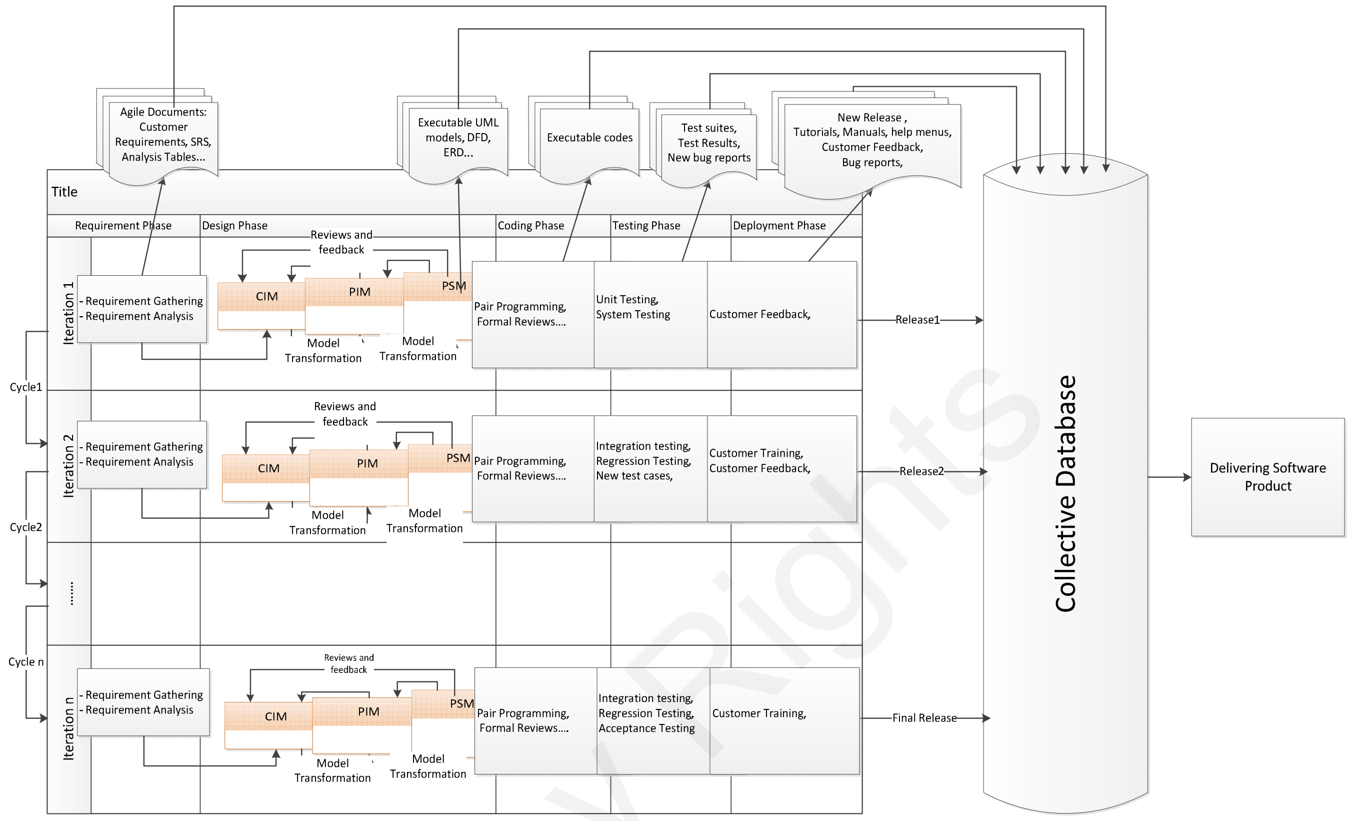

Figure 4: Empirical Framework of Agile MDA and Agile Methodologies

Two important issues should be followed in this framework: Continuous Integration and Continuous Testing. The project should be divided into pieces of software (iterations) that can be done in a few of days (each Agile Methodology has its own estimation criteria). Developers should begin by taking a copy of the highest level of system abstracted model from the development machine. This should be done by using document management system or centralized DB, then checking out a working copy. A document control system keeps all of a project's documents in a repository. At any time a developer can check out a controlled copy of their centralized machine. This copy on the developer's machine is called a 'working copy. Finally this will be checked in with new updates. This should cover both developing and testing deliverables. Once the first iteration is done, a new build with all its documents is controlled by the centralized repository on the development machine.

7. Project case-studies and results

Appendix A shows a list of the small project used as case-studies for testing the framework. Appendix B shows the MDA and UML models used in these project case studies.

The following tables and diagrams outline descriptive statistics of MDA and UML models in the projects' case studies:

Table 1: Modeling Diagrams in Agile Projects

Modeling Diagrams	CIM	PIM	PSM
Use case	6		
DFD	1	1	
Class diagram		3	2
Sequence diagram		4	1
Activity diagram		3	
Component diagram			3

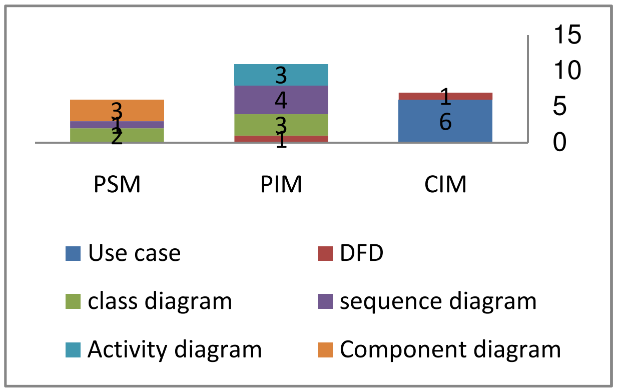

Figure 4: Modeling Diagrams in Agile Projects

Findings in Table1 and Figure5 meet with Unhelkar's findings as shown in Table3 and Table4. This is affected mainly by training when developers developed their projects (All Projects introduced in this paper are all academic junior or senior projects).

Appendix B shows that use-case diagrams are used mainly in CIM, while in PIM all UML models are used except component and deployment diagrams. PSM has Component and class diagrams. Some projects expressed their PSM as standard format of code and meaningful comments which is accepted from Agile perspective and principles. One project shows that DFD (context level only) can be used also in CIM as shown in Figure5 and Table1. Mellor provided general guidelines to how categorize UML models into CIM such as use case models and PIM models such as class diagram and sequence diagram. These PIM models should represent the executable models. He explained executable models to be as structural code with standard meaningful comments [5]. This meets results from Ambler’s except for the DFD diagram is in PIM level in Ambler’s framework as shown in Figure11 and Figure12 [15]. Unhelkar provided similar results as shown in Table3 [14]. These results can be interpreted as CIM should clearly shows system requirements of the system. This is clear when DFD level1 appear in PIM. According to Fernandes', the combination of object-oriented and functional approaches (such as DFD) can give useful results in some specific contexts [16].

8. Interpretation and conclusion

In this paper, an empirical framework for Agile MDA is introduced and tested using number of small software projects. Results of these case-studies addressed the Technology-Business problem in Agile software development process by enhancing the process of software development and management in Agile Methodologies.

The proposed framework integrated Agile Methodologies and Agile MDA. Future work is to generate new separated frameworks for each Agile Methodology (These frameworks should be derived from the proposed one in this thesis). Number of small projects should be developed as case studies projects to generate empirical forms of these frameworks. According to differences in Agile Methodologies, such work would certainly involve a changes and modifications of the major phases of framework presented in this study. These works with new frameworks will create new Architectural roles in each methodology. Other points for future work are the new capabilities of Agile Methodologies for big software projects (such as Enterprise System) and heavy processes projects within gloomy environments. This is an important issue that needs to be studied in details as potential capabilities for software projects.

9. References

[1] Boehm B., 2006, ”A View of 20th and 21st Century Software Engineering”, ACM, International Conference on Software Engineering, Proceedings of the 28th international conference on Software engineering, Shanghai, China, Pages: 12 – 29,Publisher: ACM New York, NY, USA

[2] Probst R., “Model Driven Architecture: How systems will be built”, http://www.omg.org/mda/ [last visited 2010-03-24]

[3] LazarI., ParvB.,MotognaS., CzibulaI., 2007,”An Agile MDA Approach for Executable UML Structured Activities”, http://citeseerx.ksu.edu.sa/viewdoc/summary?doi=10.1.1.104.1083,

[4] Doernhoefer M., 2007, ”Surfing the Net for Software Engineering Notes”,ACM SIGSOFT Software Engineering Notes archive, Volume 32 , Issue 2, Pages: 8 – 17, Publisher: ACM New York, NY, USA

[5] MellorS.,”Agile MDA”,

http://www.omg.org/mda/mda_files/Agile_MDA.pdf [last visited 2010-04-16]

[6] Pham H., Mahmoud Q., Ferworn A., Sadeghian A., 2007, ”Applying Model-Driven Development to Pervasive System Engineering”, IEEE, Proceedings of the 29th International Conference on Software Engineering Workshops, Page 193, Publisher: IEEE Computer Society Washington, DC, USA

[7] Miller J., Mukerji J., 2003,”MDA Guide”, OMG, 2003, http://www.omg.org/cgi-bin/doc?omg/03-06-01[last visited 2010-04-21]

[8] Ambler S., 2010,”The Agile Scaling Model (ASM): Adapting Agile Methods for Complex Environments”,ftp://ftp.software.ibm.com/common/ssi/sa/wh/n/raw14204usen/RAW14204USEN.PDF [last visited 2010-01-12]

[9] Mellor S., Balcer M., 2002,”Executable UML: A Foundation for Model-Driven Architecture”, Publisher: Addison Wesley, 2002, ISBN: 0-201-74804-5

[10] Klimes C., Prochazka J., 2008, "New Approaches In Software Development", http://www.elearning-conf.org/2010/Preliminary_Program_EL_2010.pdf, [last visited 2011-01-18]

[11] Fowler M., 2004, "UML Distilled: A Brief Guide to the Standard Object Modeling Language", Third Edition, Publisher: Addison Wesley, 2004, ISBN 0-321-19368-7

[12] Pavlova N., 2007, “Several Outlines of Graph Theory in Framework of MDA”, Publisher: Springer, 2007, 25-36, DOI: 10.1007/978-0-387-70802-7_3

[13] Motogna S., Lazar I., Parv B., Czibula I., 2009, “An Agile MDA Approach for Service-Oriented Components “, ACM, Electronic Notes in Theoretical Computer Science (ENTCS) , Volume 253 Issue 1

[14] Unhelkar B., 2003, "Process quality assurance for UML-based projects", publisher: Pearson Education, Inc., 2003, ISBN 0-201-75821-0, Chapter1, page 51

[15] Ambler S., 2004,” A Roadmap for Agile MDA”, www.agilemodeling.com/essays/agileMDA.htm [last visited 2010-04-03]

[16] Fernandes J., Lilius J., Truscan D., 2006, "Integration of DFDs into a UML-based Model-driven Engineering Approach", Publisher: Springer, Software and Systems Modeling , Volume 5, Number 4, 403-428, DOI: 10.1007/s10270-006-0013-0

[17] OMG Unified Modeling Language (OMG UML), 2010, “UML 2.3 - Infrastructure”, Version 2.3, OMG Document Number: formal/2010-05-05,http://www.omg.org/spec/UML/2.3/Superstructure [last visited 2011-04-18]

Appendix A: List of Projects case studies

The following projects are developed at AIU University (www.aiu.edu.sy). All projects are developed as Academic junior and senior projects. The original projects were 18 projects. Only 6 projects are selected only for those followed the framework completely.

Table 2: List of Projects Case Studies

Project Title	Process Model	# Team Members
Hospital Management Information System (HMIS)	XP	3
Hospital Information System	Scrum	3
E-learning System FOR AIU university	XP	3
Stock Market Information System	XP	3
Al-Shefaa Hospital Information System	Scrum	2
School Information System	XP	2

Appendix B: MDA models in Projects case-studies

Table 3: MDA models in Projects case-studies

		MDA models		
Project Title	Process Model	CIM	PIM	PSM
			UML models	
Hospital Management Information System (HMIS)	XP	use-case	class diagram, sequence diagram	Component diagram
Hospital Information System	XP	use-case, DFD (Context Level)	class diagram, sequence diagram, DFD (level 1)	Component diagram
E-learning System FOR AIU university	XP	use-case	class diagram, sequence diagram, Activity diagram	
Stock Market	XP	use-case	sequence diagram, Activity diagram	class diagram
Al-Shefaa Hospital Information System	Scrum	use-case	class diagram, sequence diagram	Component diagram
School Information System	Scrum	use-case	Activity diagrams	class diagram, sequence diagram

Appendix C: UML models in Agile projects

The following tables summarize the UML diagrams and the modeling aspect of a software solution represented by them [14]. These were used as guidelines for modeling in the case studies' projects.

Table 4: UML diagrams in practice

UML diagrams	Model representing
Use case diagrams	functionality from user's viewpoint
Activity diagrams	the flow within a use case or the system
Class diagrams	classes, entities, business domain, database
Sequence diagrams	the interactions between objects
Collaboration diagrams	the interactions between objects
Object diagrams	objects and their links
State chart diagrams	the lifecycle of an object in real time
Component diagrams	the executables, linkable libraries, etc.
Deployment diagrams	the hardware nodes, processors, and optionally, corresponding components
Package diagrams	subsystems, organizational units

Further to this description of the focus of the UML diagrams, the following table shows the relative and importance of each of the UML diagrams in each of the modeling spaces and to each of the major modeling roles within the project. While project team members can work in any of these modeling spaces using any of the UML diagrams, good quality models will result by understanding the importance of the diagrams with respect to each of the modeling spaces. This is shown in this table.

Table 5: Importance of UML diagrams to respective models

UML diagrams	(Business Analyst)	(Designer)	(Architect)
Use case diagrams	*****	**	*
Activity diagrams	*****	**	*
Class diagrams	***	*****	**
Sequence diagrams	***	*****	*
Collaboration diagrams	**	*	
Object diagrams	*	*****	***
State chart diagrams	***	****	**
Component diagrams	*	***	*****
Deployment diagrams	**	**	*****
Package diagrams	***	**	****

* = least important, ***** = most important

Table 3 and Table 4 will be used in the projects case studies in order to standardize modeling perspectives.