



UNIVERSITY OF APPLIED SCIENCES DARMSTADT

AS PART OF THE WAI SEMINAR
SUMMER SEMESTER 2016

The Way to Agile Modeling and its use in eXtreme Programming

Anwar Benhamada

supervised by
Prof. Dr. Inge SCHESTAG

June 15, 2016

Abstract

This research involves explaining the essence of Agile Modeling and how it reacts with the agile methodologies like eXtreme Programming, Scrum and Crystal Clear. In the case of eXtreme Programming, the challenge of Agile Modeling is to integrate the different phases of the adopted methodology. eXtreme programming is taken as an example of an agile methodology with the goal to clarify the common points with the agile modeling. It is compared between the agile modeling practices and the eXtreme programming values and practices in order find a potential fit and build a relationship based on practical experiences from the founder of the agile modeling.

Contents

1	Introduction	4
2	History of AM and Agile Manifest	6
3	Components of Agile Modeling	7
3.1	Agile Modeling values	7
3.1.1	Communication	7
3.1.2	Simplicity	7
3.1.3	Feedback	7
3.1.4	Courage	7
3.1.5	Humility	8
3.1.6	Beyond Motherhood and Apple Pie	8
3.2	Agile Modeling Principles	8
3.2.1	Core Principles	8
3.2.2	Supplementary Principles	9
3.3	AM Practices	9
3.3.1	Core Practices	9
3.3.2	Supplementary Practices	10
4	What is eXtreme Programming?	11
4.1	4.1 eXtreme Programming values, principles and practices	11
4.1.1	4.2. The potential fit between Agile Modeling and eXtreme Pro- gramming	12
5	Conclusion	15
	References	15

Erklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die im Literaturverzeichnis angegebenen Quellen benutzt habe. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder noch nicht veröffentlichten Quellen entnommen sind, sind als solche kenntlich gemacht. Die Zeichnungen oder Abbildungen in dieser Arbeit sind von mir selbst erstellt worden oder mit einem entsprechenden Quellenachweis versehen. Diese Arbeit ist in gleicher oder ähnlicher Form noch bei keiner anderen Prüfungsbehörde eingereicht worden.

Darmstadt, den

Unterschrift

1 Introduction

Software engineering has been going in the last two decades through tremendous changes, not only in the way the final product is brought into being but also how the developer deals with the internal and the external factors of his work.

“The primary goal of software development is to build systems, in the most effective and efficient manner possible” [Mar14, p.4]. A system can be produced through the traditional waterfall model which is based on the principle that the work progress flows from the top to the bottom.

Another modeling procedure is the Agile modeling (AM) which I’m focusing on in this research. Firstly what’s AM ? And why is everyone talking about?

“Agile Modeling (AM) is a practice-based methodology for effective modeling and documentation of software-based systems. At a high level AM is a collection of best practices, depicted in the pattern language map below. At a more detailed level AM is a collection of values, principles, and practices for modeling software that can be applied on a software development project in an effective and light-weight manner.” [Amb14a]

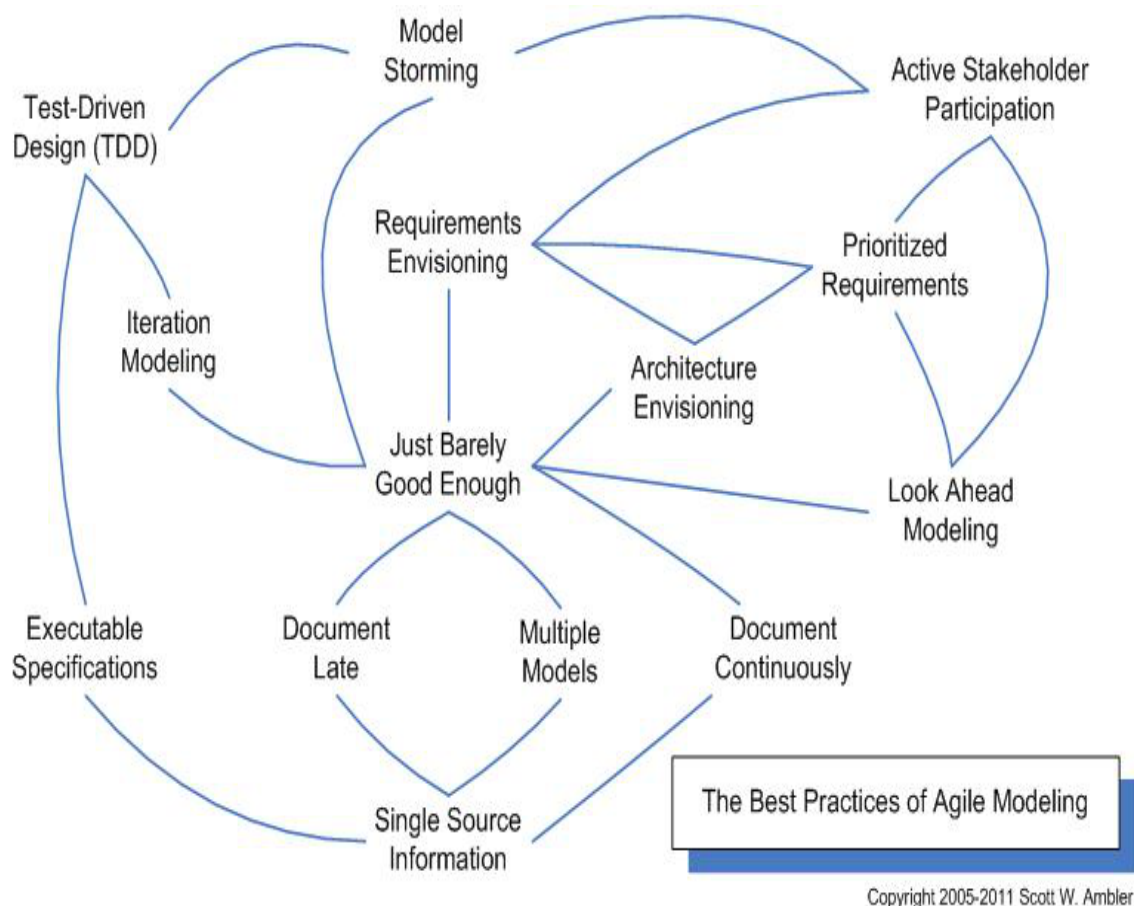


Figure 1. (Scott W.Ambler) The Best Practices Of Agile Modeling [Amb14a]

Since the meaning of the AM is understood, the question now is what's not AM?

AM is an attitude, not a prescriptive process (Based on[Amb02])

AM is not a cookbook that have to be followed by word, it describes a style of modeling and not how to model.

AM is a supplement to existing methods, it is not a complete methodology

The main focus of the AM is on modeling, the second focus is documentation. AM should be used to support teams using agile methodologies such as eXtreme Programming, Scrum and Crystal Clear and not as a separate methodology.

AM is effective and is about being effective

AM describes effectiveness as the absolute minimum to suffice a specific job, that means that the team should always document only what they need and have a clear purpose for that.

2 History of AM and Agile Manifest

The development of agile modeling was led by Scott Ambler starting in the autumn of 2000. It initially was called **eXtreme modeling**(XM) but at the suggestion of Robert Cecil Martin was renamed to AM in the spring of 2001. The book *Agile Modeling* was published in 2002 by *John Wiley Press*. Work on the methodology continues at The Agile Modeling Home Page. [Wik16]

AM is based on the Agile manifest which is a bench of principles that the developer can lay on while trying to get his job done. Following the principles is not always easy but when the team decides to adopt them, they will master the progress of the future software.

Principle 1: Individuals and practices over processes and tools (Based on [Amb02, pp. 6–7])

This doesn't mean that processes and tools should be eliminated, it simply means that a good a face to face discussion should be prioritized on rigid workflows.

Principle 2: Working software over comprehensive documentation

Traditional software development often produces extensive documentation before the program is released for an initial testing, at the end it would be better having a program than a book describing it.

Principle 3: Customer collaboration over contract negotiation

Initial guidelines are a good basis to start but by defining the exact details of the project before it starts will limit the customer. Team and costumer should collaborate to find the best solutions for the assignment.

Principle 4: Responding to changes over following a plan

Nothing goes entirely according to plan, instead of remaining on the same plan, it's effective to make adjustments as the situation changes.

3 Components of Agile Modeling

The challenge of the AM was to find a high-level values, principles and practices to manage a coherence for the Programmer adopting it. It is a difficult task to simplify those components in a way that the reader actually understand them like they really are, that's one side, the other side they should be detailed to cover all aspects and commonly errors made by users.(Based on[Amb02, p. 9])

For these reasons the founder Scott Ambler has put values regulate the work and to be a foundation to refer to when the software modeling is not going well.

3.1 Agile Modeling values

3.1.1 Communication

It's not easy to define communication, in the *Merriam Webster's collegiate Dictionary 10th Edition* [Mer99] communication is defined as “ a process by which information is exchanged between individuals through a common system of symbols, signs or behavior” In the real software development world a lot of misunderstanding results from the fact that the team members have different views and approaches to the same point or problem, so communication will probably guarantee that same things have the same meaning by all team members and eventually try to find a quick middle path.(Based on[Amb02])

3.1.2 Simplicity

Simplicity is certainly one of the most important values, it follows the famous rule *KISS (Keep It Simple Stupid)*. For that some actions must be avoided to not complicate the software and making it harder to develop, test and maintain. These is a list of some of the actions to avoid: - Applying complex patterns too soon - Over- architecting the system to support potential future requirements - To develop a complex infrastructure

3.1.3 Feedback

The best way to get a serious evaluation of the software is to get an internal feedback from the workteam or an external one throughout a survey. The internal feedback is regarding the models, models are abstraction of how people will work with the system. In order to obtain a feedback regarding a model, one (or more) of these methods can be used:
Develop the model as a team - review the model with your target audience - implement the model - acceptance testing

3.1.4 Courage

Since the agile modeling or the agile software development in general is new to most people, there will be a fear to change the current situation to adopt the agile modeling. It's easier to follow a hierarchy of several superiors and wait days and even weeks to accomplish a simple task, but that's the point of agile modeling which relays on the principles that the members of the development team should have the courage to make decisions even if they are not very comfortable in taking this step.

3.1.5 Humility

Agile modelers are persons who are ready to say that they don't know everything. This characteristic is so important so they can be able to learn new skills, concepts from other team members including the junior person on the team. They understand that every member has more experience in coding or testing or he/she is better at requirements modeling. The humility to ask for help when someone has no clue is required in AM. The humility to respect of the people working with the agile modeler is also required, calling managers or other workteam members with inappropriate names should not happen.

3.1.6 Beyond Motherhood and Apple Pie

So the agile modelers seek simplicity, always waiting for a feedback, they know how to communicate and stay humble. After all the agile modelers are people, so they can't follow these values all the time, but they should try to set an agile mindset in their work and build a culture that supports agile and effective development efforts. The values are used in combination with the principles and the practices in order to achieve a sustainable strategy to develop softwares.

3.2 Agile Modeling Principles

When the AM values (communication, simplicity, feedback, courage, and humility) are combined to produce the modeling principles. They are divided in core and supplementary where in the first is indispensable and the second is good to fulfill but not a must.

3.2.1 Core Principles

Software is your primary goal (Based on[Amb02, pp. 27–43])

The primary goal is to produce high-quality software that adapts with the requirements of the client, writing massive documentation and drawing models can give a feeling that the team is in progress but that's not true, the real progress is to deliver a working software in first place and then work to deliver a software that meets with the goals set in first place. All activities that don't contribute in delivering a software should be avoided.

Model With A Purpose

In the stage of modeling many developers draft their artifacts without precising a goal behind it, so with time this operation becomes ambiguous and very difficult to manage. What is meant by "A Purpose" is that the developers should clarify two main points, firstly the definition of a valid purpose for creating a model, secondly define the audience receiving that model.

Maximize Stakeholder Return on Investment

The project stakeholders are investing their money and are waiting from the developers to fully satisfy their needs, in other words the software should have a high ROI (Return on Investment). The stakeholders have the final say in resources investment. Other principles of the Agile Modeling are : Enabling the next effort is your secondary goal Travel Light Assume Simplicity Embrace Change Incremental change Multiple Models Quality Work Rapid Feedback

3.2.2 Supplementary Principles

The supplementary principles are concepts which help the effectiveness of the agile modeling, their adoption is not required for the full functionality of the agile software development. The supplementary principles are:

- Content is more important than representation
- Everyone can learn from everyone else
- Know your models
- Local adaptation
- Open and honest communication
- Work with people's instincts

3.3 AM Practices

Practices are what the developer or the modeler actually apply on his work, the complementarity between the core and supplementary practices is the key to adopting agile modeling.

3.3.1 Core Practices

To get a real agile modeling environment the founder Mr Scott Ambler recommends that all core practices should be adopted if they fit well with the organization's culture.

Core practices are organized into four categories and have practices within them:

1. Iterative and Incremental Modeling (Based on [Amb02, pp. 44–59])

- Apply the Right Artifact(s)
- Create Several Models in Parallel
- Iterate to Another Artifact
- Model in Small Increments

The iterative and incremental modeling AM focuses on the goal behind the every model, it recommends that whenever a change takes place in model, all related models should be updated automatically

2. Teamwork
- Model With Others
 - Active Stakeholder Participation
 - Collective Ownership
 - Display Models Publicly

Teamwork is a key component in AM, the interaction between the team members themselves or between them and the stakeholder are trivial for the AM.

3. Simplicity

- Create Simple Content
- Depict Models Simply
- Use the Simplest Tools

AM recommends to keep both models and tools as simple as possible, all additional aspects should not be added unless they are justifiable.

4. Validation

- Consider Testability
- Prove it With Code

Agile modelers and developers consider writing a testing for software at the early as soon as possible, preferably before even coding. This gives the chance to control all the aspects of the model or the software and be able to correct any existing errors.

3.3.2 Supplementary Practices

Similarly to the supplementary principles of AM, the team may adopt the supplementary practices. Supplementary practices are organized into four categories and have practices within them:

1. Productivity (Based on [pp. 61–71][Amb02])

- Apply Modeling Patterns Gently
- Reuse Existing Resources

The patterns should be used in such a way as to implement the minimal functionality that the modeler need but that makes it easy to refactor. The reuse of existing resources is a great benefit for the modeler and also for the stakeholder (*Maximize Stakeholder Return on Investment*).

2. Documentation

- Discard Temporary Models
- Formalize Contact Models
- Update Only when it Hurts

The models don't have to be perfect to provide value that is why it's recommended to model only when needed. The effort of updating of the models is in most case more painful than not having the model updated.

3. Motivation

- Model to communicate
- Model to understand

The modeler should model to explore the problem space and be able to test whether his model satisfies the requirements of the project. Model is also the way to communicate, the modeler should have in mind who needs the model and the goal behind that model.

4 What is eXtreme Programming?

As mentioned earlier in section 1, agile modeling is a supplement to existing methods, it is not a complete methodology. eXtreme programming, and Crystal Clear are the widest used among those methodologies because they showed best results and have been tested by a large number of agile developers.

eXtreme Programming (XP) is the best-known of the agile methodology of the agile methodologies. It was developed by three participants of the agile-alliance meeting, Kent Beck, Ward Cunningham and Ron Jeffries, in the late 1990s, and became popular in 1999 (first edition of [Bec04]). XP stresses customer satisfaction. The main goal of eXtreme Programming is to reduce the cost of change. In traditional system development methods, requirements are determined in the beginning and often fixed from that point on. [Kur08]

4.1 4.1 eXtreme Programming values, principles and practices

The eXtreme programming values, principles are almost identical with the ones of agile modeling so they won't be analyzed in depth.

The eXtreme programming values are :

- Communication
- Simplicity
- Feedback
- Courage
- Respect

The values of the XP and the AM are similar and represent the agile philosophy. The value simplicity is often understood as avoiding documentation, but the documentation here is reduced through the verbal communication between team members.

Principles are :

- Humanity
- Economics
- Self-similarity
- Improvement
- Diversity
- Reflection
- Flow
- Opportunity
- Redundancy

- Failure
- Quality
- Baby steps
- Accepted responsibility

The practices are what the XP teams will be doing day-to-day. As in agile modeling the practices are divided into core and supplementary practices.

Some of the core practices are:

- Sit together
- Whole Team
- Informative Workspace
- Energized Work
- Pair Programming
- Stories
- Weekly Cycle

(Based on [Bec04])

4.1.1 4.2. The potential fit between Agile Modeling and eXtreme Programming

As already seen in the previous chapter, the similarity of the AM and XP are obvious but the question is how to apply every AM practice within XP ?

AM Practice	Fit With XP
Active Stakeholder Participation	This practice is simply a new take on XP's On-Site Customer practice. AM uses the term project stakeholder in place of customer and focuses on the concept of their active participation, hence Active Stakeholder Participation and not On-Site Stakeholder.
Apply Modeling Standards	This is the AM version of XP's Coding Standards practice.
Apply Patterns Gently	This practice reflects the YAGNI principle to the effective application of patterns within your system, in conformance to XP's practice of Simple Design.
Apply the Right Artifact(s)	This practice is not explicitly described by XP principles and practices although is very much aligned with XP philosophies of "if you need it do it" and using the most appropriate tool or technique for the job at hand.
Collective Ownership	AM has adopted XP's Collective Ownership practice.
Create Several Models in Parallel	This is a modeling-specific practice. XP developers can clearly work on several models – such as CRC cards, acceptance test cases, and sketches – if they choose to do so.
Create Simple Content	This is complementary XP's Simple Design practice that advises to keep your models as simple as possible.
Depict Models Simply	This is complementary XP's Simple Design practice that suggests that your models do not need to be fancy to be effective, perfect examples of which are CRC cards and user stories.
Discard Temporary Models	This practice reflects XP's Travel Light principle, which AM has adopted, explicitly advising you to dispose of models that you no longer need.
Display Models Publicly	This practice reflects XP's (and AM's) value of Communication, principle of Open and Honest Communication (adopted by AM), and reflects its practice of Collective Ownership.
Formalize Contract Models	This practice is not currently reflected within XP, well perhaps in its "if you need to then do it" philosophy. This practice was included in AM to provide guidance for how to deal with the very common situation of integrating with other systems.
Iterate to Another Artifact	This practice explicitly states, in a general form, the practice of XP developers to iterate between working on various artifacts such as source code, CRC cards, and tests.
Model in Small Increments	This practice supports XP's iterative and increment approach to development. Both XP and AM prefer an emergent approach to development and not a big design up front (BDUF) approach.
Model With Others	This is the AM version of XP's Pair Programming practice.
Prove it With Code	This is the AM version of XP's Concrete Experiments principle. In fact, it was originally called Concrete Experiments although was renamed when it was evolved into a practice.
Reuse Existing Resources	This concept is not explicitly included in XP, although it clearly isn't excluded either. XP developers are practical, if there is something available that can be appropriately reused then they will likely choose to do so.
Single Source Information	The goal of storing information in a single place reflects the XP concept of traveling light.
Update Only When it Hurts	This practice reflects AM and XP's Travel Light principle, advising that you should only update an artifact only when you desperately need to.

Table 1. Applicability of AM Practices on an eXtreme Programming Project.[Amb14b]

The above table shows the possibility how to apply agile modeling practices on a project which already adopted eXtreme programming as a methodology. Not all practices of AM reflect in XP but they are all most likely reflected in the XP philosophy.

5 Conclusion

From the above sections, it's made clear that the agile modeling can be a moving factor in the software development process, the adoption of its values, principles and practices are the key for its success. The supplementary principles and practices will participate in the effectiveness of the process but can be not adopted if they do not match with the organization's culture of the agile modelers or developers team. The values of the agile modeling are inspired from the agile manifest and come to support an agile methodology and not as a seperate one. The XP advices to reduce the effort in documentation and concentrating on producing a working software as soon as possible, the AM advices to be more effective when documenting and modeling. The agile modeling has showed a compatibilty with the eXtreme programming which proves that AM can be a great support for the XP.

References

- [Amb02] Scott W. Ambler. *Agile modeling. Effective practices for eXtreme programming and the unified process*. Wiley computer publishing. New York: J. Wiley, 2002. xvi, 384. ISBN: 9780471202820 (cit. on pp. 5–10).
- [Amb14a] Scott W. Ambler. *Agile Modeling (AM) Home Page: Effective Practices for Modeling and Documentation*. 9.10.2014. URL: <http://agilemodeling.com/> (visited on 06/14/2016) (cit. on p. 4).
- [Amb14b] Scott W. Ambler. *Agile Modeling and eXtreme Programming (XP)*. 9.10.2014. URL: <http://agilemodeling.com/essays/agileModelingXP.htm> (visited on 06/14/2016) (cit. on p. 14).
- [Bec04] Kent Beck. *Extreme Programming Explained: Embrace Change: Embracing Change*. Addison Wesley, 2004. ISBN: 978-0321278654 (cit. on pp. 11 sq.).
- [Kur08] Karl Kurbel. *The making of information systems. Software engineering and management in a globalized world*. Berlin: Springer, 2008. 1 online resource (xii, 591. ISBN: 3540792600 (cit. on p. 11).
- [Mar14] Robert C. Martin. *Agile software development, principles, patterns, and practices*. Pearson new international ed. Martin, Robert C., (author.) Harlow, Essex: Pearson, 2014. 1 online resource (iv, 524. ISBN: 9781292025940 (cit. on p. 4).
- [Mer99] Merriam Webster. *Merriam-Webster's collegiate dictionary*. 10th ed. Springfield, Mass.: Merriam-Webster, 1999. 38a, 1559. ISBN: 9780877797135 (cit. on p. 7).
- [Wik16] Wikipedia, ed. *Agile modeling - Wikipedia, the free encyclopedia*. 6.06.2016. URL: <https://en.wikipedia.org/w/index.php?oldid=723991649> (visited on 06/14/2016) (cit. on p. 6).