Using an Agility Scorecard

Using predictive and adaptive attributes to select the most appropriate management methods for each phase of new product development

Adaptive Project Traits	Value	Predictive Project Traits	Value	Weight	Adaptive Score	Predictive Score
Cheap to fail	1	Expensive to fail	0	10%	10%	0%
Senior developers	1	Junior developers	0	20%	20%	0%
Requirements change often	1	Requirements do not change	0	30%	30%	0%
Small number of developers	1	Large number of developers	0	10%	10%	0%
Culture thrives in chaos	0	Culture that demands order	1	10%	0%	10%
Phase Considerations	0	Phase considerations	1	10%	0%	10%
Industry considerations	1	Industry specific considerations	0	10%	10%	0%
	5		2	100%	80%	20%

Prepared By: Greg Rosenberg

Team: #5

Course: Marketing 548 Fall 2009

Table of Contents

Research Question

My research paper seeks to clarify when is it most advantageous to apply an Agile development method to the new service development process, and how to create a decision model to assist managers in selecting the correct method for each phase of the new product development life cycle.

The activities commonly found in the fuzzy front end and the following five phases of the new product development life cycle have been documented very well in the ninth edition Crawford and Benedetto book. What was missing from their research was a model of how to manage the new product development phases.

To fill this gap in new product development literature, I developed a model for selecting a project management method for each phase of the new product development life cycle. I will focus on developing new software services but these concepts can be applied to products and services in other industries as well.

My research will start with developing an Agility scorecard that calculates the Agility index of a NPD phase. The Agility index of the NPD phase is used to select the most compatible project management method for the phase. My Agility scorecard is based on common attributes that describe project management methods, in terms of their predictive versus adaptive nature. I will extend this model to include NPD phase, industry, and situational specific traits to consider when selecting a project management method. I will demonstrate how to use these attributes to select the most compatible project management method for each phase of new software development, using a case study from the health care industry.

At the fuzzy front end of new product development, the project pipeline looks like the opening of a funnel filled with iterative processes. As you progress through the five phases of new product development the funnel narrows as products become more defined and the outcome becomes more linear and predictable. At each phase in new product development, a management method should be chosen that best fits the attributes of the phase. I developed a model that uses predictive and adaptive attributes to select the most appropriate management methods for each phase of the new product development process.

I will limit the scope of my paper to the software industry since the Agile method was born out of the software industry's need for a process that was consistent with the ways software developers actually performed their work.

With most new products also having a service component, it is becoming harder to separate a product from a service. Even though some characteristics of new service development are more common in the software industry than in other industries, these concepts and models can be applied to the development of new products or services in other industries. This is due to the fact that all modern project management methods in the software development industry are derived from classic scientific management principles.

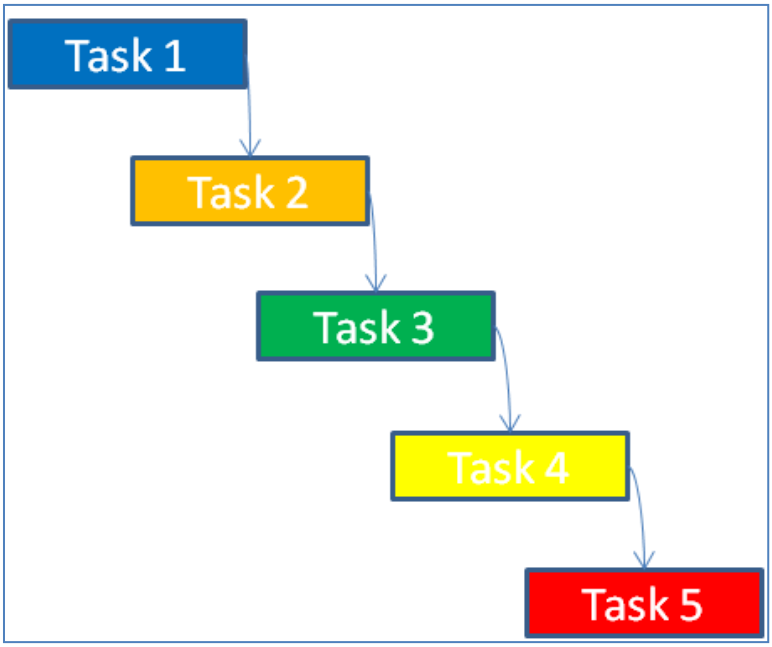
Spectrum of Project Management Methods

Before we can create an agility scorecard we need to review some general project management methods to help guide our discussion. Project management methods are commonly classified by where they fall in the spectrum of Agility. At one end of the agility spectrum we have predictive plan driven methods like PMI and Waterfall. At the other end of the spectrum we have adaptive methods like Agile.

Waterfall's Predictive Flowchart

Traditional predictive project management methods like Waterfall can be visualized using a waterfall shaped flowchart. This reveals the linear and predictive nature of the method.

With Waterfall (Royce 1970) project management methods each activity is broken down into smaller distinct tasks. These tasks are plotted along a timeline in an attempt to create a detailed schedule of work.

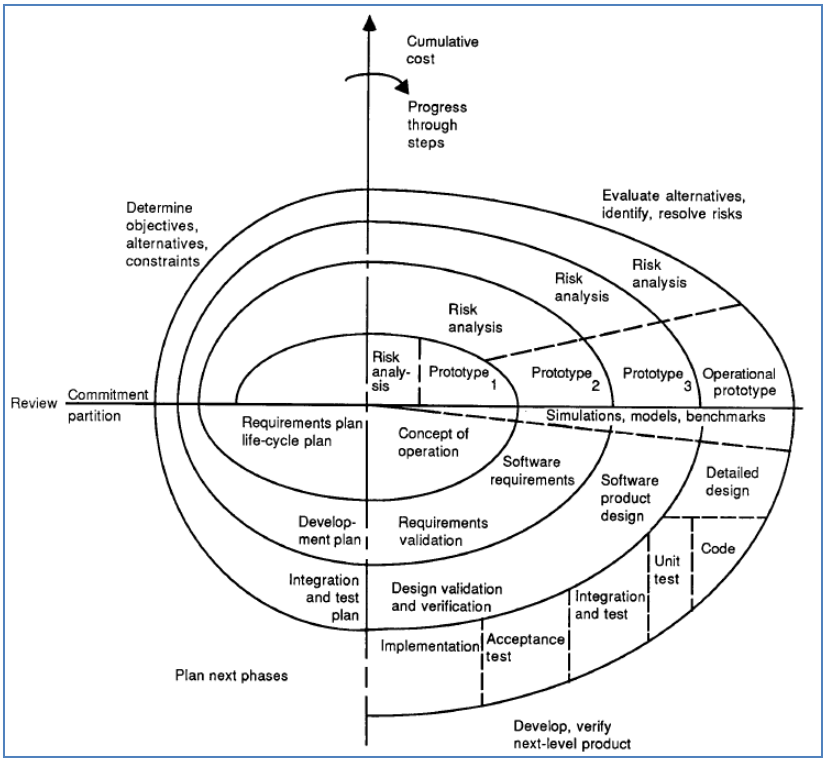


These tasks are completed in sequential order with the output of one task becoming the input for the next task. This process then takes the form of a waterfall shaped flowchart. Like a real waterfalls, the work flows in only one direction from start to finish, in a linear and predictive manner.

This flowchart is commonly displayed as a Gantt chart produced using Microsoft Project. Henry Laurence Gantt (1919) was an American mechanical engineer who is credited with the invention of the Gantt chart. Gantt's work focused on managing large civil construction projects where the work needed to be broken down into small manageable tasks that could easily be scheduled.

Agile Adaptive Spiral Diagram

Adaptive project management methods like Agile can be visualized using a spiral (Boehm 1987) diagram. This reveals the iterative and adaptive nature of this method.



Each circle in the spiral diagram represents one iteration of work to be completed. In each iteration the entire software development life cycle is repeated.

Rather than grouping all the design, development, testing, and deployment activities together into one time phases like in a Waterfall style flowchart, they are repeated iteratively for each feature to be developed.

To do this, each customer requirement is documented as a user story. User stories are then translated into required features that must be developed for the application to satisfy the customer's requirements.

The customer works in close collaboration with the development team. As each feature is developed, the customer communicates directly with the development team to provide feedback to be used in the next iteration. This improves the quality of the software because the developers don't have to wait until the project is nearly complete to know if the customer's needs have changed.

The adaptive nature of an iterative method comes from the direct communication with the customer throughout the development process. As the customer reviews each newly developed feature they develop a better understanding of what is possible. This leads to changes in the customer's requirements. Programmers can easily adapt to changing customer requirements because they are incorporated into the next iteration of work.

What would be considered scope creep in a predictive Waterfall method is valuable feedback that leads to better software in an adaptive Agile project management method.

Introducing Agile Methods

The software development industry has created a variety of highly iterative and collaborative development methods collectively known as Agile (Cohn 2004) methods. These highly adaptive Agile software development methods replaced traditional predictive methods like Waterfall. Waterfall relied too heavily on upfront planning for use in situations with more unknown knowns than known knowns.

Traditional methods like Waterfall focused on detailed project planning, scheduling, and monitoring but focused very little on the process by which the value would be created. The high probability of encountering unknown unknowns throughout the software development process significantly reduces the benefit from investing in upfront planning.

Other key attributes of the Agile development model include the emphasis on face to face communication over written documentation, use of highly skilled cross functional teams, and delivering working prototypes with each iteration of work.

The earliest research I found that specifically focused on describing and recommending iterative software development methods was in a 1968 report from Brian Randell and F.W. Zurcher at the IBM T.J. Watson Research Center. The adaptive software development method was first introduced in a paper by Edmonds in 1974. Some early Agile methods include Scrum in 1995, Crystal Clear, Extreme Programming in 1996, Adaptive Software Development, Feature Driven Development, and Dynamic Systems Development Method in 1995.

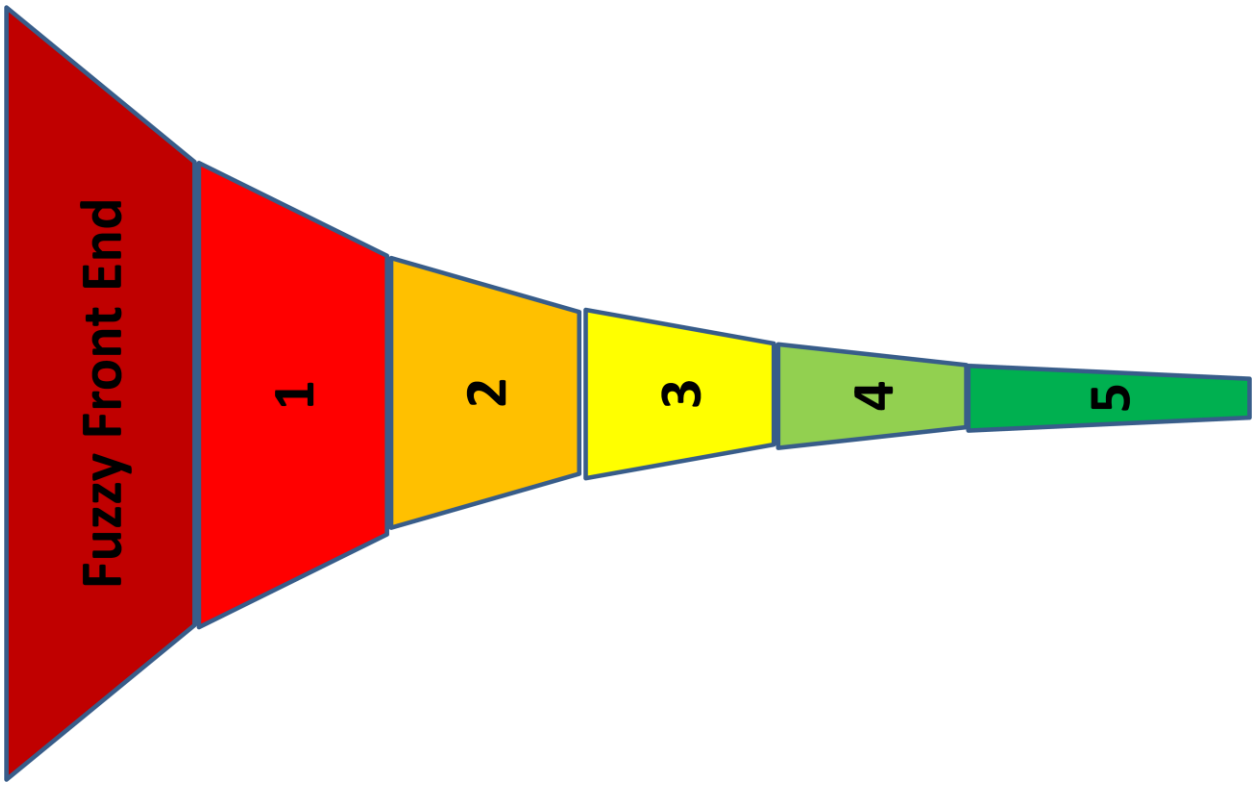
While these represent some of the first writings on applying iterative and adaptive project management methods to the software development industry, they all stem from general use of the principles of scientific management pioneered by the likes of Taylor, Shewhart, Deming, and others.

These popular project management methods are commonly referred to as Agile methods since the Agile Manifesto was published in 2001. The Agile Manifesto contributors have created the Agile Alliance, to promote the values and principles of the Manifesto for Agile software development methods.

By (Laplante) 2004, predictions that traditional predictive methods like Waterfall were losing ground to adaptive Agile methods had started to appear. Today there are more than 6 Agile project manager job postings for every one Waterfall project manager posting. I'm in no way suggesting Waterfall methods will not continue to be used with great success, just that there is growing interest in Agile software development methods.

From this research I developed a model for selecting a software development method based on matching the predictive and adaptive attributes of the new product development phases to the attributes of the methods. My model divides the development methods along their predictive versus adaptive attributes. On one end of the spectrum you have the Waterfall method, representing a traditional predictive development model. At the other end of the spectrum you have Agile methods, representing highly adaptive development models.

Managing the FFE and NPD Phase



Different phases of the new product development process may require a different project management method. The Agility scorecard can be used to select the most compatible management method for each phase by measuring the phase in terms of its predictive versus adaptive nature. My goal is to match the attributes common between the phase, and the project management method. It's from this commonality that my model bases the selection of the attributes to be included in the agility index scorecard.

At the fuzzy front end of new product development, the project pipeline looks like the opening of funnel filled with many chaotic, iterative, and collaborative undertakings and very little is predictable at this point in the new product development process. These attributes make the phase a good fit for an adaptive project management method such as Agile.

As you progress through the five phases of new product development, the funnel narrows as products become more defined and the outcome becomes more predictable, making it a good fit for a predicative project management method such as Waterfall.

There are also industry specific attributes to consider. The funnel shaped FFE diagram represents a typical new product development life cycle where by the fifth phase "launch" the process has become more predictive than adaptive. However in the software development industry the development phase is a very iterative, collaborative, and adaptive phase. These attributes make the fourth phase of software development a better fit for an adaptive project management method such as Agile, than a predictive method like Waterfall.

Agility Index Scorecard

Adaptive Project Traits	Value	Predictive Project Traits	Value	Weight	Adaptive Score	Predictive Score
Cheap to fail	1	Expensive to fail	0	10%	10%	0%
Senior developers	1	Junior developers	0	20%	20%	0%
Requirements change often	1	Requirements do not change	0	30%	30%	0%
Small number of developers	1	Large number of developers	0	10%	10%	0%
Culture thrives in chaos	0	Culture that demands order	1	10%	0%	10%
Phase Considerations	0	Phase considerations	1	10%	0%	10%
Industry considerations	1	Industry specific considerations	0	10%	10%	0%
	5		2	100%	80%	20%

Now that we have some common language to guide our project management discussion, let's review how I created my Agility scorecard for selecting the most compatible project management method for each phase of the new product development process.

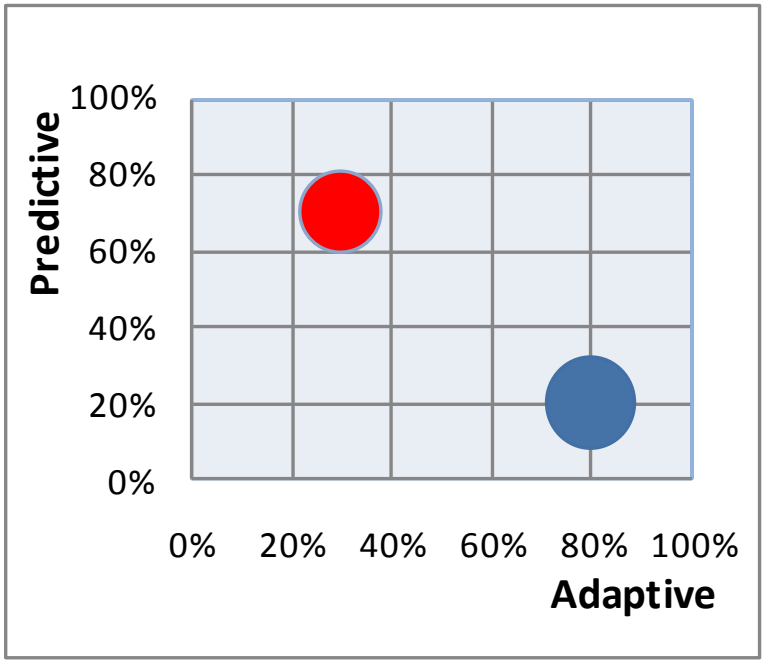
I started with Boehm and Turner's list of project traits, Next I added new product development phase and industry specific considerations, Finally I weighted each factor and scored them to get the Agility index of the new product development phase.

My Agility scorecard measures attributes directly related to new software development and will not be appropriate for other types of new services. As with any scorecard, the attributes measured and how heavily they are weighted should be tailored to meet the needs of your situation. This will require some subject matter expertise in the industry you are managing.

This sample Agility scorecard shows a phase of new product development with 80% adaptive traits and only 20% predictive traits. This suggests that we should choose a project management method with more adaptive than predictive attributes.

The key to understanding an agility index is that project management methods are not purely predictive or purely adaptive. Rather they fit somewhere along the agility spectrum with adaptive at one end and predictive at the other. By matching the common attributes between a phase to the attributes of a project management method, the most appropriate method can be selected.

Graphing the results of an Agility scorecard allows decision makers to visualize the method that best fits each phase of the new product development process.



In this graph a circle represents a phase in the product development process. The red circle is a phase with an adaptive score of 25% and a predictive score of 75%. Since the phase has strong predictive traits, it may benefit from Waterfall's predictive strengths more than Agile's adaptive strengths.

The blue circle is a phase that has an adaptive score of 80% and a predictive score of only 20%. Since this phase has strong adaptive traits, it may benefit from Agile's adaptive strengths more the than Waterfall's predictive strengths.

Now that we have matched the phases of new product development to project management methods using common attributes, we can discuss what led to the creation of my agility index scorecard.

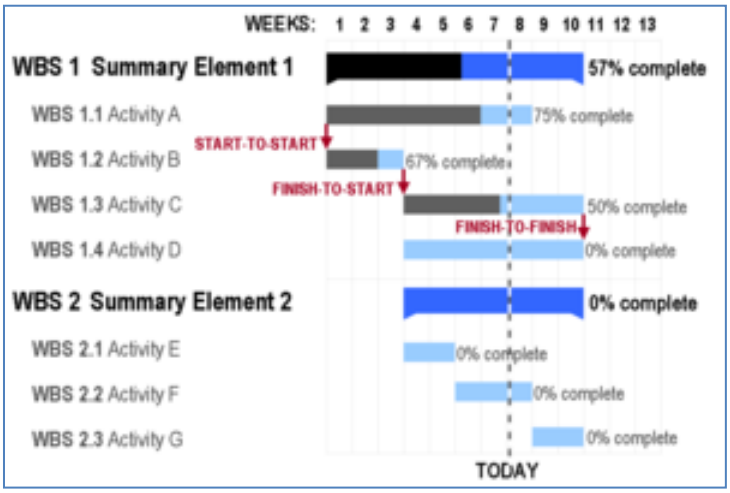
Project Management Taxonomy

To develop my scorecard I began by researching the history of project management methods. From this research, I noticed a pattern developing where project management methods were being divided along the spectrum of predictive versus adaptive attributes. More recent project management literature has focused on finding the right balance between adaptability and predictability.

Year	Author	Key Contribution
1911	Taylor	Scientific Management: The Principles of Scientific Management was first published in 1911.
1919	Gantt	Gantt Chart: Used for managing predictive projects.
1920	Shewhart	Shewhart Cycle: (SPC) Developed statistical process control and was a mentor of Deming that inspired the Deming cycle.
1950	Deming	Deming Cycle: (Plan, Do, Check, Act) This was an iterative PM process.
1969	PMI	PMI Certification: The Project Management Institute was formed in 1969.
1970	Royce	Waterfall: Although Royce did not use the term "waterfall" in this article, his model was used to coin the term.
2001	Martin	Agile Manifesto: Practitioners of Lean software development methods drafted the values and principles of Agile methods.
2002	Loch & DeMeyer	Managing the Unknown - A New Approach to Managing High Uncertainty and Risk in Projects: Choosing a PM method based on a least risk scenario.
2004	Boehm & Turner	Balancing Agility and Discipline: 5 key attributes that categorize a NPD phase as adaptive or predictive.

Modern day project management methods in the software industry stem from scientific management methods developed in the early 1900s. No discussion of scientific management would be complete without Fredrick Taylor (1856-1915), the father of scientific management.

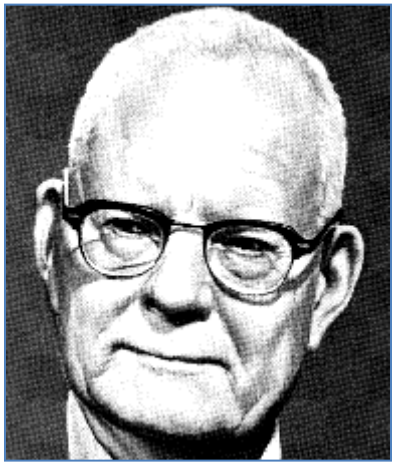
In his book (The Principles of Scientific Management 1911) Taylor started by quoting President Theodore Roosevelt "The conservation of our national resources is only preliminary to the larger question of national efficiency." Taylor wrote that the solution to inefficiency lays in systematic management, rather than in searching for extraordinary people.



Henry Gantt (1861-1919) is credited with the invention of the Gantt chart. Gantt charts are bar charts that visually communicate a project's predicted schedule. Some Gantt charts also show the dependencies between activities. Gantt charts can be used to show current schedule status using percent complete shadings and a vertical "TODAY" line as a point of reference. Although now regarded as a common charting technique, Gantt charts were considered revolutionary when they were introduced.

Walter Andrew Shewhart (1891-1967) was an American physicist, engineer and statistician, sometimes known as the father of statistical quality control. Shewhart's work pointed out the importance of reducing variation in a manufacturing process and the understanding that ad hoc continual process adjustment in reaction to non conformance actually increased variation and degraded quality. To put it another way, Shewhart understood that manufacturing is a science not an art. This is the foundation of continuous improvements methods widely adapted today.

Deming (1900-1993) is credited with creating an early iterative management method referred to as the Deming cycle or PDCA (plan-do-check-act).This is an iterative four step problem solving process typically used in business process improvement. It is also known as the Deming cycle, Shewhart cycle, Deming

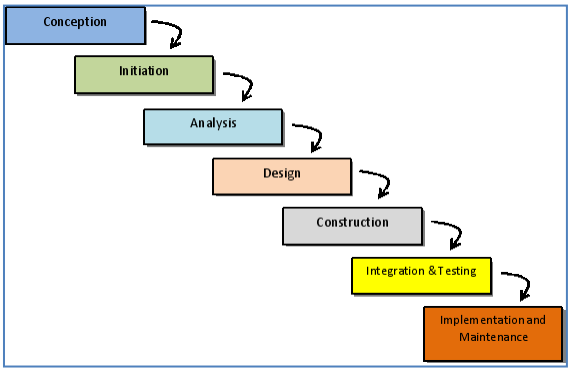


wheel, or plan-do-study-act. There is some debate over whether Deming or Shewhart should be credited with the iterative project management method described in the Deming Cycle.

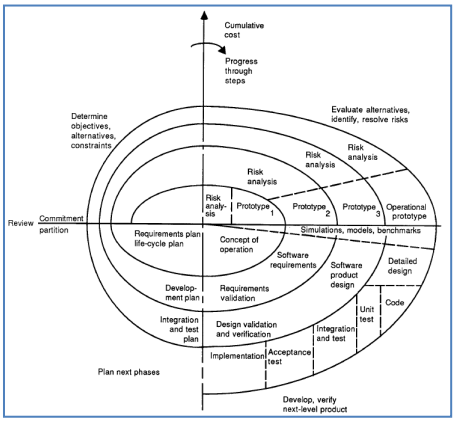
I think it's clear that both men contributed to our understanding of how iterative project management cycles reduce waste and improve quality. Deming was more specific is his assertion that with every iteration there is an opportunity to adapt to change in order to improve efficiency and quality. That's not to say that the Shewhart's cycle wasn't adaptive, just that Deming refined his mentor's ideas further.

PMI "Project Management International" was established in 1969 and today is the world's leading organization for the project management profession. They serve practitioners and organizations with standards that describe good practices, credentials that verify knowledge and experience, and resources for professional development, networking and community.

The first formal description of the waterfall model is often cited as an article published in 1970 by Winston W. Royce (1929–1995), Although Royce did not use the term "waterfall" in this article. Royce was presenting this model as an example of a flawed, non working model. This term has often been used in writing about software development as a way to criticize a commonly used software practice.

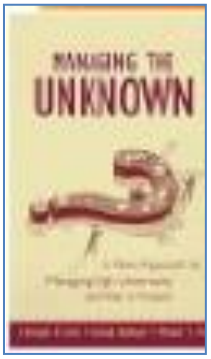


The Agile Manifesto (February 2001) is a statement of the principles that underpin agile software development. It was drafted at The Lodge at the Snowbird ski resort in the Wasatch Range of mountains in Utah. Representatives of various new methodologies (such as Extreme Programming, Scrum, DSDM, Adaptive Software Development, Crystal, Feature Driven Development, and Pragmatic programming) met to discuss the need for lighter alternatives to the traditional heavyweight methodologies.



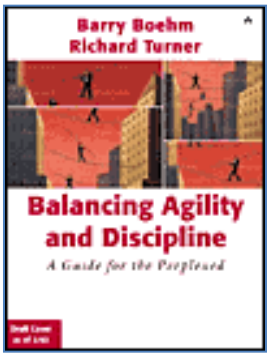
The 17 authors of the manifesto were: Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland and Dave

Thomas.



Loch and DeMeyer (2002) describe when to use an adaptive or predictive management method by focusing on project risk management. They describe project risk management as four conceptual steps: identify risks beforehand; classify and prioritize them according to their probability of impact, manage them with a collection of preventative, mitigating, and contingent actions that are triggered by risk occurrence, and embed these actions into a system of documentation and knowledge transfers into other projects.

My research focuses on using predictive and adaptive attributes to select the most appropriate management methods for each phase of new product development. Loch and DeMeyer's approach was similar in that they viewed selecting the most appropriate project management method in terms of managing the risk to project.



Boehm and Turner (2004) wrote "Balancing Agility and Discipline: A Guide for the Perplexed" in the hope of clarifying the perplexity about the roles of discipline, agility, and process in software development. They objectively compare and contrasted the traditional, plan-driven approaches to the newer, agile approaches and presented an overview of their home grounds, strengths, and weaknesses. They then described a risk-based approach to aid in balancing agility and discipline within a software development project.

Boehm and Turner's work provides the foundation from which I identified common attributes of project management methods and new product development phases. Their work on the spectrum of project management attributes the key to understanding how agility impacts project management method selection.

Spectrum of PM Attributes

The following six attributes of project management methods are commonly used to categorize methods by their predictive and adaptive attributes. By categorizing project management attributes we can create six unique dimensions to be analyzed when choosing the most appropriate project management method.

Predictive	Adaptive
Breaks requirements down into smaller steps	Phases not always predictable, distinct, onetime steps
Predicts how long each step will take to complete	Doesn't rely on trying to predict very far into the future
Structured for predictable units of work	Structured to plan for the unknown unknowns
Project timeline sets the teams priorities	Customer feedback from each iteration sets the team's priorities
Large teams follow detailed project plans	Small teams are in direct communication with the stakeholders
Focuses on progress reports and other non value adding artifacts	Focuses on building working prototypes in short time intervals

Predictive Process Methods

Traditional project management models such as Waterfall, attempt to break the customer's requirements down into smaller steps. Managers try to predict how long each step will take to complete and what resources will be required. The more predictable a phase is, the more it can benefit from the predictive attributes of traditional project management methods.

Adaptive Process Methods

The design and development phases of the software deployment life cycle are not always predictable, distinct, onetime phases. They are fraught with unknown unknowns and iterative in their very nature. To effectively manage them requires a software development method that thrives in these harsh conditions.

Agile methods don't rely on trying to predict very far into the future. Instead their strength comes from their ability to adapt to unpredictable situations. It's not that Agile models lack structure or planning; rather they are structured to plan for the unknown unknowns in the new product development process. It is from the tremendous agility this method provides that the name Agile was derived.

Agile methods use customer feedback from each iteration to set the team's priorities for the next iteration. Small teams are in direct communication with the stakeholders and build working prototypes

in short time intervals. This combination of direct communication and rapid prototyping make Agile methods very adaptive to changing customer requirements.

Strengths and Weaknesses

Method	Strengths	Weaknesses
Waterfall	Project planning and scheduling	Requires predictable projects
	Monitoring project progress	Difficultly managing changes in requirements
	Managing very large project teams	Works better with standardized work than custom solutions
		Skilled employees don't like being micromanaged
Agile	Adapts to changing requirements	Budgeting time and costs
	Built to handle unknown unknowns	Requires more of the customer's time and involvement
	Works well with small teams	Harder to manage larger teams
		Doesn't work well with junior programmers

Adaptive Agile development methods are not better than a predictive Waterfall development method. In many situations they would even be totally inappropriate. By understanding the strengths and weaknesses of both methods, new product development managers can create a model to help them to select the most appropriate method.

The selection of any method should begin with matching the method's strengths to the needs of each phase of the new product or service development process. The exact needs will be dictated by the phase of new product development and other unique situational factors.

Agile's Critics

Agile's supporters may dismiss their critics as simply not understanding Agile's methods and the benefits they bring to the phases of new product development. In Matt Stephens's and Doug Rosenberg's article (Extreme Programming Refactored, 2003) they critique the XP Agile method and came up with the following criticism.

- A methodology is only as effective as the people involved, Agile does not solve this
- Often used as a means to bleed money from customers through lack of defining a deliverable
- Lack of structure and necessary documentation
- Only works with senior-level developers
- Incorporates insufficient software design
- Requires meetings at frequent intervals at enormous expense to customers
- Requires too much cultural change to adopt
- Can lead to more difficult contractual negotiations
- Can be very inefficient if the requirements for one area of code change through various iterations, the same programming may need to be done several times over. Whereas if a plan were there to be followed, a single area of code is expected to be written once.
- Impossible to develop realistic estimates of work effort needed to provide a quote, because at the beginning of the project no one knows the entire scope/requirements
- Can increase the risk of scope creep due to the lack of detailed requirements documentation
- Agile is feature driven; non-functional quality attributes are hard to be placed as user stories

Matt Stephens and Doug Rosenberg were not trying to discredit Agile project management methods. Rather they understood that agile methods were not a panacea, and should only be used when they are appropriate for a new product development phase. By highlighting the faults of Agile project management methods, they revealed when it's not appropriate to select an Agile project management method.

Selecting the Appropriate Model

Some phases have characteristics that make them more suitable for an adaptive or predictive development method. In the 2004 Boehm and Turner article titled "Balancing Agility and Discipline", they listed five key traits that can be used to categorize a phase as adaptive or predictive. These traits form the spectrum of project management attributes that should be paired to the attributes of a new product development phase. It's from this commonality between the attributes along the agility spectrum that the most appropriate project management method can be identified.

Adaptive Phase Attribute	Predictive Phase Attributes
Cheap to fail	Expensive to fail
Senior developers	Junior developers
Requirements change often	Requirements do not change often
Small number of developers	Large number of developers
Large teams following detailed project plans	Small teams are in direct communication with the stakeholders
Culture thrives in chaos	Culture that demands order

It is important to understand that not every attribute will match between the phase and the method. Just because a phase would be best suited by a small team of senior developers, it doesn't mean you can't succeed with a method that relies on a larger team of junior developers working from a predictive project plan. As Donald Rumsfeld once said "you go to war with the army you have not the army you wish you had". If your agility index is 65/35 "65% adaptive and 35% predictive" you would likely benefit more from an Agile method's adaptive nature than from Waterfall's predictive nature, but that's not to say that you cannot be successful using a predictive method like Waterfall if senior developers are not available.

I extended Boehm and Turners model further by adding phase and industry specific considerations.

Phase Specific Traits

Crawford and Benedetto (2008) suggest that the new product development phases will become more linear as you progress through the five phases. In the beginning the new concept development phase involves fuzzy and iterative steps that are supported by Agile's strengths. By the fifth and final stage "launch" the project should become more predictable, suggesting a good fit for predictive methods.

Therefore a new product development phase will have attributes that make the phase more suitable for one method over another. Different methods may be used during different phases of the new product development process.

Industry Specific Traits

In the software industry, special attention should be given to the fourth phase "development" of the new product development process. It's in the fourth phase that software projects tend to be very iterative, collaborative, and requirements often change after the work has begun.

In the software industry, adaptive Agile methods are a good fit for the fourth phase of the new product development process. In other industries the fourth phase may be more predictive and therefore better suited for a more predictive Waterfall method.

Quantifying Agility Requirements

There have been many attempts to quantify the agility of projects by measuring different aspects of the development project. These include the Agility Index Measurements, similarly named Agile Measurement Index, and Agility Self Assessment tests. None of these have been widely accepted, but that is not to say I believe you can't quantify Agility. In the course of my research I have developed my own scorecard for quantifying the agility requirements of a project.

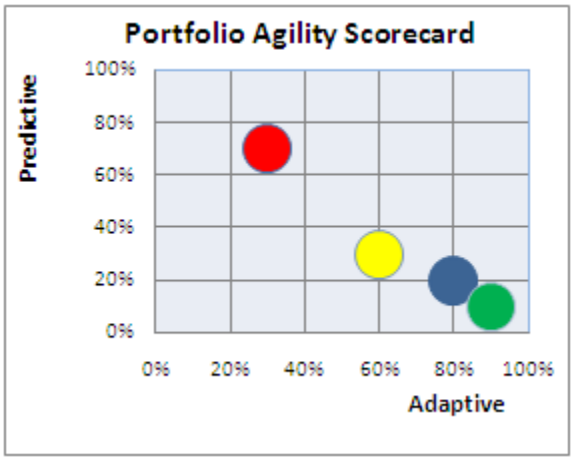
Agility Scorecard

My Agility scorecard is a useful tool for selecting the appropriate method. I started with Boehm and Turner's list of project traits, added phase and industry specific considerations, then scored and weighted each factor. As with any scorecard, the attributes measured can be tailored to meet the needs of your situation.

Adaptive Project Traits	Value	Predictive Project Traits	Value	Weight	Adaptive Score	Predictive Score
Cheap to fail	1	Expensive to fail	0	10%	10%	0%
Senior developers	1	Junior developers	0	20%	20%	0%
Requirements change often	1	Requirements do not change	0	30%	30%	0%
Small number of developers	1	Large number of developers	0	10%	10%	0%
Culture thrives in chaos	0	Culture that demands order	1	10%	0%	10%
Phase Considerations	0	Phase considerations	1	10%	0%	10%
Industry considerations	1	Industry specific considerations	0	10%	10%	0%
	5		2	100%	80%	20%

As with any scorecard you must include unique attributes specific to your situation. For example if you were working with a consultant you are taking a risk since you have no prier experience working with them. To reduce the risk you may want to insist on a predictive project management method. So in this case you may want to include time working with consultant in your scorecard.

Balancing Your Scorecard



Graphing also reveals how balanced the company's project portfolio is. A company whose Agility Scorecard chart shows more adaptive phases than predictive, is taking on more risky projects than a company with more predictive development phases. A company with more predictive phases than adaptive phases may be giving up too much reward by playing it too safe.

Companies should balance their new product portfolio by selecting new products for development that provide a balanced combination of predictive and adaptive phases. What constitutes a balanced portfolio involves phase, industry, and company specific factors.

For example in the software development industry a balanced scorecard typically leans towards the more adaptive side. In the bridge construction industry the phases on new product development tend to be very predictive in their nature. I'm proposing that there simply is no one size fits all, cookie cutter solution to selecting the most appropriate project management method. After all, common sense tells me that if I have no subject matter expertise in a given industry, then I'm not qualified to develop an agility index scorecard for that industry in the first place.

Infinity Scorecard Case Study

Now that we have covered how to use attributes common to the phase and method to develop an agility index scorecard, let's walk through a real world application of the agility index scorecard. For this exercise I have chosen a project from my work at Avamere Health Services. Avamere owns an ancillary company "Infinity Rehab" that provides physical therapy services in skilled nursing facilities.

Bob Thomas, the president of Infinity asked me to develop a data warehouse that would support the automation of his company's dashboard reports. The goals were to reduce costs, increase accuracy of information, and increase reporting cycles from monthly to daily.

Fuzzy Front End

The fuzzy front end activities included collaborating with our software vendors to automate the nightly ETL "Extract, transform, and loading" of data from our therapy and accounting applications. This data would be loaded into one analytical database where reports are be generated, without slowing down the transaction processing database. We had had mock ups of reports but weren't sure what data we get from our vendors.

During this phase it was cheap to fail so we developed several options to address both the data we would extract and the timing of the nightly SQL jobs. As we learned more about what data was available for reporting we iteratively developed mock ups of the dashboard reports that could be developed. A small team of executives, senior developers, and vender representatives were used to layout the architecture of the application. This was required because the impact of moving forward with a bad plan would be significant in later phases of new product development. Phase specific considerations matched a typical fuzzy front end found in other industries.

So far, the attributes of the phase have suggested an adaptive agility index. One exception was that the company's culture did not thrive in chaos. This represents an industry specific attribute of long term health care. Chaos in health care leads to mistakes that could be fatal and generate massive financial liabilities.

Overall this phase was more adaptive than predictive, suggesting a good fit for an Agile project management method.

Phase 1 – Opportunity Identification and Selection

Early in the project opportunities to reduce labor costs through automation were identified. Automation led to identifying an opportunity to increase our report cycles from monthly reporting to daily reporting. This increased manager's ability to respond to performance problems quickly, thereby reducing the negative impact of poor performance. Decision makers relying on monthly reporting to drive performance is like driving a car by looking through the rear view mirror. By the time you hit a bump in the road it is too late to avoid it.

Adaptive Project Traits	Value	Predictive Project Traits	Value	Weight	Adaptive Score	Predictive Score
Cheap to fail	1	Expensive to fail	0	10%	10%	0%
Senior developers	1	Junior developers	0	20%	20%	0%
Requirements change often	1	Requirements do not change	0	30%	30%	0%
Small number of developers	1	Large number of developers	0	10%	10%	0%
Culture thrives in chaos	0	Culture that demands order	1	10%	0%	10%
Phase Considerations	1	Phase considerations	0	10%	10%	0%
Industry considerations	1	Industry specific considerations	0	10%	10%	0%
	6		1	100%	90%	10%

During phase one, it was still cheap to fail since no substantial code had been developed. Senior executives and developers were required to steer the project in the right direction. Requirements changed daily as new possibilities and obstacles were discovered. The company culture in this phase and all others was understandably not one that would thrive in chaos. The phase was still heavily adaptive as concepts were being developed and selected for further development. Industry considerations were typical of new product development in most industries.

Overall, this phase was more adaptive than predictive suggesting a good fit for an Agile project management method.

Phase 2 – Concept Generation

By phase two, several concepts were generated that fit the need of the management team. The pros and cons of each were evaluated and concepts were ranked by the benefit and costs to implement. Reporting tools including; IMB's Cognos, Microsoft's SQL reporting services, and Crystal Report were evaluated. Whether to develop the application in-house or hire a consulting company was also considered.

Adaptive Project Traits	Value	Predictive Project Traits	Value	Weight	Adaptive Score	Predictive Score
Cheap to fail	0.9	Expensive to fail	0.1	10%	9%	1%
Senior developers	1	Junior developers	0	20%	20%	0%
Requirements change often	0.75	Requirements do not change	0.25	30%	23%	8%
Small number of developers	1	Large number of developers	0	10%	10%	0%
Culture thrives in chaos	0	Culture that demands order	1	10%	0%	10%
Phase Considerations	1	Phase considerations	0	10%	10%	0%
Industry considerations	1	Industry specific considerations	0	10%	10%	0%
	5.65		1.35	100%	82%	19%

During phase two it was still cheap to fail since no substantial code had been developed. Senior executives and developers were required to steer the project in the right direction. Requirements were

still changing, but less often, as the possibilities and obstacles became better understood. The company culture in this phase and all others was understandably not one that would thrive in chaos. The phase was still relatively adaptive as concepts were still being developed and selected for further development. Industry considerations were typical of new product development in most industries.

Overall this phase was more adaptive than predictive but we were starting to develop a clear picture of the scope of work and related costs.

Phase 3 – Concept/Project Evaluation

During phase three, it was clear that IBM wanted way more for Cognos licensing than it was worth. Microsoft's SQL Server 2008 included great analytical and reporting services in licenses that we already owned. Since Avamere was already a Microsoft shop it didn't make technical sense to add Linux or AIX "IMB Unix" servers to our platform. By developing the system in house we would save tens of thousands of dollars and have more control over our future.

Adaptive Project Traits	Value	Predictive Project Traits	Value	Weight	Adaptive Score	Predictive Score
Cheap to fail	0.1	Expensive to fail	0.9	10%	1%	9%
Senior developers	0.2	Junior developers	0.8	20%	4%	16%
Requirements change often	0.3	Requirements do not change	0.7	30%	9%	21%
Small number of developers	1	Large number of developers	0	10%	10%	0%
Culture thrives in chaos	0	Culture that demands order	1	10%	0%	10%
Phase Considerations	0	Phase considerations	1	10%	0%	10%
Industry considerations	1	Industry specific considerations	0	10%	10%	0%
	2.6		4.4	100%	34%	66%

During phase three, it was no longer cheap to fail since substantial code had been developed. Senior executives and developers were no longer required since the project plan was becoming more predictable. As with any software development project the requirements were still changing but overall it was becoming clear what the major requirements were. The phase was still predictable as concepts were selected for further development. Industry considerations were typical of new product development in most industries.

Overall this phase was more predictive than adaptive suggesting a good fit for a more plan driven project management method like Waterfall.

Phase 4 – Development

During phase four, the software development activities were in front and center. Prototypes generated in earlier phases were fleshed out, becoming the first generation of our new enterprise report platform.

In a typical new product development life cycle this phase would have been more predictive than adaptive. This is where considerations specific to the software development industry came into play.

Adaptive Project Traits	Value	Predictive Project Traits	Value	Weight	Adaptive Score	Predictive Score
Cheap to fail	0.1	Expensive to fail	0.9	10%	1%	9%
Senior developers	1	Junior developers	0	20%	20%	0%
Requirements change often	1	Requirements do not change	0	30%	30%	0%
Small number of developers	1	Large number of developers	0	10%	10%	0%
Culture thrives in chaos	0	Culture that demands order	1	10%	0%	10%
Phase Considerations	1	Phase considerations	0	10%	10%	0%
Industry considerations	1	Industry specific considerations	0	10%	10%	0%
	5.1		1.9	100%	81%	19%

In the software development industry, it's typical for the fourth phase to be very iterative, collaborative, and for the requirements to change after the programming has begun. Switching project management methods for just this one phase is justifiable for three reasons. First, the phase's attributes have changed to suggest a more adaptive methodology. Second, this is by far the most expensive phase so it has the most to lose or gain by how it is managed. Third, this phase of new product development will be by far the longest phase in the development lifecycle; therefore we will not be flip flopping back and forth anytime soon.

In the software industry the fourth phase of new product development typically benefits the most from an adaptive Agile style project management method. In other industries the fourth phase may be more predictive and better suited for predictive Waterfall methods.

In this particular case, the fourth phase clearly called for a more adaptive Agile project management method. That is not say the fourth phases of every software development project can benefit more from an adaptive method than a predictive method. I'm working on a web based compliance reporting application right now that is so predictive, due to the project sponsor's detailed project plan, that an iterative process would be totally unnecessary and inefficient.

Phase 5 – Launch

During phase five managers had to be trained on how to use the dashboard reports to better manage the facilities in their regions. We also worked with the company who hosts our server farm to deploy the new servers and set up the required security between the database, web, and email servers.

During phase five it was definitely not cheap to fail since the applications had been fully developed. Senior executives and developers were no longer required because most the work was already completed and the remaining tasks could be completed by system administrators. The number of

changes was now limited to only minor improvements to existing applications. This phase was very predictable and the remaining tasks consisted of clearly defined work that could be predictably scheduled.

Adaptive Project Traits	Value	Predictive Project Traits	Value	Weight	Adaptive Score	Predictive Score
Cheap to fail	0	Expensive to fail	1	10%	0%	10%
Senior developers	0	Junior developers	1	20%	0%	20%
Requirements change often	0	Requirements do not change	1	30%	0%	30%
Small number of developers	0	Large number of developers	1	10%	0%	10%
Culture thrives in chaos	0	Culture that demands order	1	10%	0%	10%
Phase Considerations	0	Phase considerations	1	10%	0%	10%
Industry considerations	0	Industry specific considerations	1	10%	0%	10%
	0		7	100%	0%	100%

Overall this phase was clearly more predictive than adaptive, suggesting a good fit for a more plan driven project management method like Waterfall.

Conclusion

At each phase in new service development, a project management method must be chosen that best matches the attributes of the phase. An Agility index scorecard can quantify a project management method's compatibility with the new product development phase. The attributes and weightings in your scorecard should be tailored to meet the needs of your situation. There is no one size fits all approach to selecting the most appropriate project management method. Therefore significant subject matter expertise is required to develop an agility index scorecard for any industry.

Future Research Questions

Future research needs to be done to quantify the correlation between attributes common to new product development phases and project management methods. Knowing whether common attributes are positively or negatively correlated is fairly easy, but it would strengthen the accuracy of the scorecard if we could quantify their correlation. A model also needs to be developed to guide in assigning weighting the attributes.

References

Agile Alliance, Values Statement, http://www.agilealliance.com, 2009

Ambler, Scott W., The Threat of the New, http://www.ddj.com/architect/184414798, 2001

Basili, V. R., and A. J. Turner, Iterative Enhancement: A Practical Technique for Software Development, Software Engineering, 1975.

Beck, K. Extreme Programming Explained, Addison-Wesley, Palo Alto, CA, 1999.

Boehm, B., Software Engineering, IEEE Trans. Computer, 1976.

Boehm, B. W., Software Engineering Economics, Prentice-Hall, Englewood Cliffs, N. J., 1981

Boehm, B., A Spiral Model of Software Development and Enhancement, Computer, 1987.

Boehm, B., A. Egyed, J. Kwan, D. Port, A. Shah, and R. Madachy, Using the WinWin Spiral Model: A Case Study, Computer, 1998.

Bolcer, G.A., R.N. Taylor, Advanced workflow management technologies, Software Process Improvement and Practice, 1998.

Budde, R., K. Kuhlenkamp, L. Mathiassen, and H. Zullighoven, Approaches to Prototyping, Springer-Verlag, New York, 1984.

Chatters, B.W., M.M. Lehman, J.F. Ramil, and P. Werwick, Modeling a Software Evolution Process: A Long-Term Case Study, Software Process-Improvement and Practice, 2000.

Cohn, Mike, User Stories Applied: For Agile Software Development, Addison-Wesley, 2004

Crawford, Merle & Benedetto, Athony D., New Product Management, 9th Edition, McGraw-Hill Irin, 2008

Distaso, J., Software Management--A Survey of Practice in 1980, 1980.

Edmonds, E. A., A process for the development of software for non-technical users as an adaptive system, General Systems, 1974

Graham, D.R., Incremental Development: Review of Non-monolithic Life-Cycle Development Models, Information and Software Technology, January,1989.

Heineman, G., J.E. Botsford, G. Caldiera, G.E. Kaiser, M.I. Kellner, and N.H. Madhavji., Emerging Technologies that Support a Software Process Life Cycle. IBM Systems J., 1994.

Hekmatpour, S., Experience with Evolutionary Prototyping in a Large Software Project, ACM Software Engineering Notes, 1987

Hoffnagel, G. F., and W. Beregi, Automating the Software Development Process, IBM Systems, 1985

Humphrey, W. S., The IBM Large-Systems Software Development Process: Objectives and Direction, IBM Systems J., 1985.

Koen , Peter A., Et. Al, Fuzzy Front End: Effective Methods, Tools, and Techniques, 2005.

Laplante, P.A. and Neill C.J., The Demise of the Waterfall Model Is Imminent and Other Urban Myths, 2004

Lehman, M. M., Process Models, Process Programming, Programming Support, Proc. 9th. International Conference on Software Engineering, IEEE Computer Society, 1987.

Larman, C.; Basili, V.R.; Iterative and incremental developments. a brief history, Computer, Volume 36, Issue 6, June 2003

MacCormack, A., Product-Development Practices that Work: How Internet Companies Build Software, Sloan Management Review, Winter 2001.

Mili, A., J. Desharnais, and J.R. Gagne, Formal Models of Stepwise Refinement of Programs, ACM Computing Surveys, 1986.

Moore, J.W., P.R. DeWeese, and D. Rilling, "U. S. Software Life Cycle Process Standards," Crosstalk: The DoD Journal of Software Engineering, July 1997

Noll, J. and W. Scacchi, Supporting Software Development in Virtual Enterprises, Journal of Digital Information, February 1999.

Ould, M.A., and C. Roberts, Defining Formal Models of the Software Development Process, Software Engineering Environments, 1988.

Paulk, M.C., C.V. Weber, B. Curtis, The Capability Maturity Model: Guidelines for Improving the Software Process, Addison-Wesley, New York, 1995.

Penedo, M.H., An Active Web-based Virtual Room for Small Team Collaboration, Software Process --Improvement and Practice, 2000.

B. Randell and F.W. Zurcher, Iterative Multi-Level Modeling: A Methodology for Computer System Design, Proc. IFIP, IEEE CS Press, 1968

Reid, Susan E. and Brentani Ulrike de, The Fuzzy Front End of New Product Development for Discontinuous Innovations: A Theoretical Model, J PROD INNOV MANAG 2004.

Royce, W. W., Managing the Development of Large Software Systems, Proc. 9th. International Conference Software Engineering, ,IEEE Computer Society, 1987, Originally published in 1970.

Royce, W., TRW's Ada Process Model for Incremental Development of Large Software Systems, 12th Interational. Conference on Software Engineering, Nice, France, IEEE Computer Society, 1990.

Sathi, A., M. S. Fox, and M. Greenberg, Representation of Activity Knowledge for Project Management, IEEE Trans. Patt. Anal. and Mach. Intell.,, 1985.

Scacchi, W. and P. Mi., Process Life Cycle Engineering: A Knowledge-Based Approach and Environment, Intelligent Systems in Accounting, Finance, and Management, 1997.

Stephens, Matt and Rosenberg, Doug, Extreme Programming Refactored, Apress L.P, 2003

Wirth, N., Program Development by Stepwise Refinement, Communications of the ACM, 1971.

Yu, E.S.K. and J. Mylopoulos, Understanding "why" in software process modeling, analysis, and design, 16th International Conference on Software Engineering, 1994.

Zave, P., The Operational Versus the Conventional Approach to Software Development, Communications of the ACM, 1984.