J. Favaro, "Efficient Markets, Efficient Projects, and Predicting the Future," *Proc. 5th International Conference on eXtreme Programming and Agile Processes in Software Engineering XP2004*, J. Eckstein and H. Baumeister (Eds.), LNCS 3092, Springer Verlag, June 2004, pp. 77-84

# Efficient Markets, Efficient Projects, and Predicting the Future

John Favaro

Consulenza Informatica, Via Gamerra 21,
56123 Pisa, Italy
jfavaro@tin.it

**Abstract.** Economic concepts have provided valuable sources of insight into important concepts underlying agile methodologies. The dynamics of capital markets are understood through the concept of *market efficiency*; an analogy is developed to *project efficiency* for understanding the dynamics of agile projects. The efficient project concept is then used to motivate the preoccupation of agile developers with dealing only with available information at any time and not trying to predict the future. Finally, six lessons of project efficiency are presented. **Keywords:** economics, efficiency, value, market, project.

## 1 Introduction

> Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
> - the *Agile Manifesto*

One of the ways in which agile methods such as Extreme Programming (XP) have differentiated themselves from other software development methodologies has been their explicit elevation of economic arguments onto an equal (or greater) footing with the more familiar technical arguments. Therefore it is perhaps not surprising that economics and finance have also proven to be a rich source of analogies for explaining the values, principles, and practices of agile methods.

Certainly the most widely disseminated of these has been the analogy of "business options," introduced both in the White Book [1] and in other publications [2], where concepts from option pricing theory are used to support the discussion of flexibility in agile methods. In another recent example [3], the concept of residual income (or Economic Profit), commonly used in financial management to monitor usage of capital resources such as inventory, helped illustrate the notions of "software inventory" and "software in process" currently being promoted in the agile community.

In this paper an analogy is developed between agile project dynamics and the concept of efficient markets from corporate finance.

## 2 Efficient Markets

The ideas leading up to the theory of efficient markets are over a hundred years old. Louis Bachelier, in his doctoral thesis [4] in 1900, put forward the proposition that stock price movements are completely random – an idea considered so preposterous at the time that it was quickly forgotten. (This was unfortunate, because Bachelier not only anticipated the next formulation of this proposition by 53 years, he also anticipated Albert Einstein's work by five years in postulating that stock prices follow Brownian motion. As if that weren't enough, he also managed to contribute several key ideas in the field of option pricing.)

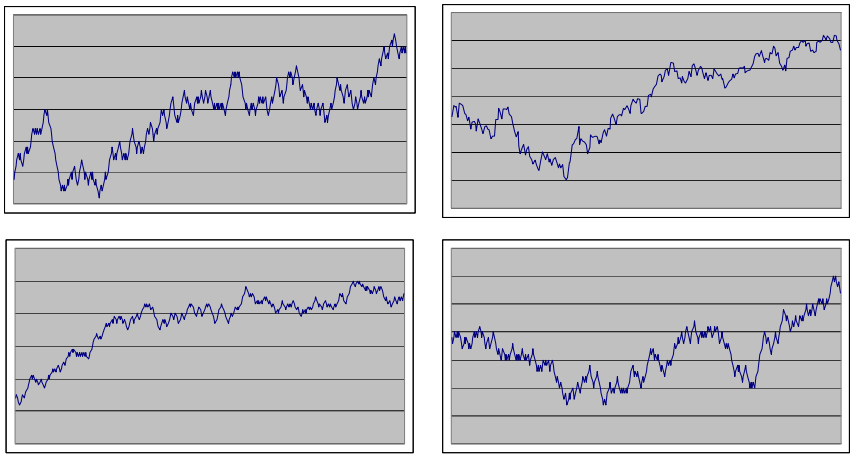

**Figure 1: Which is the real chart of stock prices?**

In 1953, Kendall performed an extensive study of patterns in stock market prices, and subsequently reported [5] on the number of patterns he had found: none. To his own amazement, he had been unable to find evidence of *any* regularity or cycles. On the contrary, prices seemed to walk around randomly, as though their direction were being determined by a simple toss of a coin. This report was also greeted by skepticism, even hostility (especially from those making a living from finding patterns in stock prices). But this time the conclusions were harder to ignore: time and again it has proven to be impossible to distinguish between a chart generated entirely by random coin tosses and a chart of real stock prices. Three of the four stock price charts in Figure 1 were generated for this article from a spreadsheet in which successive up and down movements were determined randomly. Which is the real chart? (The answer is given at the end of this article.)

The reason that the randomness of stock price movements was so hard to accept, of course, is that it seemed to make no sense at all: how can stock price movements be random when stocks are obviously affected in a very concrete way by profits, losses, acquisitions, mergers – in short, by significant events of all kinds? The explanation arrived in the form of the Efficient Market Hypothesis, first proposed formally by Fama in the mid-1960s [6].

In an efficient market, information travels freely among the large number of intelligent, motivated participants (as Malkiel [7] says, "money attracts brains"). As soon as any bit of information becomes available, investors pounce upon it, and its implica-

tions are quickly incorporated into the prices of stocks. As a result, the market is always completely "up-to-date" – it reflects all information that is currently available to investors.

Most importantly: not only does an efficient market reflect everything that has happened in the *past*, it also reflects anything that can currently be said about what might happen in the *future*. Legions of investors scrutinize, discuss, and analyze any new information over and over until every useful conclusion from that information has been drawn – *and also acted upon*:

> If today's direction … does indeed predict tomorrow's step, then you will act on it today rather than tomorrow. Thus, if market participants were confident that the price of any security would double next week … Why wait? [7]

This is the key insight in understanding the puzzle of random stock price movements. In an efficient market, the incessant activity (motivated by greed and fear) of investors assures that any information that clearly points the way to the future is acted upon without delay. Afterwards, only one type of information remains: *that which does not yet exist*. Its arrival must come as a genuine surprise. But the timing of unexpected new information is *by definition* unpredictable (as Paulos notes [8], it would have been extremely strange to have seen a newspaper headline in 1890 exclaiming "Only 15 years to relativity!"). And therefore, each *new* step by the stock market is taken in response to new information whose timing and impact are necessarily unknown beforehand – a random walk (also called a "drunkard's walk").

Today, more than 50 years after Kendall's report, the degree to which the market is efficient is a matter of lively ongoing debate, but the essential validity of the Efficient Market Hypothesis is widely accepted.

## 4 Efficient Projects

The Efficient Market Hypothesis turns out to provide a very good conceptual framework for gaining insight into the dynamics of agile projects. Working within this conceptual framework, we now introduce the notion of *efficient projects*.

Before proceeding, it is worth noting that the word "efficient" is used here more in the engineering sense of "completely consuming all input" than in the bureaucratic sense of "well-organized and disciplined" more commonly seen in the software engineering literature today. Another engineering definition of efficiency is "high ratio of output to input": A perfectly efficient market quickly and completely consumes information as it becomes available, converting every bit into investor action. It is this same goal of perfect efficiency that agile projects strive to attain, the rapid and complete absorption of new information and its immediate conversion into implementation.

The concept of *common knowledge* [8] is central to the functioning of efficient markets. Information is disseminated in such a way that all participants are aware of it, and moreover, are aware that others are aware of it. As a result, information is not compartmentalized. Agile projects strive to achieve rapid information dissemination and a state of common knowledge through a variety of techniques including stand-up meetings, pair programming, ruthless refactoring, collective ownership, continual re-estimation of effort and velocity, and the absence of fixed roles that tend to compart-

 

mentalize information. Rapid and complete information dissemination is coupled with techniques for rapid conversion into implementation, such as the principle of the Simplest Design That Could Possibly Work.

We can contrast this with traditional projects that might be called *inefficient projects* (in the engineering sense we are using it). At any one time, there is information that is not common knowledge in the project. On the contrary, information is segmented and compartmentalized. This prevalence of "insider information" is partially a result of roles such as, for example, a "Chief Architect," who may act as the sole custodian of much important information. Moreover, information is generally not quickly converted into system functionality. At any one time, there is much information that is not reflected in the current state of the system – much design, much implementation, much testing is still in the future.

## 4 Predicting the Future

The notion of efficient projects yields insight into the preoccupation of agile practitioners, so puzzling to many, with not trying to predict the future – expressed, for example, in the familiar YAGNI (You Aren't Going to Need It) principle. Many cannot understand why agile projects do not try to deal with the future; paradoxically, this arises from the fact that they deal so completely with the past.

In an efficient project, everything to date – requirements, analysis, design, test, everything implied by the information available – is completely reflected in the *implemented* system. (This objective is also reflected in the so-called Customer Bill of Rights [9], where the customer " … can cancel at any time and be left with a useful working system reflecting investment to date.").

The less efficient a project, the more the "future is built-in." It is built in by the design that is not yet coded, by tests not yet run, by assumptions and claims made for the future. It is more difficult to change direction because it is predetermined by the very state of the implemented system that does not reflect all currently available information.

In contrast, in an efficient project, as in an efficient market, the future literally *is* unpredictable – because the past and present have been so completely digested. It is ready to react to this unpredictable future (for example, a user decision to introduce a new story). Each new step in an agile project (e.g. an iteration) leads to the rapid absorption and implementation of new information, leaving behind no assumptions about the future, in its own form of a random walk.

## 5 The Six Lessons of Market and Project Efficiency

In their classic text on corporate finance [10], Brealey and Myers presented six "lessons" that conveyed succinctly the most important implications of market efficiency. It is instructive to revisit these lessons now from the perspective of this discussion. Each lesson is presented and summarized first in its original form for efficient markets, then in an adapted form for efficient projects. Where appropriate, quotes from the original presentation in [10] will be utilized.

**Lesson 1: No memory**

*Efficient markets have no memory*. "[In an efficient market] … the sequence of past price changes contains no information about future changes." This is the most fundamental message of the Efficient Market Hypothesis: the past does not condition the future – there are no patterns or cycles implied by past movements.

*Efficient projects have no memory*. An efficient project likewise strives to not build the future into the system. By working only for the present, the project builds in only what is necessary to handle what has happened up to now, so that there are no mechanisms that condition how new information will be handled. For example, in a web project, if the system is built to handle, say, 200 users now, there is nothing in the current implementation (e.g. "hooks") from which to infer that the system might be asked to handle 1000 users in the future. The more efficient the project, the more it will decouple its past from its future, leaving it optimally ready to react to new information.

This is much different from an inefficient project, where the past strongly conditions the future, making it difficult to change course – because so much remains to be done, based upon so many suppositions.

**Lesson 2: Trust**

*Trust market prices*. "In an efficient market you can trust prices, for they impound all available information about the value of each security. To [improve on this], you not only need to know more than *anyone* else, but you also need to know more than *everyone* else." Often managers, confident of their superior investing ability, acquire other companies simply because they think those companies are undervalued. But in an efficient market, the phenomenon of *arbitrage* ensures that the values placed on securities (and therefore companies) by investors quickly converge to the correct ones: if the available information indicates that a price is too low, investors quickly take advantage of this and drive the price up; the converse happens when the price is too high. Even when the price is not correct, it is *unbiased*: any error is just as likely to be in one direction as another.

*Trust the implemented system*. In an efficient project, you can trust the implemented system, because it impounds all available information about what the system should do. In a phenomenon similar to arbitrage, the principles of the simplest possible implementation and refactoring place downward pressure on complexity, while information such as failing tests act to produce upward pressure, resulting in an implementation whose complexity is generally appropriate for the information available. When you try to second-guess the implementation, you are not only saying that you can improve on the consolidated wisdom of the project, but that you have a better idea of where the project is headed next. But since the system impounds all available information, then even if it is not yet completely right, it is still unbiased: there is no reason to think that you have a better idea about what the future holds.

**Lesson 3: Read the entrails**

*Read the market entrails*. Since the prices in an efficient market reflect all available information, it is there that we must go for answers. For example, if the stock price of a company (e.g. Oracle) is sinking in response to the bidding war it is waging to acquire another company (e.g. Peoplesoft), it is the clearest signal available that investors are displeased with this initiative. As another example, reading the entrails of long-term versus short-term interest rates will tell us whether the market thinks that interest rates are set to rise in the future.

*Read the system entrails*. "Ask the code," as the common saying goes. Since the implementation impounds all available information, it is there that we must go for answers. When there is a question, then look to how the system is actually implemented and performing – an addition to the suite of tests is invariably the best route. If the code smells or, for example, if the system seems to be able to get to 90% of tests passing and can't move beyond, then the system is sending a strong signal that something could be fundamentally wrong with the implementation. In an inefficient project, in contrast, where much important information remains outside the implementation, the system cannot be reliably interrogated for answers.

**Lesson 4: There are no illusions**

*There are no financial illusions*. "In an efficient market … investors are unromantically concerned with the firm's cash flows …" In recent years there have been a number of cases of "creative accounting," where reported earnings were manipulated in order to appear to make them appear higher (think of so-called *pro forma* earnings reported by many tech firms). But the incessant scrutiny of investors has invariably exposed the financial window-dressing and kept the focus on the true cash flows of the firm (with some infamous exceptions during the years of the dotcom mania – and even those were eventually exposed).

*There are no functional illusions*. In an efficient project, customers are unromantically concerned with the functionality of the system. The unrelenting rhythm of implementation and testing in an efficient project quickly peels off any functional "window dressing" (perhaps in the form of a colorful and flashy GUI) and keeps the focus on whether the customer functionality (e.g. handling a particular set of file formats) is really implemented by the system or not.

**Lesson 5: The do-it yourself alternative**

"In an efficient market, investors will not pay others for what they can do equally well themselves." The transparency of efficient markets reveals the costs and value of operations undertaken by firms – and consequently a firm must demonstrate to the investor that it can offer something at a cheaper price than he could have done himself. For example, companies that merge or acquire others often try to convince investors that they have added value by "diversifying." But the investor can easily and more cheaply diversify on his own, simply by buying shares in several different companies. There is no reason for him to prefer the generally more costly route offered by a merger.

In an efficient project, customers will not pay others for what they can do equally well themselves. Efficient projects are very transparent: the relentless cycle of estimating and re-estimating stories leads to the customer always knowing the cost and value to him of paying to have a feature implemented within the context of the project, down to a relatively fine grain. He will therefore always have the opportunity of being aware of possibilities to acquire the feature at a cheaper price outside of the project (say, a COTS or open source tool or component that implements that feature perfectly) – or to renounce altogether, when the efficient processes in the project reveal that the cost/value relationship of the feature is not advantageous.

In contrast, in an inefficient project, the implementation is generally not feature-aligned; as a consequence there is generally little or no opportunity to separate out and evaluate features that could be provided in a more cost-effective way outside the project. The customer must simply trust the implementers and hope for the best.

**Lesson 6: Seen one, seen them all**

*Seen one stock, seen them all*. "Investors don't buy a stock for its unique qualities; they buy it because it offers the prospect of a fair return for its risk." In an efficient market, stocks are perfectly substitutable for each other: investors don't care whether their cash flows are generated by selling cars, computers, or candy.

*Seen one implementation, seen them all*. In efficient projects, customers don't buy features for the unique characteristics of their implementation; they buy them because they deliver the functionality requested at a fair price, whether it is implemented with objects or with acorns. Agile methods support this view by being relatively technology-neutral: although certain technological categories are recognized to be generally effective (just as certain market sectors are recognized to be generally profitable), agile methods focus on delivery of features at the promised cost and consider the supporting technologies to be essentially substitutable for each other.

## Conclusions

The notion of efficient markets is central to modern corporate finance: it is the primary mechanism through which the value of capital assets is determined. The notion of efficient projects can help agile project managers understand the mechanisms that lead to the production of software with measurable value. Agile project managers don't try to predict the future, because they strive to have projects that completely impound the past and present. This leaves them free of the baggage of the past and present, and ready to confront an unpredictable future. Of course, neither markets nor projects are ever perfectly efficient all of the time – but the concept provides agile developers a way of understanding what they are trying to achieve.

The upper right-hand chart in Figure 1 tracks the S&P500 index from 20 November 2002 to 19 November 2003.

## References


1. Beck. K., Extreme Programming Explained: Embrace Change, Addison-Wesley, 1999.
2. Erdogmus, H. and J.M. Favaro, "Keep your options open: Extreme Programming and the economics of flexibility," in Extreme Programming Perspectives, M. Marchesi, G. Succi, D. Wells and L. Williams, Editors: Addison-Wesley, 2003.
3. Favaro, J.M., "Value-Based Management and Agile Methods," Proc. Fourth International Conference on Extreme Programming and Agile Processes, Genoa, May 2003.
4. Bachelier, Louis, Théorie de la spéculation, Annales scientifiques de l'Ecole Normale Supérieure, 3° série, 17 :21-86, 1900.
5. Kendall, M.G., "The Analysis of Economic Time Series," Part I. Prices, Journal of the Royal Statistical Society 96 (1953), pp. 11-25.
6. Fama, E. F., "Random Walks in Stock Market Prices," Financial Analysts Journal, September/October 1965.
7. Malkiel, B.G., A Random Walk Down Wall Street, W.W. Norton, 1996.
8. Paulos, J. A., A Mathematician Plays the Stock Market, Basic Books, 2003.
9. Jeffries, R., et al., Extreme Programming Installed, Addison-Wesley, 2001.
10. Brealey, R. and S. Myers, Principles of Corporate Finance, McGraw-Hill, 2000.


 