Activist Investing in Strategic Software

Ross J. Pettit

Activist Investing in Strategic Software

Ross J. Pettit

This book is for sale at http://leanpub.com/activistinvesting

This version was published on 2019-05-21

Contents

Introduction

> The most interesting opportunity for maximising IT value lay in the discretionary IT investment.
>
> To put things in perspective, if the entire $350m discretionary IT investment [of this firm] had been retained as profit instead of spent on projects, the company's earnings per share would have risen, creating more than $5bn of additional shareholder value.
>
> – Richard Bhanap, Managing Director, KPMG Europe[1]

This is a book about investments in software, whether a capital investment by a large company, a research and development investment by a small company, or investment in a software-based product by a startup. Specifically, this is a book about governing those investments. Why does this matter?

Software, specifically custom software, is the principal means through which Information Technology has an impact on a business. E-mail, data centers, and ERP software are important technologies to a business, but they are utilities, no different from water or electricity. Utilities are costs to be minimized, not investments that yield a return. When a firm invests in developing software, either as a product or for its own operations, it is making a strategic investment in itself that the leadership believes will create a competitive advantage. These are not investments a firm needs to make: investments in software compete with acquisitions, distributions to shareholders, and other uses of corporate cash. As the introductory quote makes clear, to be an attractive use of cash, the returns on a software investments have to be compelling.

[1] Bhanap, Richard. The Best Return from your Budget Financial Times, May 28, 2008.

Companies and venture capital firms invest a lot of money into software, ranging from mobile applications that allow customer self-service and engagement, to algorithms that automate critical business workflows, to data analysis of spending patterns[2]. Yet software investments tend to exceed expectation in the wrong way. According to Standish, nearly a quarter are canceled prior to completion[3]. Research by Oxford University suggests one in six IT projects costs three times as much as forecast[4]. With only one project in three meeting time and cost expectations[5], investing in software development is clearly not without risks.

Traditional governance offers little protection against investment failure, for two reasons.

The first limitation of traditional governance is the lifecycle. Traditional governance follows a coarsely-grained phase-gate lifecycle that is designed to do two things: restrict outlay until the investment is more thoroughly researched and defined, and, if it is a capital project, keep it compliant with guidelines for capital investing.

[2] On average, companies spend just under a third of their software budget on custom software. Gartner estimates total worldwide spend on custom development to be $107 billion in 2015. See Hammond, Jeffrey. Forrester DataByte: Spending on Custom Software in 2010 Forrester, January 21, 2010, and Gartner Worldwide IT Spending Forecast Gartner Market Data Book, Q1 2015.

[3] CHAOS Report, The Standish Group, 2009.

[4] ‘Black swans’ busting IT budgets. BBC, August 26, 2011.

[5] CHAOS Report, The Standish Group, 2009.

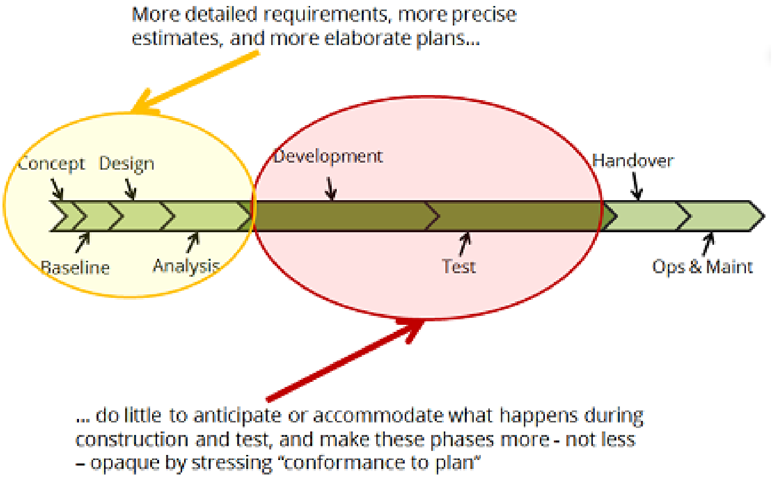
More detailed requirements, more precise
estimates, and more elaborate plans...
Concept
Design
Development
Handover
Baseline
Analysis
Test
Ops & Maint
... do little to anticipate or accommodate what happens during
construction and test, and make these phases more - not less
- opaque by stressing "conformance to plan"

Figure 1: A traditional phased-delivery approach to software development

This approach to investing is more financially than operationally biased. The early phases call for business cases, requirements, specifications, and plans. These are proxies for the one thing that can deliver results to the investor: the actual software. It is difficult to disprove the quality, completeness and accuracy of things like specifications and plans, so investments enter the largest and most expensive phase - construction - with more hope than hard fact. The construction phase is, for reasons we will explore in chapter 4, opaque to governance. Opacity of what happens during construction twined with a lengthy construction phase increases the likelihood that an investment accretes technical and functional problems. It is only when the investment attempts to exit the construction phase and enter test and validation that investors begin to discover the severity of problems with the software. The investor is left with the unenviable choice of either paying to correct the problems and starting the journey of being the one in six investments that sport a spectacular cost overrun, or canceling the investment and being one of the quarter that never make it.

The second limitation of traditional governance is the way governance is staffed and performed. Traditional steering committees are composed of senior managers and vendor representatives. Its responsibilities are largely super-management tasks: reviewing status reports from management, removing external blockers to execution, and confirming readiness for upcoming milestones. Because it is staffed with the most senior people responsible for delivery, it is an in-bred political body, not an independent board adjudicating the performance of the people delivering the asset and changing the course of the investment as necessary. Defaulting into self-regulation and equating governance with management adds nothing but a periodic checkpoint that expedites damage control negotiations when things go wrong.

In practice, software development is characterized by poor execution supplemented by ineffective governance. There has been a lot of attention given over the years to improving the prior, things like code patterns, automation, code quality analysis, Agile practices, and Lean concepts[6]. There is merit in each of these: it is intuitively obvious that better code, closer relationships among people in business and technology, and greater efficiency make for better execution and, ultimately, results. Patterns, Agile and Lean have culminated in Continuous Delivery[7], which sets the bar for the delivery and operation of software investments much higher. Although there has been substantial attention given to execution, there has been comparatively little attention given to improving governance of software investments.

It is wishful thinking that improvements in development practices like Continuous Delivery and Agile will eventually change the prevailing software investing models from within. Accounting rules, financial demands, legacy assets, operating pressures, deteriorating subject matter expertise, narrow responsibilities, and a shortage of skilled developers will co-opt, restrain, or undermine well-meaning

[6] This is a long list, including familiar titles such as Martin Fowler's Patterns of Enterprise Architecture, Mary and Tom Poppendieck's Lean Software Development, Bret Pettichord's Lessons Learned in Software Testing: A Context-Driven Approach, and many more.

[7] Humble, Jez and Farley, David. Continuous Delivery. Addison Wesley Professional, 2010.

attempts to redefine how a business operates. Agile and Lean can have significant impact on execution and, as we will see in later chapters, make for a strong foundation of good governance. But until they can offer up something more than vague platitudes about how companies should function[8] they will not get very far changing a company's investment structures.

Improving investment outcomes requires improving investor practices. I have written this book to help investors in strategic software understand and implement those improvements.

To that end, this book presents policies, structures and behaviors that lead to better investment performance and business innovation through technology. It is intended for people who have responsibility for the definition, finance and delivery of strategic software investments, to give them the knowledge and confidence to act as advocates for the success of those investments.

This book is just as much about behaviors as it is mechanics. Organization, policy and process do little good if there are not inquisitive people who fulfill a duty of curiosity. In the world of investing, the activist shareholder offers reference cases for effective governance behaviors, specifically in the way they challenge results and leadership as well as the way they re-define expectations of performance. The mechanics of governance and the behaviors of activist investing are described in detail in this book because, as we will see throughout this book, they go hand-in-hand with one another: governance is more effective with activist investors and investor activism is more effective with good governance in place. Conversely, bad governance engenders value destruction and stunts investor activism.

It is my hope that after reading this book, the strategic software investor will better understand the nature of corporate governance,

[8] There's no value in drawing attention to well-intentioned pontification. Although it is pontification - advice on how to run a business from people who have never run a business before isn't going to be much more - it is well intentioned. Even if the mechanisms are poorly thought out, their intent is aligned with the outcomes I am advocating in this book.

its value and why good governance is important to any investment; the mechanical processes of governance, and how to recognize the presence of good and bad governance practices; and the role of the individual investor, specifically how to behave in a thoughtful and responsibly aggressive manner with respect to the people consuming capital and those who committing capital.

Software investments are commonly measured in terms of execution, specifically time and cost of delivery. Management will default to using these measures out of reasons of familiarity (they are what have always been used) and comfort (they are easily controlled and manipulated). These measures are proxies at best and of secondary importance at worst to an investor's ultimate goal of return or yield. Good governance imposes an investing paradigm on management; the activist investor elevates management's goal from *compliance with a plan* to *best use of capital based on all available information today.*

The quote at the beginning of this chapter makes clear that the people who commit corporate capital to a software investment have very high expectations. Communicating and reinforcing those expectations are functions of governance. Seeing that execution is aligned with those expectations is a function of an activist investor.