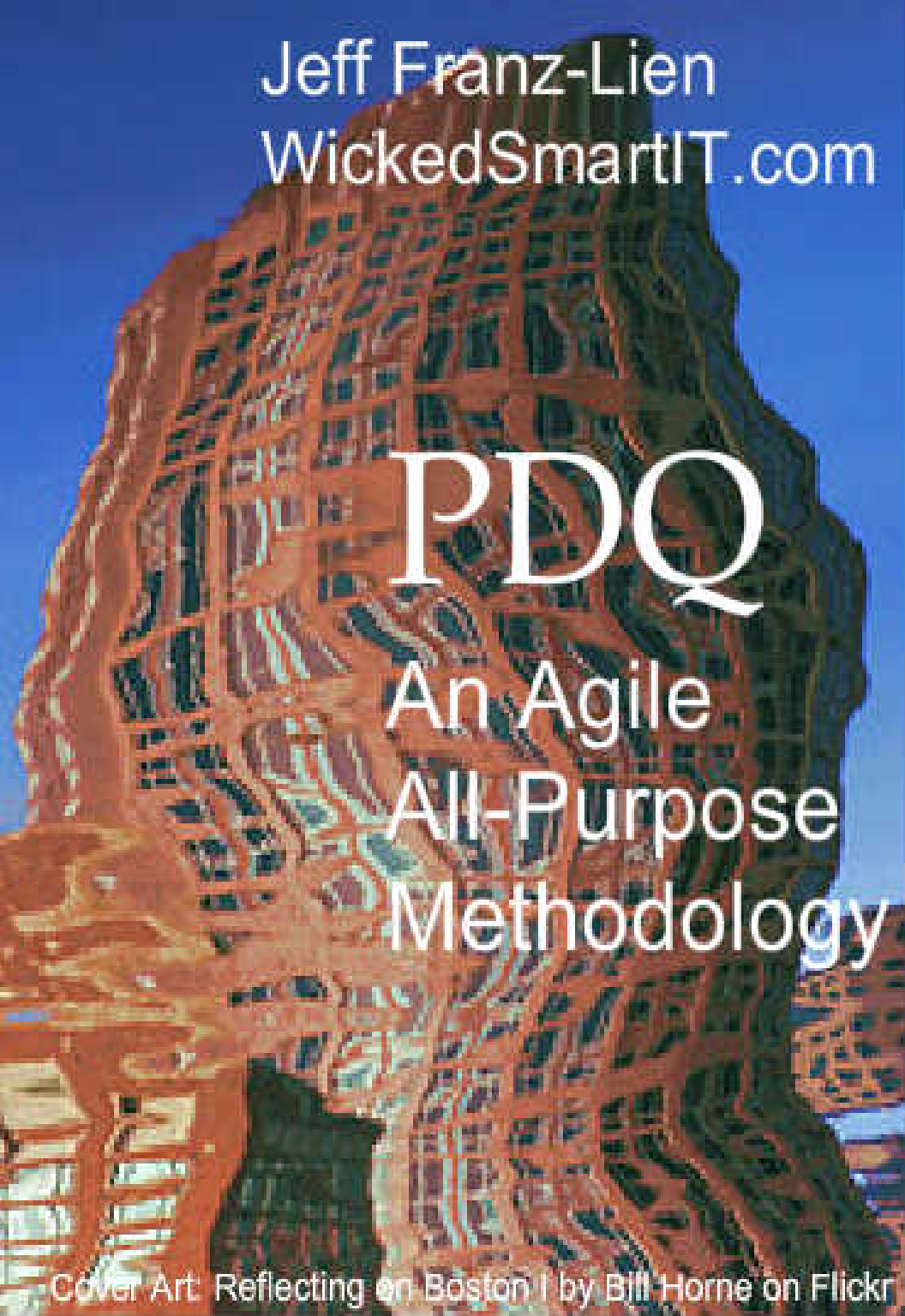
Jeff Franz-Lien
WickedSmartIT.com
PDQ
An Agile
All-Purpose
Methodology
Cover Art: Reflecting on Boston I by Bill Horne on Flickr

PDQ

Pretty Darn Quick: An Agile, All-Purpose Methodology

Jeff Franz-Lien

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

This version was published on 2013-03-30

This is a Leanpub book. Leanpub empowers authors and publishers with the Lean Publishing process. Lean Publishing is the act of publishing an in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

Tweet This Book!

Please help Jeff Franz-Lien by spreading the word about this book on Twitter!

The suggested hashtag for this book is #WickedSmartIT.

Find out what other people are saying about the book by clicking on this link to search for this hashtag on Twitter:

https://twitter.com/search/#WickedSmartIT

Contents

Chapter One: The Genesis of PDQ **1**

Project Description 1

First Application of PDQ 2

PDQ Results 3

The Invention of PDQ 4

Goals for PDQ 5

Research Design 6

Project Evaluation 7

Chapter One: The Genesis of PDQ

Project Description

In the field of Information Technology, agile methodologies abound (Larman, 2004), though all of these were developed for software development and none for IT infrastructure projects. Unfortunately, the activities in the software development life cycle and the infrastructure development life cycle are very different. While I discovered some techniques from agile software development methodologies could be adapted to infrastructure projects, there were many gaps requiring input from other sources. To address the absence of an IT infrastructure methodology, I developed PDQ (Pretty Darn Quick) specifically for an ERP platform migration project. PDQ was quickly cobbled together from a handful of accelerated project management techniques, put into practice immediately, and until now was mostly in my head. The methodology was very successful, helping us deliver in less time than anyone expected.

As the developer of PDQ, I was heavily involved in its first use, though my overall responsibility for the infrastructure department and its portfolio, prevent me from getting this involved in the execution of every project. I have a technical project manager on my team, but the size of our

portfolio also precludes him from heavy involvement in every project. Therefore, the project lead role must often be delegated to a senior technical analyst while the PM coaches and steers as required. My own ongoing role is sponsorship of infrastructure projects, providing resources, periodic steering, and removing roadblocks as required. I believe that PDQ, in a refined, documented form could help project teams learn to independently seek agile solutions and implement them in an agile manner. This Project will be such a document, first outlining and critiquing PDQ in its current state, then reviewing the available literature for possible improvements, and finally reintroducing PDQ in a refined state suitable for any IT infrastructure project.

First Application of PDQ

In the eight months from August 2009 to March, 2010, my IT technical team undertook a project to select, procure, and prepare the infrastructure for an upgrade to our ERP application. In determining the path we would take, the major alternatives were:

- keep ERP on our existing minicomputer systems which had reached end-of-sales and would be end-of-support within a few years,
- procure Itanium servers, a hybrid architecture which would allow us to keep running ERP on familiar UNIX and gradually migrate to Linux or Windows, OR

- retire our minicomputer systems and migrate ERP to our existing virtualization architecture including Intel blade servers, Linux O/S and VMWare.

Given its mission-critical status, any proposed change to ERP is taken very seriously. As I was to discover, everyone involved including consultants, our ERP vendor, other IT management, managers of other departments, and my own technical staff had very strong opinions about how we should or should not proceed. Some feared there was insufficient time to change the architecture and some feared that Linux and virtualization were too immature for mission-critical applications. Over the eight month project, we explored the three options and selected option C, developed a proof-of-concept installation, completed a detailed design, purchased software and hardware, and completed the final installation.

PDQ Results

On April 1st, 2010, right on time, my technical team delivered a working, upgraded ERP environment on the new platform to our Systems Analysts to begin data conversions and interface configuration. Between April 1, 2010 and go-live on January 4, 2011, the business and IT teams conducted a great number of tests and fixed the anticipated number of issues. The upgraded ERP system went live over New Year's weekend, followed by the quietest post-implementation period we have ever experienced. The ERP

platform migration was deemed a great success. Onlookers were impressed by the speed of our delivery. Moving from UNIX minicomputers to virtualized blade servers resulted in large savings ($600,000 per year), improved performance, and achieved remarkably greater flexibility.

The Invention of PDQ

As we began work back in August 2009, I recognized that to deliver an ERP platform by April 1st would require accelerated project management techniques. Waterfall methodology with its cautious "do one step at a time, do it very well, then move on" approach would easily have taken twice as long as the time available. I was taking a series of project management courses in the Master of Science in Information Management (MSIM) program at Aspen University. From my studies, I pieced together various project management concepts relating to speed and efficiency and assembled them into a cohesive methodology that I immediately applied to the project. These concepts included time-boxing, progressive elaboration, rolling-wave planning, daily scrum meetings, parallel development, and prototyping. I called the resulting methodology PDQ Project Management. In trying to validate PDQ, I examined Agile and other accelerated methodologies but found these purpose-built for software development. Given the urgency and importance of my ERP project, I had no time to study or adapt these methodologies.

Goals for PDQ

One challenge with inventing a methodology was communicating its workings to the team and making it take root while the project was in flight. Many of PDQ's principles ran counter to the established organizational culture and often met with resistance. While senior management has been promoting innovation and agility for some time, transformation of the organizational culture takes time and coaching. Documenting PDQ as part of the Project will certainly help my teams understand agile concepts and run their future projects more efficiently and with less support. However, while PDQ methodology was effective, it was tailor-made for one particular project. As such, the methodology may not transfer well in its current form to other projects. This Project will address these two needs by a) documenting PDQ so it is accessible and reusable, and b) broadening the applicability of PDQ to include all manner of IT infrastructure projects through study, assimilation, and adaptation of other agility perspectives and techniques.

An agile methodology specifically for IT infrastructure projects fills a definite need within my organization. In documented, academically reviewed form, PDQ is more likely to take root and flourish. Project leaders will be able to read about PDQ and apply it themselves. The IT management team, auditors, and other stakeholders will also have an opportunity to review and better understand this strange new methodology used by the IT infrastructure unit.

The detailed objectives for this project are as follows.

1. To review and improve PDQ methodology so it is reusable on other IT infrastructure projects of various shapes and sizes.
2. To make PDQ accessible to project teams in my department so they can learn and apply the techniques.
3. To incorporate knowledge areas in PDQ that are integral to IT infrastructure projects but poorly covered by other agile methodologies which were narrowly developed for the needs of software development projects. An example is software and hardware procurement which is not a prominent part of software development projects.
4. To declare possible future research extensions of this project. These could include further development of PDQ, case studies of PDQ use on individual projects or particular project types, or in-depth studies of PDQ methodology elements.

Research Design

To achieve these goals, I will first present a case study describing what PDQ is, why it is needed, and how it was used on the ERP platform migration project. Next, I will review the literature and document any concepts or paradigms that could either enhance PDQ or help illustrate

its principles to others. I will then restate PDQ, assimilating any enhancements arising from the literature review. Finally I will articulate possible research extensions of this Project. To help ensure suitability and buy-in, the project was developed in consultation with the CIO.

Project Evaluation

The new PDQ methodology has already been tested successfully on one major project, the ERP platform migration. In reviewing project results, the CIO remarked that not only did the platform switch save millions of dollars and increase agility, but was delivered on time and within a remarkably short timeframe. Had we stayed with the status quo platform instead, these very compelling benefits would have been pushed several years down the road. Therefore, the chief success of PDQ was allowing us to seize a compelling opportunity before the available window closed. Further testing and refinement of PDQ on upcoming projects would be interesting and valuable but does not fit the timeframe available for the Project. Therefore, scope of the Project will be limited to documenting and refining PDQ to facilitate its use on subsequent IT infrastructure projects.