50% off !
Click here to buy
Limited availability

AGILE PROJECT MANAGEMENT FOR GOVERNMENT

Leadership skills for implementation of large-scale public sector projects in months, not years

"A stream of practical advice."
Tom Gilb, the 'grandfather' of evolutionary project management.

Extract of Introduction and Chapter 1 Case Study

Brian Wernham

“A forceful, evidence-based argument
for the role of agile within government.”

Andrew Bragg, CEO,
Association for Project Management (APM)

“An enlightening insight into grand
failures and successes in government projects.”

Neil Coutts, Director,
Project Management Institute (PMI)

“A wonderful collection of real case studies.
A stream of practical advice. As broad a scope
as one could hope for. A view sorely needed in a
field long dominated by dogmatic developers and
code-centric softcrafters. I found myself ‘hooked’ on
this and read it cover to cover in a weekend.”

Tom Gilb, the ‘grandfather’
of evolutionary project management.

“A unique, insightful, and
readable guide to agile government
from a leadership perspective”

Dot Tudor, Winner of
Best Agile Coach Award 2011

Agile Project Management for Government

Brian Wernham

Maitland & Strong *London – New York – Sydney*

Agile Project Management for Government
First Edition, Paperback – published 2012
Maitland and Strong
ISBN: 978-0-957-22340-0

Maitland and Strong
21 Berridge Mews
West Hampstead
London

Contents

Guest Foreword – Dot Tudor

Agile Coach of the Year 2011

When Brian asked me to review this book and write a foreword, I approached the task with some trepidation. So much has already been said about agile approaches. How is this book going to add to the already-burgeoning body of knowledge? However, I need not have worried. This book takes a fresh approach and is really useful to anyone wanting to introduce agile into any large organization. It is a must-read for anyone engaged in large-scale projects and trying to change the organizational culture to nurture and not stifle agility.

The book is evidence-based and I appreciated the presentation of the case studies early in the book with sufficient detail for me to understand why and how they worked. The book then reassesses the 2001 Agile Manifesto from a completely new angle, taking a leadership perspective, which will help to support leaders in establishing the agile culture from the top down, the middle out as well as the bottom up. It does not give precedence to any one particular source of agile best practice guidance: agile is a family of approaches, which came together after the signing of the Agile Manifesto, primarily because they had so much in common. In the years since that challenge to the world of tight, directive project management and over-zealous focus on process maturity, there have been many successes and failures.

In some organizations the pendulum has swung from the very bureaucratic and process-focused approaches to almost total anarchy – and often back again. This book argues that management control and

Agile are not incompatible, and gives practical guidance on how to generate a culture of management and leadership in which Agile teams can work effectively. A culture is needed that fosters creativity and a sustainable pace of collaborative work, while keeping sufficient governance and the strategic focus on the business need and deadlines.

Brian's book is the first of its kind, to my knowledge, that looks at the application of the agile approach specifically within Government projects. It recognizes the constraints and restrictions, which are an integral part of the large, controlled, transparent, and necessarily auditable world of government. These case studies show that agile government does work. It also argues that for governments to succeed in their quest to gain advantages from an agile approach to projects, the focus must be on adopting nine specific Agile Leadership Behaviors.

The book considers government, in the UK and the US and elsewhere. However, I believe that its conclusions and advice are equally applicable to any large organization. Multi-national corporations managing multicultural teams across the world face many of the same problems and will find this book useful and practical. The nine Agile Leadership Behaviors are equally applicable as a focus for their levels of management and facilitation. The concept of "light-tight" control is a simple but effective reminder that Agile does not mean chaotic and loose, but requires empowerment at the solution development team level and a responsible approach to governance at senior levels.

It is about time that Governments around the world adopted the agile approach as a default for projects. Agile delivers value early and regularly. It reduces risk by providing early feedback of progress, by involving the right people at all levels and by embracing change.

This book adds to the body of knowledge on agile approaches by considering its use on large-scale projects in big organizations. It gives practical, experience-based advice for its effective integration into the complex cultures of such organizations.

Dorothy J Tudor, Technical Director, TCC Ltd July 2012
Agile Coach, Sandbach, UK, and the World

Executive Summary

Top officials on both sides of the Atlantic have too often failed to provide agile leadership. The seductive siren call of huge fixed price contracts to deliver technology usually ends up in disaster. In one case a supplier is fired. In another there is simply a resigned acceptance by a government of a flawed solution. Government customers and their suppliers can end up in death-embraces – where neither party can admit that a project is undeliverable. As one newspaper commentator succinctly stated:

> "Yet another outsourcing company collects profits when all goes well and the state picks up the pieces if the company fails. Soon much of the state may be too atrophied to step in."[3]

Governments must stop pretending that the business risks of large project failure can be managed by suppliers. Governments must manage these risks. They must stop trying to outsource mission critical work to be built in large, indigestible deliveries.

There are agile success stories out there: US Veteran Affairs, the FBI, the UK Ministry of Defense, the UK Government Digital Service, Housing Benefits in Australia – all over the world these pockets of excellence demonstrate that governments can be agile. For example, the New Zealand government instituted a disaster compensation system within three days of the Christchurch earthquake. The team responsible for the software used an agile approach to visually track their work on a continual basis. Releases of working software were scheduled on a half daily and sometimes even hourly basis. The system paid out more than AU$200m, and ensured economic continuity in the face of a natural disaster.[4]

I present cases of projects that governments around the world have implemented successfully using agile approaches, such as a safety critical defense project, a benefits payment project, and a federal criminal and homeland security project among many others.

I am proposing that the spread of **agile thinking in governments will be accelerated by the adoption of 9 specific Agile Leadership Behaviors that I identify in this book.** These 9 Agile Leadership Behaviors are a necessary foundation that will pave the way for agile success. If the concept behind a project is bad, then the approach should change. Cancelation of the project early with little harm done may be the best decision. The ability to change direction when facts are uncovered that upset prior ideas is a fundamental characteristic of an agile approach.

Also identified here are 6 Barriers to Agile Success. Agile thinking addresses the current *addiction to process* and mega-project mania to reduce risk and deliver on time. By thinking differently about how they agree their objectives up front on projects, go about procurement, and carry out project audits, governments around the world will overcome these 6 Barriers to Agile Success.

This is the first large scale research that has been published on agile project management for government, and I have been helped enormously by Chief Information Officers in governments around the world. They have seen how agile can deliver, and they want the call for change to be loud and clear.

There are many sources for best practice guidance on the agile approach. I have chosen to describe three in this book because they have different perspectives and strengths: the Dynamic Systems Development Method framework from the not-for-profit DSDM Consortium; the Scrum method described in the writings of Schwaber and Sutherland; and the eXtreme Programming (XP) techniques developed by Kent Beck.

I give examples of how these have been combined to get the rounded approach to agile that government needs. By reading this book you will be exposed to just enough jargon to enable you to sit down and talk to agile experts and ensure that your team processes will work under your leadership.

But although best practice guidance can help project processes, it is no substitute for leadership!

Introduction

The agile approach is best summed up as being a way of incrementally delivering change so as to get the earliest possible benefit, get feedback early on what works, and change direction accordingly. I argue in this book that governments around the world have for many years been doing the exact opposite with their technology developments. They have commissioned large projects that progress in a predetermined and unfaltering course, deliver late (if at all) and provide little or no benefit.

I have decided to lay out these arguments in the first part of this book using the Harvard MBA case study approach to compare and contrast the agile and non-agile approach. I avoided the classic book structure of 'history, theory, examples' because the first question people have been asking me when I told them about this book was "Can agile be used in governments?" Therefore I have turned that classic sequence of explanation on its head.

I start with fully attributed examples of government success stories. These are from around the world, including the USA (where the Federal Government is in the vanguard of demonstrating success with use of agile on some huge projects) and also the UK and Australia. These are real stories and are fully referenced. The case studies in this book actually happened and are fully attributed.

The central tenet of the agile approach is that we must be scientists. When we start a project we have a hypothesis that the outcome will be beneficial. We must test that hypothesis as the project progresses. Regular delivery of testable product provides the basis for ensuring that our projects are on the right track.

Much is made of the word "agile" in government today. "Government

IT needs to be more agile, more responsive and more accountable to the citizens" says the US Government. [5]

In the UK the government has vowed "to be more agile, more fleet of foot".[6] So, is agility anything more than a nebulous concept? If a government wants to be more agile what must it do?

Tom Gilb was one of the first to propose an incremental, agile approach to developing software. Rather than have large, clumsy, slow and ultimately risky projects that took years to complete, he proposed an *evolutionary approach* he termed "Evo":

> "Evo is a technique for producing the appearance of stability. A complex system will be most successful if it is implemented in small steps and if each step has a clear measure of successful achievement as well as a 'retreat' possibility to a previous successful step upon failure. You have the opportunity of receiving some feedback from the real world before throwing in all resources intended for a system, and you can correct possible design errors" [7]

In a 1985 paper, "Evolutionary Delivery versus the 'Waterfall model", Gilb introduced the EVO method as an alternative of the waterfall which he considered as "unrealistic and dangerous to the primary objectives of any software project". Gilb based EVO on three simple principles:

- ♦ Deliver something to the real end-user
- ♦ Measure the added-value to the user in all critical dimensions
- ♦ Adjust both design and objectives based on observed realities. [8]

Figure 1 shows a simple conceptual model that helps put this book into the context of government strategy. The outside bubble represents the desire by politicians and top management to be both *lean* and *agile*.

Lean Government has all unnecessary and wasteful 'fat' trimmed off. This is a process that not only boosts efficiency but also increases quality of output. Lean initiatives are generally internally initiated and maintained.

Agile Government is able to change direction quickly due to unforeseen or unforeseeable circumstances. This reduces risks of failure. Just as an athlete may fall attempting to jump over a hurdle that is set too high, in an agile world we set the hurdles at a comfortable height and at regular intervals. *Agility*, then, corresponds to setting short, realistic targets and reacting fast to changing circumstances.

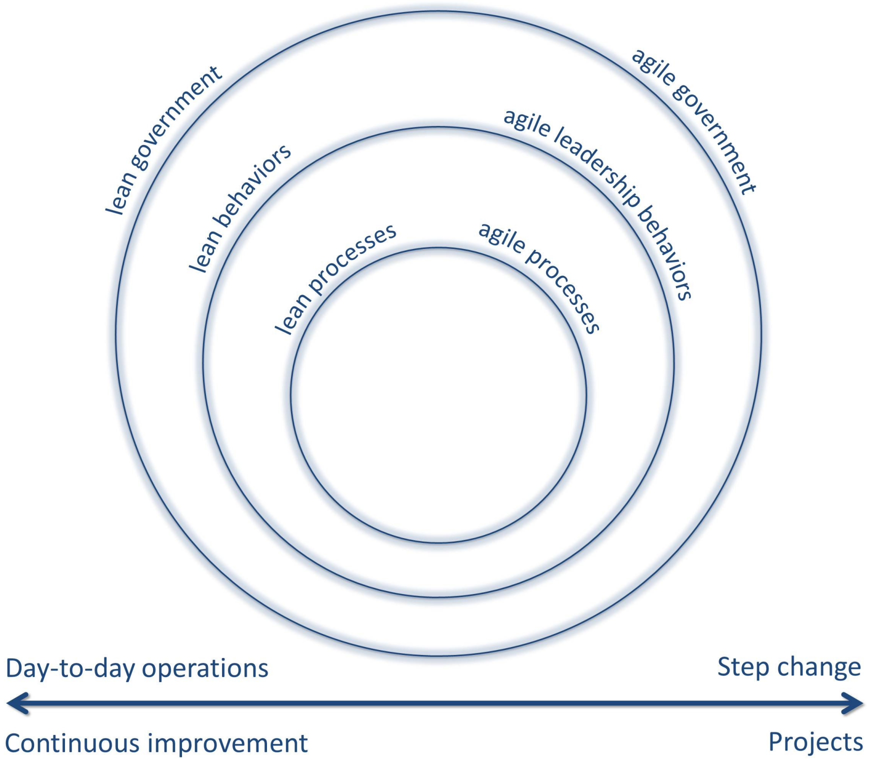

Figure 1: What does *agile* mean?

For example, the White House 25 Point Plan to increase quality and efficiency in US Government Information Technology (IT).[9] The UK Cabinet Office has published an IT Strategy that has similar objectives, with five out of the 14 points specifically relating to the adoption of agile approaches in development.[10]

When Vivak Kundra was sworn in as Chief Information Officer (CIO) for the US Government in 2009 he inherited $27bn (and that is billion not million!) in IT projects that were behind schedule and over budget. His

$1bn cancellation of the Military Human Resources System was just one of several actions he took to try to take control of a spiraling, out of control IT project budget. From 2001 to 2009, IT spending nearly doubled, growing at an annual rate of 7 per cent. But from 2010 onwards Kundra capped the IT Budget. Spend was forecast to rise to $104bn by 2013, the new forecast was just $79bn – a saving of $25bn per year.[11]

His 25-Point Plan called for a "modular approach to development using an iterative development process". It intensified previous attempts to move to an agile approach.[12]

Agile behaviors reduce the reliance on premature agreement of detail before development work commences. The proponents of this approach (often called *Agilists*) argue that as development gets underway new requirements appear that were not considered previously. Conversely the development teams discover problems (and opportunities) that can inform strategic decisions. They say that one should not imagine that a detailed specification for a system can be written years in advance of the development taking place and being implemented for use.

On the face of it, adopting an agile approach appears to be at odds with typical government bureaucratic approaches. I argue that although the turnaround to a new way of thinking will be a challenge, there is already evidence of success.

This book is about the adoption of agile in government and how to overcome the barriers to its introduction. The application of this book is relevant at local levels, not just central government. Some geographically local projects are of a staggering size. The Mayor of London, for example, spent £161.7m in setting up a congestion charge plan for the city.[13] Public bodies have planned significant technology projects. The US National Digital Information Infrastructure and Preservation Program were allocated $100m in funding from Congress in 2000.

The trans-Atlantic interaction between technology developers and project managers in the US and the UK is a central theme of the book. There has been a continual and fruitful interaction between the Governments of the US and the UK in the development of computers. The US Navy played a pivotal role in the British development of the first large-scale vacuum tube driven computer at Bletchley Park in England, which broke encoded Nazi war messages.[14]

Other authors have argued that agile processes can be scaled up to large projects.[15] But I propose here that a bottom-up push by *agilists* will take time and will run into organizational inhibitors. What is needed is leadership, especially at the strategic level. Although many agile concepts are complementary to existing approaches, and there has been more continuity in the development of project management approaches than many recognize, a change in leadership thinking is needed. It is the emphasis and strategic philosophy of management that needs to evolve to encourage agile and allow it to thrive in a government environment.

In delivering large projects in both public bodies and large corporations, I have had to work hard to make large, inflexible procurements more incremental and customer focused. Leadership of others, such as lawyers and procurement executives, played a crucial role in steering these projects to success.

My experiences in leading large teams in the USA (in North Carolina and New York) and in Canada mirrored those in the UK and Europe. I led projects that would now be termed agile that delivered the first automatic code generators for Windows-based computers. These projects revolutionized user-friendliness, decreased training requirements, and reduced error rates by replacing mainframe terminals at large public and private sector organizations. The key was flexibility in setting the team goals, and agreeing what the business was going to realize by delivering working solutions incrementally from an early stage.

Any practical project manager will know that it is better to deliver an imperfect solution early than wait forever for perfection. The banker J.P. Morgan is reputed to have said "I want it Thursday, not perfect!" The trick is to know what level of imperfection can be handled by the business and traded off against the early realization of the benefits of the solution that the project is going to deliver. Bill Gates knew this when deciding on the right moment to release the replacement for Windows 3.1. He called it Windows 95. It was officially named for the year of its release (1995), but my sources at the time told me that it was named after an internal slogan "95% ready, not 100% perfect".

Part I then, does not start with theory – it contains proof of success.

It tells stories of effective use of agile in the face of huge challenges. These stories are provided to give you the incentive to say "Yes – we can also be agile! Tell me how I can lead my colleagues so that they can also have agile successes!" In these cases, I see how projects around the world have used popular agile best practice guidance to achieve agile success. As previously stated, I focus on three sets of best practice: the DSDM framework, the Scrum method and eXtreme Programming techniques.[16] I have chosen to examine these three in this book because they have different perspectives and strengths, and, as we shall see later, they have been used together to great effect:

- DSDM provides an agile framework that can be applied to any type of project. It can be used to run IT or non-technology projects such as incremental construction or engineering. The DSDM framework provides practical guidance on agile governance processes, operational implementation, and project management together with team-level structures and techniques.
- Scrum is a method which provides guidance on technology development via a set of processes and practices at the team level. The Scrum method takes an unpretentious, empirical approach to the development of products which is easy to follow.
- eXtreme Programming (XP) techniques help IT developers work together, become more productive, and create high quality computer software.

There is a growing body of opinion that these three can be used to contribute to success on large government projects. Recently Craddock, Richards, Tudor, Roberts, and Godwin have proposed a promising approach for using the DSDM framework with the Scrum method:

> "One or more aspects of the DSDM Agile Project Framework may be used to supplement Scrum … where they make the use of the Scrum (method) easier (or) more effective."

As we shall see, where management has in mind a time and budget limited project to deliver change into operations, the DSDM framework may be successfully used as a wrapper around the Scrum method to create a hybrid of the best of both sets of guidance. This ensures that all those who may be impacted by the new system (the *stakeholders*) are engaged with appropriately. In the same way, when using the DSDM framework and the Scrum method together on a project involving IT development, it can be useful to include some XP techniques because the Scrum Method does not give guidance on specific software development techniques.

Many organizations embarking on agile projects for the first time feel that they have to make an exclusive choice, and adopt one set of best practice guidance only. This can lead to a very limited set of processes and some blind spots. I suggest here that you should consider using the best of all three of these sets of Best Practice, and Incorporate Good Agile Thinking from Elsewhere Whenever Possible. **Keeping an Open Mind to New Ideas and Fresh Evidence Is a Great Agile Leadership Quality.**

Part II then proceeds to give guidance on the leadership dimension. It explains the genesis of the Agile Manifesto and the related 12 Agile Manifesto Principles which define what agile is, and what it is not. I make the argument that it is the leadership perspective, not the process perspective that is most critical, and I propose 9 Agile Leadership Behaviors that you should follow.

Your adoption of these behaviors will reduce risk and encourage the use of the agile approach in your organization. Each of the nine chapters that follow examines one of these leadership behaviors in the context of government regulations, rules, unhelpful and inconsistent 'best practice' guidance, and organizational inertia.

More evidence of agile project successes around the world is provided and contrasted with the problems of the traditional *waterfall* approach on past government projects. The waterfall approach to project management requires each step of a project to be completely finished before proceeding to the next. For example, design then development then testing before use can start.

I put forward two main arguments. First, that agile project management provides a much better way of running most technology projects than waterfall approaches. Second, that without agile leadership, governments cannot become agile. Some practical advice is given in these chapters on specific improvements to make in the use of the DSDM framework and the Scrum method in your organization. At the end of each chapter I provide a list of agile leadership exercises you can do right now – even if you are working in a waterfall environment!

Part III of this book identifies 6 Barriers to Agile Success. These are the potential blockers to the adoption and spread of the agile project management approach. The 1990s was dominated by a desire to use large complicated design methods. The last decade was dominated by huge prime supplier outsourced contracts which crushed any incipient agility in many government offices. And procurement, regulations, and outdated approaches to project audit still remain the main inhibitors of agile adoption in government.

One interesting piece of news I discovered while carrying out the research for this book is that agile is being introduced into high-schools in New Zealand. Final year students are required to understand and practically use an iterative development lifecycle and the concepts of test-driven development of technology. Students will be required to demonstrate agile teamwork.

The new syllabus is very broad – ranging from how mp3 players work to e-commerce and the impact of technology on society. Although traditional technical skills are being taught, the stress is on getting a "taste of the discipline to find out if it suits them or not." The syllabus is now implemented, and researchers are tracking the students through the system to assess the results.[17]

The output of the first agile graduates into work and higher education is expected shortly not just in New Zealand, but from schools around the world. Governments need to be ready to make use of their knowledge, energy and enthusiasm....

Part I

Stories of Agile Success in Government

When I discuss the concept of the agile approach with leaders in governments in different countries, I get a lot of interest. These people understand the scale of culture change that is needed if the agile approach is to spread throughout government, and they want to know how to convince their colleagues. That is why I wrote this book.

The best way for me to convince you to adopt the 9 Agile Leadership Behaviors in the middle part of this book is to start with some real-life agile success stories. Whether you are a trainee or a senior director or politician, these case studies will provide you with the evidence that you need to lead your teams to agile success.

These events actually happened. I present a *warts and all* account of each one. Each is fully attributed – no anonymous case-studies appear in this book. I haven't selected small, experimental projects. These are all large, hairy beasts. As we progress though each case, I will introduce agile concepts and some jargon.

I start with a straightforward success story where the agile approach was adopted by the UK Ministry of Defense to develop a battlefield system to reduce the risk of friendly-fire in the wake of a series of friendly-fire incidents involving US and UK servicemen. After that we delve into an ultra-fast agile implementation at the US Department of Veterans Affairs which was on time and was a success, despite teething problems. I then tell the tale of the recovery of the failing Sentinel project at the FBI which was saved by a switch to an agile approach. Then a case from 'Down Under' where the State of Queensland in Australia has proved the worth of agile in parts of its organization, in stark contrast to its recent $1.2bn failure of the new Health payroll system.[18] Finally, I give an overview of how the UK Met Office has taken the best from agile and non-agile best practice to implement what they call "Just Enough Project Management", and how it has led to project successes.

Chapter 1

Case Study at the UK Ministry of Defense

There can be no substitute for the clear, positive ID of targets linked to unambiguous confirmation of precise location. The passage of positional data relating to both the target and the nearest friendly forces should be mandatory.

Board of Inquiry Report,
Ministry of Defence, 2004

We shall see proof in this chapter that an agile approach can incrementally deliver large mission and safety critical technology solutions. In this case, it did so quickly and is now on its way to protect the lives of coalition service personnel. It shows how the UK Ministry of Defense (MoD) successfully developed a new, improved battlefield system in the space of 18 months by using the DSDM framework.

In relating the case, I will give some concrete examples of the concepts behind agile.

We will see how the DSDM framework was used to provide governance and a project management approach to ensure that things got done on time and within budget. This is because it is important for you to be exposed, at least at an overview level, to some of the essential jargon that *agilists* use, and to get a gist of the processes they are advocating. In later chapters, I will describe the Scrum method and XP techniques which are also popular methods and are complimentary to each other and the DSDM framework. As mentioned in the introduction, each of these three sets of best practice addresses development project issues at different levels. But always bear in mind that one of the arguments of this book is that although best practice materials such as these are helpful, it is the leadership of management and those inside the teams that really make projects like the one in this chapter a success.

At the end of the case study I ask some probing questions that should prompt you to refer back to the text and provoke you into thinking more deeply about how you can adopt the agile leadership lessons therein.

Case Study Background

On January 14, 2009, Captain Tom Sawyer, 26, of the Royal Artillery, and Corporal Danny Winter, 28, of the Royal Marines, tragically died in a 'friendly fire' incident in Helmand province. A subsequent investigation revealed that they were killed by a heat-seeking missile fired by coalition forces in bad visibility while they were providing mortar ground support.

Many such incidents have occurred during combat operations in Afghanistan. This incident increased the number of British troops killed by friendly fire in Afghanistan operations to six. Incidents of friendly-fire are usually due to a lack of *situational awareness* of the combatants, not due to a lack of precision in the weaponry. Responsibility for command and control of fire is dispersed to individual units in the heat of battle, and the knowledge of who friendly units are and where they are situated is vital to those responsible for fire control in modern, fast-moving battlefield situations.

Poor situational awareness in combat is a key risk factor, often leading to friendly fire deaths. The board of inquiry into the killing of Lance Corporal Matthew Hull in Iraq 2003 found that the co-ordination between battlefield units and air units was lacking due to poor situational awareness.

The CIDS Project

The US, UK and other NATO forces have been developing and improving Combat Identification Systems (CIDS) over many years. In 2009 the UK Ministry of Defense (MoD) initiated a project to create a Combat Identification Server (CIDS). The CIDS was needed to tightly integrate close air support with shared situational position information.

A contract was awarded to General Dynamics to develop the CIDS to be in place by July 2010.[19] It needed to provide autonomous, accurate near real-time force tracking and location information to direct fire away from coalition troops. General Dynamics had only 18 months to integrate their "Net-Link tactical gateway" with specialist technology supplied by its subcontractors, Rockwell Collins and QinetiQ. Every few seconds, CIDS would integrate data from all the friendly forces in a battlefield and distribute it back to all the nearby unit commanders.[20]

Project Kick-Off and the Foundations Phase

To meet their objective of an 18 month implementation of this lifesaving software, the MoD chose an agile approach. They believed that complex military technologies could be better delivered without delay or unexpected cost overruns using agile.

A decision was made to use the DSDM framework because it gives guidance on the process for agile supplier delivery to a customer. The customer does not need to be a third-party – it could be the technology department within an organization. The important point with DSDM is that because it uses a *product centric approach* (see page 227) it has the potential to be used to formalize payment milestones with suppliers based on product deliveries.

The first phase of a DSDM project is called the *Foundations* phase. On the CIDS project the team analyzed the theoretical payment plan in the contract during their Foundations phase and found that it did not match reality. Both the MoD and General Dynamics recognized that a *win/win* situation was needed, and that traditional contract renegotiation would take time, and could lead to a deterioration of relationships before the development had even begun. They agreed to start work, and use evidence of progress to amend the scope of the required solution to fit with the planned timescales.[21]

How DSDM Avoided the Pit-Falls of Waterfall Projects

DSDM requires *just enough design up front* (EDUF), and the Foundations phase should be as short as possible, while still ensuring an essential understanding and clarity of structure of the overall solution and to create an Agile plan for delivery. This initial Foundation phase created an architecture that gave both the MoD and General Dynamics an assurance that minimum acceptable performance levels could be achieved.[22] For example, tracking information on the position of friendly forces needed to be collated from a minimum of 15 different units in any battlefield. The architecture also had to be flexible enough to allow near-real time position information to not only artillery units, but also to nearby aircraft. [23]

The approach ensured that test and evaluation of the solution was a "constant and regular activity", and allowed the development team and the stakeholders to gain more confidence with each iteration.

The project would run from February 2009 to July 2010. Overall plans were agreed at the end of the Foundations phase, which took three months. Then the Exploration and Engineering phase started with three iterations, each about 3–6 months long:

- Iteration 1: Create a simple version of the software that could deal with one friendly force position
- Iteration 2: Extend the software to process multiple position information
- Iteration 3: Make the solution robust and fast enough to deal with the operational number of request responses and to interface with systems from other coalition partners.[24]

The MOD planned practical demonstrations for June 2010, before final deployment took place in July 2010.

DSDM stresses the need for scalability from the smallest project to the very largest. It concentrates on governance and structures around incremental project outputs. It was first published in the UK in 1994 as an alternative rapid development method, which would avoid the pitfalls of the traditional waterfall approach.

Waterfall projects are segmented into discrete phases, each dependent on the completion of the previous phase, but without feedback or iteration. When using a waterfall approach, one cannot start a phase until the previous Has Been Completed. This Leads to a Series of One-Way 'Gates' (see Figure 2). Once One Has Committed to Swimming Downstream, It is impossible to return to an earlier stage without a lot of effort –similarly difficult to attempting to swim up a waterfall. In contrast to doing just enough design, a waterfall approach requires a grand design in detail before any solution building commences. A waterfall approach is appropriate for some civil engineering projects that are monolithic in nature, such as building a skyscraper, but in technology projects a waterfall approach will tend towards what Kent Beck called 'Big Design Up Front' (BDUF) when describing a fundamental problem of the waterfall lifecycle – that it relies upon pinpoint accuracy and perfect logic at every step if it is to produce a workable solution.[25] Kent's argument, and one that I emphasize in this book, is that we should aim for Enough Design Up-Front (EDUF), not BDUF.

Although DSDM started as a proprietary method closely controlled by a small consortium, in 2007 the decision was taken to make the method more openly available. The manual is now available to all for free on the Internet at www.dsdm.org and training may be bought from many suppliers (subject to training body certification requirements of the DSDM consortium). [26]

At 202 pages, the DSDM handbook may not at first glance appear to reflect the ideal of a 'light-weight' method. However, it supplies the role and process definitions often required for large projects by government regulations, and thus provides a useful template for a project management framework. It is process and output orientated, and gives seven main steps for every DSDM project, creating 43 products – each described in the handbook with some detail. Prior to the Foundations step, the customer (perhaps with the help of expert suppliers) should carry out the Feasibility step.

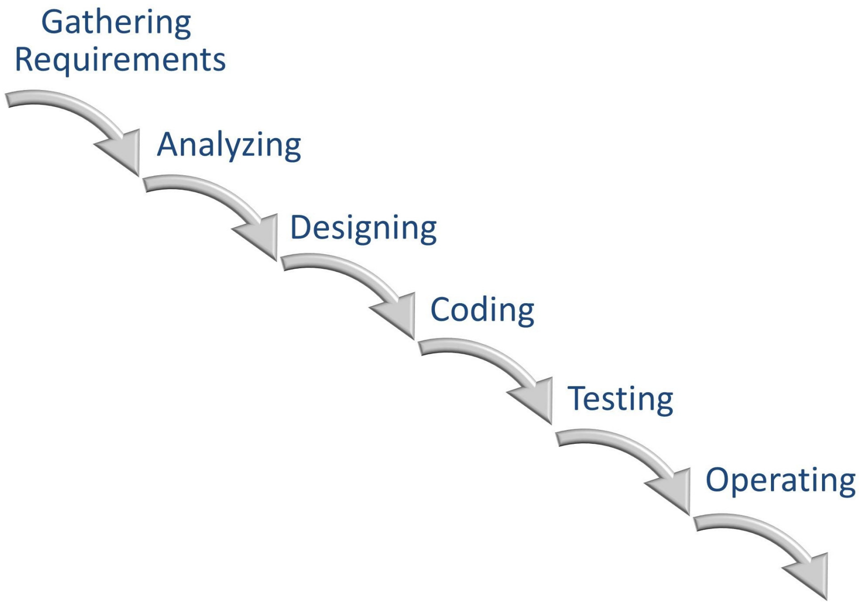

Figure 2: An example of a waterfall lifecycle

The method gives guidance as to the level and approach needed to produce an outline business case containing enough information to make a decision, but no more. If the project is given the go-ahead, then in the Foundations phase of the project this business case is expanded just enough for internal needs and government regulations. After the project is finished, DSDM gives advice on collecting lessons learned, evaluating the project performance against expectations, and monitoring the business performance of the solution against the business case.

One of the strengths and flexibilities of DSDM is that it gives guidance on how the iterative development work of Exploration and Engineering should be carried out alongside Deployment. It encourages flexibility in how these could be combined together, or omitted. Some projects initially need to iterate Exploration and Engineering many times, building models of different solution options, before proceeding with the iterative Engineering of a solution and its deployment. For example, if one month iterations are being followed, but updates to end-users are restricted by a wider organizational policy to once every three months, then only every third iteration will include a Deployment step.

Iteration and feedback is the core of DSDM, and it makes it very different from waterfall approaches. Its strength is that it presents agile concepts from a management point of view, using terms that traditional project managers understand while avoiding a waterfall approach. Like many methods, though, it has little to say about leadership behaviors. Processes and outputs are defined that are amenable to 'traditional' project management techniques, but have an agile approach. For example:

- ♦ Quality planning is used to define the necessary levels of acceptance for project outputs – this provides a description for each output that can be objectively tested and audited (see *definition of done* later)
- ♦ Requirements planning is used to maintain a Prioritized Requirements List (PRL), with mandatory release dates defined for all mandatory requirements, and tentative release dates for others
- ♦ Earned Value Analysis (EVA) can be carried out to compare the actual versus estimated development effort originally expected for each product feature in the PRL, thus providing feedback on the accuracy of the original estimates and the productivity of the team. (EVA is a technique that is controversial with agilists, as discussed further in Part III).

Thus a high level of compatibility with traditional formal management techniques can be achieved, but coming from the direction of flexibility and iteration, rather than upfront, detailed plans that become set in stone as *baselines* to be measured against.

The DSDM framework is an agile approach and guards against cost and time overruns by turning the *baselining* model on its head. In a waterfall project, a detailed *baseline* for the scope of a project needs to be agreed upon – supported by detailed design assumptions and theoretical estimates. Hence the phrase Big Design Up-Front (BDUF). If the estimates are inaccurate (and of course they often are because they are made before work begins and actual progress starts to be measured) the only variables left in the equation are cost and/or timescales.

Stakeholders flex their muscles and ask for additional *nice to have* features which cause the required amount of work to increase: a situation known as scope creep. This is why so many waterfall projects go over time and cost. Since the baseline is fixed, these mutually dependent parameters are allowed to run out of control. And what is more, waterfall projects usually implement one risky, disruptive, large change to operations: the *big-bang* approach which we will encounter again and again in the stories of large project failure in this book.

DSDM is an agile method and therefore has a different philosophy from the waterfall approach. When it is used as the project management framework to guide the team, only the central core of solution features is identified at the outset. The scope is allowed to change, in a controlled manner, as the inevitable mis-estimation of time and cost becomes clear. The opposite of scope creep takes place – scope is reduced if difficulties are encountered, rather than time and budget being increased. The project comes in on time and cost because DSDM fixes these variables, and instead re-scopes the Features to Be Delivered. in Effect, There Is Zero Time or Cost Contingency, but There Is Contingency in the Scope of Requirements (see Figure 3).

At its Simplest, Features left out of one iteration are simply deferred to the next iteration. This can work both ways: if better than expected progress is made, then features that were only on a wish list for an iteration may be included – some delight and surprise for the stakeholders!

DSDM suggests that no more than 60% of the work expected for each iteration of development should be on features classified as *Must Haves*. About 40% of the remaining work is split between *Should Haves* and *Could Haves*. The *Should Haves* are features that would be painful to leave out, but a workaround could be found for them otherwise a Must Have would be compromised.[27] *Could Haves* are features that bring additional value-add and business benefits, but can be delayed for future work without any immediate downside. To complete the picture, and to ensure that limitations to scope are understood, some requirements are classified as *Won't Haves*.

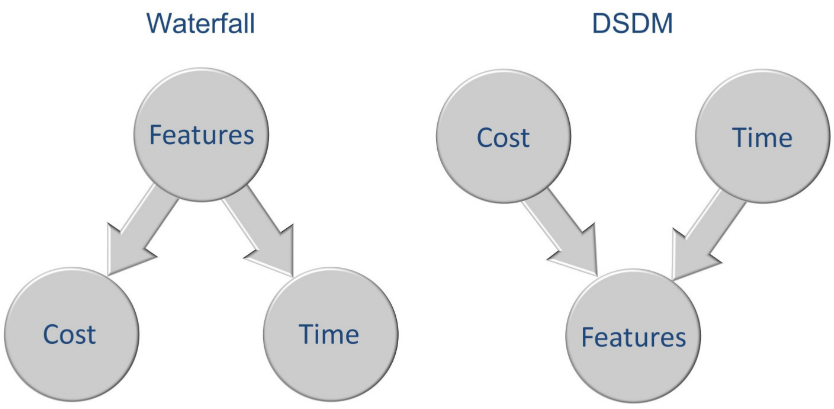

Figure 3: Waterfall: Features are the driver — DSDM: Cost and time are the drivers

Of course, 60% is a rough rule of thumb. As the project progresses, the team's *velocity* will be *calibrated* against the PRL. Each PRL item can be sized using the idea of *story points* rather than notional person-days. These story points are a relative measure of the size of each item. This concept cuts away the idea that plans can be accurately estimated in detail up-front. Progress is measured as the number of story points per day per team member, not the number of person-days notionally assigned to a set of detailed tasks at the start of a project.

It is only when the team gets going that the actual rate of progress of that particular set of people, technology and problem domain can be determined – by feedback from actual experience, rather than conjecture and theory.

The actual percentages should be reviewed with regard to the predictability of the overall scope of the project and the calibration of the velocity of the team. If the scope is well understood, in a stable business environment, and the target technology has been previously used, then perhaps a lower percentage of requirements could be in the tentative category of 'Could Have's'. However, it is tempting to make simplifying assumptions and start to move back towards traditional fixed-scope estimating. The risk is that the assumptions may be false, and development will then be more problematical than expected. It is better to achieve an over-delivery of output features than promise too many mandatory features and not deliver.

Requirements Planning in DSDM

A key control in DSDM is the list of requirements or Prioritized Requirements List (PRL). It lists all the requirements and states, which are most needed for the upcoming iteration. Every requirement is prioritized into four categories, referred to by the acronym 'MSCW'. Often this is pronounced and written as '*MoSCoW.* These are the *Must Have, Should Have, Could Have* and *Won't Have* requirements. This technique, which is central to DSDM, helps create flexibility and *Agileness* by three tricks:

- ♦ Priorities are set within the framework on iterations which are *timeboxed* – the deadlines are immovable. The team has delegated responsibility as to which *Should Have* and *Could* Have features they will deliver. Of course at the core of their work are the *Must* Haves. If a delivery is to be deployed into live use, rather than as a prototype demonstration of capability, the *business sponsor*, who is the executive responsible for the success of the project, and the team members, which includes key users, should create a joint *deployment plan*. Decisions are set at the lowest level possible so as to reduce the cycle time in decision making and ensure that quality and delivery timescales are met.

- ♦ Priorities are set for each iteration and change as the project progresses. For example, features that are *Should Haves* for one iteration may be promoted to *Must Haves* for the next iteration.

- ♦ Quality is protected: if an essential feature in the emerging

> solution is not of sufficient quality (it functions incorrectly, is unusable, or cannot meet capacity or other performance requirements) then it can be descoped from delivery for that iteration. Mike Cohn notes that the *could have* requirements items in an DSDM Prioritized Requirements List work as a *feature buffer* which can be sacrificed as required so as to ensure deadlines are met.[28]

DSDM ensures that the necessary governance is in place so that if any *Must Haves* are likely to fail these tests, a Business Sponsor is in place and responsible for decision making. In these cases it is usually necessary to change the deployment plan. Effort is always focused on ensuring that the highest priority features are of adequate quality. Research indicates that on average only 45% of features of technical solutions are used to any great extent, so a ruthless approach to descoping *Could Have* requirements is needed if the overall project is to produce early benefits and have a robust business case.[29]

A Solution to 'Friendly-Fire'

Forward Air Controller engagement scenarios and acceptance criteria were developed with real-life military operators collaborating in the Exploration and Engineering iterations. These tests were indexed against the requirements itemized on the PRL.

For example, air requirements such as interfaces to and from the "Link 16" air intelligence system were *must have* requirements, whereas armored situational awareness systems, such as the Force XXI Battle Command Brigade and Below (FBCB2) system was categorized as being a *Should Have* requirement.[30]

Developing the CIDS in Timeboxes Within Each Increment

Each of the 3–6 month long increments was divided into timeboxes of a month. In each timebox, the team, which included domain experts, was encouraged to get on with the work without interruption. Deadlines were sacrosanct. If within a timebox it became apparent that any of the *Must Have* requirements were at risk, then the team self-organized the redeployment of people away from development of solutions for lower priority requirements. This is the essence of Agility: decisions are delegated to the lowest possible level – to those closest to the work at hand.

A key discipline of Agile is that deadlines cannot be extended. Any issues become evident immediately and a positive attitude towards failed tests is encouraged. It is better to *fail early* and rectify an error, than to hope for the best and carry on regardless. In this case, through a *requirement trading* process the MoD agreed that a few *Should Have* and *Could Have* requirements could be descoped from the PRL. The important factor was that agreement was achieved without any penalties being incurred on the supplier, or any cost or schedule overrun for the customer.[31]

The supplier project manager from General Dynamics led a multi-disciplinary team comprising staff members from the other sub-contractors (Rockwell Collins and QinetiQ) and also MoD staff and their specialist technical advisors. The MoD designated an overall *Business Sponsor* (responsible for the success of the whole project), and a *Business Visionary* (responsible for decisions on day-to-day issues). Risks were recorded on a *risk register* and linked to items on the PRL to provide a means for prioritization and replanning. Again, the overriding concern was to maintain an iron grip on cost by flexing the delivery of functions to deal with risks as they emerged.

On this project the potential for 'tit-for-tat' negotiations over points of detail and costing were considerable. Trust had to be built up between a potentially suspicious customer not used to an agile approach, and a supplier under pressure to deliver.

The MoD procurement division initially proposed a severe penalty clause to guard against the possibility that the supplier would not deliver all the requirements – even the *Could Have*s. However, a collaborative approach was agreed upon, following the DSDM principles of *fixed cost* rather than *fixed scope*. Any difficulties encountered were to be resolved by *requirement trading*. In effect the project contingency was held in the *Could Have* requirements which could be traded out if unworkable or too onerous.[32]

Laboratory Integration Tests

As planned, at the end of summer 2009, integration tests started to take place at the UK Battlespace Laboratory. The Battlespace Laboratory is an independent body managed and owned by the MoD. It brings together government, industry, and military coalition partners from across the world to collaborate on highly realistic simulations of battlefield conditions.[33]

A test took place at the Battlespace Laboratory at the end of every 3-6 months in line with the development team delivery iteration schedule. The aim was to work towards a final defense demonstration servicing 50 interoperating battlefield positions. This testing was at Technology Readiness Level 6 or TRL6 (explained in more detail later on page 120) and had to be seamless and free of any significant bugs. A realistic demonstration to the military was carried out at the end of each laboratory test to increase their confidence.

Testing was based on a number of scenarios. In agile teams each of these is called a *user story*, but given the special context of the battlefield the team used the term *vignette*. One vignette, for example, was based on a land battle group carrying out counter-insurgency operations. The simulated mission was for the UK to coordinate a multi-nationality NATO attack on an insurgent compound in a desert storm. Proof was needed that the system would be able to provide friendly force ID to all units (including artillery and attack aircraft) in a difficult environment.

The team found that an agile approach instilled a discipline of delivery into the formal test environment at the end of every iteration come what may. An agile approach of immoveable deadlines ensured that intensive use could be made of the Battlefield Laboratory on the expected dates, thus making best use of an expensive and limited facility. A focus on interoperability was the key to the development. Although various items were re-prioritized for each iteration, in the end, the flexibility and discipline of the DSDM framework adopted meant that important requirements were not sacrificed and CIDS and the other battlefield systems all linked up to each other successfully.[34]

Joint US and UK Interoperability Testing

Full coalition interoperability testing with all coalition partners at TRL8 (the highest technology readiness level) was to take place the next year at the next "Bold Quest" coalition Combat ID (CID) capability assessment organized by the Joint Forces Command for 2011. These demanding exercises are aimed to enhance situational awareness, targeting, and minimize "collateral damage and fratricide".

Rather than wait for that event, the MoD decided to carry out a previously unscheduled trial to prove the system at TRL 7 well in advance of "Bold Quest".

So, in Norway, in August 2010, the new UK CIDS system was demonstrated side-by-side with the US CIDS by joint coalition ground and air forces – and all were able to successfully communicate with one another.[35] Both static tests and dynamic tests were undertaken using known positions and mounted and dismounted Norwegian soldiers exercising controlled scenarios.

Over 90 dynamic user friendly force information requests were made from the ground and the air using seven different systems.[36] Throughout the exercise period, the CIDS proved to be highly reliable, providing friendly-force position data within an average of three seconds to within five meter accuracy.

In addition, even though the Danish aircraft had not been specifically prepared for the exercise when they arrived they were immediately able to make successful requests friendly force ID using their Link 16 technology.[37]

Conclusions

In this first of five case studies we have seen how an agile approach delivered a large mission, and safety critical technology solution, and we explored some of the concepts behind agile, and the DSDM framework.

Questions

1. The MoD had indicated their inexperience with agile approaches. What risks did this represent to their business case?
2. What strengths and weaknesses are there in the application of DSDM to the CCID project outlined above?
3. The MoD procurement division was keen to 'nail down' the suppliers to a fixed specification. What may have been their thinking? How would you draw up an agile contract that would fairly hold a supplier to account for poor performance? How would it also ensure that the customer is held to its responsibilities? (More on this in Chapter 2.)
4. The Agile Manifesto Principles expect projects to iteratively deliver working solutions and have a natural preference for shorter rather than longer timescales between iterations (see Table 2).
5. Did the CIDS project meet these criteria? Could more have been done to make the project more agile?
6. Look at the Henson's presentation of the CIDS project plan (see Endnote 38). Compare it with the waterfall lifecycle (see Figure 2). What similarities are there? What differences?

Chapter 2

Case Study at the US Department of Veterans Affairs

With no significant bugs reported … operation nearly flawless – a stunning and an unpredicted success what are the implications for failing IT programs across government?[39]

Roger Baker,
Assistant Secretary for IT,
Department of Veterans Affairs

Some of the most widespread uses in technology in government are for claims and payment processing systems. Governments and their national economies depend on these mission critical administration systems, their accessibility, and their capabilities. In this case study I will show that these massive systems can be developed and incrementally implemented using an agile approach.

This case study describes how the US Department of Veterans Affairs (VA), implemented a large education claims processing solution quickly and with no major technical bugs. The project was a success, but I include some of the criticisms of the rushed operational implementation. The fast implementation of such a large system would not be without some pain, and it is instructive to examine how these problems occurred and how they were tackled.

In this case, I show how an agile approach provides a natural fit for the need for phased implementation where there is a policy directive that must be implemented urgently. This case examines how attitudes at national audit organizations, such as the US Government Accountability Office (GAO), can sometimes inhibit the adoption of agile. It highlights the need for a deeper understanding of agile concepts in the audit community, especially the focus of auditors on the processes used for testing, rather than the practical outcomes achieved.

This success story is the tale of using an agile approach to rapidly and successfully create the operations needed to implement major pieces of related Congressional legislation. Use of the system LTS has supported VA in delivering over $19.3B in educational benefits to over 760,000 Veterans, Warfighters, spouses and children. The number of claims processed has risen from 200,000 applications a year to about one million a year. Over $5bn is now disbursed annually as a result to those who have served their country.

Background

Historically, VA had experienced significant IT development and delivery difficulties. Individual directors at more than 1,000 sites controlled over 97% of the IT budget. Systems and processes could not share information across the department, and management could not be sure that data backups were being carried out.

On May 3, 2006, the home of a Department of Veterans Affairs (VA) data analyst was burglarized, and an employee's personal laptop and external data storage device were stolen. Stored on this equipment were the names, birthdates, and Social Security numbers of approximately 29.3m veteran personnel and their spouses.[40]

A strategic review decided that from 2007 a policy of centralization of IT would be put in place called the "One VA" policy. Its objective was to ensure security and consistency of data and processes across the organization and to centralize IT.[41]

On June 30, 2008 President Bush signed the Veterans Educational Assistance Act ("Post 9-11 GI Bill") into law. This bill added a new "Chapter 33" to a section of the United States Code. The "One VA" policy helped ensure that an integrated approach to implementing the bill could be taken. The law echoed the original 1944 "GI Bill" (officially entitled Servicemen's Readjustment Act) which was intended to support ex-servicemen and women in education and training after their active service in the Second World War was completed. The 2008 bill offered substantial financial support to those eligible, including payment of full state-university fees, housing allowances and book expenses. VA was responsible for implementing the necessary processes and starting to make payments from August 1, 2009 – just 18 months after the Post-9/11 GI Bill was passed into law.

Not only was time of the essence, but the bill was complex. It had intricate support and eligibility requirements, including the right to transfer benefits to dependents, and a special "Yellow Ribbon" program for support for study at private universities. Right from the start, the need to encompass likely changes and additions was obvious. And indeed, even though the bill had bi-partisan support in Congress and the Senate, it has subsequently been amended and changed several times, including a second version to the bill which widened eligibility to members of the National Guard.[42]

VA not only needed a fast approach to developing a solution to processing these claims, they knew that they needed future flexibility for the inevitable new benefits that would need to be paid in years to come.

The "Chapter 33" Solution

VA had to implement new interim operational processes almost immediately, using spreadsheets, and manual workarounds, together with the existing systems for disbursing monies. A project was set up to automate these interim processes in four interim increments, each delivered at the end of each quarter of 2010.[43]

The waterfall project approach could not have catered for the short timescales for full implementation, and would have left the interim processes running for over a year. An agile approach was needed to develop a system so that VA personnel would handle the complexity of the rules that had to be administered. The project would also need to cater for changes to the details of the regulations as they were agreed during 2010. For example, in 2011 the Post-9/11 Veterans Educational Assistance Improvements Act was passed that required a 60 day turnaround time between approval and implementation into production.

The replacement system was called the Chapter 33 Long Term Solution (LTS). The main problem the team faced was that large-scale agile practices needed to produce traditional Government artifacts and be assessed at gate checkpoints. For example, it met the VA's need for formal independent testing of money systems, and traceability of bug fixes. The implementation of the Chapter 33 requirements was one of the first implemented under VA's Program Management Accountability System (PMAS) which required projects to deliver new functionality every six months or less. LTS was planned to deliver a major feature on average every two to three months.

The project developed and implemented the first two of the four releases of the LTS software on March 31, 2010, and June 30, 2010 as planned. The regional processing offices were provided with key automated capabilities to prepare original and amended benefit claims on time. Legislative changes and housing rate adjustments that happened during the development were also incorporated. All new benefits from the start of the 2010 academic year were supported by the new system, and no significant bugs were reported. VA stated that operation was "nearly flawless" and a "stunning and unpredicted success". It stated that:

> "(Although VA had) one of the worst track records in systems development (as amply documented over the years by the VA's Inspector General and the GAO) we have been able to achieve a stunning success"[44]

This was a large project – $84.6m was spent on the first phase, and the GAO congratulated VA for taking an agile approach:

> "VA has demonstrated key agile practices that are essential to effectively managing its system development … the department has ensured that teams represent key stakeholders and that specific agile roles were fulfilled … The department has also made progress toward demonstrating the three other agile practices – focusing on business priorities, delivering functionality in short increments, and inspecting and adapting the project as appropriate."[45]

Impact on Operations

The hurried timescales of the new legislation led to staffing challenges. No-one knew what the take-up by veterans would be. There were even predictions that service people might leave the forces early to take advantage of the plan. Over 750 new staff members were hired to process the claims, but this proved inadequate as they were inexperienced. The interim system did not automate many processes, the office space was inadequate, and staff turnover as a result was high. In the end VA not only sanctioned overtime for staff, but actually mandated a minimum of 24 hours of overtime per person per month, including weekends. A contractor was hired to process over 150,000 other education claims and staff members were reassigned from other functions.[46]

When claims had started to come in, under the interim processes, the average time to process a claim was over 80 days, against a target of 24 days. The handling of supplemental claims overwhelmed the call-center to begin with and the GAO was critical of the advice given to applicants and the handling of emergency payments.[47] Despite these difficulties, in the first year there was a payment accuracy rate of 96% against a target of 95%.[48]

Problems with Interfaces to Other Systems

You will notice in this and some of the following case studies, that data conversion, the creation of smooth data links to other systems, and performance problems cause a lot of problems in technology projects. In this case, the link to the Defense Identity Repository was deployed seven months late, in October 2010, after many problems, but it was deployed. Interfaces are a common factor in delaying many projects, but are an unavoidable fact of life.

A lesson here is that when planning a project, the data conversion programs and interfaces should be among the first modules to be built and tested, not among the last. This ensures that any problems with existing data can be identified and tackled early on, for data usually takes months or even years to be 'cleaned'. For practical reasons, it is very useful in performance testing to have a full sized database available early in the project. And finally, interfaces are complicated, for example an old system may use a longer format for postal addresses than the new one, causing truncation of some lines of information unless special *data cleaning* programs are developed.[49]

Conclusions

VA managed to put in place the operations to cater for a complex, and changing piece of legislation by using an agile approach. The development progressed at a fast pace, however, there were some problems with implementation which provide a lesson for future agile projects.

The operational teething problems highlighted in this case are not unusual for any rushed project. Most of them derive from the use of the interim system before the LTS was phased in. They show the importance of careful planning for business transition to new ways of working. The problems were understandable, given the speed of implementation which was driven by the need to implement support for urgent legislation, and to do it fast. However, the risks associated with the short project timescales need to be managed carefully. The Scrum method pays no specific attention to interfaces, data conversion, and user implementation planning.[50] To make sure that these areas are integrated into a holistic project approach, a service-oriented framework, such as DSDM, can bring value. It covers the need for focus on interfaces to other systems, user training, using agile to develop new business processes, and how to implement a solution smoothly into Business as Usual/operations.[51] Sometimes an agile project must work alongside waterfall projects, or be part of a large waterfall program of work. This can sometimes be an inescapable fact of life. For example, an existing contract with a supplier may have been drawn up as a result of waterfall procurement. In these cases a superstructure of waterfall project management using standards such as ANSI 99-001-2008 or PRINCE2™ may be appropriate. These aspects of project management need to be planned hand-in-hand with technical development so that the technical solution is implemented smoothly. Where large scale business change is required, a program approach may be adopted to ensure that communications management and benefits management are effective. Relevant program management guidance can be found in the guidance issued by the US Project Management Institute (PMI) in its Standard for Program Management. The UK Association for Project Management (APM) and the UK Cabinet Office also provide useful guidance on strategic governance of programs which has a more external focus than the PMI materials and may usefully be used in conjunction with them.[52]

Despite these teething difficulties, which were overcome, the overall program has been seen as a great success. In the four years since 2008, over 745,000 applications have been processed and $19bn disbursed by the new technology. Senator Jim Webb was pleased with the outcome. He recently said that:

> "The Post-9/11 G.I. Bill is the best veterans' educational program in history." [53]

Questions

1. How was such a large, new operation enabled by the "One VA" initiative?

2. The new system was implemented incrementally. In what ways might this have helped or hindered cut-over from the spreadsheets and other office tools that operations had initially used to process each application?

3. Implementation needed to be in short timescales. Was enough thought given to getting operations ready for the size and scale of processing needed and the number of staff members needed?

4. Look at the more detailed description of the phasing in of the new system provided by the GAO (see Endnote 54). What would a big-bang cut-over to the new system have looked like?

5. Read the GAO's report detailing some other implementation problems (see Endnote 55). Could additional measures have been taken to make implementation smoother for emergency payments and to keep operational costs minimized?

Index

Agile Project Management for Government

"A wonderful collection of real case studies … a stream of practical advice … as broad a scope as one could hope for … a view sorely needed in a field long dominated by dogmatic developers and code-centric softcrafters … I found myself 'hooked' on this and read it cover to cover in a weekend."

- Tom Gilb, the 'grandfather' of evolutionary project management.

Large government technical development projects self-destruct almost every day despite intense political scrutiny and detailed audit.

But these failures need not continue.

Agile Project Management for Government demonstrates how agile leadership reduces the risks of failure in Government. Essential reading for leaders in government and those with government clients.

- 5 major case studies of agile government success in the US and UK and around the world
- The 9 Agile Leadership Behaviors for success
- The 6 barriers to agile government and how to overcome them
- How to successfully lead your projects using DSDM, Scrum and XP

Essential reading for leaders in government.

"A forceful, evidence-based argument for the role of agile within government."
Andrew Bragg, CEO, Association for Project Management (APM)

"An enlightening insight into grand failures and successes in government projects."
Neil Coutts, Director, Project Management Institute (PMI)

"A unique, insightful and readable guide to agile government from a leadership perspective"
Dot Tudor - Winner of Best Agile Coach Award 2011

Maitland & Strong

#APMFG

www.maitland-and-strong.com

ISBN 978-0-9572234-0-0

9 780957 223400 >