Reflections on PRINCE2 from an Agile perspective

(c) Allan Kelly – www.allankelly.net

Prologue

Having completed the first part of PRINCE2 training course I am now aPRINCE2 certified. practitioner. However, I do not expect to ever use PRINCE2 in anger. For me this is largely an academic exercise. I am an advocate of Agile development methods. In this paper I hope to show where Agile and PRINCE2 differ, and where they match. I will also give a brief outline of the method for completeness, and to enlighten those who have not encourntered it before.

Overall PRINCE2 offers a solid framework for managing projects. However this framework is based on a set of assumptions – explicit or tacit – many of which are not held by Agile project management methods. This means that while some parts of PRINCE2 may be useful, in the Agile world things must be done differently.

Introduction

PRINCE2 is a formalised project management supported by the British Government. In adheres to similar principles and practices as those promoted by the Project Management Institute in America. Projects are defined as:

> "A management environment that is created for the purpose of delivering one or more business products according to a specified Business Case." (Commerce 2005, p.7)

Or:

> "A temporary organisation that is needed to produce a unique and predefined outcome or result at a prespecified time using predetermined resources." (Commerce 2005, p.7)

While supporters of the method claim it can be used for any sort of project its IT are clearly visible. This article will confine itself to PRINCE2 in an IT setting.

The method is designed to be tailored to the specific needs of organizations and projects where it is applied. For those who must show PRINCE2 compliance but wish to take an Agile approach to software development there are several it is possible options for reconciling PRINCE2 and Agile methods, however, none are perfect.

In this article we will briefly review the history of PRINCE2, present an overview of the method, critique the method and then examine how we might reconcile the Agile approach to project management with the more formal PRINCE2 method.

History and usage

PRINCE2 is a project management methodology owned, and originally created for, the UK Government. Today the method is managed out of the UK Office of Government Commerce which regards the method as a commercial product.

In its current, version two, form the method has existed since 1996. PRINCE (version one) dates from 1989 and was originally a project management technique aimed

specifically at IT projects. PRINCE itself developed from a method known as PROMPT dating from 1975. (PROMPT also had a mid-life revision to PROMPT II).

Although anyone can buy the PRINCE2 books and start using the techniques few do. The PRINCE2 manual (Commerce 2005) is both an instruction book on how to use PRINCE2 and the official standard. Anyone who has ever read an official standard will know they are not the easiest thing to read or learn from. As a result the manual suffers from trying to be all things to all people.

Most people take one of the many formal training courses and pass certification exams at the end of the course. The APM Group is responsible for administering these exams and marketing the method. There are two levels of exams; the first *Foundation* exams simply test knowledge of the method while the second, *Practitioner* exams, test use of the method. Both exams are multiple choice based (i.e. answer A, B, C, etc.) although the second uses some surprising complex questions. (The second exam used to be an essay based question but this has been replaced.)

As one might expect of a method originating in the public sector it is widely used in Government projects. However even in the UK the method is only advised and not mandated so even Government projects may not use it in full.

Increasingly the private sector has taken up the method, or at least, the private sector appears to have taken up the method. Undoubtedly some private sector organizations are using PRINCE2 however, if anecdotal evidence is true, very few organizations are using in the full most formal sense.

It appears that private sector organizations are using PRINCE2 qualifications as a filter for staff involved with project management. Selecting a project manager is a difficult business, possession of a PRINCE2 qualification is seen to imply that the individual knows about project management and has a professional approach.

Undoubtedly someone holding a PRINCE2 qualification knows something about project management. At a minimum they know enough about how PRINCE2 suggests you run a project to pass an exam. However the qualification does little imply that the holder can actually manage a project. Like all qualification they prove you can pass the qualification, not that you can put that knowledge to practical use. Passing a PRINCE2 exam, and obtaining the qualification has more to do with knowledge of PRINCE2 than with project management.

Training and certifying people in PRINCE2 is now a big business. Individuals have learned that obtaining this qualification enhances their employment prospects so they are prepared to pay upwards of £1,500 for five days training and two exam passes. However, this also means individuals are focused on passing the exam rather than necessarily improving their project management skills. Consequently schools may focus on teaching to the exam to ensure people pass rather than training people to be better project managers.

It would be unfair to dismiss the PRINCE2 exams lightly. Most people who take the course and pass the exams will certainly be better project managers for doing so. You can't sit through five days of training and two exams and not learn something. Much of the advice is good and the method helps focus analytical thinking.

If PRINCE2 was the last word on project management this would not be a problem, passing the exam and project management would be the same thing. However there is

a limit to what one can learn in five days and the PRINCE2 authors intentionally limit the scope of the method. So, while a PRINCE2 qualification may be necessary it is not sufficient.

More troubling the method has some limitations which may make it unsuitable in every situation. Applying the method, or even parts of the method, when it is not applicable may do more harm than good. For example, as argued below, PRINCE2 is highly risk averse – probably stemming from its public sector roots. The approach is foreign to many private sector environments.

Economists tell us: profit is the reward for risk. Rather than avoiding risk many companies wish to take on risk. In such an environment adopting PRINCE2 could be fatal. Employing those who use PRINCE2 thinking may limit the risk a company embraces and thus reduce the profits.

Overview of PRINCE2

The eight processes, eight components and three techniques and how they fit together is the basis of the PRINCE2 Foundation certificate. These 21 elements are not as intimidating as they might seem at first. On the one hand they provide a comprehensive, breakdown of everything you need to think about to manage a project. On the other hand they say very little about what you actually do to get a project delivered on schedule, on budget and with the features requested.

Eight components

The eight components are basically things you need to use and manage in order to deliver the project. Most of these will be familiar to any IT practitioners. They are:

- **Business Case**: there is no point in undertaking any project unless someone wants it. Who ever wants it should be able to demonstrate that the benefits of the project make it worthwhile. Ideally a business case should be written by the business, unfortunately all too often it is left to people in the IT department who do not necessarily understand what the business needs. In PRINCE2 the business case is both a reference document to be consulted when deciding what to build and how to build it; and it is a changing document that is updated as the project progresses.
- **Organization**: every project, every company, every Government department needs some kind of organization. Even if this is a flexible organization or self-organizing team there is an organization of a type. So it is reasonable to consider the project organization

  PRINCE2 has some things to say about the organization of projects. All projects have four levels of management – shown in Figure 1. The top level is external to the project, this is corporate management or a programme management board. They set the wheels in motion to create the project and may stop it but they have little to do with how the project is managed or run. In effect they are God, omni-present but also absent.

  The second laver is the Project Board led by the Project Executive. The Executive is ultimately responsible for everything on the project, they may the key decisions and authorisations. The Executive is assisted by a Senior Supplier and a Senior User to create a Project Board of three individuals. This enshrines other

organizational assumption PRINCE2: that all projects use the customer-supplier model.

Many projects are indeed organised along customer-supplier lines, for example, in the UK the Government has engaged to develop and manage income tax systems. The Government is the customer, EDS are the supplier. In the private sector too many development effort are organized this way. PRINCE2 further assume that even when the supplier is not an external body the development process will still be managed as customer-supplier model. So, if Microsoft were to adopt PRINCE2 the development groups would be suppliers to the sales and marketing departments.

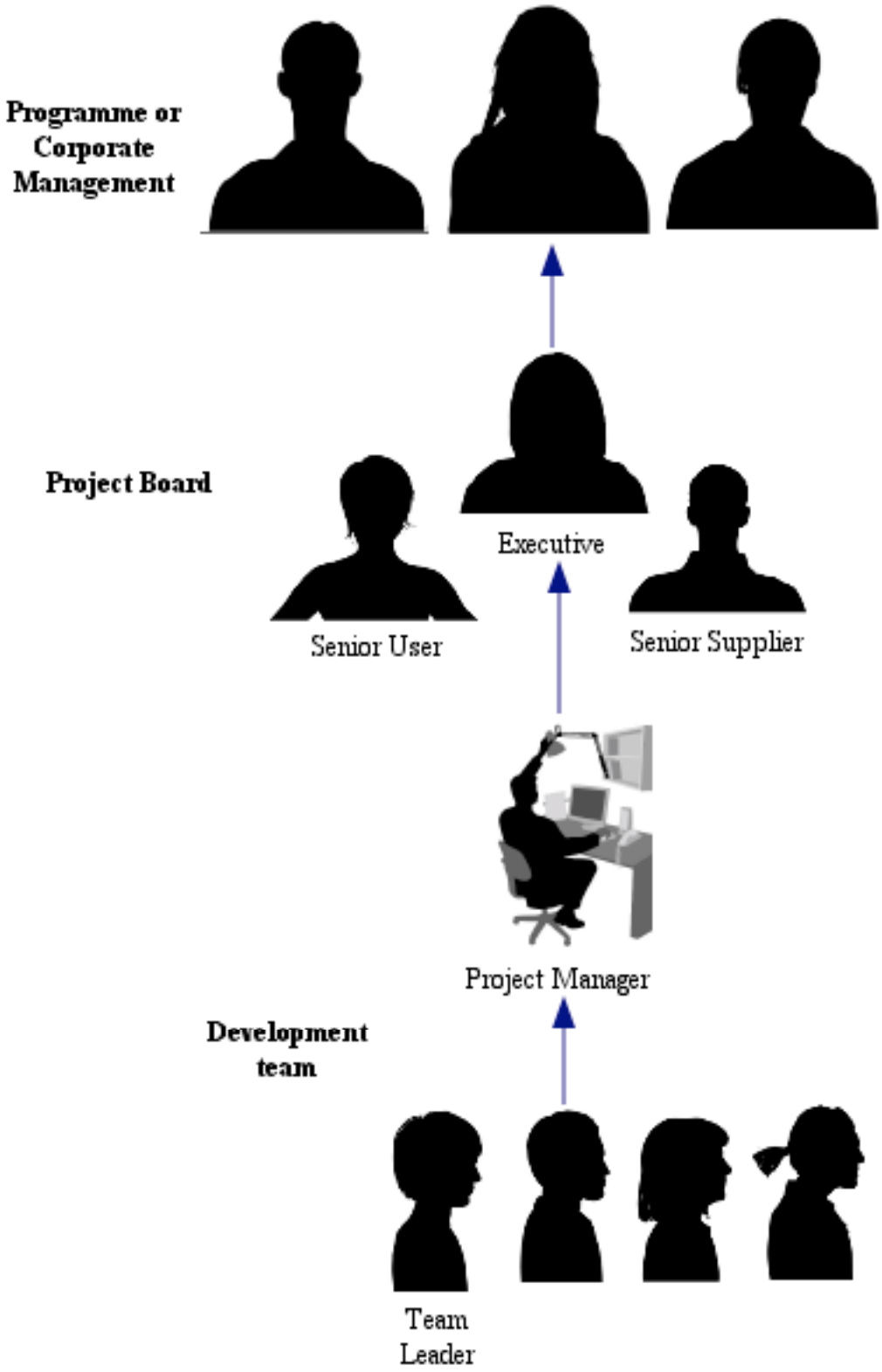

Figure 1 - Basic organizational structure

Next in the hierarchy comes the Project Manager and below them the Team Leader or Team Leader who actually direct the people doing the work. In some cases the Project Manager may double up as the Team Manager. This would normally happen on small projects where the Project Manager has the specialist (technical) knowledge needed to direct the team.

There may be additional people and roles added to this model to support these layers. The Project Manager may be assisted by *Project Support* and the Board my delegate aspects of their work to *Project Assurance*.

Finally, PRINCE2 assumes that all project organizations are temporary structures. This fits with the definition of a project as having specific start and end dates.

Configuration Management: Configuration Management is concerned with the versioning of documents and products so we can uniquely identify items, trace backwards for audit and roll back changes.

Most people in software development will have used a configuration management system such as SCCS, ClearCase, CVS, Subversion, PerForce, etc. If you have then you will know what is involved in using such a system and the benefits. You probably also understand the benefits of versioning documents which would otherwise not be in source code control.

Basically the PRINCE2 configuration management component and process steps describe a manual source control process. This is necessary both to justify the use of a configuration management system and to explain to people unfamiliar with systems what is going on.

These first three components are fairly easy to grasp an intuitive. The next five get more complicated:

- **Change Control**: PRINCE2 has both a *Change Control component* and a *Change Control Technique*, more on the latter below. Change Control, unlike Configuration Management, deals with requests for change and authorisation to make changes.

  PRINCE2 assumes that the starting point of a project – the business case, the requirements, etc. – are well defined. It also accepts that things will change, requirements are not fixed, new ones will emerge and existing ones disappear. Such changes can, and will, have an impact on what work is needed, when the project is delivered, how much it will cost and how well it meets customer needs.

  Key to the method is ensuring that all these changes are managed. Any (intentional) deviation from requirements and project goals needs to pass through the Change Control mechanism. No work should be undertaken on a change unless it has been properly authorised.

- **Risk Management**: PRINCE2 defined risk as 'uncertainty about the outcome [of the project]' and points out that this can be positive as well as negative. The important point is that a risk might happen, or it might not, and it has certainly not happened yet. As you might expect the Project Manager is responsible for identifying risks, gathering information and analysing the risks, however it is the Project Executive who is ultimately responsible for risk.

  PRINCE2 lays down, in detail, how Project Manager are supposed to identify and evaluate risks, and how they may draw up counter measures. Within certain tolerances Project Manager may act on their risk assessments, beyond these tolerances they must refer matters upwards to the Project Board.

- **Quality**: PRINCE2 takes a very broad view of quality, it is not just the quality of a given feature or aspect of a product, but it is the ability of the entire product to meet customer needs. As such many aspects of product specification and capabilities are encompassed by *quality*.

  Again quality is the responsibility of the Project Board, specifically the Senior User who, as the representative of those who will use the product, has a special interest in the output. Actually checking on quality may be delegated to one or more Quality Assurance staff who are independent of both the developers and Project Manager. However, the Project Manager is still tasked with managing quality issues.

- **Controls**: Running throughout PRINCE2 is the concept of 'management by exception'. This is the idea that when planning a project tolerances can be assigned to activities, while the activity stays within tolerance the Project Manager (or Team Leader) is free to arrange the work. Once activities break tolerances problems must be passed up the hierarchy for decisions and potential actions.

  In order to ensure we are within tolerances the Project Manager is tasked with regularly assessing the state of the project and deviation from plan. Therefore, much of the project managers work becomes monitoring, assessing and updating plans to reflect changes. The irony here is that there is very little the Project Manager can actually control, beyond updating plans, reporting and escalating there is very little they can actually do to change things. Work is conducted by one or more Work Teams while major decisions are made by the Project Board.

- **Plans**: Plans run throughout PRINCE2: Project Plan, Project Quality Plan, Team Plans, Configuration Management Plan, Communication Plans, Stage Plans and Exception Plans. One of the three Techniques (below) is, as one might expect, *Planning*. Much of the Project Managers work is the creation of plans, monitoring of progress against plan.

  Plans are considered to be 'the backbone of the management information system required for any project.' Without plans management by exception could not operate because exceptions are raised when work differs from plans.

The eight components are nothing out of the ordinary really, anyone who has worked on a software project will immediately see where they fit in. They are in effect a breakdown the aspects of a project.

Three Techniques

PRINCE2 suggests three techniques which may be used at various times during a project. These are: Product Based Planning, Change Control and Quality Review.

The **Quality Review technique** will be immediately familiar to anyone who has read about or conducted a formal code review. The creator of a document (or similar artefact) distributes copies to nominated reviewers. A review meeting is then held where the reviewers pass comment on the document and a scribe notes down the changes suggested. The meeting is chaired by another individual who is neither a reviewer, producer or scribe.

All documents are supposed to comply with a description written in advance. This descriptions specifies the content and form of the document to be produced. For

regularly reoccurring documents, such as reports and requirements, this should not be a problem – especially if the company concerned already has templates. However where templates do not exist or documents are ad hoc this adds an additional step to be performed before the document is creationed.

The **Change Control technique** will also be quite familiar to anyone who has worked on an IT project. Suggestions for change start life as Project Issue. When reviewed issues may be the basis for Risks, Exceptions, off-specification report, clarification or the creation of a Change Request.

The Project Board is normally responsible for evaluating Change Requests and approving, or not, the change. However the board can delegate Change Authority to nominated individuals (e.g. the Project Manager) or a board comprised of suitable people.

The **Product Based Planning** technique is exactly what the name suggests and suggests planners first identify the end products of a project then refine this list, identifying interim products and dependencies. This has much in common with the Plan Backwards pattern (Prince and Schneider 2004).

In addition the technique recommends a basic diagramming notation to use when drawing the breakdown. As products are identified it is expected that a description will be written for each one. Later the breakdown diagram is transformed into a Product Flow Diagram which becomes the basis for planning charts.

Interestingly PRINCE2 does not contain a planning technique or mandate use of GANTT, PERT, Critical Path or any other planning mechanism. However it does contain a Planning process. The process describes what needs to be done and what output is expected but it does not describe how to do this. Some authors see this as an omission and added it their descriptions of PRINCE2 (Hedeman et al. 2005).

Eight Processes

The eight processes until are the most involved and complicated part of PRINCE2, each is broken down into a number of sub-processes, in total there are 44 sub-process. Each process is assigned a two letter code – like SU or IP – and each sub-process is assigned the two letter code of the main process and a numeral, for example, SU1 and IP2.

Some of these sub-processes are little more than single decisions or points at which to perform a particular action. In some cases the decision is already constrained to a set of pre-determined options. Rather than present processes as multiple sub-processes the method may have been better off presenting simple check-list for project managers to review and action. While some of the sub-processes are involved many of them are quite simple, the actual action or decision is buried beneath the weight of sub-process description.

Detailing each of the 52 processes and sub-process in turn is beyond the scope of this review. Figure 2 gives a high level over view of the major processes and actions. For simplicity it is easier to say each project has a beginning, a middle and an end. In PRINCE2 terminology these correspond to Controlled Start, Controlled Progress and Controlled Close respectively.

For the sake of brevity this description will consider the most common – or most expected – process flow through a PRINCE2 project. Aficionados of PRINCE2 will

notice many omissions and abbreviations. Anyone wishing for more detail on PRINCE2 is referred to the official manuals or one of the many books on the subject.

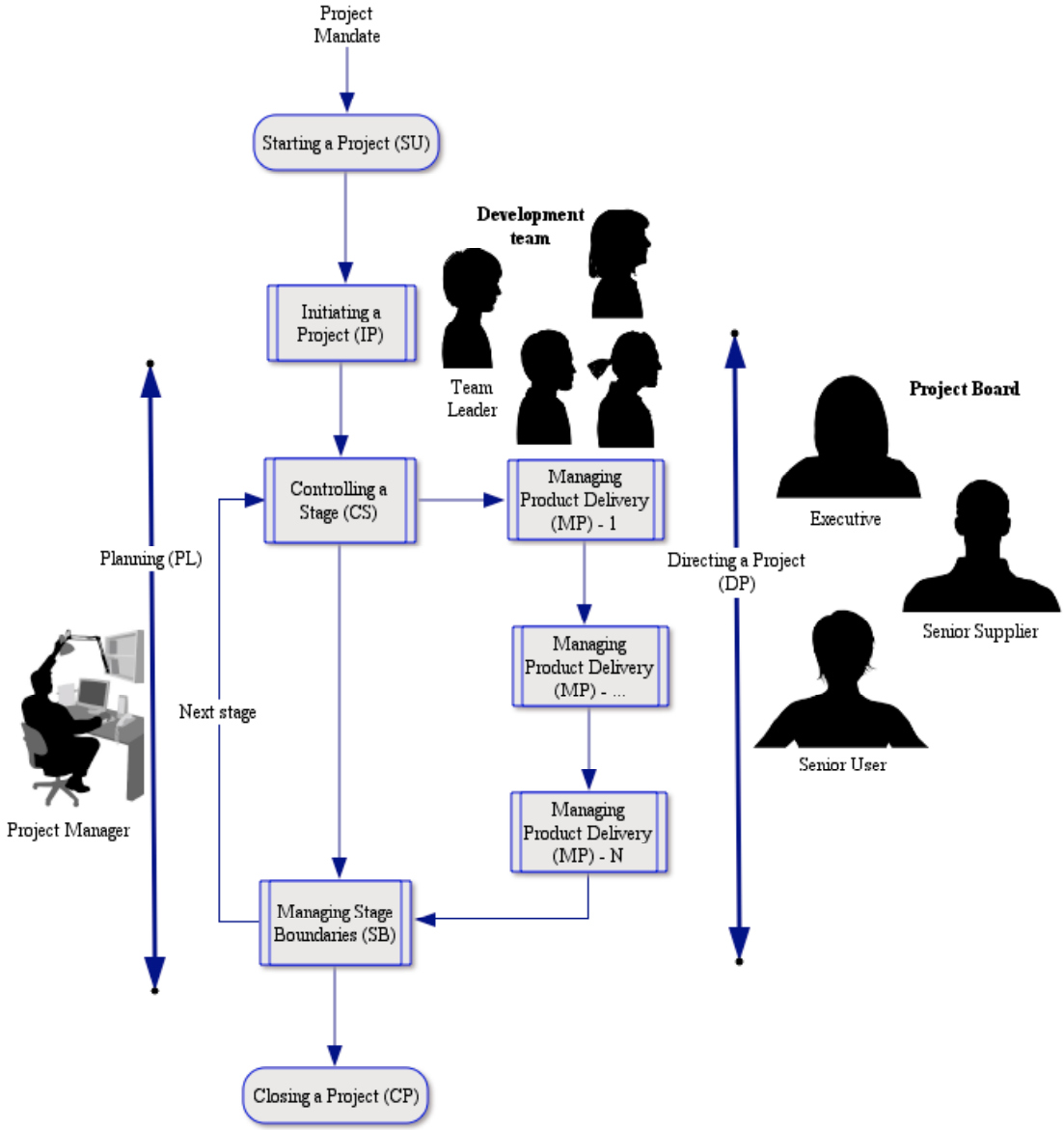

Figure 2 - Overview of PRINCE2 process flow

Before any project happens there will be some activity and someone, or some group will decide that there should be a project. PRINCE2 is right to leave these details vague. The start of a PRINCE2 process occurs when someone in authority issues a Project Mandate, this is the trigger for the first process. The mandate itself could be anything, a verbal request, a formal document or a chat in a pub. What ever it is the mandate is enough to set the wheels in motion.

Still, there is not enough information to start a project. In order to start a project there needs to be a few people and some basic agreement on what is to be done and how. PRINCE2 calls this pre-project stage *Starting a Project* or SU for short. In this stage the key people are appointed – at least a Project Manager and an Executive – and some basic details are decided. These are documented in a Project Brief – not yet a

business case but a basic outline. Some other decisions may also be made, like whether to buy an off the shelf product, outsource development or do it in house.

Importantly the first planning exercise takes place. The Project Manager creates a plan for what will happen next. The next phase is called *Initiating a Project* (IP) and it is this stage that formally begins the project. In this stage the formal controls come into play as the project team engage in more planning, formalise the business case and start to build a wider team.

Taken together Starting a Project and Initiating a Project form the Controlled Start to a project. These two process get everything planned and set up for the hard work to start.

Most of the actual work in a PRINCE2 project is grouped under the Controlled Progress banner and contains multiple repetitions of the same three processes: *Controlling a Stage* (CS), *Managing Product Delivery* (MP) and *Managing Stage Boundaries* (SB). In fact, most of the actual work in terms of coding and product creation happens inside the Managing Product Delivery.

Controlling a Stage is the process by which the Project Manager manages the project day to day. It deals with authorising work, project issues, reporting, exceptions and other routine stuff.

As mention the actual work happens in Managing Product Delivery. This stage is concerned with the technical work of creating products and is normally managed by a Team Leader. On some occasions the leader may also be the Project Manager and on other occasions one Project Manager may have several Team Leaders reporting to them.

There is no limit on how long a Managing Product Delivery process is. It could be two weeks, two months or two years. A short project might only have one two week process while a long project could have one long two year process. More likely a long project would break the delivery process down into several repeated process so there might be four processes each of six months, or maybe 24 process of one month each.

While the product delivery process is started as a result of Controlling a Stage it is closed by a Managing Stage Boundaries process. Again there is no clear definition of how long a stage is. The stage may contain one or more process instances. So, a project might be arranged as three development stages, each containing three delivery processes, each lasting one month; this would make the project nine months long.

At the end of a stage the Project Manager reviews the work done by the team and updates his plans and reports, raises any issues, risks or exceptions. The Team Leader would be expected to help the Project Manager.

And so the project continues in its steady state until one day time comes to close the project. Project closure may come about for one of a number of reasons: perhaps the work is complete, perhaps the business case is no longer valid or perhaps something goes wrong and the project must be closed.

At this point a process called *Closing a Project* (CP) is used as part of a Controlled Close. This process ties up the loose ends, terminates the development team, completes documentation, files is and writes some end of project reports.

There are two other process also in use which occur on an ad hoc, as needed, basis. The first of these is planning. As already mentioned PRINCE2 is plan centric so periodically the Project Manager or Team Leader need to go into planning mode and make use of the *Planning* (PL) process.

The second ad hoc process is undertaken by the Project Board. PRINCE2 assumes that the board are, on the whole, too busy to meet regularly so they are only involved when the Project Manager decides she needs their help. When this happen the Project Board acting within the *Directing a Project* (DP) process. As the name implies they do not *manage* but rather *direct* a project.

There are a number of reasons why this predictable flow might be broken. Changes to the flow are usually the result of Project Issues. This collective title covers a lot ground by the important ones are: Risks, Change Requests, Exceptions and Off-Specifications.

Risks and Change Requests have already been explained when discussing components. The Project Manager is responsible for recording and tracking these issues. On occasions she may need to consult the board about such issues, the board may choose to change the parameters of the project or even close the project entirely. In either case the Project Manager will need to update or re-plan as necessary and then action the plans.

Off-specifications could equally be described as bugs. These are issues which arise because the product that has been developed does not meet the specification in some way. As a result some rework is required which might involve re-planning or changes to the existing plans.

PRINCE2 copes with off-specifications, risks and changes via two methods. Firstly each stage of the project is assigned a tolerance. This is a margin around a plan which the board are happy to accept. Provided the execution stays within this tolerance the project and Project Manager can carry on as before.

The second method is provision of a change budget. This is a sum of money set aside to pay for changes. A Project Manager may need to consult the board before they can access these monies or the board may delegate control to a Change Authority – which could be the Project Manager herself or a board which includes the Project Manager.

However, there are occasions when a project moves out of tolerance or when there is no money in the change budget to pay for changes. On these occasions the Project Manager must raise an Exception Report. This report details the problem and suggests options for action, it is passed to the board who can then decide on the appropriate course of action.

Adding all the additional sub-processes and possible control paths to Figure 2 would not only complicate it immensely but would leave it looking like the worst possible spaghetti code. Such a diagram would only go to show how unlike programming project management is. There are many alternative routes through a project and project management is still a go-to based activity.

Other elements

There are a few other elements of PRINCE2 which it is worth mentioned although they do not fall into the processes, components or techniques classification.

Readers may have noticed the lack of meetings. PRINCE2 is not a meetingless method but it is devoid of any regular or scheduled meetings. This is because the method is driven on a 'management by exception' basis rather than a schedule. On the whole any process, stage or meeting in PRINCE2 can be as long, short, regular or irregular as the team decides as long as management is event not time driven.

Secondly PRINCE2 projects maintain a lot of process documentation. The multitude of plans have already been mentioned, as have the business case and project brief. Contained in these documents, or extended by them are also the Project Approach (specifying how the project will be achieved) and the Project Mandate.

Although requirements documents are not mentioned the quality criteria and plan cover most of this information – indeed the requirements would probably form part of the quality criteria.

A number of reports are expect to be written during the PRINCE2 processes. Team Leaders provide the Project Manager with a Checkpoint report, while Project Managers provide the Project Board with a Highlight Report and on occasions End Stage and Exception Reports.

In addition PRINCE2 suggests Project Manager maintain an Issues Log for Project Issues, a Risk Log, a Quality Log, Lessons Learned Log and a Daily Log for anything that might otherwise be missed.

Helpfully PRINCE2 also suggests a filing system for maintaining these plans, reports and logs. Reports, plans and logs should be filed in one of the Project File, Stage File or Quality File.

A distinction is drawn between those products, such as the Project Plan, which are created to manage the project and those which are created as part of the customer delivery. The former are considered Management Products and the latter Specialised Products.

As mentioned above PRINCE2 assigns tolerance to many aspect of the plans. Such tolerances allow deviation from the plan without triggering an Exception Report or the involvement of the Project Board. There are six types of tolerance which may be varied:

- Time – variation in the date on which a product may be delivery, or the time it takes to build a product.
- Money – variation in the amount spent on some piece of work.
- Quality – boundaries of acceptable quality of a delivered product.
- Scope – additional, or reduced product scope.
- Benefits – variation in the benefits delivered by a product while delivering the business case.
- Risk – amount of risk that may be incurred during some section of the project.

Normally tolerances are set in terms of time and money. These are comparatively easy to quantify, some of the other tolerance criteria, for example quality, may be difficult to decide in advance.

Finally, it is accepted that PRINCE2 will need to be tailored to work in many environments. Some guidance is given on where and how users may tailor the method but on the whole this is left to users.

Critique

The purpose of critiquing PRINCE2 is two fold: firstly to help readers understand when and where PRINCE2 is applicable and help them decide whether their organizations will benefit from its introduction. Secondly, only by understanding when PRINCE2 is not applicable can we look for, and possibly create, an alternative.

The name PRINCE2 is comes from 'PRojects IN Controlled Environments (version 2).' Even here thinking diverges from the Agile approach.

Projects – or Products and Programmes?

PRINCE2 helpfully provides two definitions of a project. The first could be used to describe almost any business environment:

> "A management environment that is created for the purpose of delivering one or more business products according to a specified Business Case." (Commerce 2005, p.7)

In any business one would hope to find an environment created by the management, which delivers product(s), and does so for the greater good of the business. For our purposes the second definition is more enlightening:

> "A temporary organization that is needed to produce a unique and predefined outcome or result at a prespecified time using predetermined resources.' (Commerce 2005)

Many software development efforts are indeed projects. A team is set up, they create a software product and the team is disbanded. However only in a small number of cases does software development cease entirely. One of the myths of software development is that is comes to an end. Often work continues, maybe at a reduced level with fewer resources, but software needs continual modification.

These modification are euphemistically called 'maintenance' but it has long been known most of the effort expended on a software product occurs during this phase of its life. As a rough guide we may consider 20% of the effort to be during initial creation and 80% to be during the life-time. Consequently the organization needed to support a software product is not temporary.

Part of the reason for this imbalance is that many of the demands on a software product only come to light when the product is shown to potential users, and when the software is actually in use. Of these demands corrective fixes are the most obvious but they are far from being the only changes needed – see (Kelly 2008) for more on this subject. Indeed, it can be argued that changes that occur later in a products life are more valuable.

Further, once created and in use the software product becomes part of the infrastructure of an organization. It is expensive to create (and maintain) and changes the organization, sometimes for the better, sometimes for the worse.

The difficulties in recognising forecast benefits is part of the reason software development efforts are so often late and over budget. Another factor is the way in which demands (requirements) on a project change simply because creating the product allows learning to occur which changes what was ask for.

The changing nature of the software, the demand to meet deadlines, demands to meet budget, changes to technology and unforeseen problems all mean that the resources used during the course of a software development effort can fluctuate widely. In fact, the resources assigned before work begins has a significant effect on the final outcome (Conway 1968).

For example, we may start an effort by hiring a development team: a project manager, a business analyst, a database administrator, a couple of programmers and a tester. This pre-supposed that we will build a piece of software ourselves, and that it use a database. This need not be the case. If we find a month in that we can buy a ready made product and configure it we could reconfigure the team but we are just as likely to continue with our original plan. Indeed, we may even fail to notice that such a product exists.

Putting these arguments together we see that:

- Software development organizations are far from temporary
- Outcomes are hard to foresee and likely to change over time
- Delivery seldom occurs at a pre-determined time
- Resources are not pre-determined and if they are then our options are likely to be limited

In other words, software development does not fit the definition of a 'project' as used by PRINCE2.

The creation, and life-time management, of a software product may well be better considered as a Product or a Programme (we use the English spelling here to avoid ambiguity.) Products have a different life-cycle to a Project, we continue to develop, manufacture and sell them while they can demonstrate a reasonable return on investment. Programmes share many characteristics of Projects but they have no end date, they too continue while they deliver value.

We tend to assume software development efforts are projects for two reasons. Firstly, that is how the subject is taught and the literature written but the reality is different. Secondly, we lack good alternative models of how to do it so we fall back on the project model.

Many project management techniques are useful when managing a Product or Programme but the assumption of an end-date introduces a short-term view which is damaging. Of course there is a need to balance a short-term and long-term view but this is a tension that needs managing itself, not assuming away.

Controlled Environments?

There is an ambiguity in PRINCE2's use of the term 'Controlled Environment'. Does it imply that the project happens in a pre-existing *controlled environment*? Or that the use of PRINCE2 will create a *controlled environment*?

If we assume that the project will happen in a controlled environment then we already have an advantage. Any project (or product or programme) conducted in a controlled environment has a significantly higher chance of success than one conducted in, shall we say, *ambiguous environment*. Therefore the value of any project management technique is lower.

So, it seems the second description is more suitable: that by the use of PRINCE2 we will create a controlled environment. This itself is quiet a bold claim – and may be why PRINCE2 shy away from making the claim directly. If PRINCE2 really can create a controlled environment then it is certainly worthwhile.

This leads to the question: can PRINCE2 create a controlled environment? The eight components, eight processes, umpteen sub-processes and three techniques described in the manual effectively model a controlled environment. Still, we need to move from our *ambiguous environment* which lacks these elements to the *controlled environment* with these elements.

However such a change is non-trivial. Deploying the whole methodology is no small undertaking itself and one that is not dealt with by the literature. If we can create this environment then we will have our controlled environment. In other words, the PRINCE2 methodology is the controlled environment.

The problem here is that few environments resemble the PRINCE2 environments. Most projects are conducted in ambiguous environments: goals are not clear, key decision makers are absent, staffing changes, technology advances, forecast benefits changes, costs increase, and so on. In particular we may find that the leaders at different levels disagree on what the project is try to achieve, how it is to do this and even what state it is in.

Thus we have a paradox, PRINCE2 describes a controlled environment but says nothing about how to bring it about. If you start in a controlled environment you don't need PRINCE2, but if you need it then you need to step outside the method.

The controlled environment of PRINCE2 will certainly help successful project delivery. However, the problem most software development efforts face is not a controlled environment but an ambiguous environment.

PRINCE2 therefore serves two purposes. Firstly it is a reference model from which we van measure deviation and borrow ideas for practice. Second, for those projects that do exist in a controlled environment it is a means to keep the environment controlled.

More is less

PRINCE2 started life as PROMPT, a method devised by Simpact Systems Ltd. around 1975. At the time it was used for Government information systems. Over time the method has been generalised to take in non-IT projects and to cover non-Government work.

As with any generalisation this has diluted the focus of the method because it needs to take in a wider audience. Similarly the generalisation has had the effect of enlarging the method description. Concepts common in the IT arena (e.g. configuration management) need to be incorporated and described in detail. This in turn adds to the amount one must learn before using the method.

In other words, *more is less*. As PROMPT became PRINCE2 and incorporated more applications its size grew but its focus was diluted.

Perhaps making matters worse PRINCE2 is now commercial product completing with others like the Project Management Institute's (PMI) Project Management Body of

Knowledge (PMBok). Both methods need to increase their market which leads to further dilution of focus.

Someone looking to adopt a project management method is now faced with choosing between two all-purpose, market endorsed, methods. However a better, more suitable option, might simply be to adopt a niche specific project management technique. Such a technique would be more focused, easier to learn and require less (if any) tailoring. However, finding practioners experienced in using the technique may be more difficult. Organizations adopting global standards, and aiming to swap people between roles might make this more difficult still.

Assumptions

The foundation of PRINCE2 contains a number of assumptions which while valid in some context are not always true in others. If we use lean manufacturing (Womack et al. 1991) and lean product development (Kennedy 2003) as our context then some of these assumptions are troubling. Other research and studies since PROMPT in 1975, and PRINCE2 in 1989 also raises questions about some of these assumptions.

Customer/Supplier relationship: This assumption is deep rooted in PRINCE2. Even when the company developing the software is also the eventual user and beneficiary PRINCE2 assumes the relationship can be managed as a supplier-customer one. At first sight this seems like a minor assumption but it is actually major. Making this assumption is to creates two groups that have different objectives, thus the assumption and drives a wedge between the two groups thereby creating an adversarial relationship.

The need for the customer to police the supplier appears at other times too. Audits are considered a common occurrence, and quality assurance may be tasked with assuring the product delivered by the external supplier and policing the relationship.

However, when those developing the software and those taking delivery all work for the same organization they share objectives. The need for one part of an organization to police another part of the same organization is a symptom of larger problems and a sign that goals are not aligned. Such policing costs both time and money.

People are not addressed: PRINCE2 concerns itself purely with the technical aspects of project management. It deliberately has nothing to say about people issues or how to manage people. It would be difficult to add a method of people management to PRINCE2 or any other methodology but by passing the issue a major component of real-life management is ignored.

Authority: There are four documented layers of management in a PRINCE2 project and the method assumes there is a direct reporting line from one to another. PRINCE2 relies on direct report and authority to manage the project. Yet in many organizations reporting lines are not clear, and in other organization the exercise of authority is not part of the culture. Truly temporary project often run on a form of matrix management. Individuals have two bosses to report to, answer to and take instructions from.

It is well documented that knowledge workers – a group which definitely includes software developers – do not respond well to authority (Davenport 2005). So although individuals may be able to point to an organisation chart, and their position on it such a chart, and their position, do not imply military style command and

control. In any organization where authority, and management by decree, is not the accepted way of working PRINCE2 is going to struggle.

PRINCE2 also assumes the **Project Manager is part of the customer organization**. The supplier may also appoint a one but the customer Project Manager is the primary one. Not only does this confuse reporting lines but it also assumes that the customer organization has plenty of Project Managers to spare, and that they are experienced in managing the domain in question – in our case software development.

Overhead

Applying PRINCE2 is not free, while the PRINCE2 manual talks a lot about managing project costs it is silent on how much a PRINCE2 itself might cost to implement. A project following PRINCE2 in its entirety is going to run up a large bill in terms of training, document creation, document management, meetings and following the process. Although advocates of the method are keen to point out it can be used with small projects the method costs are likely to overwhelm most.

The cost of using PRINCE2 is non-trivial and before it is used – especially in full – there needs to be some cost/benefit analysis performed. This analysis needs to consider the downside of such a structured project management technique as well as the actual costs of running the project.

In truth all projects (and products and programmes) need managing. Unfortunately management tends to take a back seat, senior staff are often much happier to pay for programmers, or even testers, than they are for product and project managers. They are right to be sceptical, such staff can add significantly to costs. But when these staff are correctly deployed, ensuring the right thing is to be built and that it is built right then they more than pay for themselves. So comparing the cost of PRINCE2 with the cost of not-managing a project is unfair, rather we need to compare it with lighter weight methods.

While most projects probably do not need the whole of the PRINCE2 method. But deciding what is needed, and what can be removed, is the work of a skilled practitioner. There can be a tendency to use a method in its entirety rather than tailor it.

Money is not the only cost to consider either. Many of the PRINCE2 practices will, in practice, prove time intensive. For example, it is not sufficient that a document be created, all documents must be accompanied by a description to specify their structure and content. True one could skip this step – through tailoring – or on a large project a template may exist but it is still a necessity.

Documents, or other products, are also subject to review. PRINCE2 adopts the traditional code-review style review process almost verbatim. Reviews are meetings, products are reviewed by one or more reviewers, and there is a chair person and scribe to take notes. Once reviewed products are changed an possibly reviewed again.

Such a review process is time consuming for all participants and slows down the creation of documents. Research shows that the review process has little effect on the outcome and that more informal reviews can have similar benefits. (Votta 1993; Votta and Porter 1997). In short, the Pareto principle seems to be at work, 80% of the review benefit can be obtained form 20% of the effort.

Exception handling: PRINCE2 includes an extensive exception handling mechanism (not completely unlike that found in Java or C++) for taking action when, well, exceptions happen. Like in programming languages this involves "throwing" an exception and having it "caught" and acted on. As in programming languages this is an expensive operation. Reports must be written, decisions taken and plans produced.

When conditions truly are *exceptional* this might be a price worth paying. However, in environments where there is a lot of change, risk or uncertainty the exception process could be involved regularly. In such an environment the cost of simply managing exceptions could be high.

Management by response to exceptions, so called **Management By Exception**, is cornerstone of PRINCE2 thinking. This stems from the belief that senior managers are short of time and therefore should not be "bothered" with anything other than the exceptional. Managing the project day to day can be delegated to more junior managers, like the Project Manager. However, this set up can itself create uncertainty because there is no rhythm to the project. Actions only happen in response to events so work and management continues in the default.

A potentially downside of management by exception is that managers only get to hear about problems, not successes. While there are regular reports our busy managers are less likely to read routine reports, or absorb the details, than they are an exception report. An exception report, by definition requires their action so they pay attention.

Some years ago London Underground realised that passengers perceived the tube service to be poor because the only announcements they heard where when something was wrong. Now travellers are routinely told "There is a good service on all lines." They are reminded that, most of the time things work and work well.

Of course the opposite may also hold. Managers who receive no exception reports may believe that everything is on track. Little by little, problems may be arising which do not warrant an exception report but still endanger the ability of the project to deliver.

Risk aversion over risk embracing: many of the mechanisms in PRINCE2 are designed to reduce and manage risk. Taken together this produces a methodology that is highly risk averse. Overall PRINCE2 attempts to cope with risk by reducing it and avoiding it, however there is little recognition of the cost of these actions or the cost of avoiding risks.

If, as economists tell us, profit is the return for risk, PRINCE2 runs the risk itself of removing all profit by the excessive management and reduction of risk. There are some risks we don't want but removing risks costs. Such aversion to risk can probably be traced back to the methods roots in Government projects. Even here price needs to be considered and sometimes it is better to take the risk.

Plans lead to goal displacement: as stated above PRINCE2 is plan heavy. Plans are put in place to assist the delivery of the Business Case – which can itself be seen as a plan. The creation, monitoring and updating of plans is no *free-lunch*, they can be expensive to create, monitor and update. Particularly when there are many interlocked plans this can be a time consuming exercise.

There is a danger than the actual goals of the project – delivering the business case – can be displaced by delivery of a plan. The objective of the project is not plan fulfilment but delivery of the product to meet the business case. However, for

individuals and teams in a stressful and possibly large project, then plan fulfilment can displace the ultimate goal as the focus of their activities. When a project encounters turbulence and is in danger of failing it is quite natural that people focus on doing their small part, they isolate themselves from the bigger problems. If the project should fail they can reasonably argue they fulfilled their plan.

Focusing on plan fulfilment can lead to additional problems. Side effects may appear as plans are met in unexpected ways or by cutting corners. So called *satisficing* may occur when staff meet the plan but go no further, if a team completes a plan in less time than expected they may decide to look busy to use up the remaining time. Alternatively, faced with extra work (e.g. exception reports) for failing to meet a plan they may over estimate the amount of work needed to complete a task.

A second problem with plans is that they cannot keep up with changes and developments, such problems increases the more plans a project has. PRINCE2 uses plans and Change Control to ensure that only authorised work is undertaken and when it is the work occurs in a controlled fashion. When changes are happening rapidly it may not be possible to update, authorise and action plans fast enough. Of course it would be possible to add more project support to help the Project Manager with this task but this again increases cost.

Process compliance as goal displacement: in the same way that plan fulfilment can displace goal fulfilment so can process compliance. Individuals and teams focus on following the process, in this case PRINCE2 rather than achieving the project objective. This is a rational response when people come to believe the project will miss its goal. In such circumstances people anticipate consequences of project failure, in order to avoid blame they seek to show that they did what was required, they followed the process. Succeeding in following the process displaces the actual project goal as the objective.

Summary of Critique

Before deciding to use PRINCE2 you should satisfy yourself that it is applicable to your organization and context. If, in your environment, some of the issues identified above apply then you might be better looking at an alternative management method. However, if one or more of these critiques does not apply then you may well be adviced to use PRINCE2.

So, if you are developing software products in a project based environment, and if your organization needs to move people between different projects – some of which may not be IT related – and if your environment is fairly controlled then PRINCE2 could be the right thing for you.

Risk transmission

Of course we would all like to avoid risk, but taking risks is what brings rewards. Remove the risk and you remove the reward. Yet it is not always obvious how the two go together.

Suppose I have an idea for a piece of software: I lock myself away for six months and emerge with a great product which sells like hot cakes. In fact I have taken a lot of risk: I haven't analysed my market, talked to potentially customers, looked at

competitors, engaged with a user interface designer, etc. etc. Had I done so I would have removed a lot of the risk.

However, to have done so would have cost money. Suppose I borrowed the money to do this, this would have changed the operation risks into a financial risk. ~If the product failed I would have owed a bank money.

Alternatively I could taken investment from venture capital company. In return they would have shared the profit. So while I reduced my risk I also reduced my return.

However, in removing these risks I created another. These thing take time, a month to negotiate the investment, a month to talk to customers, another to survey the market and competitors and a fourth to design the interface. All this costs to I need more venture capital so I have to sell more equity and get to retain less profit.

More importantly, all told my product now reaches the market four months later. That is four months during which time a competitor could enter the market, or the market could change, demand could go elsewhere.

So while we want to avoid risk sometimes we just displace it elsewhere. We remove project risk but increase commercial risk. I am not saying that charging headlong at risk – damn the torpedoes – is the right thing to do but sometimes it over caution can be as dangerous.

Advantages

Before we rush to bury PRINCE2 lets pause for a moment and consider advantages it brings. Despite what I have said above there is a lot that is worthwhile in PRINCE2. After all, PRINCE2 is built on *best practice* so even if the method has flaws there must be some parts that are worthwhile. Neither can one doubt the intentions of PRINCE2 and those who devised it. There is a genuine attempt to create a better way of managing projects.

It is possible to view the PRINCE2 methodology, with its array of processes, sub-processes, components and documents as a dissection of project management activity. In effect, it decomposes the whole activity into a number of discrete steps. This is valuable because it allows analysis of the activity. As other authors (Koskela and Howell 2002) have pointed out, the theoretical foundation of project management is somewhat lacking. Consequently the kind of dissection PRINCE2 provides is valuable in understanding what project management actually *is*.

Having broken the activity down into component parts these are then reassembled into a methodology. So at the very least PRINCE2 helps us better understand what project management is, and provides a better understanding of the Project Managers role.

The start of all projects, and products, tends to be messy. PRINCE2 is right to identify a formal start to a project. The **project mandate** and **project brief** are both useful concepts for getting a project formally started. The **project initiation document** (PID) is another useful idea which all projects would benefit from. However it should be accepted that some projects will start before the PID is complete. The need to overlap phase of the project starts very early. Delaying the start of a project is itself a risk.

Although the management levels have been critiques above the hierarchy also implies **delegation**. The project board is responsible for making the *big decisions* and

delegates day-to-day management of the project to the project manager. This manager in turn is able to delegate delivery to a team manager. Similarly, quality assurance may be delegated. Embracing delegation acknowledges two fundamental ideas. Firstly that the top level people do not have the time to become involved in every decision. Secondly, delegation implies trust. Those delegating must be prepared to trust those receiving the authority to act.

The **project board** itself is useful concept that should be embodied wherever possible. This body brings together the key people – in this case three – who are responsible for project delivery. The board is both responsible for ensuring the project is delivered and benefits recognised. In order to do this the board is empowered to make whatever decisions are necessary.

Several of the PRINCE2 project practices are also worth recommending for any project. While plans as a whole are elevated to a privileged position the **product based planning** approach has much to recommend it. The identification of specific products should be the starting point of any project plan. From here we can plan-backwards to discover the interim deliverables. In fact, this process should probably be extended far beyond the point at which PRINCE2 suggests. Rather than considering the products, the attributes – or features – of those products could be identified and regarded as interim deliverables in their own right.

Although most of PRINCE2 is revolves around plans - their creation, execution and keeping the project close to the plan – the method does recognise that plans are seldom accurate, things change. Only the very near term plans have any chance of being accurate, the longer the elapsed time the more inaccurate a plan will be.

The PRINCE2 solution to this is to regularly revision plans, re-plan and create new plans. An alternative would be to remove many of the plans. Updating plans, re-planning and creating plans is a time consuming exercise. Rather than constantly plan and update plans we may seek to remove some of the plans.

Although PRINCE2 claims to be the sum of best practice in the project management the method accepts that on any project there will be **lessons learned** which could improve the next project. Both the recommendation to keep a **lessons learned log** and produce a lessons learned report at the end of the project should be honoured in all projects. However, as recommended in the method there are two problems.

Firstly the lessons learned report owned and created by the project manager. This could lead to a one sided view of the lessons and project as a whole. There is a need to open this activity up and make it more inclusive. Retrospective techniques which can do this are well known and readily usable (Derby and Larsen 2006; Kerth 2001). Anyone implementing PRINCE2 would be well advised to both respect the lessons learned log and report and supplement it with these techniques.

Secondly PRINCE2 gives little guidance on how we should regard our lessons learned with regard to the method itself. Since PRINCE2, by its own definition, is the collection of *best practice* what are our lessons learned? Can our lessons learned supersede PRINCE2 processes and techniques? And if so do our lessons now represent best practice?

We are faced with the problems of determining who decides what is best practice. The term itself, *best practice*, implies that these practices are the best, the pinnacle of practice. Perhaps rather than regard PRINCE2 practices as the absolute *best* they are

better characterised as *best to date practices.* With this small change we can resolve our dilemma.

PRINCE2 allows for changes resulting from *lessons learned* or elsewhere by accepting that method **tailoring** can, and should, occur. This provision should be regarded as another advantage of the method.

However, this too is something of a double edged sword. If an organization has adopted PRINCE2 as a corporate standard in the hope of allowing people to move between projects then excessive tailoring could defeat this objective. A project that has tailored the method excessively will find it more difficult to absorb people from outside the project. Similarly, a project that has learned lots of lessons, and adapted its process will also find it takes longer to absorb new people.

Again we need to supplement the method to overcome this problem. In a corporate environment, where project lessons are being learned and processes tailored it is important that active communication occurs between all those managing such projects. There is a need to share learning between groups.

PRINCE2 and Agile, together?

At first sight PRINCE2 seems anathema to Agile, the two methods are based on a different view of the world. But, job adverts like these are increasingly common:

> "Web Project Manager - Amersham, Buckinghamshire A PRINCE2, AGILE Web Project Manager is required to join my client at their offices in Amersham, Buckinghamshire. ..."
>
> "Project Manager with PRINCE2 qualification or experience who has worked in an Agile Development Environment overseeing the progress and delivery of software"

To be taken seriously as a Project Manager – at least in the UK - people often need PRINCE2 qualifications. Anecdotally evidence suggests some companies have adopted PRINCE2 as a corporate standard by want their development teams to be Agile.

So how can we reconcile this position?

Fortunately PRINCE2 contains several mechanisms which will allow us to use Agile techniques while being PRINCE2 compliant. Unfortunately the two approaches are ultimately based on different principles. Maintaining PRINCE2 compliance while limit the benefits that Agile teams can achieve.

Plug in Agile

PRINCE2 concerns itself with the management of projects. It has nothing to say about how the work is actually done. Performing the work is actually encapsulated in a process called 'Managing Product Delivery' or 'MP' to use PRINCE's approved abbreviations. Control of the work within the process is the domain of the Team Leader. The leader is responsible for interfacing with the project manager, negotiating work commitments, reporting on progress and managing the development team.

Provided the Team Leader accepts work and reports progress in a PRINCE2 fashion they are free to run the team as they see fit. This means the team is free to work in an

Agile fashion. The team could probably adopt most of the original XP (Beck 2000) without any problems. *Pair programming, testing* (including *test driven development*), *simple design, refactoring, collective ownership, metaphor, continuous integration* and *coding standards* should present no problems.

The team could probably do an approximation of the *planning game* and *small releases* provided it was masked from the PRINCE2 processes. The Team Leader would still need to accept a Work Package from the Project Manager which would form the basis for the planning game. The larger the work package is, and the longer it is expected it to take, the greater the scope for using Agile scheduling techniques.

For example, if the work package is only expected to take two weeks to complete the Team Leaders is largely constrained in their use of Agile scheduling. A work package expected to take one month could be scheduled as four one-week iterations, provided iterations are not rigidly scheduled. Larger work packages, say one with an expected duration of three months could constitute an entire Agile project in their own right.

Paradoxically an Agile Team Leader may find themselves lobbying the PRINCE2 Project Manager to divide the project into fewer larger work packages to increase the ability to use Agile techniques. Yet Agile Project Management advocates smaller units of work which can be delivered discretely.

Following Agile principles the development team should aim to make the package releasable at all times. However PRINCE2 limits the scope of the Project Manager to take early delivery of working software. So while elements of the package could be released in increments during the work stage they should not.

Conflict may arise when customers can see working software but the project process does not allow release of the software. Such software constitutes inventory – to use a Lean term – which has been paid for but not generating revenue. This combination will reduce the return on investment from the project.

The XP practice of a *40-Hour week* would probably work most of the time but if the team were not able to complete the Work Package during the scheduled time there may be pressure to work overtime. Of course this may happen whether XP or PRINCE2 is used or not. However, traditional, plan driven, project managers tend to demand overtime and weekend working when projects fall behind schedule so this might present some challenges. It might prove difficult to avoid overtime working if the it looks like the Work Package will not be completed during the stage.

Ironically, overtime working could be seen as a breach of PRINCE2 planning. If a schedule is specified in term of completion dates then overtime can be used to squeeze more work into the same timeframe, e.g. complete by 1 July. But if deadlines are specified in terms of time worked, e.g. complete within 100 hours, overtime cannot squeeze more work because a overtime is still time spent. Consequently overtime working may still breach project tolerances. (One option would be to specifying project tolerance in terms of hours worked *40-Hour week* practice.)

This becomes clearer when developers are paid hourly rather than salaried or daily. When hourly payment is used any extra work results in an extra cost. Higher project costs may be met either from the project contingency budget (if one is provided) or from project cost tolerance. So paying developers by the hour will also help protect the *40-Hour week* practice.

Similarly, *on-site customer* would require some negotiation. Although PRINCE has little to say about the requirements process there is an assumption that requirements can be stated up front and any variation is managed through a *change request, project issue* or *exception*. As long as the on-site customer only elaborated on existing requirements and did not vary the requirements things should work well.

Unfortunately many of the XP values are in conflict with PRINCE2. According to PRINCE2 communication is managed through a communication plan and communication channels align with management hierarchy. By XP standards this would reduce the communication flow to the team.

For similar reasons *feedback* would be reduced. This is especially true in the case of possible requirements changes or problems which arise. Both of these are subject to management control under PRINCE2, teams should not take action until authorised to do so.

As noted above PRINCE2 is risk averse. This is at odds with the XP value of courage and embracing change. In more strict PRINCE environments teams may be required to raise project issues or exceptions before undertaking large refactoring – which might interfere with the refactoring practice too.

It should also be obvious by now that the XP value of *simplicity* will also come into conflict with PRINCE2 procedures.

While the XP practices are broadly compatible with PRINCE2 the principles could give rise to conflict. The second edition of Extreme Programming (Beck and Andres 2004) puts a greater emphasis on principles and values and less on practices thus increasing the potential for conflicts.

Still, within the development team it should be possible to work around many of these problems. In part this will depend on how skilled the Team Leader is at PRINCE2 game-play and how willing the Project Manager is of XP within his environment. A Project Manager could, if they so wish, probably make life very hard for an XP team in a PRINCE2 environment.

Costs of plugging in Agile

So it seems possible to plug many of the Agile practices, if not principles, into PRINCE2. Now the question becomes is it worth it? What advantages might we see?

Using XP within the Managing Product Delivery process is likely to lead to improved code quality. Test-driven development, simple design, refactoring and collective ownership are likely to lead to better code and fewer bugs. Thus in turn leading to less rework and greater productivity.

Set against these benefits there would be increases costs in terms of planning and interfacing. The XP planning game would be played in addition to PRINCE2 planning leading to duplication of work. The Team Leader would also need to 'translate' PRINCE2 documents and plans to XP and vice versa.

Improved quality is only part of the benefit from XP, or any other Agile method. The larger benefits of Agile development come from flexibility; the ability for the business to change its mind and for the team to respond to a changing environment. If XP is only plugged into Managing Product Delivery then PRINCE2 mechanisms such as

issue, risk and exception handling would still be used to control change. The business would never benefit from the development team's ability to respond rapidly.

In an environment where change was occasional, or happened slowly, this might not matter. However in such an environment almost any development method would work well. The real challenge for any development method is how it works when change is rapid. The pairing of PRINCE2 and XP in a rapidly changing environment is likely to lead to frustration and conflict. Project Managers would be cast as the *bad guys*, constantly demanding formal change requests approvals and holding teams back from racing ahead before changes are approved.

Evolving to Agile

A second option to for combining PRINCE2 and Agile takes a more evolutionary approach. PRINCE2 mandates Project Managers to keep a Lessons Learned Log and produce a Lessons Learned Report at the end of the project. These may be used as mechanisms for introducing Agile techniques to an existing PRINCE2 project.

These provide some opportunity to make modifications to the process. Overtime teams may agree to log lessons which lead them to modify their process as they go. For example, a team might note the difficulty in meeting deadlines and decide to adopt a time-boxed approach.

However this approach has several draw backs. Firstly the log is passive and the report is not written until the project closes. This is too late for the project although it might benefit a subsequent project. If this happens we a second problem appears, changes and modifications will happen gradually, the rate of learning and change is slow, change by a thousand lessons learned may take a long time.

More troubling is perhaps the fact that the Lessons Learned Report is, according to PRINCE2, written by the Project Manager. Without wishing to slight Project Managers, one should remember that Project Managers have a vested interest in showing that the Project Management of the project was satisfactory. It is not in a Project Managers own interest to write Lessons Learned Report which suggests a radical break with existing project management practice.

Making a larger group responsible for collectively producing the Lessons Learned Report would help offset this tendency. The more familiar a team is with PRINCE2 the less likely they are to make radical changes. There will be a natural tendency for the report to extrapolate form the past, recommendation may well be limited to minor changes to process and practices.

So, while it is not impossible to imagine a Lessons Learned Report recommending the adoption of some Agile practices it is unlikely. Many Agile practices represent too radical a deviation from PRINCE2 norms.

Tailoring PRINCE for Agile

Capturing quality improvements from Agile techniques is relatively straight forward within PRINCE2 because many of the practices occur below the radar. If we are to capture the benefits that come from Agile responsiveness we need to tailor PRINCE2.

Improving responsiveness will lead to a consequent increase in risk. As already pointed out, profit is the return for risk, if we want to profit from Agile processes we need to accept a higher level of risk than previously prevailed. An environment that

will not tolerate this risk will probably not tolerate Agile methods. Attempts to introduce Agile project management into this environment will most likely lead to conflict.

This tailoring exercise will show how PRINCE2 could be combined with the Blue-White-Red Agile method outlined elsewhere (Kelly 2007, 2008). Using Blue-White-Red for this illustration has two advantages. Firstly, as the originators of the method I know it well. Second Blue-White-Red is particularly open to adaptation. Unlike other methods I will not fall fowl of the breaking rules.

(For future reference I will call the resulting process Blue-White-Red/PRINCE2.)

Tailoring starts by introducing some rules and assumptions:

- The development team will release once per month. This release will be to a live server from where customers can access the software.
- The development team will normally complete four iterations between releases. Some months the team will complete five iterations, depending on the calendar. Occasionally the team may skip the release altogether, e.g. at Christmas and New Year.
- The project board will meet two weeks after each release, review progress, reports, exceptions and other issues. It will also authorise continued work on the project.
- Project Board will delegate Quality Assurance. On most projects this will be an individual or group of individuals. There role will encompass functional testing and conformance to expectations.
- The development team will be fixed in size for the short run. Any increases in staffing must be approved by the Project Board. Should anyone leave the team Project Board approval is required for replacement. Since staffing costs are the major element of the project budget this will effectively freeze the budget. Some budget will be made available to the Project Manager for ad hoc expenditure.
- The board may shut down the project at its regular meeting but agrees to provide the team with notice at least one month in advance. It is reasonable fair to say that, barring catastrophe, the board and team will foresee the end of a project several weeks in advance. In the event of catastrophe shutting the project down immediately is probably the right thing to do.

While PRINCE2 has no requirements for regular meetings, releases or stages neither does it prevent them. Therefore all these assumptions are PRINCE2 compliant.

Note that these assumptions are only made for this discussion and show how we might bring Blue-White-Red and PRINCE2 together. Individual projects may vary them, e.g. the team may complete six one week iterations between releases, provided these assumptions stay within both the PRINCE2 and Blue-White-Red frameworks.

Next I want to introduce a new role to the process. This is the Product Manager, in some environment this role may be called Product Owner or Business Analyst. the board will delegate all Change Authority to the Product Manager. At the regular board meeting the Product Manager will brief the board on changes he has authorised and outline any upcoming possibilities.

It might seem a drastic move to delegate all Change Authority to the Product Manager but this is necessary if the team is to be responsive. Remember that PRINCE2 assumes the Project Board are busy people, if the development team is to be responsive it needs immediate answers. Since the board will meet with the Product Manager regularly the risk is minimised. Further, it is entirely possible that the Senior User, or even Executive could double as the Product Manager.

The Product Manager's role is extensive. One the one hand the Product Manager must be out talking to customers, discovering their needs and their views on the product. If the product is sold commercially – rather than being developed in house or for one customer only – then this manager also needs to be watching the wider market. This involves watching competitors, knowing their products and being aware of changes in the market.

On the other hand the Product Manager must work with the development team. Both to understand what the technology is capable of and what could be done – an information flow from the team to the manager – and also advising the team on features under development – a flow from the manager to the team. When implementing any feature there are multiple decisions to be made and options to be considered. The Product Manager needs to be available for the team. Only where the team are very experienced in the domain will they be able to act autonomously.

In addition the Product Manager needs to understand the corporate strategy and ensure that the product is developed in line with the strategy so it satisfies future demands. At the same the product may well play a part in determining that strategy so the Product Manager may need to take part in corporate strategy formation.

The Product Manager will also be charged with keeping and maintaining the Business Case. This responsibility will move from the Project Manager (in PRINCE2) to the Product Manager. Further the Project Plan will be replaced with the Product Roadmap.

The Roadmap differs from the Plan in a number of ways. Items on the roadmap are listed in priority order rather than date order. Speculative dates may be forecast for items but no specific dates will be given. Conversations around features will be based on priority not date. This changes the nature of the conversation because moving to a higher priority automatically moves other items to a lower priority. The priority ordering shows clearly that there is no free-lunch.

Neither will the Roadmap show who will undertake which work. This level of detail is simply not required on the Roadmap so no names will be specified. When names are specified in advance it becomes more difficult to reallocate tasks thereby reducing team flexibility.

Instead of focusing on delivery to plan the development team – including Product and Project Managers – need to focus on value delivered. In our revised process progress needs to be measured not against the Product Roadmap but directly against the Business Case. Each release needs to show value added. Everyone involved with the project needs to understand this. It is the Product Managers responsibility to show that value is being added.

The value added approach also results in another change which has a long term effect. While the team can continue to show value added there is no reason to close the development. If the team complete the work envisaged in the original Business Case but continue to deliver added value then it makes sense to keep the team working.

If the team stop adding value, or the value added falls below the cost of the team (please a reasonable rate of return) then it makes sense to close the project down. This could happen before or after the original Business Case is met.

Alternatively, in a resource constrained environment, if another product team can is adding more value then it may make sense to transfer some, or all, of the resources form one product to another.

The observant reader will notice we are using the language of products not projects here. In part this is a reflection of these changes. It also demonstrates that we do not know when, or if, the product will be judged complete. Consequently the development team need to prepare for the future.

The Project Board still need to set tolerances for the project but with these changes the nature of tolerances changes:

- Time: work has now been time-boxed into one-month periods. The team are expected to show value added over month period. If a specific feature must be completed by a specific date then presumably there should be an associated value provided this value is high enough the feature will be prioritised and worked on. If there are features with higher value then it makes sense to do these first.

  It is conceivable that some features will be requested for reasons other than value added. In these cases the requester should be able to demonstrate a non-financial return that justifies prioritising the feature above financial value adding features. This is quiet acceptable – provided the Product Manager and Project Board can be convinced.

  Items may move between time boxes and but the criteria for allocating items to time box is based on value not deadlines. Some items may have a high value if completed by a given date and now if delivered later. However these items must still be prioritised against others and comparing value is the way to do this.

- Money: tolerance for additional expenditure is zero. The team operate in a steady state.
- Scope: there is a high tolerance for scope changes provided they can show value added. The Product Manager has authority to increase, or reduce, scope as they see fit. However, they will have to answer to the Project Board each month, thus their ability to massively change the scope is limited.
- Quality: high internal quality should be maintained at all times. Changes to external quality (i.e. customer facing quality) will need to be authorised by the Quality Assurance representative in conjunction with the Product Manager where appropriate. This will allow for a faster resolution of issues. Again the monthly board reviews will limit any major deviations from intention.
- Risk: as already stated the corporation and Project Board must accept a higher level of risk than is normal. Day to day risk management can still be undertaken by the Project Manager and reports made to the board. The monthly board meetings will serve as the main regulator on risk.
- Benefits: by emphasising value added rather than conformance to plan benefits have been placed centre stage. There is no upper limit on the additional benefits that may be gained. Under performing forecast benefits is taken very seriously. If

value add fails to meet expectations, especially if this happens in several consecutive release, the whole product should be reviewed and potentially closed.

In general the approach to tolerance changes in two ways. Firstly, tolerance is tightened, there is no more money available, failure to deliver benefits is considered very seriously. The second change is that rather than management by exception we are managing by rhythm. Those working on the project are trusted more between Project Board meetings but are held to account regularly.

The change of approach to tolerances and change in focus from delivery to plan to delivery of value may prove too much for some organizations to stomach. These are, undoubtedly, large changes to the PRINCE2 way for working. They are, however, standard approaches in Agile development.

As with straight PRINCE2, the developments working day-to-day are free to use what ever practices they like. In line with Blue-White-Red, XP and other Agile methods they are expected to use Test Driven Development, Refactoring, Pair-Programming – or at least code reviewing, continuous integration, simple design, etc. The removal of management by exception and the extra authority vested in the Project and Product Manager should allow teams to take more risks and take courage.

Of the three PRINCE2 techniques Product Based Planning is still valid. Only now, instead of the product plans being used as the basis for project plans they are the basis of value delivery. The intermediate Project Plan is eliminated.

The Change Control technique has been changed significantly by the delegation of authority and the emphasis on value add. The approval of a Change Request does not guarantee that the change will occur. Approval is simply the first step, next the change must now demonstrate its value and fight its way to the top of the priority queue.

The critique above has cast shown the need to reform the Quality Review technique. Instead products requiring review should be reviewed in an informal desk check. The objective is to apply the Pareto principle. Perform the 20% of the check that delivers 80% of the benefit.

Before looking at the processes in detail we need to consider the documentation produced by the method - Management Products in PRINCE2 terminology. As already stated the Project Plan is replace by the Product Roadmap and a prioritisation process. The Project Quality Plan and Quality Log should be devolved to the QA resource on the project. They should be encouraged to make automate the application of the quality criteria wherever possible and minimise the amount of documentation produced to support quality.

The Product Manager will take over responsibility for product related issues inline with their management of Change Requests. If this where not done then we would need to decide what was a Change Request and what was an Issue. If one person manages both then the debate is curtained.

The Project Manager may continue to own other project issues or may decide to pass this work, particularly for technical issues, to the Team Leader. The Project Manager will retain responsibility for the Risk Log but with the revised project risk profile and a greater acceptance of risk there should be less analysis and management of risks.

Similarly the Project Manager should retain responsibility for Exception Reports but with the revised reporting framework there should be fewer exceptions. Those that are raised will normally be handled in the monthly Project Board meetings.

In order to broaden participation in, and make more use of, the lessons learned the lessons learned log will be replaced by an interactive retrospective session after each release. This should last between one and two hours and will give all team members a chance to contribute to the lessons learned. Between times team members are free to make their own notes and record their own lessons learned.

As part of the retrospective the team should collectively agree on a few changes to make to their working practices, process or environment. These changes will be enacted immediately and a report made to the Project Board. Where the team need higher authority to change something or need funding for a change, e.g. demolishing an office, the board will we asked to approve the change.

Product, or project, initiation will still be started by the Project Mandate and the early stages of the effort will still need to work through the business case, product requirements and such. However, there should be less reliance on documentation and management products and a greater emphasis on using advanced planning sessions to examine options and prepare for project. Requirements and business case will be expected to evolve as the work effort develops.

The need for Team and Stage plans will be removed by the process mapping which we will now examine.

Figure 3 shows the revised process flow, this is itself not very different to the generic PRINCE2 flow outlined in Figure 2 however there are some changes to the processes.

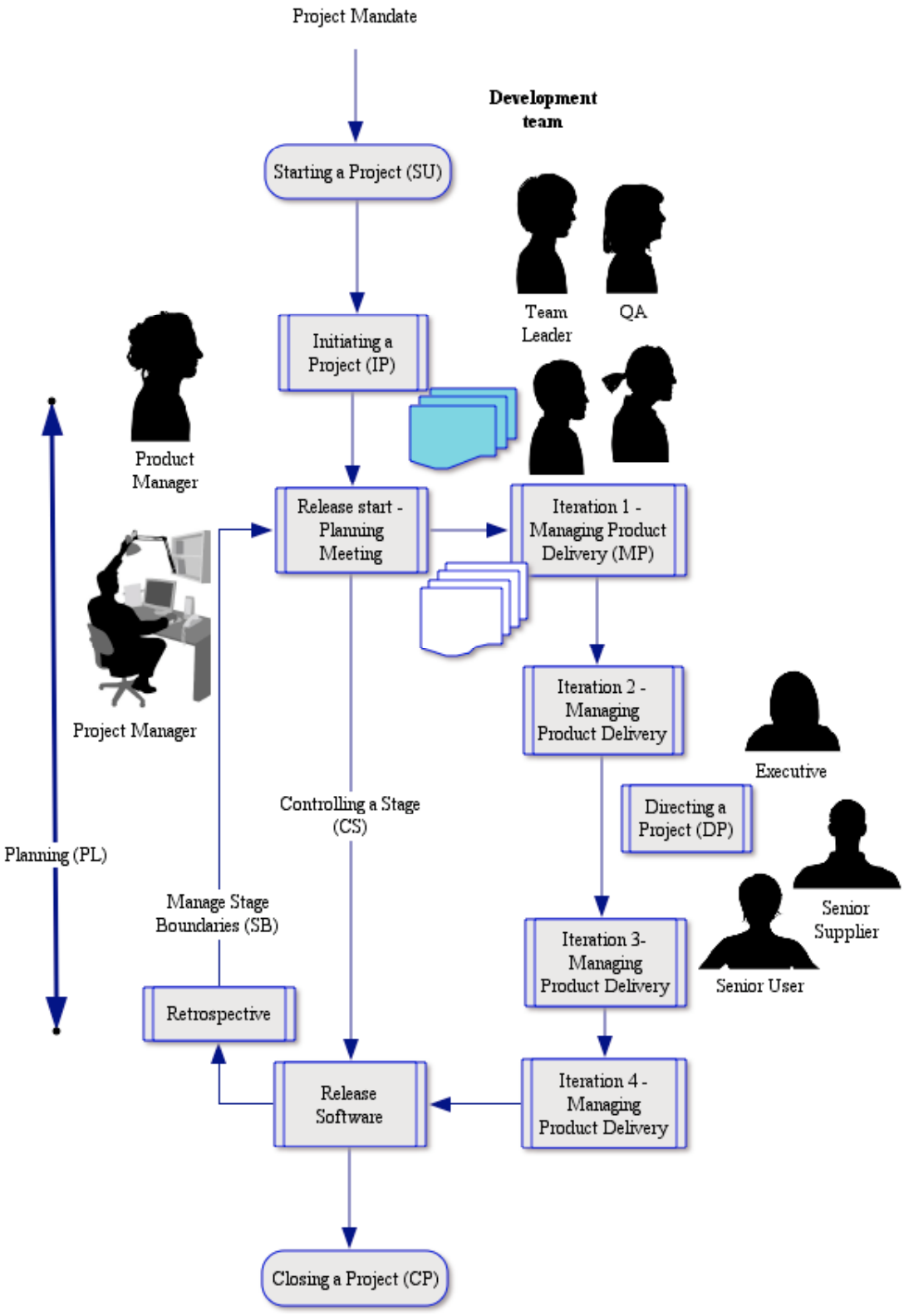

Figure 3 - Process Flow for PRINCE2 complaint Blue-White-Red

Product development efforts start in the same way. Project Mandates are handed down and some initial appointments are made. These appointments then formalise what is required and gather the team. Although the SU and IP stages remain broadly the same the amount of work involved should be reduced thanks to the documentation

changes. In addition there is an extra person, the Product Manager, to share the work load so these stages should be accomplished more quickly.

When the team is ready for work to begin they hold a planning meeting. Subsequent planning meetings are held after each software release.

Prior to the planning meeting the Product Manager will have reviewed product prioritises and created blue feature cards for each major user facing piece of work. (In SCRUM terminology this is the product backlog.) In the planning meeting the developers break the cards down into functional pieces of work which are documented on white task cards. Together the Product Manager and team will agree priorities. The work that is accepted into the release constitutes the Work Package.

For each white task card developers estimate an effort value in abstract points. The first time this is done the estimation is very abstract. Over time the team converge on a shared understand of what a point is worth. Future planning meetings start by reviewing the work done and counting the points. This then provides a guide for the amount of work which will be conducted in the next iteration and release.

With the features defined, the work understood and estimated the team can then prioritise the work. The team undertake to do the work in the order the Product Manager requests and the Produce Manager undertakes to listen to the advice of the team. So the team can advise of any dependencies in the work, or potential time savings. At the end of the meeting the work for the next release is defined by the cards.

In effect the team have created their own Stage Plan which defines the work to be done. The Team Plan is irrelevant because the team will now do the work in priority order. Except in special circumstances individuals are not assigned to pieces of work. To do so would break the priority order.

The team now have a plan but they need to break this down into an iteration plan. As we stated above we assume four one-week iterations for each release. The planning meeting is therefore repeated at the start of each weekly iteration. The first planning meeting for the release will probably take longer than the subsequent planning meetings. It might be necessary to re-estimate work, re-prioritise work or find alternative ways of addressing features where problem arise. The final iteration planning meeting will likely need to consider the release procedure in detail.

During the meeting the Product Manager, supplemented by the Project Manager and Team Leader are empowered to assign priorities, make changes and deal with issues and risks as they arise. Once the meeting is complete the Project Manager will be left to perform Controlling a Stage while the Product Manager has time to visit customers and work with the team.

Around two weeks after the release, half way through the next release the Project Board will convene for its Directing a Project process. The Product Manager and Project Manager will report the outcome of the recent release, outline the work currently in progress, raise any issues or exceptions, relate the outcome of the last retrospective and inform the board of subsequent changes. Most importantly they are expected to demonstrate to the board how the team is adding value to the company and product.

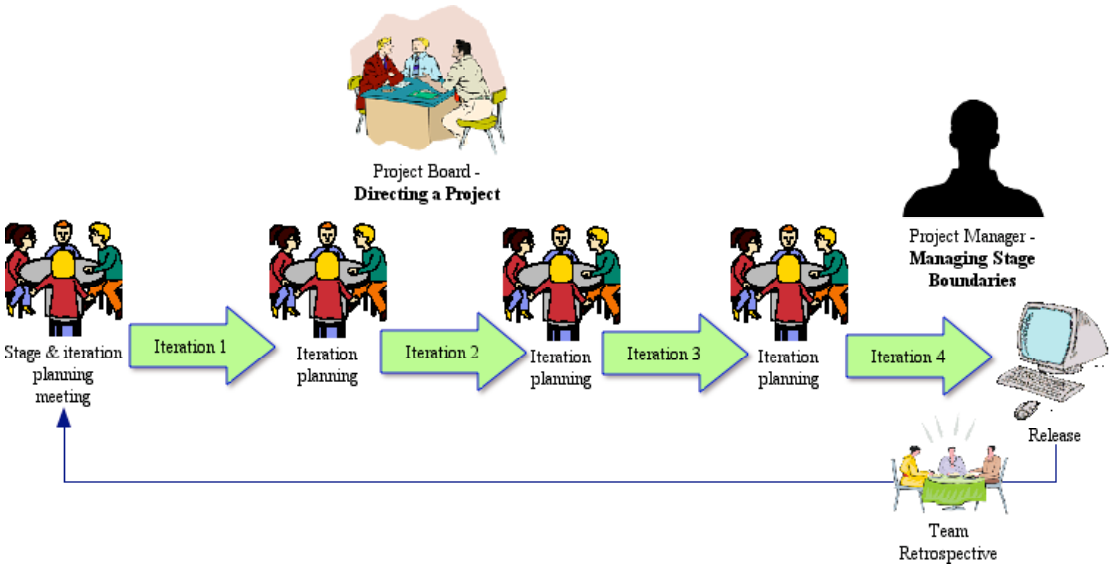

Figure 4 - A stage cycle with four iterations and one release

At the end of four iterations the release, or Stage to use the PRINCE2 term, will be complete. The team will release the software to the live system and celebrate. They will hold a retrospective and consider how to improve their work. The Project Manager will also conduct a Managing Stage Boundaries process to complete the release and prepare for the next planning meeting and Project board.

Nominally the Planning Process continues to operate throughout the cycle however most of the planning now occurs within the Planning Meetings for the release and iterations.

In the event that a Project Board meeting decides to close the project the team is allowed to complete the current release and undertake one more release to put affairs in order. Knowing the next release is the last one will change the Product Manager priorities for work. After this the closure process is largely the same as the normal PRINCE2 process.

It should be clear here that in this model the Project Managers role is drastically reduced. Some of his work is passed to the Product Manager, some is eliminated and the rest reduced. At first sight it might be tempting to combined the Project Manager role with that of the Product Manager, this temptation should be avoided simply because it risks overloading the Product Manager.

Product Managers have a lot of responsibilities and work to do: creating the business case, liaising with the project board, the development team, talking to customers, monitoring the market, calculating value and so on. The more fully, and more competently these tasks are undertaken the better the work of the development team. Skimping on these tasks is easy but will ultimately reduce the value delivered by the project.

Combining the Project and Product Manager roles not only risks overloading the Product Manager but will lead to a reduction in the quality of their work. A loss of quality in this role will impact the project and product elsewhere.

If the Project Manager role needs to be combined with another it is better to combine it with the Team Leader role. This may not be possible, if for example the development team, including the leader, work for a separate supplier organization.

When a corporation demands that project still appear like a PRINCE2 project on the outside a dedicated Project Manager will probably be needed to "fake" the documentation required. (For a wider discussion on faking development processes see Parnas and Clements (2001).)

The Product Manager, QA and Team Leader roles are important in this model. Teams and product will vary but there should be at least one Product Manager for every seven developers, on some products one for every three developers will be more appropriate. Similarly there should be at least one QA person for every seven developers and normally more. The Team Leader will most likely combine their role with some developing but the larger the team, the larger the product, the less likely this is.

Tailoring

> "the way in which PRINCE2 is applied to each project will vary considerably, and tailoring the method to suit the circumstances of a particular project is critical to its successful use." (Commerce 2005, p.9)

The PRINCE2 manual says little about how to go about tailoring PRINCE2. This create a problem: when is tailoring *tailoring*? And when is creating a new method or process?

To put is another way: how far can tailoring go? Of the approaches outlined above, *Plug in Agile* requires little, if any, tailoring to PRINCE2. Evolving a PRINCE2 process to an Agile one – no matter how unlikely – may result in a tailored process from within. The third approach arguably goes beyond tailoring and outlines a new method.

Blue-White-Red/PRINCE2 is proposed a tailoring of PRINCE2 but it might also be viewed as a entirely different method, one which is not compatible with PRINCE2. After all, the replacement of *management by exception* with *management by rhythm* and removing project plans are pretty major changes. What is needed is somekind of test which can determine whether Blue-White-Red/PRINCE2 is a tailored adaptation or something else all together.

One of the PRINCE2 sub processes, called Setting up Project Controls (IP4), covers some of this ground. During project start up managers are asked to:

- "Establish the level of control and reporting required by the Project Board for the project after initiation
- Develop controls that are consistent with the risks and complexity of the project
- Establish the day-to-day monitoring required to ensure that the project will be controlled in an effective and efficient manor
- Identify all interested parties and agree their communication needs." (Commerce 2005, p.60)

This is to include:

- "Allocating the various levels of decision making required within the project to the most appropriate project management level
- Establish any decision-making procedures that may be appropriate ...

- Consider the size of Work Packages to be used ....
- Establish monitoring mechanisms ....” (Commerce 2005, p.61)

In making these decisions the following is suggested:

> “The next points are there to reinforce the motto: ‘Not too little, not too much’:
>
> - Are the controls appropriate to the risk, scale and complexity of the project?
> - Is the level of formality established appropriate to the risk, scale and complexity of the project? ...” (Commerce 2005, p.62)

These considerations allow for most of what is needed to tailor PRINCE2 into Blue-White-Red/PRINCE2. Any project tailoring will need approval of the appropriate Project Board. If the Project Board feel the Blue-White-Red/PRINCE2 is appropriate to the risk, scale and complexity of the project then the method may be used. In fact, the Project Board could probably go even further if it was so minded.

I should point out at this stage that I am not an expert in PRINCE2 tailoring. Neither have I read the only book I know that deals with this subject *Tailoring PRINCE2* (Commerce 2002). I haven’t even tried Blue-White-Red/PRINCE2 in anger; I have only proposed it as a possible PRINCE2 compliant Agile method. There are people out there who know far more about PRINCE2 than I know – or ever want to know!

It seems we can argue that Blue-White-Red/PRINCE2 is a tailoring of PRINCE2. And we can argue that it is not. However, for your project – or product, or programme – the people who decide this are your Project Board. If they accept Blue-White-Red/PRINCE2 is appropriate for the scale and complexity of the project, and importantly, they accept the risk profile then you can use it.

Still, there are two bodies apart from the Project Board who could invalidate this decision. The first is the corporation or programme managers to whom the Project Board report. Simply, if they decide to disagree with the Project Board they can overrule them, even replace them, if they wish. So if your organization is big enough to have a layer above the Project Board then you probably need to convince this group too.

The second group are your corporate auditors. Some organizations audit their process, not all organizations but some. It is a fair bet that if your organization audits your processes then they care about what process you are using, after all, why bother spending the money on a process audit if you don’t care?

For example, I once worked on a project that was audited for ISO-9000 compliance. This was important to the company because some of their customers wanted their suppliers to be ISO-9000.

A few years later I worked for an organization that decided to adopt CMM as a model. The company had been convinced that CMM level 3 compliance would improve the development process and therefore profitability. The company changed procedures and was duly audited as CMM level 2. However, rather than working toward level 3 the company quietly dropped the whole programme. Adopting CMM had been disruptive, expensive and did not deliver the benefits claimed.

It is difficult to guess what an auditor would do faced with Blue-White-Red/PRINCE2. Quote them the PRINCE2 manual, as I have above, the tailoring description, also above, and authorisation from the Project Board might be enough to

pass audit. Alternatively, the auditor could call in a PRINCE2 expert who recoils in horror at the tailoring I have suggested.

Ultimately unless someone wants to implement and audit Blue-White-Red/PRINCE2 we won't know. Until then I will claim that PRINCE2 can be tailored to manage a Blue-White-Red Agile team.

Marrying PRINCE2 and other Agile methods

It is relatively easy to imagine XP and PRINCE2 working together because XP mainly concerns itself with development techniques and says relatively little about project management and requirements. Other Agile techniques, specifically SCRUM (Schwaber and Beedle 2002), are likely to have more conflict with PRINCE2 because they cover more of the management issues.

Lean software development (Poppendieck and Poppendieck 2003) probably faced the greatest number of challenges because the underlying principles are even more different to PRINCE2 than other Agile methods.

For organizations that must be PRINCE2 compliant and want to use Agile methods DSDM, or the DSDM consortium's new framework, is worth looking at. Since both PRINCE2 and DSDM trace their origins to UK Government projects this is hardly surprising that DSDM claims to be PRINCE2 compliant. In the past the DSDM consortium has levied charges for use of DSDM which has limited its use. DSDM use and some materials are now free to use but you might want to check the terms and conditions first.

Conclusion - Reconciliation and lost benefits

Despite the attempts here, and by the DSDM consortium, to reconcile PRINCE2 and Agile software development the two will ever be fully reconciled. The fundamental starting position and assumptions of the two approaches are different and they are build on different principles.

PRINCE2 is rooted waterfall development, incorporates an adversarial customer-supplier model that needs change control, formal processes and auditing, and is plan centric. In short PRINCE2 assumes directed processes control.

Agile rejects just about every assumption PRINCE2 makes. Agile assumes an inherently complex environment in which defined process control cannot be achieved, instead empirical process control is used to direct (Schwaber 2003) activity. Rather than a plan based client-supplier organization a co-operative model with emergent behaviour is assumed. Instead of measuring progress again plans the only true measure of success is delivered, working, software.

To put it more succinctly, PRINCE2 assumes a directed process with contingent planning in advance. Agile assumes an emergent process with just-in-time decision making if and when required.

It is possible to use both techniques in tandem but both will be compromised. It is not possible to realise the full benefits of Agile if you use it in a PRINCE2 environment. Neither will you see the full benefits of PRINCE2. You can combine them but it will cost.

The difference are perhaps most apparent when the role of plans is considered. Agile does not reject planning entirely but it is not plan driven. The plans that are made are short term, they are acted on and discarded. Agile avoids long range plans because work is change driven rather than plan driven.

PRINCE2 on the other hand is proudly plan-centric. The method accepts that reality will diverge from plans and therefore sets up a mechanism for updating the plans to reflect reality as it un-folds. Vast swaths of documentation detail how, and when, plans are to be systematically updated to reflect and monitor reality.

The further into the future the plans project the more change the plan will suffer. The more detail a plan contains the more deviations from plan there will be. The larger the plan the more work is required to maintain the plan.

Yet ultimately customers do not take delivery of a plan. A plan is a temporary object used to co-ordinate work. What is important is the quality of the co-ordination, not the quality of the plan. Eventually all plans are consigned to the waste bin, therefore, much of the effort spent updating them is ultimately wasted.

The solution therefore is to minimise planning. Rather than constantly update plans don't create them in the first place, and limit plan life span to a period when it is useful. This sounds radically but in large part it is what Agile methods try and do. The plans that do exist are short term, these are the only type of plan which stands any chance of matching reality. Long-term plans present the illusion of control but do not present control.

Perhaps one mistake frequently made is confusing *control* with *management*. Control is about directing, telling people what to do, switching things on and off, deciding what should be and making it so. Control works when you use a TV remote, or when you put your foot on the break.

But management is something different. Management is messy world of ambiguity, coping with change and getting stuff done. It is one-third psychology, one-third economics and one-third energy – with a fair bit of chance thrown in too.

You can manage a software development effort but you cannot control it. Things happen which are beyond control. It is only worth expending so much effort on trying to control a software project, additional effort is wasted because the process is emergent. This is where plans really fail. They present the illusion of control not control itself.

For some people this argument will be unacceptable. Some organizations need and expect plans. In these organizations, despite the failing of plans, they are a necessary evil. Such environments are unlikely to embrace Agile development techniques. These organizations may well benefit from the PRINCE2 approach, any use of Agile techniques will run counter to the culture and more likely than not create conflict leading to problems.

However, organizations which value change and adaptability over planning and predictability will benefit from Agile techniques. For such organizations a PRINCE2 approach will run counter to the prevailing culture and introducing it will, most likely, result in conflict and problems.

Therefore, the question that needs to be asked is not: *which is better?* but *which is more suited to our environment?*

For many organizations even this will be the wrong question. The question needs to be: *which type of organization do we wish to be?* Predictable or adaptive?

As use of Agile methods grows on one hand, and on the other use of PRINCE2, PMI and similar grows on the other; then more and more managers will be asked to reconcile the two approaches. Fortunately the two approaches can be reconciled. Unfortunately the result is sub-optimal and looses benefits from both.

Epilogue

The thing I find funniest about PRINCE2 is that it is marketing itself as a general, all-purpose project management method. For anyone who has worked in IT, and particularly software development, the approach and roots are obvious. Yet who outside the world of IT believes that the industry presents a good role model for project management?

IT projects are notorious for project failures. So why would anyone want to manage projects the way IT does? Perhaps there is a hidden agenda: rather than improve IT delivery the goal is to make the rest of the world sink to the same level, so IT projects do not look so bad.

It would be understandable if IT had fixed its problems. Maybe in 10 years time people will look back and say: "2008 was the year it changed. IT is now successfull." Then I could understand the rest of the world wanting to manage projects the way we do. We would have learned out lessons and could teach them to others. But that point has not been reached yet.

If, come 2018, that statement can be made then I think it will be building on the success of the Agile methods.

This is not to say the IT does not have things to teach the wider world. IT is a twentieth century success story, it has developed rapidly and overcome many obstacles. In fact this author has argued that software developers are the prototype of future knowledge workers (Kelly 2008). However, the IT world still faces many problems and is evolving rapidly. Anyone wishing to learn from IT needs to choose carefully from the successful methods and techniques pioneered. At the time of writing, project management is not one of our success stories to be copied.

Links

While the internet is full of adverts for PRINCE2 training courses there is noticeably less information on the PRINCE2 process itself. Other introductions and overviews can be found at:

- http://en.wikipedia.org/wiki/PRINCE2
- http://www.jiscinfonet.ac.uk/InfoKits/infokit-related-files/prince2-overview
- http://www.spoce.com/knowledge-base/prince2/what-is-p2.aspx

There have been a other attempts to describe how PRINCE2 and Agile can be brought together. Most of these centre on DSDM (Richards 2007), one exception is Craig Cockburn blog:

- http://www.siliconglen.com/news/2008/01/prince2-agile-common-sense.html

However, others have also questioned the rational and effectiveness of combining the two approaches, for example:

- http://www.btt-research.com/waterfall_projects.htm

These links are good at the time of writing (May 2008) but may change over time.

About the author

Allan is a London based consultant and interim manager specialising in Agile adoption. After 10 years at the code face he came to believe that many of the problems faced by development teams are not in the code but in the management of development efforts.

His first book, "Changing Software Development: Learning to be Agile" was published by John Wiley & Sons in early 2008. He has worked as Product Manager, Project Manager and Development Manager, holds a BSc in computing, an MBA in management and is a certified PRINCE2 Practitioner. More information and writing at http://www.allankelly.net.

References

Beck, K. 2000. Extreme Programming Explained: Addison-Wesley.

Beck, K. and C. Andres. 2004. Extreme Programming Explained: Embrace Change. Second Edition: Addison-Wesley.

Commerce, Office of Government. 2002. Tailoring PRINCE2. London: TSO (The Stationary Office).

Commerce, Office of Government. 2005. Managing Successful Projects with PRINCE2. Fourth Edition. London: TSO (The Stationary Office).

Conway, M.E. 1968. "How do committees invent?" Datamation(April 1968).

Davenport, T.H. 2005. Thinking for a Living. Boston: Harvard Business School Press.

Derby, E. and D. Larsen. 2006. Agile Retrospectives: Pragmatic Programmers.

Hedeman, B., G. Vis van Heemst and H. Fredriksz. 2005. Project Management Based on PRINCE 2: Van Haren Publishing.

Kelly, A. 2007. "Blue White Red - an example agile process." ACCU Overload(81).

Kelly, A. 2008. Changing Software Development: Learning to Become Agile: John Wiley & Sons.

Kennedy, M.N. 2003. Product Development for the Lean Enterprise. Richmond, VA,: Oaklea Press.

Kerth, N.L. 2001. Project Retrospectives. New York: Dorset House.

Koskela, L. and G. Howell. 2002. "The underlying theory of project management is obsolete." In PMI Research Conference.

Parnas, D.L. and P.C. Clements. 2001. "A rational design process: How and why to fake it." In Software Fundamentals: collected papers of David L. Parnas, ed. D.M and Weiss Hoffman, D.M.: Addison-Wesley.

Poppendieck, M. and T. Poppendieck. 2003. Lean Software Development: Addison-Wesley.

Prince, M. and A. Schneider. 2004. "Patterns for the End Game." In EuroPLoP, eds. K. Marquardt and D. Schutz. Irsee, Germany: UVK Universitatsverlag.

Richards, K. 2007. Agile Project Management: Running PRINCE2 Projects with DSDM Atern: TSO (The Stationery Office).

Schwaber, K. 2003. Agile Project Management with Scrum: Microsoft Press.

Schwaber, K. and M. Beedle. 2002. Agile Software Development with SCRUM: Addison-Wesley.

Votta, Lawrence, G. 1993. "Does Every Inspection Need a Meeting?" ACM SIGSOFT Software Engineering Notes 18(5).

Votta, Lawrence, G. and Adam Porter. 1997. "What Makes Inspections Work? ." IEEE Software 14(6).

Womack, J.P., D.T. Jones and D. Roos. 1991. The machine that changed the world. New York: HaperCollins.