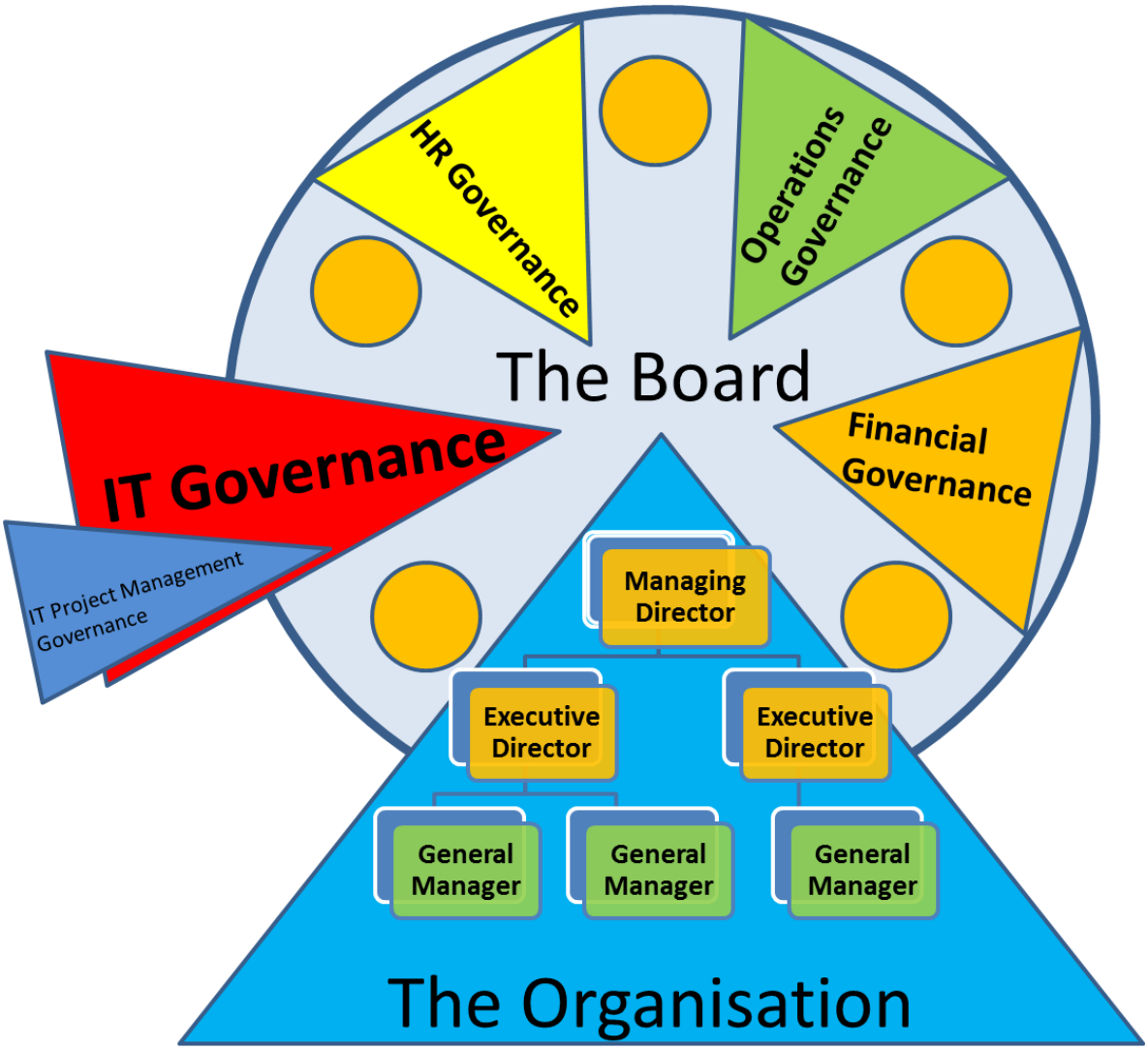
PROJECT MANAGEMENT GOVERNANCE:

Alignment with IT Governance



Richard Willis

Table of Contents

Table of Contents i

List of Figures iii

List of Tables iii

The Need for IT Project Governance 1

Evidence of Failure 1

Top 10 IT Disasters 2

Why IT Projects Fail 4

Objective 5

What is ISO/IEC 38500:2008 Corporate Governance of Information Technology? 5

AS/NZS 8016:2013 Governance of IT enabled projects 5

Benefits of using AS/NZS 8016:2013 Governance of IT enabled projects 6

IT project governance model 7

Evaluate 8

Direct 9

Monitor 9

Six principles for good governance 9

Causes of project governance problems 10

Project Governance Objectives 10

Risk 10

Organisational versus Project Structure 10

Stakeholder and Ownership 10

Principles of Effective Project Governance 11

Principle 1 11

Principle 2 11

Principle 3 11

Principle 4 11

Project Management Governance Model 11

Garland's Model – Roles and Responsibilities 13

Project, program and portfolio management 14

Governance Templates 16

PRINCE2 17

PRINCE2 Elements 18

The Principles of PRINCE2 18
The Themes of PRINCE2 19
The Processes of PRINCE2 19
PRINCE2 Maturity Model (P2MM) 21
Overview of P2MM 21
Attributes 22
Project Management Body of Knowledge (PMBOK) 24
The Project Life Cycle 24
Project management knowledge areas 25
ISO 21500:2012 Guidance on project management 27
Process Management Processes 27
Subject Groups or Knowledge Areas 28
Agile Project Management 29
Non-Agile 29
Agile Software Development 29
IT Project Management Approaches 29
Project Management Approaches vs Project Governance Attributes 30
Project Governance Attributes Median Score 31
Conclusion 33
Appendices 35
Appendix 1 – Project Governance Policy 35
Appendix 2 – Terms of Reference and Modus Operandi of Project Governance Bodies 40
Appendix 3 – Role Descriptions 45
Appendix 4 - Project Management Methodologies 53
Non-Agile 53
Agile 67
References 78

List of Figures

Figure 1 - Key Elements of a Governance Framework for IT 7
Figure 2 - Governance Model for IT Projects 8
Figure 3 - Project Board 12
Figure 4 - Project Governance Model 12
Figure 5 - Project Management Team Structure 13
Figure 6 - Portfolio, Programme and Project Management 16
Figure 7 - OGC best-practice guidance 17
Figure 8 - PRINCE2 Structure 18
Figure 9 - PRINCE2 Principles 18
Figure 10 - PRINCE2 Themes 19
Figure 11 - PRINCE2 - Processes 19
Figure 12 - The PRINCE2 Processes 20
Figure 13 - Maturity Levels 22
Figure 14 - Example assessment of Process Perspective 23
Figure 15 - Project Life Cycle (PMBOK) 24
Figure 16 - Project Management Knowledge Areas used during Project Life Cycle 26
Figure 18 - The project governance structure 37
Figure 19 - phases in a project governed by waterfall management 53
Figure 20 - Illustration of project progression with the use of V-Model management 55

List of Tables

Table 1 - ISO 21500 and PMBOK Guide Process Groups comparison 27
Table 2 - ISO 21500 Subjects and PMBOK Knowledge Areas 28
Table 3 - Project Management Frameworks vs Project Governance Attribute 30
Table 4 - IT Governance Maturity focus areas 32
Table 5 - Advantages and Difficulties of the Spiral Process Model 63
Table 6 - Software Risk Management Plan 63

PROJECT MANAGEMENT GOVERNANCE

The Need for IT Project Governance

IT supports the core functions of most organisations.

Large IT projects not only fail more, they also deliver less. According to the McKinsey/Oxford study Bloch, Blumberg, and Laartz (October 2012) 50% of IT, projects with budgets of over $15 million run 45% over budget. Additionally, they are 7% behind schedule and deliver 56% less functionality than predicted. That means, for example, it requires spending $59 million to achieve at least $15 million in benefits.

According to a survey by cloud portfolio management provider, Innotas, half of all businesses had an IT project fail over the last year (Florentine, 2013). The primary reason for this failure, according to 74 percent of respondents, was a lack of resources to meet project demands.

Evidence of Failure

In 2012, only 39 % of all projects included in Standish Group's Chaos report (The Standish Group, 2013) succeeded by being delivered on time and on budget with the required features and functions. Although this is an increase from a year earlier, it still portrays a pretty bleak picture of global IT project successes and failures across just under 50,000 global initiatives.

According to the Chaos report, 18% of these projects failed because they were cancelled prior to completion or delivery.

An Australian case in point was on 6 August 2013 when Virgin Australia passengers were delayed and experienced flight cancellations after the Sabre booking and check-in system used by the airline crashed worldwide. The Sabre system outage also affected other airlines worldwide, including Etihad, American Airlines, Alaskan Airlines, and JetBlue.

Sabre's website stated its technology connected 350,000 travel agents to more than 400 airlines, 100,000 hotels, 25 car rental companies, 50 rail providers, 13 cruise lines and other global travel suppliers. In January 2013, Virgin moved to the Sabre booking system after its previous system Navitaire suffered from a meltdown which resulted in the system being offline for 11 days. The Sabre system outage cost the airline an estimated $15-20 million in lost earnings (O'Sullivan, 2013).

More recently Australia's biggest department store group, Myers, web site crashed on Boxing Day 2013, its busiest shopping day of the year. Mr Brookes, the outgoing chief of Australia's biggest department store group, said the company was "really disappointed" and apologised that its website suffered technical difficulties and prevented online purchases, but he did not believe there would be any negative impact on profitability even though Myer had invested tens of millions of dollars into improving its website and online sales functions in previous years to cater for online shopping.

Despite investing so much into its technology, Myer has suffered problems during busy periods, including an earlier web site crash in June when heavy customer traffic triggered a website failure half an hour after the start of the annual stocktaking sale (Liew, 2013).

While the value of IT in delivering solutions and applications to the business has certainly increased, the C-level perception of IT as a cost centre has not shifted.

Connolly (2014) reported to CIO Australia that a lack of governance, bad reporting, constantly changing specifications, and security breaches were responsible for some of the most damaging enterprise IT disasters in Australia and overseas in recent years.

Top 10 IT Disasters

According to Connolly (2014), in ascending order the top 10 enterprise IT disasters that have taken place in Australia and abroad in recent years are:

10. BBC Digital Media Initiative

UK broadcaster, the BBC, launched its Digital Media Initiative in 2008 in an attempt to build a digital production system to change the way workers created, used and shared audio and video content. It was halted in mid-2012 after the BBC Trust launched an internal review. The review found that the BBC showed serious a weakness in project management and reporting, and a crippling lack of focus on business change. In total almost £100 million was spent on the entire project. As a result of the failure and losses, the CTO, John Linwood, was fired over the debacle in January of this year.

9. Distribute.IT Hack

In June 2011, an attack on the domain registrar Distribute.IT occurred which also impacted the University of Sydney, NBN Co retail customer, and Platform Networks. Approximately 4800 websites and data from 4000 customers were lost. The attack by a NSW (Cowra) truck driver was so damaging it put Distribute.IT out of business and as a result the company was subsequently taken over by Netregistry Group.

8. HealthSMART Modernisation Program

In mid-2008, the Victorian Government unveiled its HealthSMART program to modernise and replace IT systems across the Victorian public health sector. According to an Auditor-General's report, by October 2013, implementation costs for the ICT system rollout had blown 150% more than the original budget of $58.3 million. The report also suggested that the absence of appropriate controls and effective mitigations at certain sites could pose serious safety risks to patients.

7. MyKi Smart Card System

The Victorian Government's Myki public transport smart card system was plagued with delays and cost blowouts. The system, along with HealthSMART and the Regional Rail Link contributed to around $2 billion in cost overruns for the Victorian Government. Myki was estimated to have cost $1.4 billion alone.

6. State of California's ERP Deployment

Due to huge delays and cost blowouts, the State of California terminated a contract with SAP in February 2014 for the $371 million rollout of ERP software intended to overhaul the state's payroll system. In 2010, SAP was engaged after the original supplier BearingPoint had been terminated. Later in November 2013, California's State Controller filed suit against the SAP. In addition to this, Deloitte Consulting and SAP were also sued by Marin County, California in a separate case related to a software rollout.

5. Healthcare.gov

The Obama administration's malfunctioning Healthcare.gov insurance-shopping website, part of the Obamacare program, went live in October 2013, however, only 30% of its users were able to sign up

for healthcare services. The US government did work hard to fix the system, but by December 2013 government officials said 25% of the applications sent from the site to private insurers contained errors that were caused by the website.

4. Australian Customs Service

An integrated cargo system at Australian Customs Service, which went live in October 2005, was a huge failure. The move to the production of the Imports module of the Customs' project was deemed a failure of corporate governance of IT. The results were catastrophic as cargo was left unprocessed, shipments over a Christmas period were delayed, and although individual parts of the system worked, it failed as a whole despite customs having invested between $200-250 million into the project.

3. Queensland Health's Payroll System

The LATTICE system responsible for paying Queensland Health's 78,000 staff and $210 million in salaries every two weeks was planned to have been replaced. This system replacement was complex and covered 206 individual allowances across 13 awards and 5 industrial agreements. From the outset of the project, it was clear that it was in trouble and between early 2008 and March 2010, IBM Australia, the prime contractor, submitted47 change requests to the government's shared services provider, CorpTech, due to poorly defined business requirements.

The Queensland Auditor-General's report stated that during October 2008detailed planning revealed that the program had been severely underestimated and as a consequence, its revised implementation cost estimates significantly exceeded the original tender proposal.

Despite the problems from the beginning, in 2013 the system went live, which subsequently left thousands of workers unpaid and underpaid for a number of weeks. A total of $120 million was overpaid to more than 61,000 staff and over a period of just 8 years, the eventual cost to the State of Queensland is expected to be $1.2 billion.

2. Office of Personnel Management

For the past 37 years, around 600 staff employed by the Office of Personnel Management in the United States has been processing the retirement papers of US government employees. Since 1977, US government administrators have reportedly paid out more than $100 million in an attempt to automate this paper-based process, however, it proved to be fruitless.

Again between 1987 and 1996, approximately an additional $25 million was spent on another failed system. Later in 1997, an effort to revamp the system using internal resources before hiring contractors was attempted. However, by 2007 the Retirement Systems Modernization Program was not working with a reported 18% success rate during 'stress tests'. According to sources the system apparently "had trouble synthesising information from so many sources and calculations based on so many laws."

Finally in 2008, the system eventually went live before breaking down and being scrapped. In total more than $106 million was spent and the paper-based filing system still remains today.

1. U.K. National Health Service System

The number one project disaster determined by Connolly (2014). . This project was initially launched in 2002 only to be scrapped by the UK Government in September, 2011.

This nine-year debacle under the National Programme for IT was way over budget and years behind schedule due to a number of different issues including technical issues, issues with vendors and constantly changing system specifications.

In early 2012, one of the primary suppliers, CSC, made a $1.49 billion write-off against the botched project. A year later in a 2013 report it was claimed that the failed project had cost UK taxpayers an estimated £10 billion to date with the final bill expected to be "several hundreds of millions of pounds higher".

Why IT Projects Fail

Based on industry norms, less than 50% of IT projects finish on time and on budget. Discussions with experienced CIOs, consultants and project managers indicate there are many reasons for the failure of IT projects; however a number of common conclusions can be drawn:

- **Fuzzy goals:** Many large projects fail because their goals are not clear.
- **Over-optimism:** Salespeople and internal project champions both want their proposal to succeed.
- **Complexity:** Major IT projects have a high degree of complexity due to new technology, the myriad of interfaces with other systems, data conversion, or because project teams have to compete for resources with other projects.
- **Weak 'ownership':** Large projects often have multiple executives, each with slightly different agendas as stakeholders.
- **Governance:** There is a lack of IT governance.

The Standish Group research indicates smaller projects (based on Agile or Waterfall methods) have a much higher success rate (76%) than larger projects that have an average success rate of 10%. Additionally, many industry specialists agree that delivering in small doses produces positive results.

Objective

The focus of this paper is on the role of IT Governance in IT project successes.

The report will specifically examine IT governance relating to the acquisition principle of ISO/IEC 38500 (2008) Corporate Governance of Information Technology. The acquisition principle relates to any IT decisions for new initiatives, and continuation of existing systems and capabilities. This principle relates to the entire lifecycle of an IT investment. An IT investment (either maintenance or new work) is delivered using a project management approach.

The object is to determine the extent of the alignment between attributes relating to project management governance and IT governance's acquisition principle.

Five well known and respected project management (PMBOK, ISO 21500, AS/NZS 8016:2013, PRINCE2 and Agile) governance approaches will be compared to the IT governance acquisition principle.

This research is to investigate the potential relevance of these Project Management approaches for governance with an emphasis on identifying how they can be applied or extended within the context of IT governance and the subsequent development of an IT governance maturity model.

What is ISO/IEC 38500:2008 Corporate Governance of Information Technology?

The ISO/IEC 38500:2008 Corporate Governance of Information Technology standard provides a framework, vocabulary and six principles for good Information Communication Technology (ICT) governance which includes the governance of information technology and communication technology. The six principles are:

1. **Responsibility** Establish clearly understood responsibilities for ICT. management;
2. **Strategy** Plan ICT to best support the organisation's strategy.
3. **Acquisition** Acquire ICT for valid reasons.
4. **Performance** Ensure that ICT performs well whenever required.
5. **Conformance** Ensure ICT conforms to legislation and policies.
6. **Human behaviour** Ensure ICT respects human factors.

It is within the Acquisition principle that IT projects are initialised and implemented. From an IT governance perspective the Acquisition principle requires that the acquisitions are to be evaluated, provided with direction and continuously monitored. Better application of the Acquisition principle would result in more IT projects being successfully completed. Recently to support ISO/IEC 38500:2008 acquisition principle, AS/NZS 8016: 2013 Governance of IT enabled projects, has been released.

AS/NZS 8016:2013 Governance of IT enabled projects

"Most organizations use IT as a fundamental business tool and few can function effectively without it. IT is also a significant factor in the future business plans of many organisations." ((ISO/IEC 38500, 2008).

The objective of (AS/NZS 8016:2013) 'Governance of IT enabled projects' (Standards Australia Limited/Standards New Zealand2013) is to improve the business outcomes of projects that involve investment in new or changed IT capabilities. These projects are often referred to as 'IT enabled

projects' or 'IT projects'. This standard is relevant to both individual IT enabled projects and project programs to achieve business objectives; it is based on.

Spending on IT can represent a significant proportion of an organisation's overall commitment expenditure of financial and human resources. However, it is often the case that a return on this investment is not fully realised and therefore the adverse effects on organisation's strategic and operational success can be significant.

Governance of IT, including significant investments in IT, is part of sound corporate governance. Examples of IT investments include hardware, software, mobile devices, apps, cloud services, digital and social media. This standard is intended to be used by the governing bodies and executive managers of organisations, including owners, board members, directors, partners and senior executives. Governance in this context is not IT management, but it is supported by an organisation's management system.

To achieve an improvement in business project outcomes that involve new or changed IT capabilities, the standard proposes a framework comprising of definitions, principles and a model for effective governance of IT projects.

Benefits of using AS/NZS 8016:2013 Governance of IT enabled projects

The application of AS/NZS 8016:2013 assists the governing body in balancing strategic value opportunities and risks arising from IT investments. An organisation needs to establish and maintain a good governance framework consisting of strategies, policies, decision-making structures, and accountabilities to deliver improved return on IT investments.

Good IT project governance includes:

- Prioritising projects of greatest value to the organisation's objectives, ensuring management ownership of, participation in, and control of organisational change.
- Understanding the requirements for change management.
- Applying thorough management processes throughout the project lifecycle.
- Conforming diligently to all obligations.

The size, complexity and nature of the organisation will dictate the actual governance framework that needs to be established. The main elements of the governance framework should be as shown in Figure 1 – Key Elements of a Governance Framework for IT.

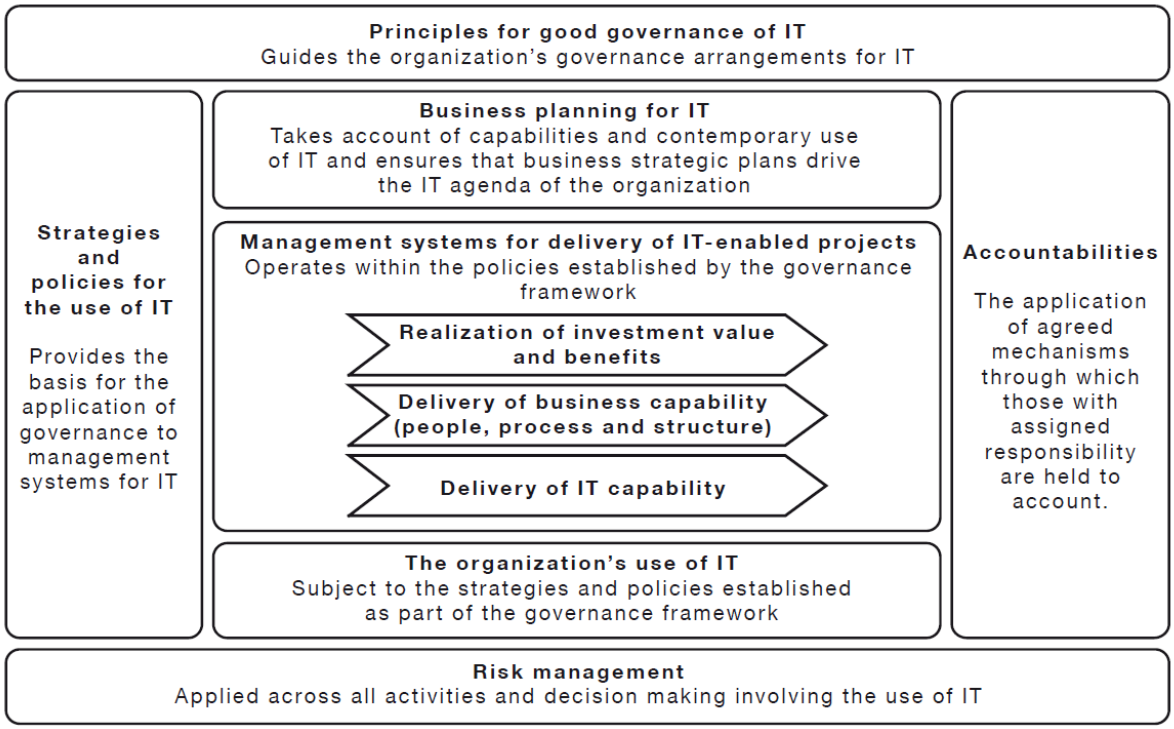


Figure 1 - Key Elements of a Governance Framework for IT

Source: AS/NZS 8016:2013 Governance of IT enabled projects

The governance framework elements in Figure 1, are equally relevant to individual projects and programs of projects.

The two most prominent systems for Project, Program and Portfolio management are Project Management Body of Knowledge (PMBOK) and Projects in Controlled Environments (PRINCE2). Of these, PMBOK is described as a framework and PRINCE2 as a methodology.

AS/NZS 8016:2013, PMBOK and PRINCE2 all advocate a governing body needs to be established. This governing body will ultimately be accountable for the success of all the projects the organisation undertakes. In addition, this governing body may delegate aspects of the governance to the organisation's managers, however, it must be highlighted that all accountability will remain with the governing body.

IT project governance model

The Governance of IT Projects Model (AS/NZS 8016:2013) uses the same three main tasks as the ISO/IEC 38500 model proposes:

- **Evaluate** - Make strategic judgments regarding current and future use of IT.
- **Direct** - Use of plans, strategies and policies to ensure IT investments meet business objectives.
- **Monitor** - Routinely examine project performance.

Figure 2 – The Governance Model for IT Projects shows how IT projects should be governed based on ISO IEC 38500-2008.

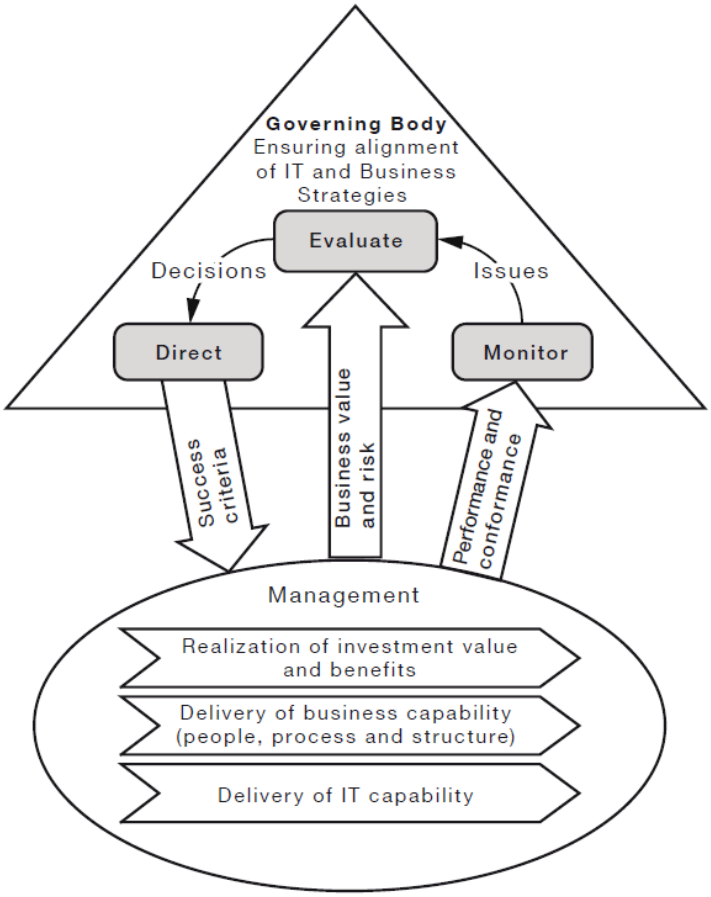


Figure 2 - Governance Model for IT Projects

Source: AS/NZS 8016:2013 Governance of IT enabled projects

Evaluate

Evaluation of project investments from a strategic and business value perspective should be undertaken at all levels continuously. The governing body or its delegates should ensure the proposed project management methodology is appropriate for each project, such methodologies may include:

- PRINCE2
- Agile
- Scrum
- Waterfall
- Extreme
- V Model.

Direct

Taking into consideration the size of the organisation, the number of IT projects and the organisation's appetite for risk, the governing body should establish a system of control and oversight of IT projects.

The system should include:

- Policies
- Processes
- Roles
- Project selection criteria
- Benefits analysis
- Risk management.

Many of the methodologies previously referred to will also include aspects of direction.

Monitor

The governing body should measure the performance of the processes for governance of IT projects. The key considerations of the monitoring task should include:

- Milestones
- Issues identification
- Interdependencies between projects
- Change management
- Stakeholder engagement
- Resource management
- Assumptions
- Risk management.

Six principles for good governance

ISO/IEC 38500 six principles express preferred behaviour to guide decision making.

The six principles are:

1. **Responsibility** - Individuals and groups responsible for achieving business value from IT
2. **Strategy** - Capability of IT projects to innovate and align with business strategy
3. **Acquisition** - Cost benefit analysis to ensure good decision making
4. **Performance** - Extent to which IT project service and quality outcomes meet business needs
5. **Conformance** - Extent of compliance with legislation, regulations, standards, policies, and procedures
6. **Human behaviour** - Development of organisational culture through motivation and trust

AS/NZS 8016:2013 refers to the acquisition principle and how it applies to the governance of IT projects however the standard does not propose how, when or by whom the principles would be implemented.

Causes of project governance problems

The causes of project governance problems are all interrelated. There is generally no single cause of governance failure (Garland, 2009). Governance problems can be categorised as:

- Unclear project governance objectives
- Risk aversion and Organisational structure issues
- Stakeholder and ownership issues.

Other factors such as skills, competencies, personalities, and political environment also contribute to project problems; therefore any project governance framework must also address the above factors.

Project Governance Objectives

Efficient and effective project decision making should be the primary objective of project governance. Problems occur when decision making is reduced by an overemphasis on stakeholder involvement through increasing numbers of forums and committees. This is usually accompanied by increased reporting through excessive organisational and project structures, resulting in poor accountability and timeliness (Garland, 2009).

Risk

Different organisations have individual risk appetites and risk cultures. From a project governance risk perspective, a culture of risk aversion can mean project decisions can be too slow and justifying documentation grows, resulting in slow and poor quality decisions. In the worst case, this can result in ‘paralysis by analysis‘ where continual analysis of every possible decision to reduce potential risk results in no decisions being made at all (Garland, 2009).

Organisational versus Project Structure

Project structures are short term and designed to deliver project outcomes. Organisational structure services the ongoing operations of a business, however, from a governance perspective; project needs cannot be met by organisational structures. A project committee, sometimes known as a project steering committee, project board or project control group, must be established at the project initiation stage. The purpose of this committee is to consolidate key project stakeholders and ensure key decisions affecting stakeholders are made. It is recommended to limit the size of a project committee to 6-8 people. In addition to this, the committee must also have the necessary authority (Garland, 2009).

Stakeholder and Ownership

Key stakeholder support and project ownership is essential for success. This is achieved through the establishment of a project committee. Ownership can be a fundamental failure unless the project is ‘owned’, ‘driven’, ‘championed’ or ‘sponsored’ by the committee (Garland, 2009).

Principles of Effective Project Governance

Garland (2009), suggest that effective project governance can be achieved by the application of four principles that differentiate between stakeholder management and project decision making.

Principle 1

For the project to succeed ensure the right person is appointed as a single point of accountability and responsibility. Choosing the right person will also ensure clarity of leadership, clarity of decision making and timeliness of decision making.

Principle 2

Ownership of the service or asset delivered by the project determines who owns the project. The ownership of the project does not necessarily reside with those delivering the service or asset/project output which places the organisation at the centre of project delivery and ensures the project governance framework maintains a service delivery focus.

Principle 3

Ensure separation of stakeholder management and project decision-making activities, which are two separate activities. This will prevent decision-making forums from becoming clogged with stakeholders, resulting in laboured or ineffective decision making.

Principle 4

Ensure a divide between project and organisation governance structures. This divide will reduce the number of project decision layers and will not assume that project decisions will follow organisational lines of command.

Project Management Governance Model

The application of the four principles proposed by Garland (2009), can guide the establishment of an effective a project governance model.

1. Appointing the right people (Principle 1)

A decision making board consists of four players: Senior User, Project Owner, Senior Supplier, and Project Director.

2. Ownership (Principle 2)

The Senior User is a representative of those who will use the asset or service. The Project Owner represents the core business of the organisation. The Senior Supplier represents those who will deliver either the services to the project or the asset itself. To support the Project Owner, who may not have the time or the project management expertise, is the Project Director (owners/client's representative). Figure 3 – Project Board, shows the combination of these four roles as the key decision making body.

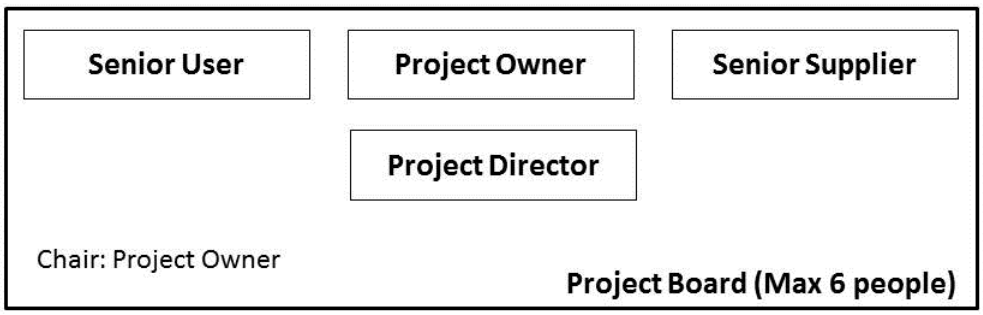


Figure 3 - Project Board

Source: (Garland, 2009)

Figure 3 – Project Board, shows most of the important project stakeholder positions.

3. Separating Stakeholder Management and Project Decision Making (Principle 3)

The Project Manager and their project team are answerable to the Project Board. Each of the board members has the responsibilities of stakeholder and relationship management. While the Project Board does have some power, it does not have the authority to make major investment decisions.

Garland (2009) refers to this group as the Investment Decision Group; PRINCE2 calls them the Corporate or Programme Management (PRINCE2: Managing Successful Projects with PRINCE2).

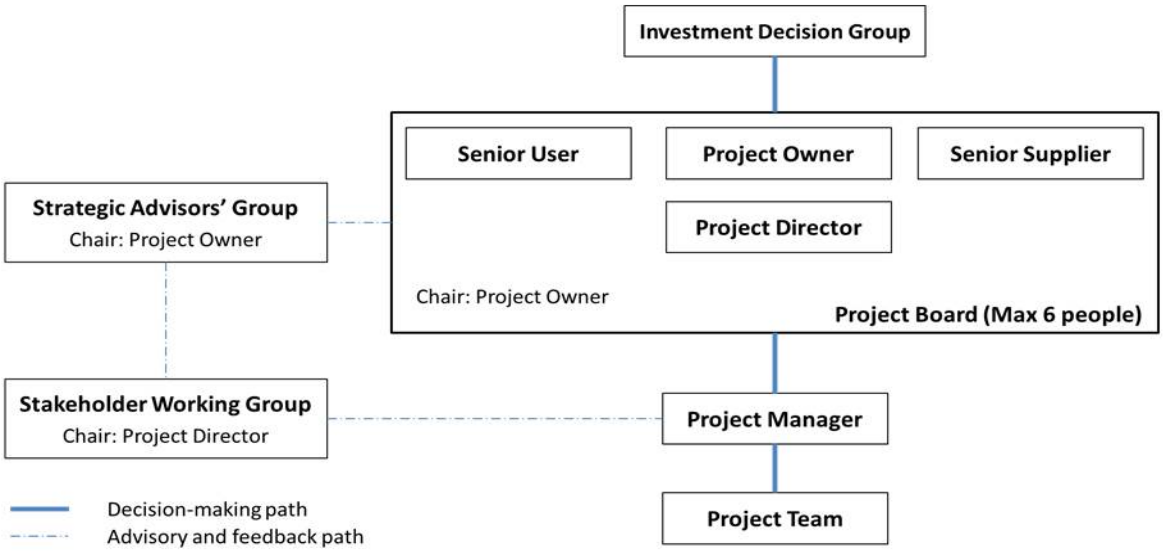


Figure 4 - Project Governance Model

Source: (Garland, 2009)

4. Ensure a divide between project and organisational structures (Principle 4)

Figure 3 excludes other potential stakeholders who, according to Garland (2009), can be divided into two groups. The first group is referred to as the Strategic Advisor Group consisting of senior stakeholders whose support is critical to project success, but who may not be interested in the technical details of the project. Stakeholders who are interested in the technical details of the project are referred to as the Stakeholder Working Group. The relationship of these two groups to the Project Board is shown in Figure 4 – Project Governance Model.

Figure 4 presents the ideal project governance framework for complex high-risk projects. This model can also be scaled back for lower risk and less complex projects. In order to remain effective when scaling the model back, roles are merged rather than discarded.

The model proposed by Garland (2009) and shown in Figure 4 is very similar to the PRINCE2 model (Figure 5).

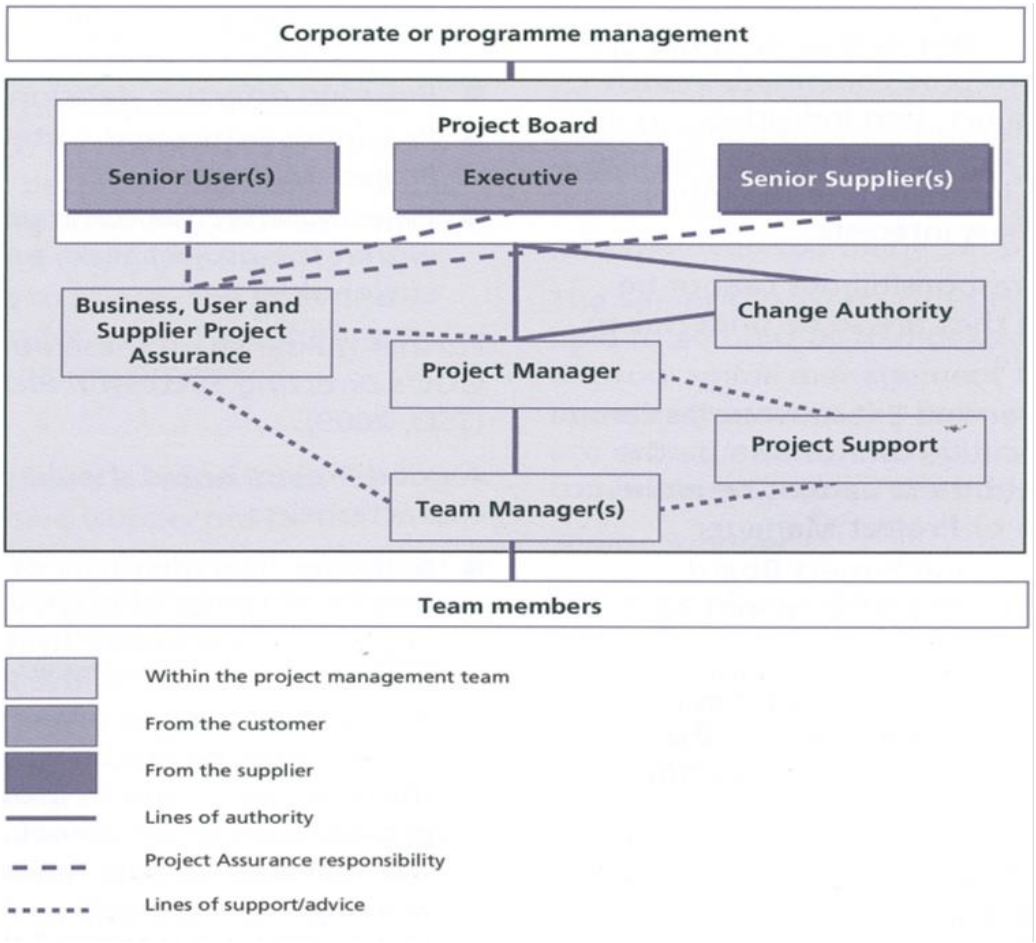


Figure 5 - Project Management Team Structure

Source: AXLEOS 2009, p33

The PRINCE2 model is more detailed and provides more day-to-day project management roles whereas Garland's model recognises that project management skills may be missing from the Project Board and compensates by the addition of the role of the Project Director.

Garland's Model – Roles and Responsibilities

Garland's model also recognises the importance of all other stakeholders with the addition of two stakeholders groups.

The Project Governance roles shown in Figures 3 and 4 – Project Board, are summarised below:

Project Owner:

- Represents the business
- Is from the organisation's business unit that will use the project outcomes
- Has a service delivery focus
- Cannot be outsourced.

Senior User:

- Represents those who will use the final product or service that the project delivers
- Represents those who the project may impact in some way (e.g. operations or maintenance activities)
- May represent an organisation that is contributing to funding of the project
- May be split between two persons if necessary

Senior Supplier:

- Represents the suppliers of services to the project
- May be delivered by an in-house provider, an external service provider, or both
- Must have the ability to commit supplier resources to the project

Project Director:

- Drives the project on behalf of the project owner
- Provides project delivery skill set to the business
- Manages service delivery outcomes for the project owner

Garland (2009) states that implementing a project governance framework is an exercise in business change management; all projects deliver change.

Project, program and portfolio management

The establishment of a sound IT governance process, based on Garland's six principles address some of the risks associated with managing IT projects, particularly as they increase in complexity.

Project, program and portfolio management are equally critical to organisation's success.

Every project should be strategically aligned with corporate objectives. However if there are many projects and corporate layers, the benefits of projects may not be realised. The concept of a program is the interconnection of related projects so they deliver benefits to the overall organisation (Garland, 2009).

A project manager manages the phases and key activities of their project. It also manages the progress of multiple projects and key interdependent phases in line with business guidelines.

Brown (2008) makes a distinction between the role of the project manager and the program manager. The environment of the program manager can be highly complex and vary from managing multiple projects to managing multiple projects in addition to their existing operational responsibilities. In addition to this, they also have accountability for profit or cost targets linked to business strategy. This contrasts with the project manager's role which is to deliver a single project within budget and

schedule constraints which are usually established at the program level. The same principles and a similar model of governance can be also used for project and program governance; however, most programs are generally more financially focussed.

An overarching program structure supports Garland's fourth principle which deals with separating project governance from corporate or organisation governance. The fundamental difference between the two is that a project has a discrete start and finish whereas programs may continue for many years with no defined end date (e.g. providing health services or road maintenance).

'A programme is a temporary flexible organisation structure created to coordinate, direct and oversee the implementation of a set of related projects and activities in order to deliver outcomes and benefits relating to an organisation's strategic objectives. A programme may have a life that spans several years.' (PRINCE2: Managing Successful Projects with PRINCE2, p. 309)

Program management is the aggregation of specific projects within a portfolio to achieve common business objectives higher than those of individual projects, but lower than business strategic objectives. Programs may be created because they contribute to a single business objective or because of client, technical or resource synergies, or a combination of these drivers.

Program managers in most cases 'direct' the activities of project managers within the parameters of the program. Use of the word 'direct' implies that the program manager has a strategic view of the projects that make up the program.

Portfolio management is essentially about investment management, whether that investment is financially defined as in a private sector enterprise or effort and resource based as in a public sector enterprise. Portfolio management involves the aggregation and total visibility of projects in an organisation so that the linkage between vision and strategic direction and project objectives and deliverables are consistent.

Figure 6 illustrates the hierarchy of project, program, and portfolio management. It is at the portfolio and program levels that governance of projects is most prevalent.

Figure 6 also illustrates that portfolio and program management belong within the operations management stream of the organisation. This means that policy generated at the portfolio and programme level will have an impact on operations, both strategically and tactically (Brown, 2008).

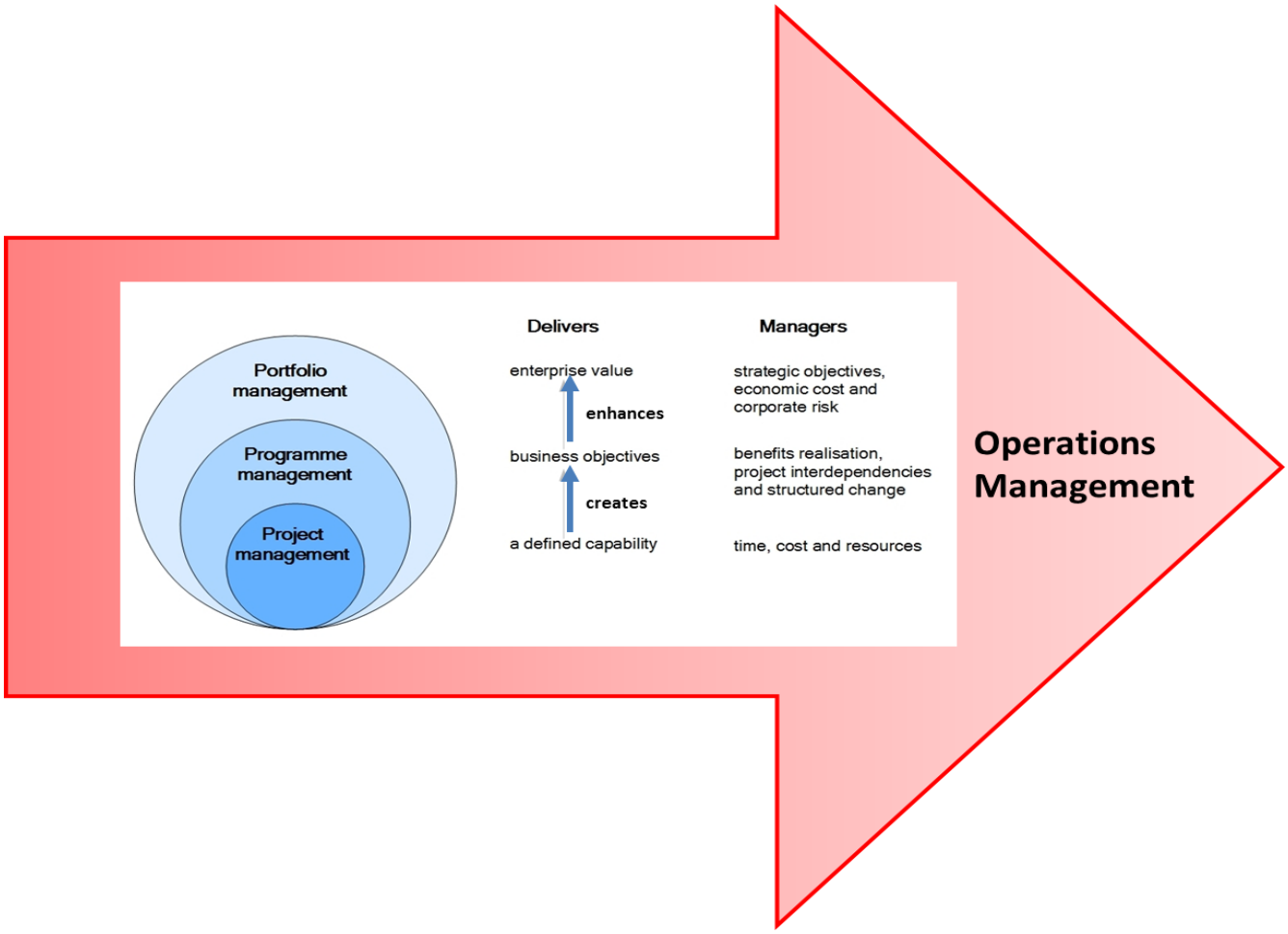


Figure 6 - Portfolio, Programme and Project Management

The increasing use of projects and programs by organisations to achieve business strategy and goals has led to the need for understanding portfolio management. Portfolio management can provide governance structure to minimise overall project costs (Koh & Crawford, 2012). However, to really understand the concept of program and portfolio management, project management must first be understood.

Governance Templates

Garland (2009) provides three useful documents:

- Project Governance Policy (**Appendix 1**)
  This policy can be modified to meet the needs of most organisations or types of projects.
- Terms of Reference and Modus Operandi of Project Governance Bodies (**Appendix 2**)
  These documents can be used as a basis to develop a project governance framework.
- Role Descriptions for project board positions (**Appendix 3**)

**** These documents are provided in full and with the approval of their author, Ross Garland.

PRINCE2

There are limited project management systems that include a governance component, but PRINCE2 does.

PRINCE2 is a controlled project management methodology that can be applied to any project regardless of the project scale, type, organisation, geography, or culture.

The Office of Government Commerce (OGC) continued to develop and improve PRINCE2 until 1 January 2014 (Murray et al., 2009) when AXELOS became the official accreditor for the Global Best Practice, which includes PRINCE2 and ITIL (Hepworth, 2014).

On 1 July 2013, AXELOS was announced as the new joint venture company that the Cabinet Office formed to deliver and commercialise the British Government's portfolio of Best Management Practice accreditation and publishing services including ITIL, PRINCE2 and other PPM products.

PRINCE2 is part of a suite of guidance systems developed by OGC to assist organisations and individuals to manage their projects, programs and services consistently and more effectively. Figure 7 outlines the structure of the OGC best practice guidance set.

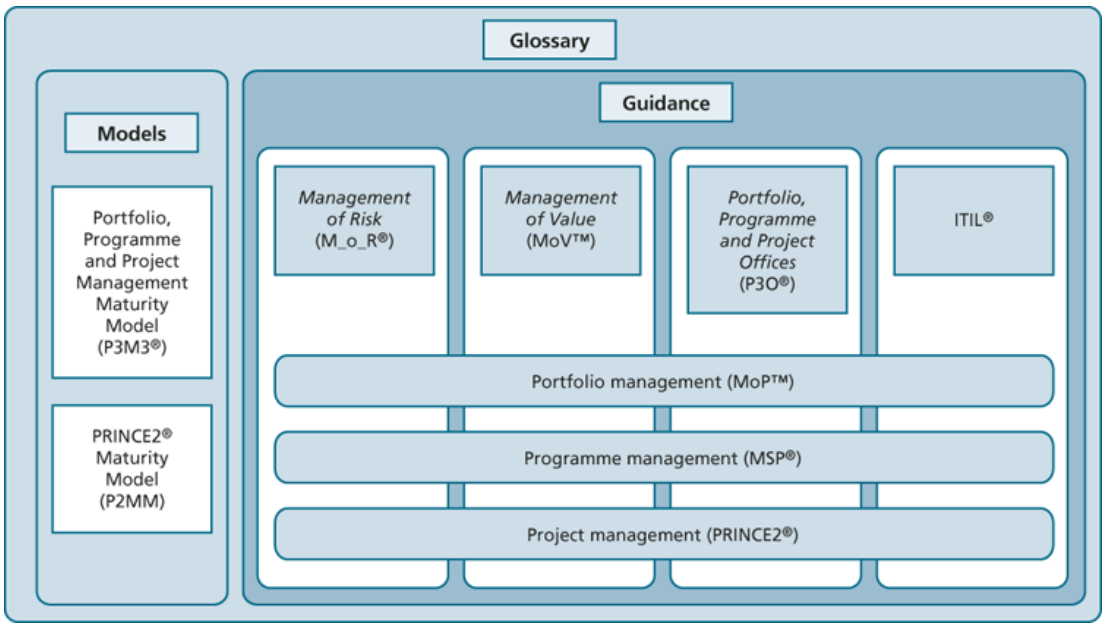


Figure 7 - OGC best-practice guidance

PRINCE2 defines a **project** as:

'… temporary organisation that is created for the purpose of delivering one or more business products according to an agreed Business Case'(PRINCE2: Managing Successful Projects with PRINCE2, p. 3).

Project management is defined as:

'… the planning, delegating, monitoring and control of all aspects of the project, and the monitoring and motivation of those involved, to achieve the project objectives within the expected performance targets for time, cost, quality, scope, benefits, and risks'(PRINCE2: Managing Successful Projects with PRINCE2, p. 4).

PRINCE2 Elements

PRINCE2 is a generic project management methodology.

It addresses the four integrated elements of project management as shown (Figure 8).

1. Principles
2. Themes
3. Processes
4. Project environment.

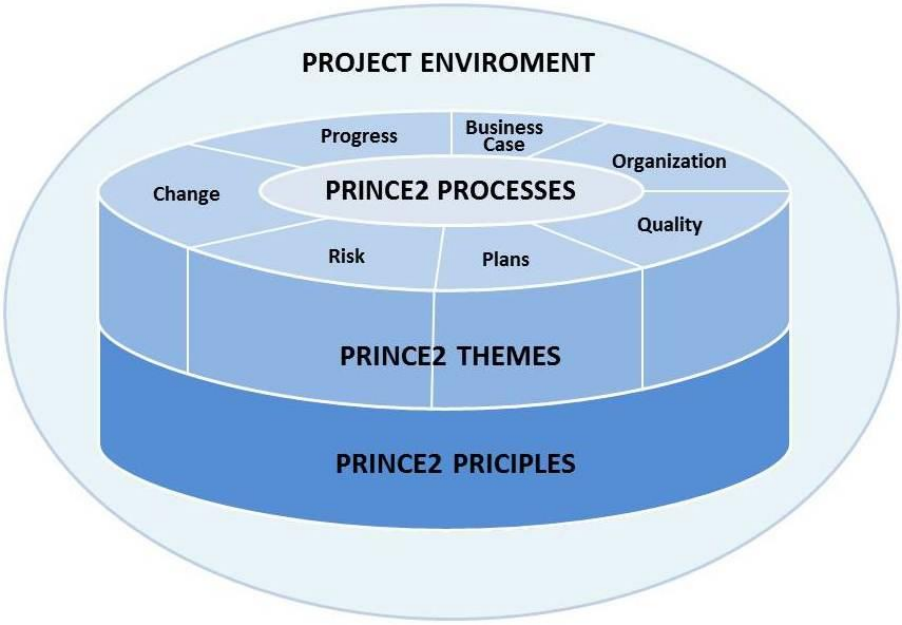
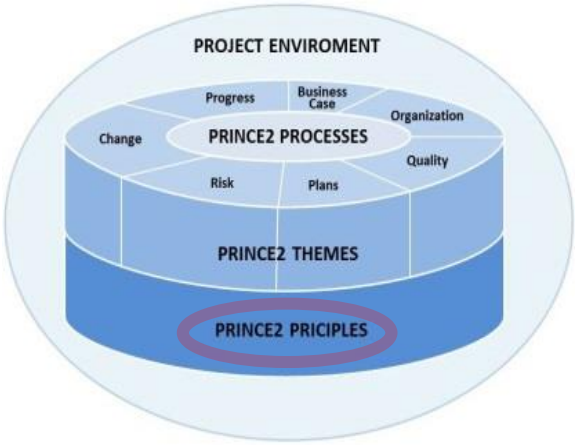


Figure 8 - PRINCE2 Structure

The Principles of PRINCE2



The seven principles on which PRINCE2 is based and shown in Figure 9 originated from lessons learned from projects. The principles can be summarised as:

1. Continued business justification
2. Learn from experience
3. Defined roles and responsibilities
4. Manage by stages
5. Manage by exception
6. Focus on products
7. Tailor to suit the project environment.

Figure 9 - PRINCE2 Principles

The Themes of PRINCE2

PRINCE2 themes (Figure 10) describe aspects of project management that must be addressed continually and in an integrated approach. All seven themes must be applied to a project, but they can and should be tailored according to the scale, nature and complexity of the projects. The themes can be summarised as:

1. **Business Case** – Idea to investment proposal
2. **Organisation** – Structure, roles and responsibilities
3. **Quality** – Quality attributes understood and delivered
4. **Plans** – Series of approved plans
5. **Risk** – Manage uncertainties in plans
6. **Change** – Manage change requests
7. **Progress** – Determine viability of plans

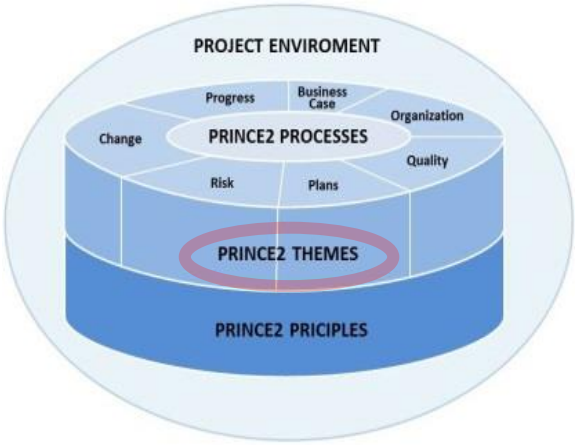
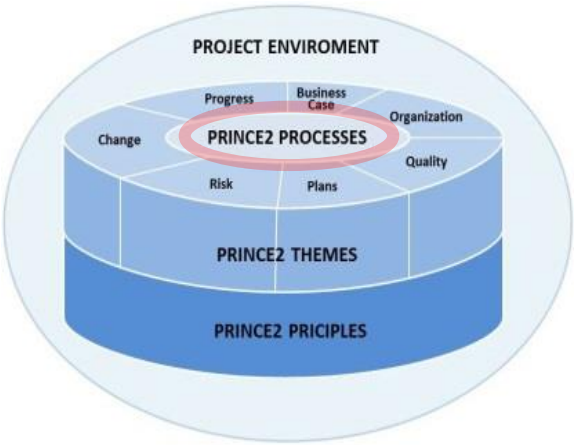


Figure 10 - PRINCE2 Themes

Many existing and proven project management techniques and tools, such as, critical path analysis and earned value analysis, support the application of the PRINCE2 themes.

The Processes of PRINCE2

PRINCE2 is a process-based approach for project management. There are seven processes in PRINCE2 (Figure 11) which provide the set of activities required to direct, manage and deliver a successful project.



1. **Starting** up a project.
2. **Directing** a project.
3. **Initiating** a project.
4. **Controlling** a stage.
5. Managing product **delivery.**
6. Managing a **stage boundary.**
7. **Closing** a project.

Figure 11 - PRINCE2 - Processes

The PRINCE2 processes (Figure 12) shows how each process is used throughout a project's life.

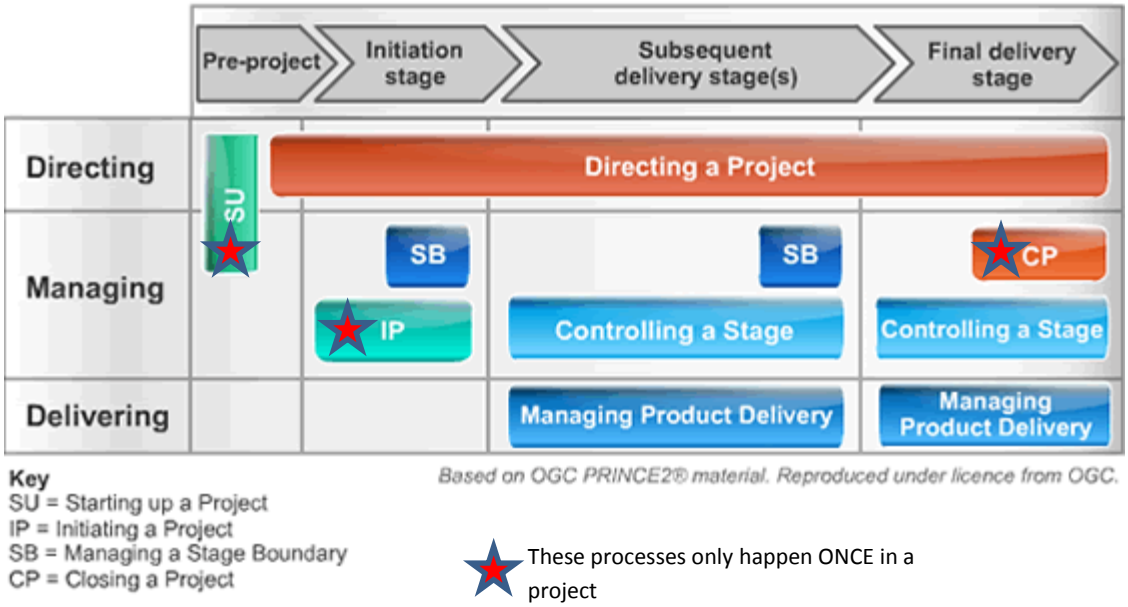


Figure 12 - The PRINCE2 Processes

Research undertaken by Queensland University of Technology (Sargeant, Hatcher, Trigunarsyah, Coffey, & Kraatz, 2010) found the major strengths of PRINCE2 to be:

- Assuring continuing project viability
- Extensive guidance offered on project governance
- Expansion of the tolerance concept to encompass six areas
- Comprehensive definition of roles and responsibilities
- Product-based planning and product-focused delivery
- Delegation of responsibilities to the appropriate level
- Ability to tailor and embed in an organisation

PRINCE2 Maturity Model (P2MM)

Overview of P2MM

Project Management Maturity Model (P2MM) is designed to allow organisations to gauge, by assessment, their maturity in the use of the PRINCE2 project management methods i.e. how well PRINCE2 is embedded in an organisation.

P2MM assessment allows organisations that deliver internal projects to identity their strengths, areas for improvement and build an action plan to improve their effectiveness to deliver the full benefits of using the structured project management approach of PRINCE2.

A possible marketing advantage for organisations that provide a project management service, in addition to the above benefits, they will also be able to provide evidence to their clients and prospective clients of their level of maturity in the use of PRINCE2.

The P2MM (UK, 2010) uses the same structure as the Portfolio and Programme and Project Management Maturity Model (P3M3) from which it is derived using:

- A five-level maturity framework to characterise the levels of organisational maturity
- Seven process perspectives covering key aspects of organisation-wide project management
- Specific and generic attributes for each level of maturity within each of the process perspectives

The five maturity levels are:

Level 1 – Awareness of process
Level 2 – Repeatable process
Level 3 – Defined process
Level 4 – Managed process
Level 5 – Optimised process

The above levels make up the structural components of both P3M3 and P2MM; they can be seen in Figure 13 – Maturity Levels which compares the characteristics of the P2MM with those of the Project Management Maturity Model (PjMM).

Maturity Level	**PRINCE2**	**Project Management**
Level 1 – Awareness of process	Does the organisation recognise projects and run them differently from its ongoing business? (Projects may be run informally with no standard process or tracking system.)	Does the organisation recognise projects and run them differently from its ongoing business? (Projects may be run informally with no standard process or tracking system.)
Level 2 – Repeatable process	Has the organisation adopted PRINCE2, but allowed the method to be applied inconsistently across projects within the organisation?	Does the organisation ensure that each project is run with its own processes and procedures to a minimum specified standard? (There may be limited consistency or coordination between projects.)
Level 3 – Defined process	Has the organisation adopted PRINCE2 and embedded it to align with other organisational processes? Can PRINCE2 be tailored to suit individual projects?	Does the organisation have its own centrally controlled project processes and can individual projects flex within these processes to suit the particular project?

Level 4 – Managed process	Does the organisation obtain and retain specific measurements on its PRINCE2 project management performance and run a quality management process to better predict future performance?	Does the organisation obtain and retain specific measurements on its project management performance and run a quality management process to better predict future performance?
Level 5 – Optimised process	Does the organisation undertake continuous process improvement with proactive problem and technology management for PRINCE2 projects in order to improve its ability to depict performance over time and optimise processes?	Does the organisation undertake continuous process improvement with proactive problem and technology management for projects in order to improve its ability to depict performance over time and optimise processes?

Figure 13 - Maturity Levels

Seven process perspectives of P2MM that are derived from P3MM focus on:

- Management Control
- Benefits Management
- Financial Management
- Stakeholder Engagement
- Risk Management
- Organisational Governance
- Resource Management

The above listed processes can be assessed at all five maturity levels.

Attributes

Embedded within the process perspectives are a number of attributes.

Specific attributes relate only to a particular process perspective.

Generic attributes are common to all process perspectives at a given maturity level; this includes planning, information management, and training and development.

Most organisations have strengths in some areas, but not in all of them. P2MM is designed to acknowledge these strengths as well as highlighting weaknesses. Figure 14 illustrates how an organisation might be viewed from the process perspective (Williams, 2010).

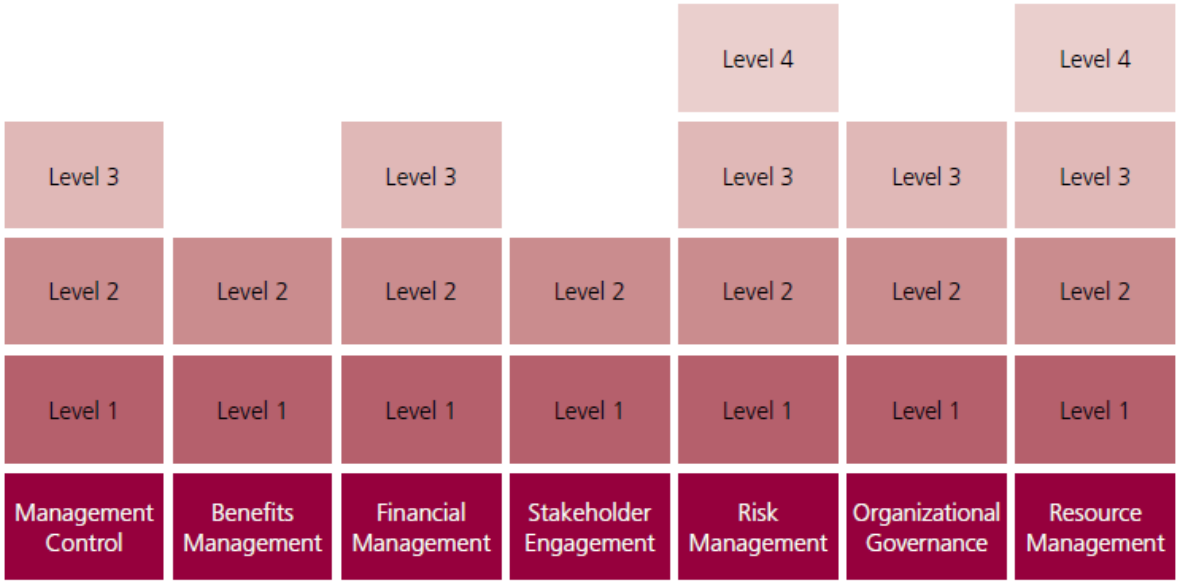


Figure 14 - Example assessment of Process Perspective

Project Management Body of Knowledge (PMBOK)

A major publication on the subject of Project Management knowledge and methods is entitled the Project Management Body of Knowledge (PMBOK) guide, which was published by the Project Management Institute (PMI, 2013).

PMBOK has existed much longer than PRINCE2. PMBOK is a framework, whereas PRINCE2 is a described as a methodology.

Framework is a loose but incomplete structure which leaves room for other practices and tools to be included but provides much of the process required. A framework allows the project team to choose their own processes, and roles for example.

A methodology is a set of principles, tools and practices which can be used to guide processes to achieve a particular goal. PRINCE2 for example is prescriptive providing detailed practices, roles and products to be used to deliver a project.

The Project Life Cycle

The project lifecycle describes the stages or process that projects move through from inception through to completion. This particular lifecycle can be demonstrated graphically as shown below in Figure 15 – Project Life Cycle.

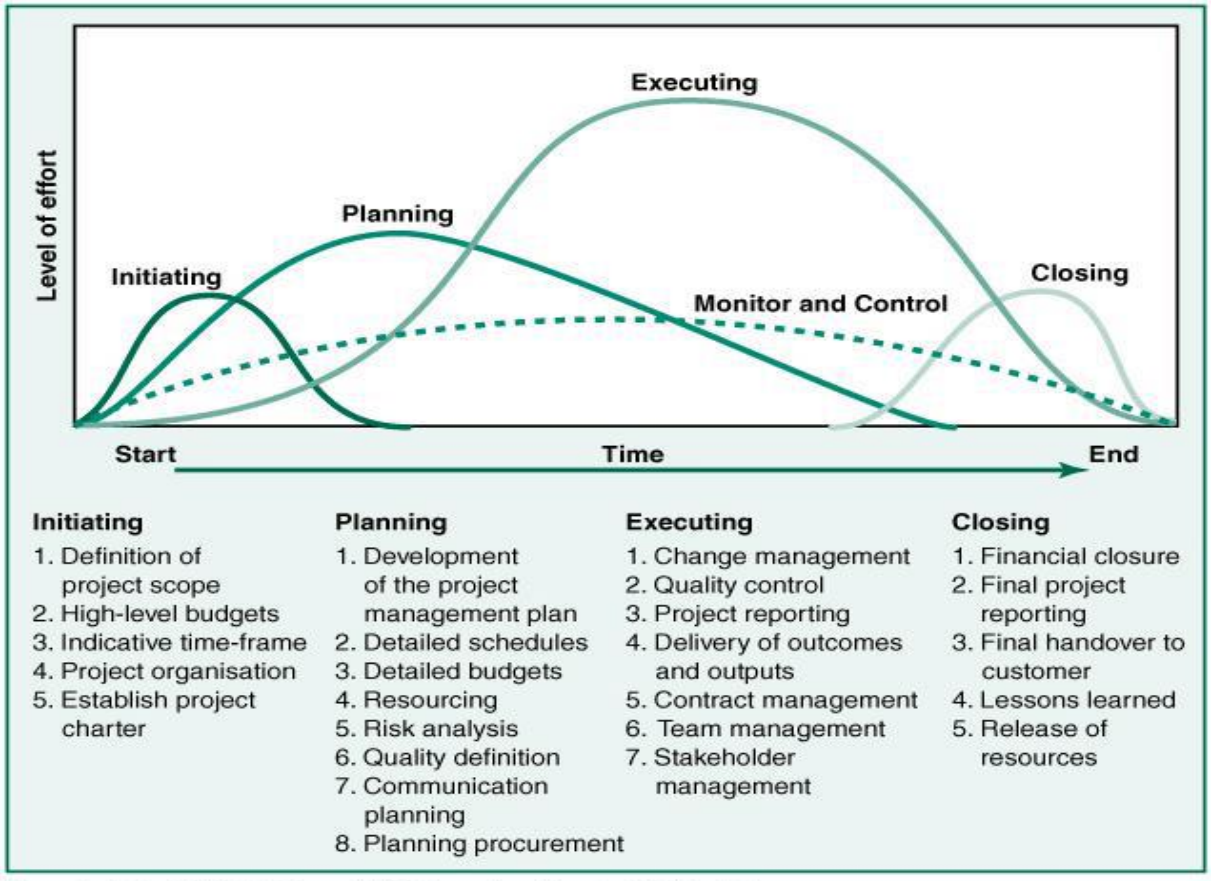


Figure 15 - Project Life Cycle (PMBOK)

Project management knowledge areas

Throughout the lifecycle of the project there are 10 core knowledge areas, which are also referred to as competencies. To attain success, those involved in projects must employ these knowledge areas which include:

Integration	This area covers the complex interactions between all the stages and functions within the project. Integration is usually a key role of the project manager or the program director.
Scope	This area is connected to defining the boundaries of the project and product specification. This area defines what work will be done to achieve the project objectives (inclusions) and what work will not be done (exclusions).
Time	Time relates to the planning and management of project time from start to end, i.e. what is done when, and in what order to complete it 'on time'.
Cost	This deals with estimating the costs for the project, forecasting and developing a project's budget.
Quality	Acceptance criteria, specifications as well as standards, procedures and regulation compliance all fall within the quality knowledge area.
Human Resources	This area looks at the skills required to deliver the project and how the team will be developed and managed. It also includes defining roles and responsibilities for the project team members.
Communications	Communications covers identifying the stakeholder and their information needs within the project. It also refers to the distribution of information and all the aspects of project reporting, record keeping and knowledge management.
Stakeholders	Stakeholders refers to anyone with a vested interest in the project who may or may not be directly involved in the project work or are in any way impacted by the project activities and outcomes.
Risk	Risks within the project are identified, analysed, and actions are taken as agreed. Risk monitoring is an ongoing requirement throughout the project life cycle.
Procurement	The Procurement area identifies what goods and services are to be provided so that the project work can be completed effectively. Procurement can cover simple purchasing through to complex contract management activities.

The below matrix (Figure 16) identifies where each knowledge area is most actively utilised during a typical project lifecycle.

	INITIATION	PLANNING	IMPLEMENTATION		CLOSURE
			EXECUTION	MONITORING & CONTROL	
Integration					
Scope					
Time					
Cost					
Quality					
Human Resources					
Communications					
Stakeholders					
Risk					
Procurement					

Figure 16 - Project Management Knowledge Areas used during Project Life Cycle

The scope area for example, in which the boundaries of the project are defined, is dealt with during the planning stage. It is also active during the monitoring and control stages where there is pressure to change or amend the project's scope that has already been agreed upon.

In PMBOK a project management office (PMO) is created that provides a management structure that standardizes the project-related governance processes and also facilitates the sharing of resources, methodologies, tools, and techniques. The PMO integrates data and information from corporate strategic projects and evaluates how higher level strategic objectives are being fulfilled. The PMO is the natural liaison between the organization's portfolios, programs, projects, and the corporate measurement systems, for example a balanced scorecard (PMI, 2013).

In a way PMO operates as a "Centre of Excellence" and only organisations with sufficient resources can sustain a PMO. These organisations, which are typically larger, would divide their projects into two areas - Programmes and Portfolios. PMBOK does not really provide any detail on how a PMO might operate, only that it should.

PRINCE2 provides valuable guidance related to the externalities of projects, namely governance and benefits, about which the PMBOK is silent. An example of such guidance is to learn how to establish an effective project governance structure and use the Business Case as the primary control over the life of the project (Rankins, 2009).

ISO 21500:2012 Guidance on project management

ISO 21500:2012 is a project management framework and was released in September 2012. However, quite a bit earlier in 1983, volunteers from the Project Management Institute (PMI) first gathered to distil the project management body of knowledge, and consequently created the first PMBOK Guide.

ISO 21500 provides a high-level description of concepts and processes that are considered to form good practice in project management.

ISO started with ISO 10006 titled 'Quality Management Systems' - Guidelines for quality management in projects. ISO 10006 was originally published in 1997 and then later updated in 2003. But it has not gained popularity equal to ISO's norm of quality of the series 9000 nor as the World leading project management standards like PMBOK Guide or Prince2. More recently the International Organization for Standardization released ISO 21500:2012 Guidance on project management

There is in fact very little difference between ISO 21500:2012 and PMBOK. .The following information can be used as a comparison:

Process Management Processes

The PMBOK Guide was the basis from which ISO 21500 was created. The next sections contain comparison of PMBOK Guide and ISO 21500.

ISO 21500 divides project processes into five process groups. Table 1 shows the comparison with PMBOK project processes in a project's life cycle.

Project Management Process Group	
ISO 21500	**PMBOK Guide**
Initiating	Initiating
Planning	Planning
Implementing	Executing
Controlling	Monitoring and Controlling
Closing	Closing

Table 1 - ISO 21500 and PMBOK Guide Process Groups comparison

The differences between these two standards' processes are minimal. In fact the only 'real' difference is the name change.

Subject Groups or Knowledge Areas

PMBOK Guide's knowledge areas have been renamed in line with subjects in ISO 21500. Their comparison is found below in Table 2.

Project Management Subject Group Knowledge Areas	
ISO 21500 - Subjects	**PMBOK Guide – Knowledge Areas**
Integration	Integration
Stakeholder	Stakeholder (Added in Edition 5 - 2013)
Scope	Scope
Resource	Human Resources
Time	Time
Cost	Cost
Risk	Risk
Quality	Quality
Procurement	Procurement
Communication	Communication

Table 2 - ISO 21500 Subjects and PMBOK Knowledge Areas

The only real noticeable difference is that the Human Resources Knowledge area has been renamed to Resource to cover both types of resources - human and other project resources.

When investigating the subject or knowledge areas, the main difference found is that ISO 21500 does not provide a description of tools and techniques. The description of each process in ISO 21500 consists of a general description and a table containing primary inputs and primary outputs. ISO 21500 descriptions are substantially shorter than those of PMBOK Guide (Rehacek, 2014).

The PMBOK Guide and the ISO 21500 standards are very close and present a set of processes that have been similarly organised into a project management stage and a project management topic. The ISO standard is more than 40 pages long and is only limited to the introduction of the processes, their

inputs and their outputs. With more than 500 pages, the PMBOK Guide describes the project management processes, their inputs, their outputs and also the associated tools and techniques.

From a project management governance perspective, PMBOK provides a far more detailed direction for managing projects than ISO 21500 despite both having a very similar structure.

Agile Project Management

Non-agile development is a straight line approach to software development - Design, build, test, and then release.

Agile software development is a group of software development methods based on iterative and incremental development, where requirements and solutions evolve through collaboration between self-organising, cross-functional teams. It stimulates adaptive planning, evolutionary development and delivery, a time-boxed iterative approach, and it also encourages a quick and flexible response to change. It is a conceptual framework that promotes foreseen tight iterations throughout the development cycle.

Non-Agile

1. Waterfall (Traditional)
2. V-Model
3. Rapid Application Development (RAD)
4. Joint Application Development/ Design (JAD)
5. Critical Path Method (CPM)
6. Critical Chain Project Management (CCPM)
7. Event Chain Methodology (ECM)
8. Benefits Realisation Management
9. EVO (Evolutionary) Project Management
10. Spiral
11. Process-based Improvement Models (CMMI, SPICE)
12. Unified Process (UP)/ Rational Unified Process (RUP)

Agile Software Development

13. Scrum
14. Scrum-ban
15. Crystal Methods
16. Lean Development (LD)/ Six Sigma
17. DMAIC
18. Dynamic Systems Development Method (DSDM)
19. Extreme Programming (XP)
20. Feature Driven Development (FDD)
21. Adaptive Project Framework/ Adaptive Software Development

The project management approaches listed above are explained in more detail in Appendix 4.

IT Project Management Approaches

There are many frameworks and methodologies to support the management of IT projects.

PMBOK and PRINCE2 are the most popular approaches to project management. PMBOK is used primarily for projects that have a physical product or clear service that is being produced. PRINCE2 is more focussed on projects of an administrative or non-physical product, such as a web site or database. However, PRINCE2 can also be used for physical projects such as roads and buildings.

The Agile and Non-Agile project management approaches outlined previously are essentially developed for delivery of IT projects and some of the Agile approaches have similarities to PRINCE2.

Project Management Approaches vs Project Governance Attributes

Project Governance Attributes listed below in Table 3 are derived from the Portfolio, Programme and Project Management Maturity Model (P3M3) developed by Office of Government Commerce in the United Kingdom (OGC, 2010).

Project Governance Attributes	Project Structure	Financial Control	Benefits Management	Stakeholder Management	Risk Management	Organisational Governance	Resource Management	Business to Project Alignment
IT Governance Attributes	Responsibility Acquisition Process Strategy	Performance Risk	Performance Acquisition Strategy	Human Behaviour	Conformance Risk	Responsibility Human Behaviour Process Strategy	Responsibility Acquisition Human Behaviour	Performance Acquisition Strategy Process
PMBOK	3	4	2	4	5	1	4	2
ISO 21500	2	3	3	3	4	1	3	2
AS/NZS 8016:2013	2	3	3	3	3	2	3	2
PRINCE2	4	4	5	4	3	4	2	4
Agile	2	4	2	4	2	2	2	3
Median Score	**2**	**4**	**3**	**4**	**3**	**2**	**3**	**2**

Rating	
1	Very Low
2	Low
3	Medium
4	High
5	Very High

Table 3 - Project Management Frameworks vs Project Governance Attribute

The IT Governance Maturity attributes are derived from ISO/IEC 38500:2008 Corporate Governance of Information Technology.

The top two rows of Table 3 show an alignment between attributes relating to project management governance and IT governance.

Table 3 – Project Management Approaches vs. Project Governance Attributes then provides a comparison between eight attributes of good project governance and five of the most common project management frameworks/methodologies. The eight dimensions that are measured for IT governance are also displayed and aligned with the eight project governance attributes.

The scores shown on the Likert scale (1 to 5) are based on this paper analysis and the author's experience as a project manager.

The following paragraphs provide a brief summary of the comparison between Project Management Governance attributes and Project Management approaches.

ISO 21500, AS/NZS 8016 and Agile are as low for the project governance attributes of project structure. PMBOK is only slightly better in its project structure. However, PRINCE2 has a well-developed structure for managing projects. The project governance attribute of project structure corresponds to four IT governance maturity dimensions of responsibility, acquisition, process, and strategy.

The project governance attribute of financial control is handled reasonably well by all five project management approaches.

PMBOK and Agile are rated low for the project governance attribute of benefits management. ISO 21500 and AS/NZS 8016 are rated medium. In contrast, PRINCE2 manages benefits management extremely well.

Stakeholder management for all five project management approaches is either rated as medium or high.

PMBOK manages its project risk very well; similarly ISO 21500 has a high rating for risk management. AS/NZS 8016 and PRINCE2 are rated as having a medium rating, while Agile has a low risk management approach to projects.

From an organisational governance perspective, PMBOK and ISO 21500 are rated very low while AS/NZS 8016 and Agile are rated low for organisational governance. On the contrary, PRINCE2 is rated significantly higher when it comes to its approach to organisational governance.

PMBOK has a high rating for resource management. PRINCE2 and Agile are rated low for resource management and ISO 21500 and AS/NZS 8016 are marginally higher performance in resource management.

Except for PRINCE2 that is rated high for business to project alignment, the other four project management approaches are low to medium.

Project Governance Attributes Median Score

A review of the Median Scores shown on Table 3 for the project management governance attributes are displayed below in Table 4. The median scores provided either a Low (2), Medium (3) or High (4) rating. The focus of further study should therefore be on Project Management governance attributes that had scores of Low and Medium.

Project management governance attributes that rated low were; Project Structure, Organisational Governance and Business to Project Alignment. Similarly attributes that rated medium were Benefits Management, Risk Management and Resource Management.

Corresponding the IT Governance attributes that aligned with low rating project management governance attributes were; Responsibility, Acquisition, Process, Strategy, Human Behaviour and Performance. Also similarly the attributes of Conformance and Risk aligned as a medium rating.

Table 4 summaries the attributes ratings and their alignment between project management governance and IT governance.

Project Governance Attributes		IT Governance Maturity
Rating - Low	**Rating - Medium**	
Project Structure Organisational Governance Business to Project Alignment ----------	 ----------	Responsibility Acquisition Process Strategy Human Behaviour Performance ----------
	Benefits Management Risk Management Resource Management	Conformance Risk

Table 4 - IT Governance Maturity focus areas

Table 4 – IT Governance Maturity areas indicate that the areas the IT governance maturity model should focus on for "Acquisition Principle" are:

- Responsibility
- Acquisition
- Process
- Strategy
- Human behaviour
- Performance

Of secondary concern is:

- Conformance
- Risk

Conclusion

Acquisition principle is a term used in the standard ISO/IEC 38500:2008 Corporate Governance of Information Technology.

The Acquisition principle involves evaluating, directing, and monitoring proposed IT investments that support business operations and ensure organisations capabilities are fulfilled whilst balancing risks and value for money.

IT investments are delivered or should be delivered using a project management approach. Thus focusing on project management governance directly reflects the level of application of the IT governance Acquisition Principle.

Existing governance approaches available to support IT Governance's Acquisition principle are inadequate.

A very high percentage of IT projects fail. Research indicates the higher the project cost the higher the potential for failure. IT projects whether for new or maintenance activities are the main delivery mechanism of the Acquisition principle.

Causes of project governance problems and therefore project failure are interrelated; there is generally no single cause of governance failure. However, governance problems can generally be attributed to:

- unclear project governance objectives
- aversion to risk
- organisational structure issues
- stakeholder and ownership issues.

Other factors such as skills, competencies, personalities and political environment contribute to project problems and any project governance framework must also address these issues.

Expenditure on IT can represent a significant proportion of an organisation's expenditure of financial and human resources. To improve the delivery success of IT projects, AS/NZS 8016:2013 has recently been released.

Governance of IT, including significant investments in IT, is part of sound corporate governance. IT investments include projects for hardware, software, mobile devices, apps, cloud services, digital and social media. AS/NZS 8016:2013 which supports and is based on ISO/IEC 38500:2008 assists the governing bodies, to balance strategic value opportunities and risks arising from the investments in IT.

The two most prominent systems for Project, Program and Portfolio management are Project Management Body of Knowledge (PMBOK) and Projects in Controlled Environments (PRINCE2). Of these, PMBOK is described as a framework and PRINCE2 a methodology. PRINCE2 is far more advanced in project governance compared to PMBOK. Based on an evaluation PRINCE2 rates most highly in project management governance and therefore supports IT governance the best.

The rating of the IT Governance attributes indicates that the areas the IT governance maturity model should focus on for “Acquisition Principle” are:

- Responsibility
- Acquisition
- Process
- Strategy
- Human Behaviour
- Performance

Appendices

Appendix 1 – Project Governance Policy

**** This document is provided in full and with the approval of their author, Ross Garland.

The following is an example of a project governance policy that can readily be modified to meet the needs of most organisations.

Overview

The policy addresses the project governance arrangements for all projects undertaken by (insert organisation's name). Its primary focus is high-risk projects as determined by (the organisation should have critter or possibly a high-level project risk assessment model that determines the risk of any particular project) and the governance framework is designed to support such projects. The policy also addresses lower-risk projects by enabling flexible governance arrangements.

Applicability

This policy is applicable to all projects. Projects that are therefore covered by this policy include (select as appropriate for the organisation: asset and non-asset solutions, change management projects and ICT and policy projects). It shall be adhered to by all employees, as well as by consultants and contractors working for the organisation. This policy is not applicable to non-capital or operational activities. It is not applicable to projects that are currently (at the time of approval of this document) being implemented or constructed.

Definitions

(Add other definitions as necessary. Any role or body that appears in the framework will need to be defined.)

Accountable Accountable means answerable to your superior.

Investment decision group *(This group normally already exists within an organisation and often does not need to be separately constituted. It may be known as a budget committee.)* The investment decision group makes the major investment decisions on a project.

Key project documentation The key project documentation is: *(adjust to suit, naming conventions and needs of the organisation. Each organisation should identify a family of documents that must be produced for each project undertaken. If not, then individual project boards should identify these documents.*

- Strategic business case;
- Preliminary business case;
- Final business case;
- Procurement strategy;
- Project completion report
- *(Add other documentation as necessary.)*

Project A project is an undertaking of fixed duration created to deliver a new, enhanced or modified service for the organisation.

Project board The project board is a committee responsible for directing the project, although a number of smaller projects could come under the same umbrella of a single project board. The core members of the project board are the project owner, senior supplier, senior user and project director. Others may be invited to attend by the project owner.

Project director The person who manages the project owner's interests in the project on a day-to-day basis.

Project manager The nominated person who leads the project team and is assigned the authority and responsibility for managing the project within the constraints of scope, budget, schedule and quality as defined by the project owner.

Project owner The person accountable for the success of the project and the chair of the project board.

Senior user The person(s) who represents the interests and viewpoint of users on the project board and supports the project owner on directing the project.

Senior supplier The person(s) who represent the interests and viewpoint of suppliers on the project board and supports the project owner in directing the project.

Strategic advisors' group A group comprised of senior advisors whose role is to provide advice and support to the project owner and project board and to monitor and report on the alignment of the project with their organisation's needs.

Stakeholder working group A group comprised of technical advisors whose role is to provide advice and support to the project manager and project team on technical matters that have an impact on their own organisations.

Project governance framework for high-risk projects

(Organisation's name) will manage high-risk projects in accordance with the following framework. Further detail on the operation of the project governance framework and policy is contained within the project governance procedure *(a document that provides greater detail around the operation of the project governance arrangements)*. The project governance structure shall generally be in accordance with that shown in Figure 15 – The project governance structure.

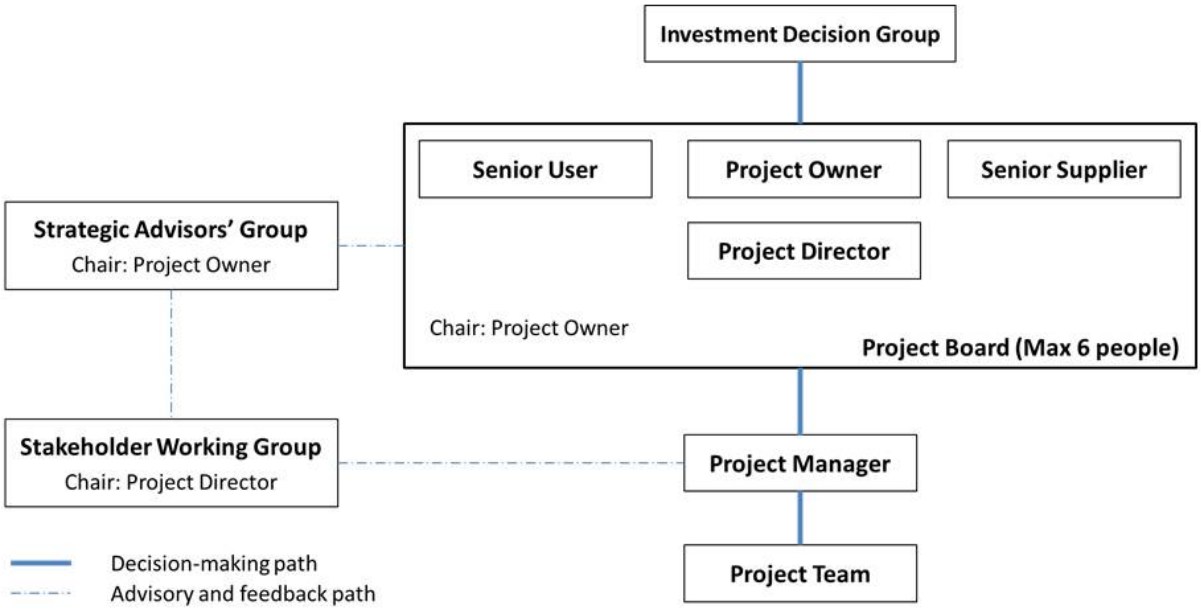


Figure 17 - The project governance structure

Role of the project board

All high-risk projects shall have a dedicated project board that shall operate in accordance with this framework. The project board is chaired by the project owner and should ideally contain no more than six people, to maintain decision–making efficiency.

The project board is responsible for directing the project. In discharging this responsibility it will approve the key project documentation and work to resolve issues escalated by the project manager and project director.

Role of the investment decision group

The investment decision group has the following responsibilities:

- approve funding of the development of a financial business case for the project;
- approve funding of the project in the project on accordance with the funding requirements of the final business case;
- approve major variations to funding;
- address and resolve issues raised by the project board
- address and resolve matters of policy raised by the project board

Role of strategic advisors' group

The strategic advisors' group represents key stakeholders who have a valid interest in the project. It is chaired by the project owner. The strategic advisors group has the following responsibilities:

- review and provide input to project documentation;
- provide advice to the project owner;
- raise issues that have an impact on their organisations involvement in the project;
- keep their host organisations or departments apprised of project developments.

Role of stakeholder working group

The stakeholder working group represents the interests of stakeholders operating at the technical level. It is chaired by the project director or project manager. The stakeholder working group is responsible for monitoring project technical developments to ensure they remain consistent with their own organisations or departments requirements.

Roles and responsibilities within the project governance framework

Project owner

The project owner is the person accountable for the success of the project and owns the service whose delivery the project will facilitate and the project business case. All projects, irrespective of their assigned risk level, shall have a single nominated project owe r. The project owner has the following responsibilities:

- determines the composition of and chairs the project board;
- chairs the strategic advisors group;
- owns the project budget;
- appoints the project director;
- provides direction to the project director and project manager.

Project director

The project director supports the project owner and ensures the project owner's business needs are being met. The responsibilities and delegations of the project director are determined by the project owner but will normally encompass the following:

- chairing of the stakeholder working group;
- assisting in establishment t of the project team;
- assisting the project owner on stakeholder management;
- acting as the main point of contact between the project manager and *(organisations name)*;
- establishing client reporting arrangements;
- managing business resources.

Senior supplier

The senior supplier is the senior representative of the project's suppliers and provides their perspective and expertise. The senior supplier is responsible for:

- ensuring the necessary supplier resources are committed to the project;
- advising and informing the project board of supplier issues;
- ensuring the quality of outputs and products provided by suppliers.

Senior user

The senior user represents the end users of the delivered service and promotes their concerns and interests. The senior user is responsible for:

- representing the interests of users;
- establishing and chairing user groups where required;
- negotiating and developing user requirements and other user documentation; and
- identifying and committing user resources for the project.

Project manager

The project manager is accountable to the project owner for managing the delivery of the project within the constraints of scope, budget, schedule and quality that are defined by the project owner. The project manager is responsible for:

- planning and managing the necessary activities to enable the project to be delivered with the above constraints;
- appointing project team members and is supported in this by the project owner and project director.

Project governance framework for medium-risk and low-risk projects

This section describes project governance arrangements for medium-risk and low-risk projects.

All projects are required to have a single nominated project owner. All projects must have a project board; however, a single project board can encompass more than one project. In such cases the membership of the project board must reflect the needs of the project and, in particular, the project owner must be chosen on the basis of representing the business or service needs that the project will deliver. The need for a project board, and whether that project board is dedicated to that project, is determined by the project owner. A single project director may act as such for a number of projects. On medium-risk and low-risk projects there may be no need for a project director and the project owner may also fulfil the role of project manager and/or senior user. The decision on combining project governance roles is made by the project owner.

The need for the strategic advisors group and stakeholder working group is dependent upon the number of stakeholders and the complexity of stakeholder relationships. Only quite complex projects are likely to have the need for a stakeholder working group. Smaller projects with fewer stakeholders may not require a stakeholder advisory group for the management of stakeholder needs where the project owner can fulfil that role on an ad hoc basis.

Policy approval

This policy was approved by *(insert name of approving body)* on *(date)* and becomes official policy on *(date)*.

Policy owner

(Insert name and position of the owner of the policy. The policy owner will normally be a corporate level group or perhaps a programme office if such an office sits across all projects in the organisation.)

Related policy and procedures

The following policies and procedures are related to this policy:

- *(Add as required.)*

Appendix 2 – Terms of Reference and Modus Operandi of Project Governance Bodies

**** This document is provided in full and with the approval of their author, Ross Garland.

The following provides high-level terms of reference and modus operandi for the various bodies that make up the project governance model. These may prove useful as a basis for organisations developing their own project governance framework.

Project board

Establishment

The project board is the key decision-making body of the project and is established at the commencement of the project. Members are appointed by the project owner, possibly with the assistance of programme management.

Membership

The members of the project board are:

Project owner: the project owner is accountable for the success of the project and chairs the project board. The project owner is the owner of the business service, the delivery of which will be facilitated by that project. The project owner owns the business case and has project budget responsibility.

Senior user: The senior user represents the interests of business, operational and maintenance users. This role is responsible for the definition of user requirements and for ensuring the project delivers to those requirements. The senior user role may also represent senior managers who have a major interest in the project ad who activities will be affected by the project. If there are multiple sources of funding for the project, a representative of a major funding body may fulfil this role.

Senior supplier: The senior supplier represents the interests of those supplying services to the project and is primarily responsible for the delivery of the project's assets. The holder of this role any change as the project moves from the business case phase to the construction phase.

Project director: The project director is accountable to the project owner for ensuring the project owner's needs are met. This role undertakes day-to-day management and makes decisions on behalf of the project owner.

Size of the project board

- Although there are four roles on the project board, certain roles may be shared or combined.
- The project owner role cannot be shared because accountability for the success of the project cannot be split.
- The project director role should not be shared or split on a major project.
- There may be more than one senior user, although it is recommended there be no more than two.
- There may be more than one senior supplier, especially when there are both internal and external suppliers or providers involved in the project. It is recommended there be no more than two.
- Business representatives should always be in the majority on the project board to ensure a service delivery focus in maintained.

- The project manager is not a member of the project board but reports into it.
- Experts can be invited to attend project board meetings; however, their input is normally obtained through the strategic advisors group or the stakeholder working group.
- Once the project board exceeds around six persons, decision making becomes less effective.

All project board members should attend all project board meetings.

Decision making

- The project owner is the chair of the project board and appoints project board members.
- The project owner is accountable for the success of the project and so all project board decisions require the support of the project owner.

The project board shall:

- Approve the terms of reference of the project board;
- Approve the responsibilities of project board members;
- Support the project owner;
- Work with key stakeholders to meet their needs and ensure their issues are addressed at the project board;
- Approve the project manager;
- Provide direction to the project manager;
- Approve the responsibilities of the project manager;
- Approve the project structure as developed by the project manager;
- Approve reporting and communication arrangements;
- Approve project documentation, which may include:
    - The project business case (and material changes to it);
    - The project management plan;
    - Feasibility studies;
    - Concept design;
    - Output specifications;
    - Options analyses;
    - The procurement strategy;
    - The project completion report and lessons learned
- Ensure project stakeholder engagement is being adequately addressed;
- Confirm the project's operating parameters and tolerances with programme management, including budget and schedule tolerances for project stages and for the project as a whole;
- Address and resolve project issues escalated by the project director or project manager;
- Escalate issues that cannot be resolved to the investment decision group;
- Approve any material changes to scope, budget, schedule or quality;
- Ratify any critical design decision.

Meetings of the project board

The project board should be convene at the commencement of the project (i.e. during the strategic assessment) and continue meeting until the project completion report has been delivered. The frequency of project board meetings should be appropriate to the scale and complexity of the project and is dependent to a large extend on the issues to be addressed. In certain circumstances the project board many need to meet weekly, but at less critical pints in the project's lifecycle monthly meeting

may be sufficient. If a project board meets too infrequently it runs the risk of becoming too remote from the project.

Record keeping

The project board provides direction to the project. Its decisions therefore need to be clear and unambiguous. On a long project it is possible that one or more project board roles could change hands over the duration and therefore it is important that the status of any document issued to the project board for approval I understood. This avoids revisiting decisions. Project board members can also use 'understudies' to ensure another member of their staff is kept appraised of the project and the decisions reached so that in the event the project board member moves form their position, continuity can be maintained until a new board member is chosen and briefed by the understudy.

When a project document is presented for approval, the project board decision should be one of the following:

- Approved;
- Approved subject to (list the changes that must be made for the document to be considered approved);
- Not approved – rework required in the following areas: (list).

A similar approach is beneficial for issues that have been addressed by the project board. Issues need to be logged and the outcome reached by the project board on each issue recorded. Issue resolution normally takes one of the following forms:

- The issue is resolved and the outcome recorded;
- Further information is requested of the project manager or project director to assist in resolving the issue;
- The project board considers the issue is a matter for resolution by the project manager and requests the project manager to advise;
- The issue is unable to be resolved by the project board and is escalated to the investment decision group for resolution;
- The issue is taken off-line by a project board member.

Investment decision group

Membership

The investment decision group is normally a pre-existing committee within an organisation. It may be referred to as the budget committee, budget review committee, expenditure (review) committee, etc. As such its membership is normally predetermined.

Terms of reference

The investment decision group will:

- Approve, or otherwise, funding the development of a final business case for the project based upon the information contained with the preliminary business case and the presentation and information provided by the project owner;

- Approve, or otherwise, funding to complete the project based upon information contained within the final business case and the presentation and information provided by the project owner;
- Approve, or otherwise, material variations to funding based upon a updated final business case and representation from the project owner and/or project director;
- Stop projects from proceeding at preliminary business case stage where the preliminary business case describes a project that is not aligned with the organisation's goals;
- Stop projects from proceeding at business case stage if the business case is not considered viable, affordable or value for money;
- Refer the preliminary business case back to the project board for reworking where I does not meet the organisation's requirements for such a document or where further information or analysis is required. (note: the investment decision group's secretariat will often act as a filter to ensure the quality of documentation provided to the investment decision group is adequate; however, this does not presuppose the necessary level of understanding and sophistication in respect of projects within the secretariat.);
- Address and resolve issues raised by the project board;
- Address and resolve matters of policy raised by the project board (note: if there is a strong programme management board, it may address escalated issues and matters of policy;
- Forward approved preliminary business cases to (financial planning function) for inclusion in forward budget programmes.

Strategic advisors' group

The strategic advisors' group shall undertake its activities as follows:

- The strategic advisors' group is chaired by the project owner;
- Strategic advisors' group members must have the authority to make decisions on behalf of their organisations. Decisions made by strategic advisors' group members are taken to represent the views of their respective organisations;
- Members are to work with an alternate so that in the event of their absence from a meeting; their view can be represented by someone with equivalent authority and understanding of the project and its issues.

Members of the strategic advisors' group have the following responsibilities:

- Review project documentation and advise the chair of any issues or concerns they have with it;
- Identify issues and risks that may impact the project;
- Raise any issues members have with the project with the strategic advisors' group initially;
- Provide advice on project issues. In particular, provide detailed advice on areas relating to their organisation's specific interest in the project;
- Work as a group to provide the project owner with a single agreed position on project issues (wherever possible);
- Disseminate non-confidential information regarding the project within members' host organisations;
- Where a stakeholder working group has been established, nominate one or more officers from their organisation to sit on that group and provide the technical advice required;
- Support the project owner.

Stakeholder working group

The stakeholder working group is chaired by the project director. The project owner or project manager may chair the group in the absence of the project director.

Members have the following responsibilities:

- Represent the interests of their organisations;
- Keep their representative on the strategic advisors' group appraised of developments and decisions taken by the stakeholder working group;
- Work with an alternate so that in the event of their absence at a meeting their views can be represented by someone with equivalent authority and understanding of the project and its issues;
- Review project documentation and advise the chair of any issues or concerns they have with it;
- Identify issues and risks that may have an impact on the project;
- Raise any issues members have with the project within the stakeholder working group initially;
- Provide advice on project issues and in particular, provide detailed advice on areas relating to their organisation's specific interest in the project;
- Work as a group to provide the project director with a single agreed position on project issues (wherever possible);
- Disseminate non-confidential information regarding the project within members host organisations;
- Support the project owner, project director and project manager.

Appendix 3 – Role Descriptions

**** This document is provided in full and with the approval of their author, Ross Garland.

The following role descriptions cover each project board member as well as the project manager's role in respect of the project governance arrangements. They may prove of use to an organisation either developing a project governance framework or structuring the governance of a particular project.

Project Owner

Introduction

The project owner is accountable for the success of the project. They are the owner of the project and must have responsibility for the project's budget.

The project owner is drawn from the business itself. The more important the project, the higher the position the project owner holds in the organisation. As a result of their business focus, the project owner views the project as a means to an end – the end being benefit that the project is designed to deliver. They own these benefits and the outcomes that the project will deliver and their ownership of the project is focused on ensuring these benefits are delivered to the business. Ownership of the project confers ownership of the documentation that defines the project, in particular the business case. The project owner role on any project should not be outsourced and should be considered a core part of the business of the organisation.

A project can have only one project owner – the role cannot be split, since accountability itself cannot be split. In the event of multiple sources of funding a project, a single project owner should be chosen with other funding parties being represented in other roles on the project board, such as senior user.

Major responsibilities

Note that if the project does not have a project director, the project owner's responsibilities will need to encompass those of a project director.

Establish the project's governance arrangements

- drive the initiation of the project upon appointment;
- establish the project board and select project board members based on the project's needs;
- recruit the project director;
- establish the strategic advisors group and stakeholder working group;
- work with the project director to identify the project manager and source project advisors;
- ensure all stakeholders understand the operation of the project governance arrangements and their role in it.

Be the primary sponsor of the project

- present themselves as the main face of the project, both internally and externally to the organisation;
- manage stakeholders with the assistance of the project director and ensure stakeholders needs are met and their issues are addressed;
- ensure ongoing stakeholder support for the project;

- ensure stakeholders are aligned with the project's objectives and that they remain so during the project;
- manage upwards rather than downwards-downwards management from the customer's perspective is the role of the project director.

Note that on smaller projects where a project director is not engaged, the project owner will have to manage downwards.

Ensure the project maintains a service delivery focus

- ensure the overall project focus is on delivering services rather than just the asset;
- ensure project costing, budget and cost controls are focused on whole-of-life costs rather than only capital costs;
- ensure the project benefits are clearly stated and that a clear plan is developed for realizing those benefits;
- ensure the overall output specification is designed around delivery of benefits;
- develop the definition of the project from the owner's perspective, ensuring clear articulation of the project's benefits, objectives, drivers and critical success factors;
- Ensure the project focuses on benefits realisation throughout its life;
- ensure the project is aligned with the goals and vision of the organisation.

Monitor and control progress

- drive the project forward and ensure that momentum and progresses maintained;
- own and manage the project budget;
- where additional funds are required, present or support the case for such funds;
- own and manage the project business case and other key project documentation such as the project plan, preliminary business case or equivalent, etc.;
- ensure project documentation is reviewed by the project board and that documents are either approved of the necessary modifications required to achieve are articulated;
- approve major scope changes and ensure the business case reflects such changes.

Focus on the main risks and issues

- ensure adequate attention is focused on risks and risk mitigation;
- where appropriate, ensure risk mitigation is adequately coated;
- resolve issues that have been escalated by the project manager and work with stakeholders where necessary to address such issues;
- Where necessary, seek independent advice on the project.

Resource the project for success

- ensure adequate project owner resources are allocated to the project to assist in defining service delivery outcomes, desired benefits, output specification, etc.;
- ensure sufficient user resources are deployed on the project for the production of user specifications, acceptance critter, etc.;
- ensure external supplier resources are adequate in terms of numbers, skills and expertise;
- ensure they allocate adequate of their own time to the project.

Maintain a strategic perspective on the project

- liaise with the programme management office (if extant) regarding resourcing, quality criteria and the positioning of the project as part of the greater programme;
- understand the strategic objectives of the organisation and ensure the project remains aligned with those objectives throughout its life;
- Maintain awareness of any broader environmental (not just green) considerations and how the project has an impact on of could be affected by such considerations.

Specific responsibilities

The project owner is responsible for:

- appointing project board members;
- Appointing the project director and agreeing their remit and delegated authority;
- Chairing the project board
- Chairing the strategic advisors group
- Agreeing all major plans and any deviations from them;
- Approving the full business case and recommending it to the investment decision group;
- Approving major project deliverables;
- Communicating information about the project organisation and stakeholder groups as necessary;
- Resolving conflicts escalated by the project team, clue t or supplier, or escalated these issues to the investment decision group;
- Resolving conflicts between project team, end users and suppliers, or escalating as necessary;
- Agreeing the project tolerances for time, quality and cost (with the programme management office if applicable);
- Providing overall strategic guidance for the project;
- Addressing the risk(s) associated with the project;
- Ensuring project quality assurance is adequate and consistent with the organisation‘s norm (possibly as defined by the programme management office);
- Providing advice and direction to the project director and project manager as required;
- Closing the project;
- Approving the end of project report and the lessons learned report.

Ideal characteristics and skills

When selecting a project owner, the primary consideration is the person’s position within the organisation and whether that position owns the service delivery outcomes the project will deliver. The person identified via this route may not exhibit all the characteristics and skills listed below; however, this is a secondary consideration that perhaps can be addressed in part by further support for the project owner, training or the appointment of a project director with the necessary skills:

- Senior commensurate with the scale of the project;
- Decisive, involved and not just a figurehead;
- Prepared to accept accountability and responsibility for the project;

- Reasonable understanding of projects and the project lifecycle;
- Good communication skills;
- Strong understanding of the business case and its development process;
- Detailed knowledge of business issues and desired business outcomes;
- In the event the project's business case does not provide value for money, be prepared to report same to senior management and, if necessary, terminate the project;
- Able to focus on the big picture and understand the project in the context of the overall programme;
- Be a good negotiator who is able to reconcile the disparate needs and drivers of users, suppliers and the business;
- Recognise when the project is in difficulty and be prepared to act to resolve the issue;
- Act in an open and honest manner in regard to the project;
- Provide senior management and programme management with an honest appraisal of the project's progress.

Project director

Introduction

The project director is tasked with guarding the interested of the project owner when the project owner does not have the necessary time to devote to the project. The project director reports directly to the project owner and the project manager reports to the project director. In this respect, the project director provides the interface between project ownership and delivery.

The project director represents the customer and acts as the main point of contact with the project manager for the day-to-day management of the customer's interests. This role is responsible for ensuring the project objectives are delivered. For this to happen the project director must ensure the project maintains a service delivery focus and will work with the project owner and project manager to ensure this is the case.

The person in this role must have adequate knowledge and information about the business to make informed decisions and must also have a detailed understanding of project management to appreciate the 'respective of the project manager and project team. On large and complex projects, this is a difficult combination to achieve because those with sufficient understanding of the business are not always project specialists. Thus it is not always possible to source project director from the business and it may be necessary to outsource this role. If the person is outsourced they will need to develop a good understanding of the business, work closely with business experts and integrate them into the project. The alternative approach of taking a business expert and training them in project delivery skills is unlikely to be effective on large and complex projects.

If the project director is seconded from the delivery side of the organisation, the contractual arrangements and personal performance arrangements must all sit with the project owner, else there will inevitably be conflicts of interests.

Major responsibilities

Protect project owner's interests

- provide project drive and momentum;
- Maintain a service delivery perspective and a focus on business outcomes;
- Keep project owner informed of issues and risks;
- Monitor project progress against plans and review project manager's reports;
- Monitor project quality;
- Define protocols for control and management of the project;
- Identify and select project advisors.

Assist project owner in management of stakeholders

- Support project owner in managing strategic advisors' group;
- Manage the stakeholder working group, working with the project manager;
- Balance the competing demands of the business, users and suppliers;
- Ensure communications protocols are in place and effective in enabling sufficient dialogue to ensure ongoing alignment of stakeholders, contractors, end users and business resources.

Manage business resources

- manage those resources providing business input to the project;
- Ensure business inputs are provided with adequate levels of quality and detail and are timely;
- Establish project team arrangements and work to foster teamwork.

Work with the project manager

- develop project reporting and communications protocols
- Work with the project manager in the development of key project documentation and review all documents that proceed to the project board;
- Keep project owner apprised of project activity;
- Work with the project manager to resolve issues and escalate those that cannot be resolved;
- Assist in the identification of risks and ensure mitigation strategies that meet business needs are developed;
- Assist in coordinating and fostering teamwork.

Ideal characteristics and skills

- decisive, with good clarity of purpose;
- Good negotiator;
- Ideally should have leadership qualities, with the ability to form a team around themselves;
- Project management skills, possibly someone who has delivered large projects themselves in the past;
- Detailed understanding of the department's business;
- Detailed knowledge of the project's objectives, drivers, desired benefits, etc.;
- Understanding of procurement models and their suitability for different project types;
- Knowledge of risk analysis and risk management;
- Detailed knowledge of all elements of the project lifecycle;

- Strong understanding of quality management principles and processes.

Senior user

The senior user reprints those who will use the final product or service that the project delivers. This usage may comprise direct usage, indirect usage such as network operations where the product or asset forms part of the network, or even maintenance or facilities management where such considerations have a significant impact on the development of the project.

The role can also represent those whom the project may significantly affect. In particular, if the project has more than one source of funding, the project owner is normally chosen as a representative of the main funding organisation and a senior user role could be used to provide a set at the project board for the second funding organisation. The senior user has the following responsibilities:

- Support the project owner and assist them in directing the project;
- Advise the project owner of any user issues that may have an impact on the project;
- Advise the project owner of the impact on users of any changes being considered by the project board;
- Assist the project owner as required in discussions with the strategic advisors' group on matters relating to user needs;
- Negotiate with the project owner and senior supplier regarding the provision of user requirements balanced against the cost of providing those requirements;
- If there is more than one senior user, liaise closely with the other ole holder for the benefit of the project;
- Ensure all users are advised of any issues raised at the project board that may affect them;
- Represent the interests of their user clients to the project;
- Resolve conflicts between users on the matters the project board addresses;
- Establish and chair user groups as necessary and ensure a consolidated user view is presented to the project board;
- Maintain a focus on the delivery of users' needs throughout the project;
- Take responsibility for the development of user requirements specifications, acceptance criteria and user acceptance testing documentation, aided as necessary by the project director and project manager;
- In developing user requirements, maintain a whole-of-life perspective;
- Ensure the quality of the user project deliverables meets the standards laid down by the project or programme management;
- Commit resources to the project as required to meet the project needs and develop user documentation;
- Assist the project manager in managing user resources on the project.

Senior supplier

The senior supplier represents the suppliers of the services to the project and provides their perspective and expertise. The senior supplier has the following responsibilities:

- Support the project owner and assist them in directing the project;
- Advise the project owner of any supplier side issues that may have an impact on the project;
- Ensure all suppliers are advised of any issues raised by the project board that may affect them;
- Advise the project owner of the impact on suppliers of any changes being considered by the project board;
- Represent the interests of all suppliers to the project;
- Where supplier contracts are with the project owner's organisation, work with the project owner to manage the activities of these suppliers and represent their interests and perspectives at the project board where these interests are not represented by a second senior supplier role;
- Resolve conflicts between the suppliers to the project;
- Provide the perspectives of all suppliers on the matters the project board addresses;
- Chair supplier forums as necessary;
- Interpret technical aspects of the project for the benefit of less technical project board members;
- Assist the project owner as required in discussions with the strategic advisors' group on matters of a technical nature;
- If there is more than one senior supplier, liaise closely with the other role holder for the benefit of the project;
- Commit resources to the project as required to meet the project needs;
- Ensure the quality of those project deliverables that are the responsibility of suppliers meets the standards laid down by the project or programme management;
- Relinquish the role of senior supplier in the event that the project has progressed in its lifecycle to a point where another organisation is better placed to provide a person to fill the role.

Project manager

There are sufficient texts covering the role of the project manager for this book not to dwell further upon it. The following, then, covers that role only from the perspective of the project governance framework. The responsibilities of the project manager in relation to this project governance framework include;

- Manage the delivery of the project on a day-to-day basis in accordance with the constraints established by the project owner and the project board;
- Liaise closely with the project director;
- Work with and support the project director in managing the needs of stakeholders;
- Provide project progress reports to the project director 9there may be a reporting line also to the senior supplier);
- Establish project control mechanisms in conjunction with the project director;
- Establish communications protocols in conjunction with the project director;
- Attend project board meetings as required ad provide reports to the project board;

- Support the project owner and project director in managing the activities of the strategic advisors' group and stakeholder working group;
- Develop the project management structure of the project/project team;
- Manage the integrated project team and, in particular, supplier resources on the project;
- Manage the production of project documentation;
- Ensure adequate project management methodologies are employed on the project ad that these are consistent with organisational and programme management requirements.

Appendix 4 - Project Management Methodologies

Non-Agile

Waterfall (Traditional)

The Waterfall development process is perhaps one of, if not the oldest methodologies used in software design, dating back as far as the 1970s. At a time when the field was in its infancy, there were no existing methodologies in place that could cater to the specific needs of technical IT projects. As a result, pioneers drew from a well-known model used in the manufacturing and construction industries. The term 'waterfall' is used to describe a sequential development process, where progress is seen as a steady downward 'flow' through the pre-determined phases or steps of conception, initiation, analysis, design, testing, production/implementation and maintenance.

According to Jurison (1999), the model emphasises the importance of remaining in a given phase until it is fully completed or perfected and the moving forward. Figure 1 illustrates this concept. In the example illustrated in Figure 1, the requirements analysis phase could not start until the feasibility study is completed, and similarly the design phase could not begin until the requirements analysis is finished.

Jurison (1999) also describes each phase as having a termination point that is clearly defined by a set of deliverables. These deliverables require review and approval by a qualified member of the organisation or project team before progress is allowed.

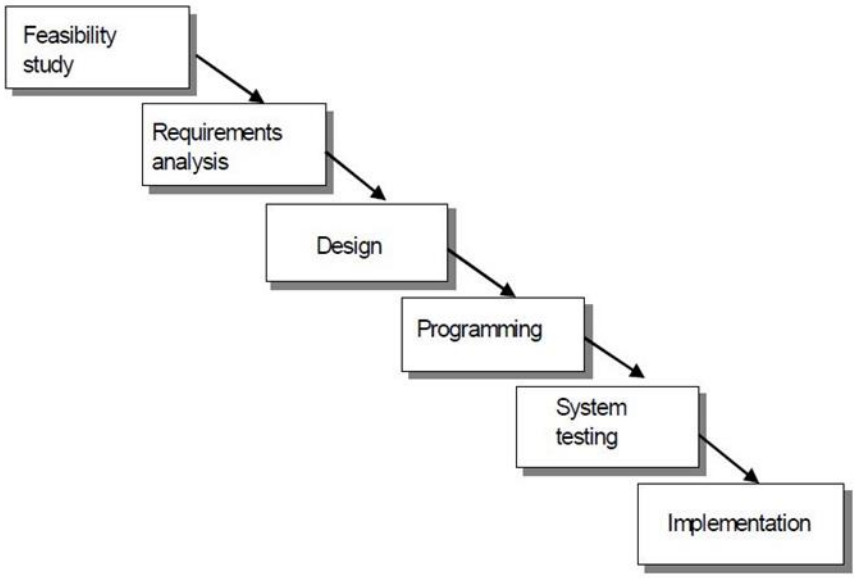


Figure 18 - phases in a project governed by waterfall management

Two key factors contributing to the model's efficacy are the rigor of following a single path to success and the emphasis on perfecting and completing each phase.

There is great benefit in the amount of discipline required and strength in the model's simplicity. By getting things right in the early stages of a project and not moving forward until this is achieved, significant financial and other resource efficiencies can be realised. According to Cusumano and Smith (1995), the Waterfall method works well in 'stable problem domains' in which problems are well defined and changes or refinements are minimal or non-existent, as well as in cases where one seeks to incorporate unplanned rework while minimising negative impact on customers and

opportunity for competitors. The development of an Annual Percentage Rate (APR) calculator application and annual tax program like Turbo Tax are good examples of projects in which the requirements (e.g.. tax laws and timelines) do not change over time, and full prediction by developers and designers seems like a reasonable possibility from the outset; the customer does not create or drive iterative changes in the application.

Another advantage of the Waterfall method is the ability to plan, manage, assign and release project resources based on effort (Armour, 2014). While other approaches focus on task-based workflow, the attainment and return of resources remains a large challenge that is often overlooked. The Waterfall method facilitates project tracking with a valuable focus on the detailed documentation of all processes (and source codes) throughout each phase, in order to protect the project from losing valuable information and causing subsequent delays if a team member were to leave and a new team member takes over.

With its roots in manufacturing and construction (industries that have highly structured physical environments with little allowance for changes 'after-the-fact', the benefits of using the Waterfall method in scenarios like the automotive industry are clear, as requirements and designs have to be complete before a car is constructed due to high material and assembly costs. However, its application to software appeared to be due to a lack of alternatives. Even the earliest literature published on the method describes it as being flawed and ineffective for software development (Royce, 1970). Agile supporters would contend that no phase of a software development project is ever really complete, as the client or end-user will often not fully understand or know what they need until they are provided with a working prototype to comment on. This aligns with Royce (1970) stance that product teams using a Waterfall approach really only have one opportunity to successfully complete each aspect of a product or project.

The inability of Waterfall management techniques to adjust to fast-moving markets and uncertain requirements could result in two negative outcomes (Cusumano & Smith, 1995):

- the project coming to a grinding halt and never moving out of the requirements phase, or
- the project moving out of requirements phase without addressing market or consumer needs.

This second case highlights the major disadvantage of traditional Waterfall techniques: the lack of an iteration and feedback mechanism to improve initial imperfections. In essence, if there is a design flaw in the design phase or a requirement flaw during the requirements phase, those flaws have a greater chance of being present in the final software application than in projects managed by different techniques.

The urge to plan everything before execution in later phases therefore leads to a need for a few key controls to be integrated in the management process. Examples of these controls include the use of a change control board to approve changes throughout the project lifecycle, a defined system for cataloguing and documenting artefacts, and detailed estimation tools and budgets that shoulder the burden of 'predictive understanding' (the desire for control and understanding before all necessary information is available).

When using the Waterfall methodology, an organisation must constantly examine whether it is able to get products to market fast enough and with sufficient quality to meet customer demands. Organisations who find their requirements constantly changing or the market driving products in multiple directions should investigate other development methodologies.

V-Model

The V-Model project management methodology is a close extension of its Waterfall/Traditional predecessor, created in response to the flaws or defects that often survive and are incorporated in the final product of projects developed with that technique. With the aim of eliminating these flaws, V-Model uses Waterfall as a starting point, before then focusing on the decomposition of requirements and the creation and alignment of associated validation steps. According to Rowen (1990), this decomposition was intended to reduce some of the complexity in large system development. Returning to Figure 1, which depicts the (distinctly linear) progression of phases in a Waterfall project, one can appreciate that any testing performed in this case must have a broad and simple focus, examining all previous work in one step or with the same measure. V-Model's decomposition of requirements and allocation of specific testing/verification efforts for each stage causes that linear shape to change, as illustrated in Figure 2.

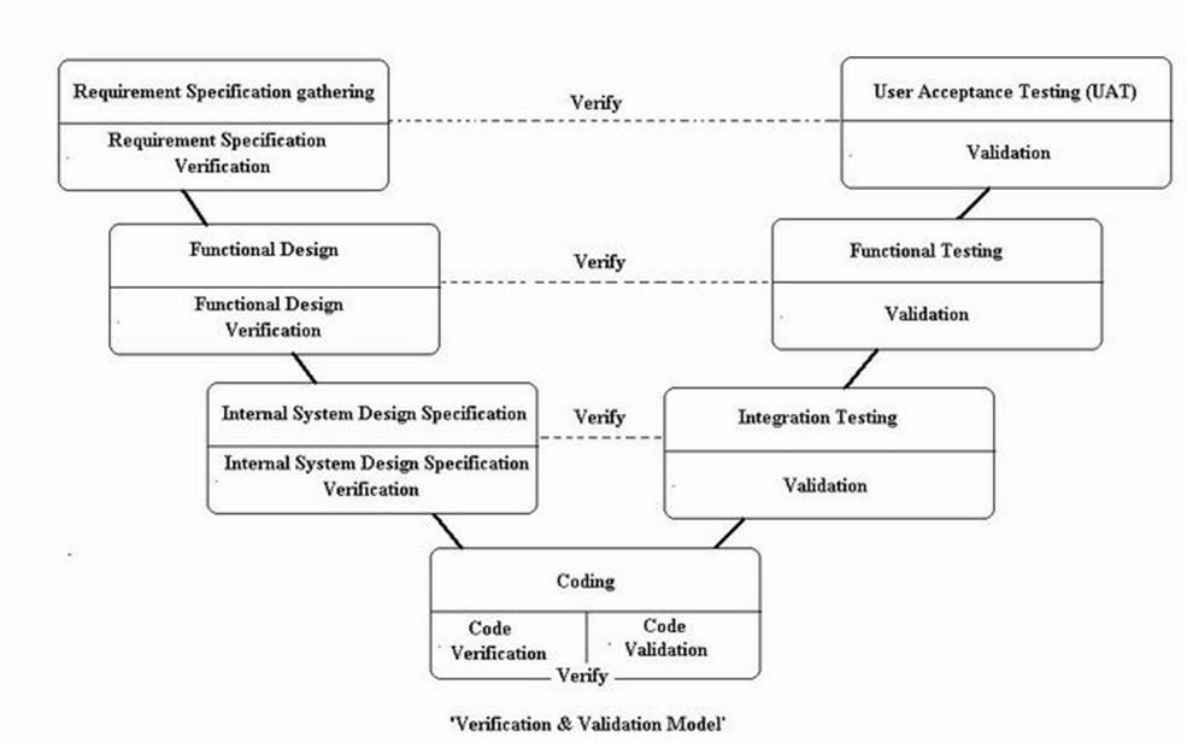


Figure 19 - Illustration of project progression with the use of V-Model management

Source: "Waterfall model vs. v-model," by Satalkar (2011)

Instead of moving down through sequential phases in a linear way, the phases are arranged in a 'V' shape, 'bent' back upwards after coding is complete. The V-shape is visualised on an x-y axis, on which the horizontal component represents the passage of time or degree of project completion, and the vertical axis reflects a 'level of abstraction'. This essentially means that as the project begins at the top of the V and progresses downward, more and more attention is paid to rigid, detailed outcomes relating to design. After coding is complete, the phases that follow proceed back upwards and thereby begin to retain an increasingly broad or abstract focus. In this way, a project is also divided into distinct verification (requirement analysis and varying levels of design) and validation portions, the latter of which incorporates unit, integration and systems testing. These improvements have allowed the V-model to become accepted as a viable tool for use in the development of both hardware and software.

A main feature and benefit of the V-Model approach is its ability to demonstrate the relationship between each phase of the development lifecycle (complete with deliverables, documents and meetings) and the associated testing phase. Testing aspects are more defined to align with the depth of the requirements and design work performed in earlier stages. The decomposition required to separate requirements and allow this type of focused testing proves beneficial not only for the project, but the development team as well. Sub-groups can be created to allow a focus on smaller program elements, while a number of inherent checkpoints are introduced in the lifecycle, stemming from the Verification and Validation processes.

Another key element Rowen (1990) highlights is the recognition of the fact that coding is not permitted to begin until the critical design review is complete.

Many of the criticisms of the V-Model are similar to those of the Waterfall methodology. The V-Model is also perceived as a representation of what a typical Project Manager might desire in an approach for other fields, but it remains inconsiderate of what software developers and even end-users might actually need, as it is often too rigid and provides little ability for response to change. Rowen (1990) suggests that the early ambiguity associated with requirement definitions could lead to a costly misinterpretation of those requirements. Further decomposition of an already ambiguous requirement could lead to massive system failures that do not manifest or become identifiable until near the end of a project. The model's encouragement of pre-emptively developing testing methodologies often leads to ineffective and inefficient analysis, as developers can only look for problems they were able to foresee from the beginning; the quality of testing is limited in this way.

As with many older methodologies, a noticeable weakness in the V-Model (and Waterfall) approach is the lack of end user involvement, which results in a failure to scale to ever-changing market demands. Though a product may meet functional requirements, elements like ease of use, aesthetics, impression and satisfaction can also be forgotten in the development process and therefore fall short of user expectations

Rapid Application Development (RAD)

A major drawback of older methods like Waterfall is that software projects take so long to develop that often the requirements will change before system development is complete. The traditional methods also rely on the very risky assumption that a rigorous planning phase will allow for the identification of all critical requirements and challenges. While this would be nice, it's very rarely seen, even with a team of the best experts.

RAD was created as a response to these weaknesses, and is attributed primarily to the work of Dan Gielan at the New York Telephone Company's System Development Center in the 1970s, and later to the work of James Martin in the 1980's at IBM.

While many methodologies will use a distinct (and often lengthy) planning phase of some form, RAD utilises minimal planning, in favour of quick prototyping. Developing software in this way means that the code and product is actually written as planning takes place, which results in increased ease for accommodating changing requirements and substantial increases in speed. There are four distinct phases: requirements planning, user design, construction and cutover.

There is some debate about whether or not RAD should be classified as an agile methodology. A key difference appears to be that RAD foregoes a focus on developing what is needed in order to explore what is interesting. Trends in software development would suggest that eventually, all users 'need' is what interests them. As an early response to and improvement upon traditional Waterfall methods however, RAD is more appropriately categorised as a non-Agile methodology.

Joint Application Development/ Design (JAD)

A number of the methodologies already discussed have grown from attempts at improving upon their predecessors. Just as RAD was developed in order to address the inherent weaknesses of traditional methods, subsequent strategies have been created to negate its shortcomings. One of these methods, Dynamic systems development method (DSDM) (discussed later in more detail) uses RAD creator Dan Gielan's own technique regarding workshop implementation as a critical feature in the prototyping portion of project lifecycle. These workshops (building on RAD strategy and becoming a key feature of DSDM) can exist separately as their own project management methodology, called Joint Application Design (JAD).

JAD workshops were originally designed to bring end-users and system developers of varying backgrounds together in a creative and productive environment. Structured meetings aim to quickly obtain all product and business requirements (and specifications) while allowing knowledgeable workers, users and IT specialists to work out any difficulties or disputes early in the development process. The resolving of issues and disputes is aided by the use of a detailed workshop agenda that has been created in advance. Good communication is essential in ensuring that no items are left out. These meetings are much more efficient than the series of separate interviews which would otherwise take place in traditional methods, and they result in systems projects that are both appealing to the user and feasible for the designer. By bringing all parties together, developers, users and experts alike are all able to retain a sense of project ownership and contribution. Key participants in JAD workshops include: an executive sponsor, subject matter experts, a facilitator, a documentation expert, and observers.

Arnie Lind, a senior systems engineer working with IBM Canada at the time, was responsible for seeing the real value in using these workshop sessions as a central project development tool. Rather than relying on developers to read about and immerse themselves in environments and businesses that required a given application, Lind realised that a great deal of time and effort could be saved in simply teaching or helping end-users themselves develop the applications. In this way, applications are not only more likely to meet stakeholder needs, but also to be understood and embraced. A government-based pilot program that saw developers paired with emergency room workers was a great success and fuelled the popularity of the JAD approach. It is generally thought that JAD is most effective when applied to projects that are small and clearly focused.

There are nine key steps in JAD:

1. identify project objectives and limitations
2. identify critical success factors
3. define project deliverables
4. define the schedule of workshop activities
5. select the participants
6. prepare the workshop material
7. organise workshop activities and exercises
8. prepare, inform and educate workshop participants
9. coordinate workshop logistics.

Care should be taken in participant selection and other planning activities as poorly constructed workshops run the risk of wasting the valuable time of professionals.

Critical Path Method (CPM)

The CPM involves applying a type of mathematical analysis (or algorithm) to the scheduling of any project with interdependent activities. It was used in tasks long before the age of software development, much like the Waterfall methodology. All activities required to bring a project to completion are listed within a model (through a Work Breakdown Structure), and notations are made regarding the dependency and relationship between each, endpoints such as deliverables and milestones, and the time they will take to complete. By using these values, one is able to determine the earliest and latest points each activity can commence and be completed without delaying the entire project. Those that cannot be delayed fall upon the longest (critical) path from a project's beginning to end, and are termed 'critical' activities. Those that can be delayed without affecting total project duration are described as having a 'float' value, the time cushion allowed before causing a delay. The longest (critical) path actually represents the shortest possible time to complete the project.

Developed in the 1950s by Morgan Walker at DuPont and James Kelley Jr., Arrows in 'activity-on-node' (AON) diagrams are used to show the dependence and relationship between each activity from start to completion. A mathematical algorithm is used to compare the duration, float and drag of activities that fall parallel to one another in the AON diagram, allowing managers to effectively manage and prioritise each activity. Two actions can be taken in order to influence natural project progression, each modifying the activities falling on the critical path. These activities can be 'fast-tracked' by re-organising and encouraging a higher number of activities to be performed in parallel, or they can be 'crashed,' having their duration shortened by adding resources. There is a limit to the magnitude of time decrease an activity can experience through crashing and re-allocating resources; this is termed the 'crash duration.' While this movement of resources will result in decreased time for an activity, an accompanying decrease in quality is also often observed.

Critical Chain Project Management (CCPM)

In recent years, the CPM has grown to include consideration for the resources related to each activity through 'activity-based resource assignments' and 'resource levelling.' This re-allocation and levelling of resources can lead to bottlenecks in the project, where delays result from an unavailability of resources at a given point. This type of delay can cause what was originally a shorter, parallel and non-critical path to become the longest path in a project, and thereby become the most 'resource-critical.'

Derived from the Theory of Constraints and the work of Eliyahu Goldratt (year), CCPM makes an attempt at protecting each activity (and the project as a whole) from delays occurring due to resource constraints. It does this by shifting the main emphasis to resources required to execute project tasks, in contrast to CPM's approach of emphasising rigid scheduling and the order of tasks. Although keeping resources levelled, CCPM requires all resources to be available for quick re-allocation, and for all activities to have flexible start times. Other features that distinguish CCPM from CPM include the lack of a search for an optimal solution, and the identification and insertion of project 'buffers'. These buffers are essentially cushions of time allocated for different elements of the project, and through monitoring the consumption rate of these buffers the health and progress of the project is actually monitored. This differs from the CPM model, where progress is judged by adherence to the set schedule of individual tasks, a type of earned-value management. Easy and effective monitoring through buffer management is one of CCPM's greatest advantages.

A 'critical chain' is determined in the same manner as the critical path in CPM, though the resources are levelled first. In some forms of the method, a Monte Carlo (statistical) simulation is used to determine the overall probability of various risks actually impacting a project outcome. CCPM has an implicit aim of removing 'bad' or ineffective multi-tasking, and the strategy has been compared to the passing of a baton in a relay race. Each task is completed with a singular focus and with as much speed as possible.

Event Chain Methodology (ECM)

ECM is another network-analysis technique, one that builds on and follows a logical progression from both CPM and CCPM.

ECM is governed by the following set of principles:

- Activities can exist in different states because they can be affected by external events. The precise moment at which an influencing event will occur is predicted using statistical probability, and in this way each activity's state is quantitatively defined.
- Event chains (events caused by other events) are expected, defined, planned for and analysed.
- After event and event chains have been defined, Monte Carlo simulations are performed to quantify the cumulative effects of each event. Separate distributions related to duration, start time and cost are often used to supplement the initial input data, which consists of risk effects and probabilities.
- The correlation between main project parameters (such as duration and cost) and event chains is analysed, allowing the identification of those chains that have the most potential to affect the project. These are deemed to be 'critical', and it then becomes a priority to mitigate their potential effects.
- Performance tracking is undertaken of the various event chains, as actual data becomes available during the project and is fed back into the simulation, and the risk and probability of events are recalculated and compared with predictions.
- Visual depictions of the relationship between activities and events are created by using arrows on Gantt charts (Event Chain Diagrams).

ECM allows for the easy visualisation and handling of phenomena such as changing resource allocation in response to events, and the unplanned repetition of activities. A main distinguishing benefit is the opportunity for creating mitigation plans pre-emptively as their own activities, which can then be placed among the activities of the larger project, ready for use and implementation if necessary.

Benefits Realization Management (BRM)

BRM takes an 'all-or-nothing' approach in which a project is only considered to be a success if the benefits that stakeholders were hoping for are actually realised. Central to BRM is the idea that no matter how technically powerful, IT alone cannot deliver business results.; For value to be created and sustained, benefits (measurable positive impact of change) need to be actively managed throughout the project's life cycle. Essentially BRM contends that benefits don't just happen because of a system, and when they do they rarely happen according to a given plan. Instead, they arise and change as a result of people learning to use that system over time.

All development teams and systems make use of BRM to some degree, whether they know it or not. Not knowing is probably a result of informal implementation or a lack of understanding of the concept. Like most other methodologies, BRM works best when actively managed, designed and engineered to improve performance by intention. Senior management is provided from the outset with a clear understanding of which results are desirable, and in what way IT will contribute to attaining them. Teams will therefore only embark on projects with well-defined plans and the means required to reach specific goals.

A variety of 'mapping' strategies are commonly used to identify and agree desired outcomes. Examples include Benefits Dependency Maps, Benefits Dependency Networks and Results Chains. Having gained recent recognition in the Media due to its implementation in government programs in the UK, BRM projects also differ from traditional methods by reaching a little further at each end of a

project's lifecycle, encompassing everything from 'concept to cash', instead of the usual design to delivery standard. BRM has been successfully used to ensure that benefits are understood and realised for large, expensive and complex projects like Software Application Packages.

Evolutionary Project Management (Evo)

Evo focuses on the delivery of maximum value to stakeholders while setting and emphasising goals related to product quality that minimise the use of development resources. Information about Evo was first published by Tom Gilb in the 1970s and used in the Cleanroom techniques of Harlan Mills at IBM during that same period.

A key Evo principle is the acceptance of evolutionary delivery as a learning cycle, in which one learns what does and does not work from reality and experience. This learning must occur early (while there is still time to make necessary changes) and the cycle should be small, in order to minimise risk and the loss associated with failed delivery. Simplicity is desired for all projects and the tasks therein, so anything of a complex nature is divided into smaller parts. Evo strategy is not only useful for software projects, but can lead to great success in other areas of IT, beyond the coding of feature and function.

Spiral

The Spiral model of IT project management is almost an early form of Adaptive Software Development. Its creation and introduction in the field allowed for an important new distinction to be made between software development methodologies and process models. While methods and methodologies focus primarily on how to navigate through project phases and represent resulting products, process models (like Spiral) strive to provide guidance for determining the appropriate order of project stages and tasks, and to establish criteria that facilitate transitions between each stage and task. Examples of the latter include completion criteria and choice/entrance criteria, for current and future stages, respectively.

Process models evolved as a means to help reduce the high frequency of software projects experiencing failure as a result of development phases being executed in an ineffective order. They do this by asking the following questions:

- What do we do next?
- How long shall we continue to do it?

Much like the V-Model, Spiral took shape as refinements were made to early Waterfall applications, especially those used in large government software projects. First described in a 1986 paper by Barry Boehm entitled 'A Spiral Model of Software Development and Enhancement', it acts more as a 'model generator' than a model itself. What this means is that the technique guides development teams through a process of adapting an ideal combination of elements from other existing models and methods (most of which can be accommodated), based on the risk patterns unique to any given project. In Boehm's own words, the Spiral Model's major distinguishing feature is that "it creates a risk-driven approach to the software process rather than a primarily document-driven or code-driven process... It incorporates many of the strengths of other models and resolves many of their difficulties" (B. Boehm, 1986). This risk-driven 'sub-setting' allows the model to accommodate any appropriate mixture of approaches to software development, including those with a specification, prototype, simulation or automatic transformation-oriented nature. A spiral model therefore blends features from other strategies, and while it certainly produces modified forms of these existing methods, by circular logic, those products can also be classified as special cases of the spiral model itself.

The most complete application of the Spiral Model at the time of Boehm's publication was the development of the TRW Software Productivity System (TRW-SPS) (B. W. Boehm, Penedo, Stuckle,

Williams, & Pyster, 1984). A famous and frequently replicated illustration of the Spiral model from Boehm's publication is included as Figure 4

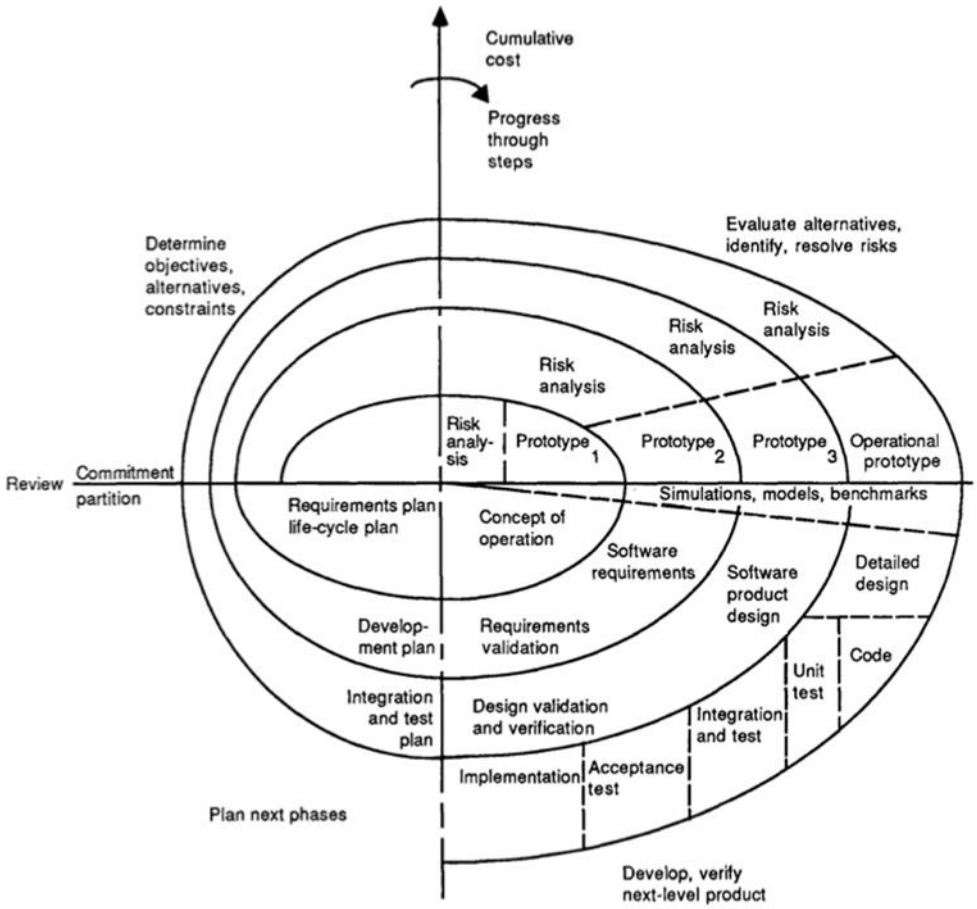


Figure 4 - Spiral model of the software process

Source: "A spiral model of software development and enhancement by (B. W. Boehm, 1988)

The radial dimension in Figure 4 represents the cumulative cost incurred in accomplishing all project steps to date, while the angular dimension is indicative of the amount of progress made in completing each cycle of the spiral. Boehm mentions the importance of noting that some artistic license has been taken with the radial (cumulative cost) dimension, in order to enhance the legibility of steps (B. Boehm, 1986). A central concept is that each cycle involves a progression through the same sequence of steps, regardless of project phase and level of product development; from concept and operations right down to each individual program's coding. Every authentic application of the Spiral Model therefore not only requires that a number of cycles be performed, but that each cycle must further incorporate the following six invariant elements:

1. All key artifacts should be defined at the same time, and not sequentially.
2. Four basic activities must be performed in every cycle:
    a. consider the win conditions of success-critical stakeholders
    b. identify and evaluate alternative approaches for satisfying the win conditions
    c. identify and resolve risks that stem from the selected approaches
    d. obtain approval from all success-critical stakeholders and commitment to pursue the next cycle.
3. Risk determines effort level. For every activity in a project, the team has to decide how much effort is needed to minimise risk. This becomes a balancing act of sorts, as certain activities will minimise risk to a point, after which added time causes risk to accrue in other

areas. One example of this would be with project testing, which decreases risk of poor market review, while simultaneously giving time for competing projects to reach that market first.

4. Risk determines level of detail.
5. Use of anchor-point milestones as points of commitment and progress indicators, through the use of key questions. Three anchor point milestones are typically used (life cycle objectives, life cycle architecture and initial operational capability), each coincidentally fitting neatly into gaps between phases of Rational Unified Process (RUP). This allows some projects to effectively use Spiral and RUP in tandem.
6. Broad focus on the system and lifecycle; the concerns and importance of which are sometimes forgotten.

"Project cycles that omit or shortchange any of these [elements] risk wasting effort by pursuing options that are [either] unacceptable to key stakeholders, or are too risky" (B. Boehm, 1986).

Common misconceptions that lead to 'inauthentic' Spiral implementations include the idea that all project activities follow a single spiral sequence, every activity in the Spiral diagram originally published by Boehm must be performed in the order shown, and that spiral is simply a sequence of Waterfall increments.

The possibility of partitioning product development plans into smaller increments or components leads to an interesting notion in the case of the Spiral model. When individuals or smaller teams are permitted to work on select phases of a project in this way, the appropriate visualisation would then be of a series of parallel spiral cycles, each adding a third dimension to the project as a whole. Use of a 'review-and-commitment' step can therefore lead to either a walk-through of a single programmer's component or a major requirements review that involves customer, user, developer and maintenance organisations.

Through its use, the Spiral model tests the hypothesis that a particular operational mission can be improved by a software effort. If improvements are indeed observed, it is likely that the model will apply equally well to both development and enhancement efforts. If the hypothesis fails at any time or for any reason (including if a superior product becomes available or if delays cause a product to miss its market window), the spiral is terminated immediately. Otherwise, Boehm tells us, "it terminates with the installation of new or modified software, and the hypothesis is [again] tested by observing the effect on the operational mission" (B. Boehm, 1986). New maintenance spirals may be initiated to test further hypotheses about software improvements from time to time, and although they are not included in Figure 4, to simplify presentation (just as initiation, termination and iteration of the tasks and products of previous cycles aren't), they are implicitly defined in the Spiral model.

Other features implicitly defined in the model include the following (B. Boehm, 1986):

- It fosters the development of specifications that are not necessarily uniform, exhaustive, or formal, in that they defer detailed elaboration of low-risk software elements and avoid unnecessary breakage in their design until the high-risk elements of the design are stabilised.
- It incorporates prototyping as a risk reduction option at any stage of development. In fact, prototyping and reuse risk analyses are often used in the process of going from detailed design into code.
- It accommodates reworks on go-backs to earlier stages as more attractive alternatives are identified or as new risk issues need resolution.

"The primary advantage of the spiral model is that its range of options accommodates the good features of existing software process models, while its risk-driven approach avoids many of their difficulties" (B. Boehm, 1986). In some situations the model can become equivalent to one of the pre-existing techniques, simply by the nature and content of its adaptations. Table 1 summarises some of the advantages and difficulties of the Spiral Process Model, as described by Boehm.

Table 1. *Advantages and Difficulties of the Spiral Process Model*	
Advantages	Difficulties
Great amounts of detail are not necessary unless the absence of such detail jeopardises the project.	Matching to contract software.
It focuses early attention on options involving the reuse of existing software.	Relying on risk-assessment expertise.
It accommodates preparation for life-cycle evolution, growth, and changes of the software product.	Need for further elaboration of Spiral model steps.
It provides a mechanism for incorporating software quality objectives into software product development	
It focuses on eliminating errors and unattractive alternatives early in the process. For each of the sources of project activity and resource expenditure, it answers the key question, 'How much is enough?'	
It does not involve separate approaches for software development and software enhancement (or maintenance).	
It provides a viable framework for integrated hardware-software system development.	

Table 5 - Advantages and Difficulties of the Spiral Process Model

Teams not yet ready to commit to full implementation of the Spiral Process Model can still elicit many of its benefits by implementing one characteristic element in conjunction with whatever their current operations may be. An example of this element, a risk management plan, is depicted in Table 2.

Table 2. *Software Risk Management Plan*
1. Identify the project's top 10 risk items.
2. Present a plan for resolving each risk item.
3. Update list of top risk items, plan, and review monthly.
4. Highlight risk-item status in monthly project reviews, and compare with previous month's rankings and status.
5. Initiate appropriate corrective actions.

Table 6 - Software Risk Management Plan

Use of a risk management plan ensures that top risk items pertaining to a project are identified early on, and that a strategy is developed for not only mitigating or responding to those risks, but to any other risk items as they surface. Use of a risk management plan has also been known to ensure an appropriate focus on early simulation, prototyping and benchmarking; risk-resolution techniques that have proven invaluable in avoiding or dealing with failure-inducing occurrences. In fact, DoD-Std-2167, the recent US Department of Defense standard on software management, requires that developers produce and use risk management plans, as does its counterpart US Air Force regulation, AFR 800-14 (B. Boehm, 1986). This and other newer software risk management techniques are paving the way for spiral model concepts to have acquisition applications and greater development capabilities.

Process-based Improvement Models (PIM)

Rico (2002) describes Software Process Improvement (SPI) as the discipline of characterising, defining, measuring, and improving software management and engineering processes, simultaneously leading to successful software engineering management, higher product quality, greater product innovation, faster cycle times, and lower development costs. These techniques are being increasingly used by IT product development organisations around the world, due largely to a realisation that improving the quality of products and services delivered to stakeholders and end users requires development process improvement (Herbsleb, Carleton, Rozum, Siegel, & Zubrow, 1994). Process improvement stems from the premise that product quality is highly dependent upon the processes used in the product's creation (Dorling, 1993). SPI is therefore in some ways 'one step removed' from other methodologies; it seeks to improve any methodology or process used to aid software development, not just the software development process itself. This is achieved through the implementation of evaluation frameworks.

An IBM division in Research Triangle Park, North Carolina provided an early example of successful SPI execution when it pioneered, mastered and used the Software Defect Prevention Process, achieving fifty per cent quality improvement the first time used, and up to ninety-nine per cent quality improvement in other instances, without product appraisal activities like Inspection and Test (Rico, 2002).

All SPI efforts have two main stages: Process Assessment and Process Improvement. The Process Assessment stage is characterised by a disciplined examination of the processes used by an organisation against a set of criteria, to determine the capability of processes to meet quality, cost and schedule goals through measures of strength and weakness (Dorling, 1993). The Process Improvement stage meanwhile consists of self-explanatory actionable steps in response to first-stage findings.

Cost has become an increasingly important factor in choosing an SPI derivative, and a number of cost models have been included here for this reason (Rico, 2002):

1. **Personal Software Process (PSP) Cost Model**: The PSP costs about 50 workdays (or 2.5 months) to execute, producing 10 000 lines of code and resulting in zero defects, repeatable project performance, and software engineering professionalism. The cost model used is 'Source Lines of Code divided by 25'. This is a custom cost model derived from a study by the Software Engineering Institute, and is considered to be more than economical for 10 000 source lines of code.
2. **Hewlett Packard SPI Cost Model**: This SPI cost model clearly indicates that Design Management (reuse) and the Software Inspection Process offer the greatest return on investment. Hewlett Packard has reclaimed $350 million in development costs by using the Software Inspection Process, as displayed in Figure 5.

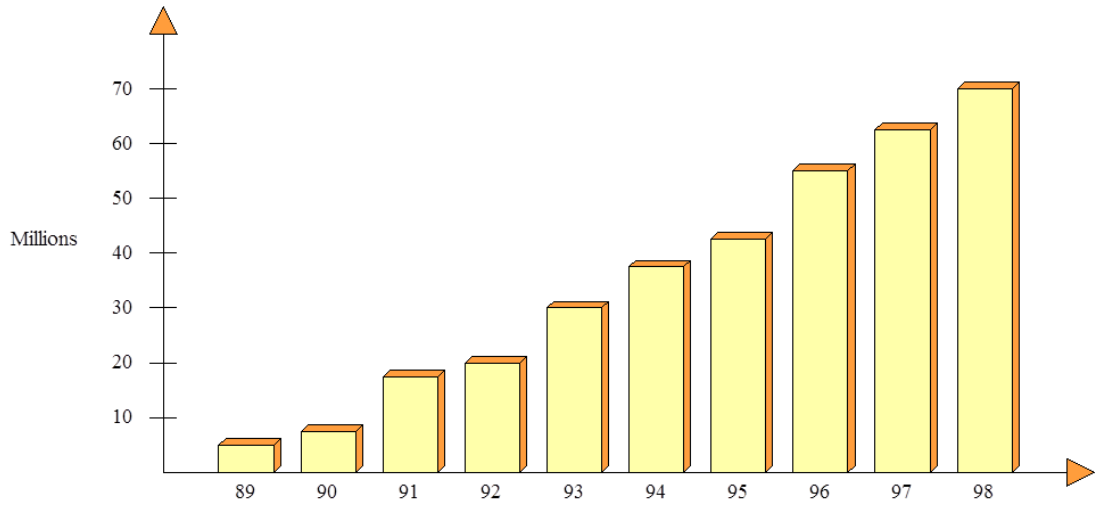


Figure 5 - Software Inspection Process Return on Investment (ROI)

Source: "Successful software process improvement" Rico (2002).

3. **Software Productivity Research (SPR) SPI Cost Model**: This SPI cost model by Capers Jones of Software Productivity Research (year?) indicates that the cost of SPI is rather negligible. This model indicates that it costs a 1000-person firm about $17 000 to achieve industry leadership. The PSP can easily be implemented for that cost.
4. **SPR SPI Effort Model**: The SPI effort model developed by Capers Jones of Software Productivity Research indicates that the time to achieve SPI is also rather negligible. This model indicates that it takes just over 2 years to achieve industry leadership. Once again, the PSP can easily be implemented in less than that amount of time.

Although SPI can be expensive or time-intensive from an organisation's standpoint (especially in terms of time taken to reach new maturity levels), some major benefits include increased productivity, cost savings for outside stakeholders, customer satisfaction and cycle time reduction. However, resistance, inertia, negative experience, lack of evidence of benefits, imposition, resource constraints and commercial pressures are major de-motivators for using SPI (Baddoo & Hall, 2003).

Software Process Improvement and Capability Determination (SPICE)

SPI objectives require that a process be predictable, under statistical control and open to continuous improvement. The lifecycle of that improvement includes phases that assess the current and target levels of process capability, prioritisation of improvement, implementation, monitoring and evaluation. These concepts are all derived from a seven-part series of international standards called ISO/IEC 15504, 'Software Process Improvement and Capability Determination'. Organisations are able to use these standards in many ways:

- in capability determination mode, to help a purchasing organisation determine the capability of a potential software supplier
- in process improvement mode, to help a software organisation improve its own software development and maintenance processes
- in self-assessment mode, to help an organisation determine its ability to implement a new software project (Paulk, Konrad, & Garcia, 1995).

Built on contributions from all identified major stakeholders, SPICE defines processes in terms of their domain, purpose and outcome (process reference model), and provides a measurement framework for evaluating the capability of those processes to reach certain levels of maturity (Dorling, 1993).

Capability Maturity Model Integration (CMMI)

CMMI is one form of SPI that meets SPICE criteria, particularly those elements laid out in its second part. The first process improvement framework of its time, CCMI was developed by the Software Engineering Institute at Carnegie Mellon University in the early 1980s, although it does not appear in published literature until 1993 (the CMM). The CMMI describes the principles and practices underlying software process maturity and aims to help software organisations improve the maturity of their software processes in terms of an evolutionary path from ad hoc, chaotic processes to mature, disciplined software processes. The maturity model written about in 1993 grew from a compilation of best practices observed in early use, with maturity levels organised as follows (Paulk et al., 1995):

Initial The software process is characterised as ad hoc, and occasionally even chaotic. Few processes are defined, and success depends on individual effort and heroics.

Repeatable Basic project management processes are established to track cost, schedule, and functionality. The necessary process discipline is in place to repeat earlier successes on projects with similar applications.

Defined The software process for both management and engineering activities is documented, standardised, and integrated into a standard software process for the organisation. All projects use an approved, tailored version of the organisation's standard software process for developing and maintaining software.

Managed Detailed measures of the software process and product quality are collected. Both the software process and products are quantitatively understood and controlled.

Optimising Continuous process improvement is enabled by quantitative feedback from the process and from piloting innovative ideas and technologies.

CMMI uses three models (constellations) of application: CMMI for development, CMMI for services and CMMI for acquisitions. Each model/constellation is defined by different process categories, and assessment of compliance for implementation is supported by two documents: Appraisal Requirements for CMMI (ARC) and Standard CMMI Appraisal Method for Process Improvement (SCAMPI) (a method description document).

A project's work products serve as the major assessment indicators for SPI methods like CMMI. Some common myths and misconceptions held about SPI and CMMI are listed in Table 2 below (adapted from (Rico, 2002)):

Table 2. SPI and CMMI: Common Myth and Misconceptions

- High quality is too expensive.
- Faster cycle times result in lower quality.
- SPI and high quality are for NASA and DOD.
- Software is purely creative thought
- Process improvement is a long journey.
- CMMI Level 3 is good enough.
- CMMI Level 5 is a utopian state.
- CMMI Level 5 costs more than Level 1.
- SPI is just a fad.
- SPI doesn't affect the bottom line.
- Metrics are too hard and irrelevant.
- Quality can't be measured.
- SPI is too expensive.
- CMMI is too heavy and complex.
- Pervasive myth of partial implementation

CMMI does have some disadvantages including a risk of increased documentation, time for implementation and occasional requirement of additional knowledge and resources. There are also major advantages including the antithesis of most elements in Table 2.

Unified Process (UP)/Rational Unified Process (RUP)

UP and RUP are extremely popular incremental and iterative strategies often used in software development. Unlike other strategies, which exist merely as a 'process', UP and RUP offer more of a modifiable framework that organisations can customise for any project range. UP and RUP are so similar in their aims, methods and results, that these modified forms often give little clue as to which served as the 'parent' management strategy. For this reason, the two titles are often used interchangeably. However, when possible, UP is used to describe elements that are common among most refinements or the generic process itself, because Rational Unified Process and its abbreviation (RUP) are trademarked by developer IBM.

A countless number of UP refinements and variations exist, all of which differ in the number of workflows/disciplines (sequences of continuous steps making up a discrete activity or task within the project). RUP itself has nine of these disciplines, while agile forms such as Agile Unified Process (AUP) and Open Unified Process (OpenUP) aim to simplify RUP and decrease that number. Other refinements include Enterprise Unified Process and Essential Unified Process.

One common UP characteristic includes the use of four phases (inception, elaboration, construction and transition), each partitioned into time-boxed segments. Most refinements also place a large emphasis on the importance of architecture and risk mitigation early in the project lifecycle. The 'executable architecture baseline' is seen as a key deliverable, created during the elaboration phase (through partial system implementation) with the purpose of validating architecture and guiding remaining development. Early risk mitigation is achieved through analysis of critical project elements and the main deliverables. An iterative method is used to create releasable increments with added functionality, and 'use-cases' (or scenarios) are examined in order to make sure that those increments capture functional requirements that will create real value.

Agile

Agile Software Development is a strategy that has informed a large group of methodologies, most importantly through the value placed on an iterative approach. In an analysis of this group of methodologies, Charvat (2003) highlight five (Extreme Programming, Scrum, Crystal, Dynamic System Development, and Rapid Applications Development) that are widely discussed in current software development circles. These methods are often referred to as being 'lightweight,' in contrast with the 'heavier' traditional

Waterfall methodologies that are thought by many to be overly regulated and micromanaged. Agile development methodologies find solutions and meet project requirements through collaboration between cross-functional, self-organising teams; teams that through tight interactions in the development lifecycle are encouraged to have a rapid and flexible response to change.

According to Larman and Basili (2003), most iterative models can be traced back to Walter Shewart's 'Plan, Do, Study, Act' of the 1930s, a system that has greatly evolved and been built upon in the years since.

The Agile approach itself was first developed and described at a February 2001 meeting in Utah, where 17 software developers gathered to discuss lightweight methods. These discussions resulted in the creation of a formal document entitled 'The Agile Manifesto', which outlines 12 important principles necessary for a method to be successful and also earn the 'agile' descriptor. These principles include the use of short timeframes ('time-boxes' typically lasting 1-4 weeks for each iteration), at the end of which a working product is shown to stakeholders as a way of minimising

risk. In this way, Agile methods use minimal and mostly short-term planning. These short iterations make for small feedback loops and frequent meetings, using efficient face-to-face communication, information radiators (physical displays that are centrally located in an office and provide an up-to-date status of software-development) and the appointment of a customer representative on the team (someone to act on the stakeholder's behalf). Much of the work is around how to decide what to do over the next few increments of time, and how to generate the most value in that time (Armour, 2014).

In comparison with traditional software engineering techniques, Agile methods mainly target complex systems projects with dynamic and indeterminate characteristics. The methods recognise that accurate predictions, stable plans and estimates are all difficult to obtain in the early stages when projects are frequently full of unpredictability. In response, the Agile approach seeks to 'inspect and adapt' by conducting tests at different stages during development, instead of separately afterwards. Stakeholders, clients and end-users therefore have recurring opportunities to calibrate releases and preserve market relevance for any project, while decreasing development costs, time to market and waste. According to Nerur, Mahapatra, and Mangalaraj (2005), true Agile methodologies must perceive the customer role as being critical in this way, in contrast with older methods like traditional and V-Model, which hold that role as being merely 'important'. Agile methods have a lot in common with RAD techniques and with CCMI, which is also iterative in nature.

Canadian software engineer Philippe Kruchten (2011) aptly described Agile practices:

> "The agile movement is in some ways a bit like a teenager: very self-conscious, checking constantly its appearance in a mirror, accepting few criticisms, only interested in being with its peers, rejecting en bloc all wisdom from the past, just because it is from the past, adopting fads and new jargon, at times cocky and arrogant. But I have no doubts that it will mature further, become more open to the outside world, more reflective, and also therefore more effective."

Following is a more detailed examination of Agile software development methodologies.

Extreme Programming (XP)

The most pervasive and commonly implemented Agile software development methodology, XP distinguishes itself by focusing on a select, consolidated group of Manifesto values, and specifying them in the form of simple practices. While other Agile techniques might seek to predict and include all practices that 'could' ever be needed, XP seeks to organise a minimalistic list that includes only definite practice requirements, and then limit a project's needs so as not to exceed their capabilities. According toLindstrom and Jeffries (2004), "[the] significance of this difference cannot be understated". Through this narrow focus, the technique is able to take these beneficial elements of traditional software engineering methods to 'extreme' levels, thereby maximising their positive aspects while allowing room for adaptation and change.

Though early forms are evident in some of NASA's Project Mercury in the 1960s, Kent Beck is largely credited with developing XP through his involvement with the Chrysler C3 project (Beck & Andres, 2005). XP involves 'cranking the knobs up to 10' on all that works and is essential, while leaving everything else out. The most frequent critique of XP is that it is too simple to work beyond a narrow set of project criteria, yet known successes continue to stretch perceived capabilities.

The 13 interdependent core practices of XP are:

1. **Whole Team**: Every contributor to the project is considered to be a member of the team. The team is divided into (two main) sections (comprised of either an individual or multiple members), each fulfilling a key role. The customer is often a real end user who is familiar with the domain and what is needed; able to act as a business representative by providing requirements, setting priorities and steering the project. The programmer implements the customer's requirements. In keeping with the theme of simplicity, most XP literature describes these two roles as being filled by an individual, when in practice, each can consist of a large group of like-minded workers who communicate with one voice. Other sections can include testers, analysts, and even a coach or manager. "Everyone on an XP team contributes in any way that he or she can; [the] best teams have no specialists, only general contributors with special skills" (Lindstrom & Jeffries, 2004). The rhythmic interaction between those filling the two main roles in an XP project is depicted in Figure 7.

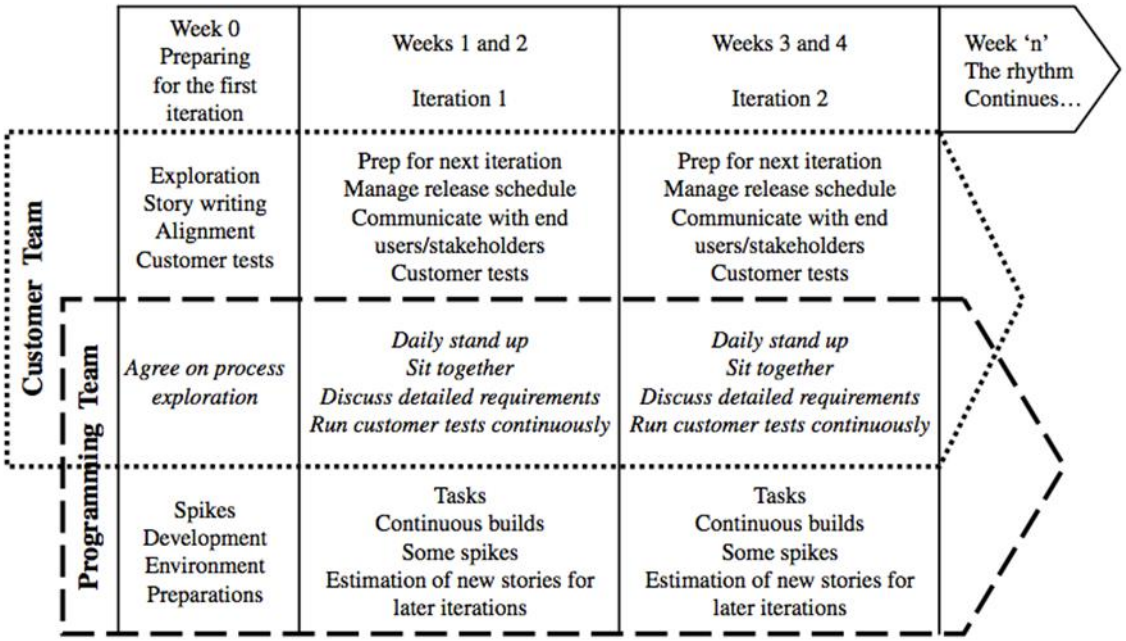


Figure 7 - Interaction of Major Roles in an XP Project

2. **Planning Game**: XP planning focuses on determining what will be accomplished by a given due date, and what is to be done after that mark is reached. In this way, an emphasis is placed on straightforward 'steering' rather than exact prediction of requirements and the length of associated time intervals. Two key planning steps are used to facilitate this strategy: release planning and iteration planning.

   In release planning, the customer team presents desired features to members of the programming team, who estimate their difficulty and cost. The customer then uses that information and their own knowledge of varying feature importance to create a revised project plan. Release plans are often imprecise, and the quantitative information laid out within does not retain reliable accuracy until work begins; the plans are often revised for this reason.

   Iteration planning is a term used to describe the direction given to the team by the customer prior to each two-week iteration. The features desired are broken down by programmers into cost estimates (retaining more detail than those in the release plan) and tasks which are then allocated to different workers for the coming iteration. The resulting software deliverables are always both useable and useful.

Lindstrom and Jeffries (2004) note that these planning steps provide excellent steering control in the hands of the customer, and a "focus on visibility that results in a nice little paradox: [on] the one hand,...the Customer is in a position to cancel the project if progress (entirely visible every couple of weeks) is not sufficient, [while on] the other hand...the ability to decide what will be done next is so complete that XP projects tend to deliver more of what is needed, with less pressure and stress" (page reference?).

3. **Customer Tests**: XP customers define one or more automated acceptance tests, which the team then builds and uses to prove to all parties that a desired feature has been implemented correctly. Automation is important to save time and ensure that the working test is performed regularly, resulting in constant improvement and no backtracking.

4. **Small Releases**: XP teams release running, tested software that delivers real business value (as determined by the customer) to both the customer and other end-users (depending on the project and obvious legalities) at iteration's end. Customers themselves are encouraged to pass these releases on to end-users if the programming team is not permitted to or has not done so already. These releases allow for regular evaluations and the obtainment of relevant feedback, while also (importantly) providing visibility; keeping things open and tangible. In some cases, releases can occur more or less frequently than an iteration span would dictate. Web projects may provide releases as often as every day, while those pertaining to 'shrink-wrap' products may be as infrequent as once per quarter. While the creation of quality output in these short timeframes may seem unlikely, standards are maintained through XP's "obsession with [both customer tests and test-driven development] testing" (Lindstrom & Jeffries, 2004).

5. **Simple Design**: The building of software with XP requires that designs be simple from the outset and remain that way; qualities that are aided by programmer testing and design improvement. Software is intended to be perfectly suited for current functionality, while always remaining prepared for what's next. XP design is neither up-front, nor one-time; it is a constant presence that informs all steps in project development, always ensuring that there is no wasted motion.

6. **Pair Programming**: XP takes the documented benefit of code reviews present in traditional software development, and implements an extreme form (continuous code review), made possible through pair programming. Two programmers (or teams of two) build all production software (or a component thereof) while sitting side-by-side at the same machine. Described in 1999 by Beck and Andres (2005) as being a "handcuffing of two programmers" (2005: page reference?), the method has been demonstrated to deliver better code in terms of flexibility, usability, maintainability and extendibility than two developers who are working individually, and in less time (Wood & Kleb, 2003).

7. **Test-Driven Development**: In order to ensure that good feedback is constantly available and utilised, XP teams practice 'test-driven development', a method in which tests are incorporated often, or performed following very short cycles of work. In this way, the code produced frequently has close to 100 percent test coverage. Every time a programming pair releases code to the repository, it must pass the many unit tests that have been created and collected for the project as a whole. This results in the provision of immediate feedback and an expectation of 100 per cent operational efficiency at all times. Another example of taking things to the extreme, XP teams will often write automated tests inside the software being developed; tests that will validate even small sections of coding. This is in contrast to the testing procedures that are characteristic of more traditional methods, which either occur separately (in parallel) or afterwards, with a focus on larger features.

8. **Design Improvement**: A process of continuous design improvement called 'refactoring' is used to ensure good design is created and maintained, allowing the delivery of business

value with every iteration. The main focuses of refactoring include the removal of duplication and "increasing 'cohesion' of code while lowering the 'coupling'" (Lindstrom & Jeffries, 2004). According to Martin (2003), manipulating these qualities in such a way has been recognised as a hallmark of well-designed code for at least 30 years. In this way, the high-quality, simplistic design achieved at the outset is maintained, allowing the XP team to sustain development speed, and in some cases, to increase it as a project moves forward. Customer and programmer tests ensure that nothing is broken while the design evolves through refactoring, making them a critical enabling factor. This is evidence of the belief that "XP practices support each other: they are stronger together than separately" (Lindstrom & Jeffries, 2004).

9. **Continuous Integration**: Lindstrom and Jeffries (2004) correctly describe the weaknesses of infrequently integrated code in their assertion that it is "buggy...created by a team that is not practiced at integration (despite that step's critical importance) and unfamiliar with the system as a whole" (2004" page reference?). These characteristics all lead to serious problems in software development projects, the biggest of which is probably a high incidence of 'code freezing': long periods where programmers are unable to work on important shippable features. The features are then 'held back' for later versions, which results in a weakening of market position and end user satisfaction. To combat these occurrences and what has become known as 'integration hell', XP teams keep the system fully integrated at all times, often building multiple times per day.

10. **Collective Code Ownership**: In much the same way that infrequent integration is undesirable for software development projects, so too is individual code ownership. Ugly, hard-to-maintain code is created with such ownership, as programmers (noticing the need, but failing to understand correct placement in code they did not originally write) often put required features in the wrong place. XP is therefore a proponent of collective code ownership, where any programming pair can improve any code at any time. Code quality is increased and defects are reduced in this way, as all code benefits from the input of many people. The main danger of people working blindly on code they do not understand is avoided by pair programming, and by programmer tests catching mistakes. Again showcasing the strength of combined principles, paired programming and collective code ownership encourage the spreading of knowledge throughout the team.

11. **Coding Standard**: In support of collective ownership, all code is written to follow a common standard. It therefore appears as if it was written by a single individual.

12. **Metaphor**: A common system of names or a simple, evocative description is used to ensure that all team members understand how a system is intended to work, where to look for a given functionality or where to place new functional elements being added. This naming system is intended to be representative of a common vision, and may or may not make use of vivid or poetic imagery. The example given by Lindstrom and Jeffries (2004) of a metaphor that could be used for an agent-based information retrieval system is as follows: "this program works like a hive of bees, going out for pollen and bringing it back to the hive." (2004: page reference?)

13. **Sustainable Pace**: XP teams aim to work hard, at a pace that can be sustained indefinitely. In contrast with older, 'death march' projects that are neither efficient nor successful in producing quality software, XP projects balance a vision, however distant, with a commitment to survival. This is achieved by submitting to overtime when it is deemed effective and aiming to maximise productivity week after week.

The core XP practices described above can be organised collectively as a cycle of activities, as illustrated by Figure 8 below. The inner circle describes the tight cycle executed by the programming team, while the outer loop describes the planning cycle that occurs as customers and programmers

interact. The loop in the middle of the diagram includes the practices that increase communication and coordination, resulting in the delivery of quality software.

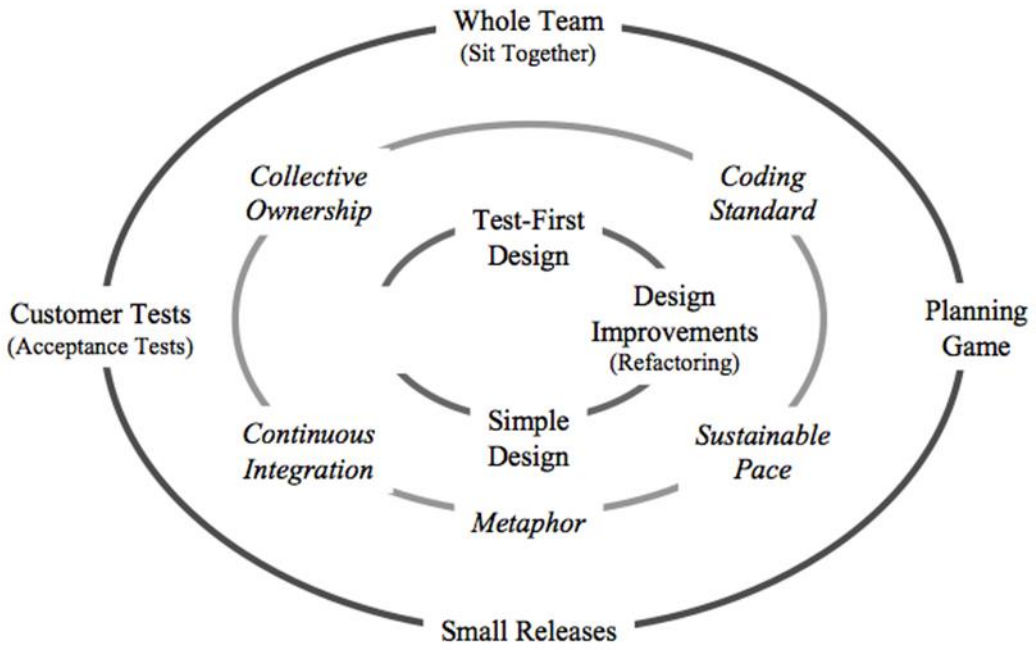


Figure 8 - Extreme Programming, Core Practices Illustrated as an Activity Cycle

In terms of project governance, a management team typically works with both customer and programmer on all but the smallest of projects, allocating resources, removing any obstacles that impede progress and managing alignment to the goals of the business. Unlike many other Agile methodologies however, XP does not specify management practices. Instead, it attempts to simplify things by empowering the customer and programmer alike to make most project decisions, creating what has been termed a 'flat management structure'. It is because of this that many XP projects are referred to as being 'self-managed'. More management is required to coordinate the efforts of different teams as projects grow in size and complexity. In terms of scalability, XP has traditionally only worked with teams of 12 or fewer members. However, this number problem can be avoided by breaking projects and teams into smaller factions, each responsible for different parts of the four most basic activities required in software development: coding, testing, listening and designing.

While teams that are new to XP are encouraged to focus on using and developing skills within these practices, they should also continue to check their proficiency and tailor the practices (adapt, add or eliminate them) to project needs as time goes on. As a team matures in its use of XP, it may become increasingly apparent to its members that the practices alone do not fully capture the methodology's essence, and that the use of an additional set of attributes, called XP Values, is an equally large determinant of success for projects developed with the technique. The explicit description of both practices (what to do) and values (how to react when those practices fail) is significant and unique, as XP remains the only methodology to make both provisions.

> "While most methods are specific on practices, and some specify principles...few combine both. It is likely true that skilled practitioners of most methods are guided by a set of values, perhaps dictated by the culture of the organization...or the leadership of the team, [but] it is the expression of the values as integral to the practices that makes XP unique. The practices leverage the values to remove complexity from the process, and guide actions on a project. XP is [therefore] a discipline of software development based on values of simplicity, communication, feedback and courage. It works by bringing the whole team together in the presence of simple practices, with enough

> feedback to enable [them] to see where they are and to tune the practices to their unique situation" (Lindstrom & Jeffries, 2004).

Jeffries, Hendrickson, and Anderson (2001) summarise the technique nicely-

> "The essence [of XP] truly is simple. Be together with your customer and fellow programmers, and talk to each other. Use simple design and programming practices, and simple methods of planning, tracking, and reporting. Test your program and your practices, using feedback to steer the project. Working together this way gives the team courage."

Scrum

Scrum and Scrum-ban are a scalable, team-based all-at-once approach to IT project management.

An early example of Scrum implementation is provided by the work of Jeff Sutherland in building the first object-oriented design and analysis tool incorporating round-trip engineering at Easel Corporation in 1993. In a thorough review of software development processes, Sutherland and his co-workers determined that the Waterfall approach and current 'all-at-once' models would fail to meet their needs. An enhanced version of rapid application development would be required: one that allowed the creation of working code to quickly follow design visualisation, through a more scalable and team-based all-at-once approach than was currently available (Larman, 2004).

The assigning of value to these criteria was motivated by the success of the Japanese approach to new product development; particularly the work of Takeuchi, Nonaka, Honda, Canon and Fujitsu (Takeuchi & Nonaka, 1986). In Sutherland's own words, "the idea of building a self-empowered team where everyone had [a] global view of the product on a daily basis seemed to be the right one" (2004). Sutherland received further inspiration from a paper written about Borland's development of Quattro Pro for Windows, by Coplien (1994). The paper described the Quattro team as "having the ability to deliver 1 million lines of C++ code in 3 months" (equivalent to 1000 lines of deliverable code per person per week, and the most productive software project ever documented at that time). This level of productivity was attributed to intensive interaction among all parties in the type of daily meetings we now consider to partially characterise Scrum. Sutherland persuaded the development team at Easel to set up its first Scrum meeting soon after reading this material.

Originally called the Holistic Rugby Approach, Scrum derives its name from the manner in which a rugby game is restarted following a minor infraction. This re-starting is mirrored in software development scenarios that recognise customers may change their minds about what they need and want over the course of a project, which is a key principle of Scrum. As with other Agile techniques, Scrum acknowledges that unpredicted challenges cannot be easily addressed in a planned or traditional manner. Instead of wasting energy and resources in that type of planning, it is accepted that responding to emerging requirements and providing quick deliverables is more valuable. This observational and empirical approach is most effective when used by a cross-functional team across a number of overlapping phases.

Much like in the game of rugby, the development team attempts (through self-organisation, co-location and good communication) to 'go the distance' as a unit and reach a common goal. Members begin by collecting and organising all product requirements as laid out by the owner, many of which may have originally been contributed by external sources like executives, stakeholders and customers; this becomes the product backlog. A number of the requirements are then taken from the 'top' of this list, as many as can be completed within a first iteration of effort. The metaphor is carried one step further by referring to these time-bound iterations as 'sprints,' which are preceded and followed by meetings to plan and review work.

As per the Scrum pattern, meetings are to be kept short (typically under 30 minutes) and require all participants to answer three key questions:

- What did you do yesterday?
- What will you do today?
- What obstacles did you encounter that you might require help with?

(Beedle, Devos, Sharon, Schwaber, & Sutherland, 1999; Herbsleb et al., 1994)

Sutherland (2004) describes the massive value that these meetings have, allowing everyone on the project team to see the status of all project aspects in real time, and the "collective neural networks of the team's mind to fine-tune or redirect efforts on a daily basis to maximize throughput" (Sutherland, 2004). The sharing of software resources in this way allows development tasks originally estimated to take days to be completed in a few short hours. A working and potentially shippable product is expected at the end of each sprint.

In terms of project governance, the Scrum approach to planning and managing projects aims to bring decision-making authority to the level of operations through the allocation of work among three core roles:

- The Product Owner represents stakeholders and acts as an accountable customer voice, ensuring value is delivered by the development team to the end-user or business.
- The Development Team is a self-organising group, usually comprised of 3-9 individuals who perform the actual software development work. The team has limited interaction with traditional project management entities, and is responsible for delivering potentially shippable increments of a product following each sprint.
- The Scrum Master does not act as a traditional project manager or team lead, but instead provides a buffer between the development team and distracting influences, maximising their ability to meet project goals and provide deliverables. The Scrum Master facilitates the rules and process of scrum by chairing meetings and challenging the Development Team to improve, becoming a 'servant-leader' of sorts. They also do not have any obligations to people management in the way that traditional Project Managers might. In fact, a correctly implemented Scrum strategy will not employ the use of a Project Manager at all, but instead will divide those duties among the three core roles, primarily between the Development Team and Product Owner.

The characteristics and inherent roles of the Scrum methodology are illustrated in Figure 6 below.

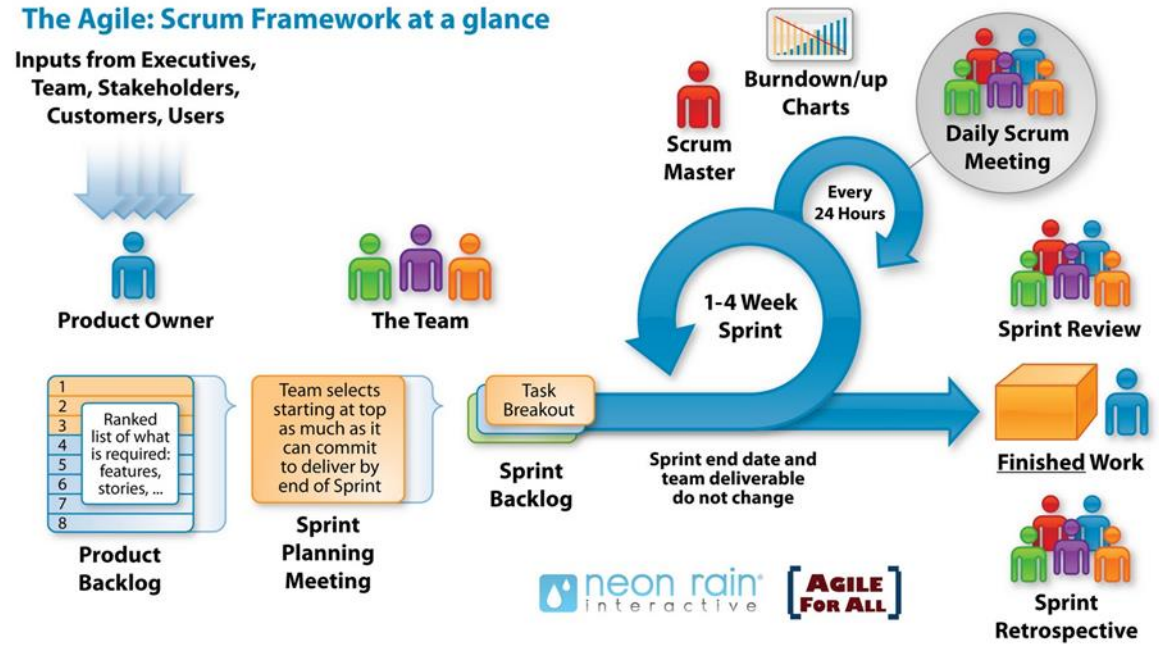


Figure 6 - The Scrum Framework at a Glance

Source: 'Art of project management: what is agile scrum?', Kumar (2011).

Easel Corporation's CEO was wary of Scrum implementation at first, having received years of projects that were supported by Gantt charts and extensive reports and none of which had been able to deliver the required functionality on time. While these other projects always appeared solid on paper, no interim view of the software's status was ever available; this made discovery of slippage in time to reforecast company revenue a rare occurrence. Scrum's delivery of working code at the end of each sprint provided a welcome change and more confidence than extensive documentation with no operational system. Through these potentially shippable increments, management saw significant step-by-step progress, and the creation of a satisfactory product even before reaching the final sprint that had been scheduled. This provided early evidence of what are now recognised as two common Scrum capabilities: meeting deadlines and delivering more functionality than expected. It also became clear that there was little evidence to support "a return to [the] waterfall type mentality" in software development, as Scrum successfully addressed all of its major flaws through "an inability to predict, inability to deliver on time, less functionality production per developer unit of time, and user dissatisfaction due to lack of involvement" (Sutherland, 2004).

Scrum-ban

Scrum has contributed to tens of thousands of projects, many of which have been undertaken by leading software development companies worldwide. It has even influenced the creation of newer 'hybrid' methodologies that draw upon its strengths, like Scrum-ban.

Scrum-ban combines the principles of Scrum with another management strategy known as Kanban. Kanban's contributions include the visualisation of separate work stages and limiting the number of unfinished work defects allowed to exist simultaneously. Tasks are categorised in various ways (most simply as being 'yet to start', 'ongoing' and 'completed') and are often posted on a large board/information radiator to illustrate stages of work for teams in the same physical environment. Each team needs to find its own flexible limiting values for unfinished work, however a common rule of thumb suggests that no team member have more than two simultaneous tasks, and that not every member can have two tasks. The time-limited sprints regularly employed in Scrum are removed as

they are of no real use, and the workflow instead becomes more continuous. The Development Team may also be more specialised, with its members having less broad inter-subject knowledge. Other practices from regular Scrum such as daily meetings are still utilised.

The combination results in a method that is especially useful for maintenance of software systems and addressing projects with programming errors or frequent and unexpected work items.

Crystal Methods

In the mid-1990s, Alistair Cockburn noticed that some software development teams were delivering successful projects without following formal management methodologies. After interviewing and studying these teams, Cockburn compiled his findings and described a new family of methodologies, called the Crystal methods. The word 'crystal' was chosen as a metaphor, where the many facets or faces of a gemstone represent various views on the central 'core' principles of software development. The inward-facing 'views' pertain to tools, techniques, standards and roles.

An important idea for Crystal method teams is that an expectation of varied skill-set allows 'process' to become secondary to community, interaction, people and talent. This stems from observed behaviour and belief in assumptions about people and human nature; that although they can have highly variable behaviour and trouble acting consistently across time and location, there is an inherent desire to take initiative and accomplish what is needed through good communication. The multi-talented team members can undertake similar and interdependent activities in a number of different ways, making Crystal some of the easiest Agile methods to apply.

The individual methods that comprise the Crystal family are divided across a colour spectrum. Larger projects and those with greater risk, criticality and importance (heavier projects) are represented by progressively darker colours. Compared with traditional management approaches, Crystal methods are more tolerant, flexible and resistant to strict and rigid processes. Cockburn found seven underlying principles common to most of the Crystal methods, and postulated that the more that are incorporated, the more a project is likely to succeed. These principles include frequent delivery, reflective improvement, close/osmotic communication, personal safety, focus, easy access to expert users, having a technical environment with automated tests, configuration management and frequent integration.

Lean Development (LD)/Six Sigma

Lean Software Development has its roots in the Six-Sigma and Lean management methodologies originally defined by the Toyota Production System (reference?). The manufacturing practice considers any resource expenditure that does not create value for the end customer to be wasteful and in need of elimination. Modifying its principles to fit the software development domain does not change that standpoint. Seven principles of the Lean methodology include eliminating waste, amplifying learning, delivering as fast as possible, deciding as late as possible, empowering the team, building integrity in and 'seeing the whole.'

DMAIC is a data-driven improvement cycle, and is the core tool that drives most Lean and Six-Sigma projects. The method is however not unique to Six-Sigma frameworks, as it can be applied to other applications for improvement. The acronym stands for 5 steps that are always required and must proceed in the order given: Define, Measure, Analyse, Improve and Control.

Dynamic Systems Development Method (DSDM)

DSDM was developed in 1994 by a consortium seeking to develop and promote a more disciplined, independent RAD framework (reference?). Initially hoping to achieve this by combining their own experiences of RAD best practices, members of the consortium settled on a method that fixes quality

and cost from the outset, as well as time, using a scope prioritisation methodology MoSCoW assigns a label - must, should could or won't have- to deliverables over the project lifecycle in order to ensure that time constraints are met. Because time and resources are fixed, in contrast with other methods, project requirements themselves become the main variable in DSDM. Other core techniques involved include time-boxing, testing, prototyping, modelling, configuration management and the hosting of stakeholders in discussion workshops (see method 4, JAD).

The most recent form of this methodology is called DSDM Atern, a name that references a collaborative bird that can travel large distances (the Arctic Tern). Other forms include Versions 4 and 4.2. Atern's approach maintains that more projects fail because of problems with people than technology, and the focus is therefore on helping people work effectively together. In this way, the method essentially becomes 'vendor-independent' and applicable for any business or environment. DSDM is most useful for addressing exceeded budgets, missed deadlines, lack of user involvement and commitment from high-level management, which many would recognise as being the common sources of failure in many IT projects. Although missing the element of meta-modelling, RUP is the most similar methodology to DSDM.

Four factors have been identified as critical to the success of projects using DSDM, including the need for:

- a supportive relationship between vendor and customer
- a commitment by management to ensure end-user involvement
- acceptance of the DSDM itself by that same management
- a stable project team comprised of skillful members who are empowered with the freedom to make important decisions, and given the appropriate environment and technology to have those decisions result in actionable steps.

Feature Driven Development (FDD)

FDD blends a number of industry-recognised practices with client-valued functionalities, or features, following the work of Jeff DeLuca at a large Singapore bank and that of Peter Coad before him (Reference?). There are typically eight best practices from software engineering that are focused on, including:

- domain object modeling
- developing by feature
- individual class (code) ownership
- feature teams
- inspections
- configuration management
- regular builds
- visibility of progress and results.

FDD is model-driven and uses a short iteration process that follows five basic activities: developing an overall model, building a feature list, planning by feature, designing by feature and building by feature. The features described in these activities are usually relatively small tasks, but it remains important to define milestones to mark and measure progress for each.

A series of six common milestones are used for each feature, reached sequentially and allowing description by percentage complete as they progress. The first three milestones (domain walkthrough, design and design inspection) are placed in the second last activity, designing by feature, and the last three milestones (coding, code inspection and promote to build) are found within the last activity, building by feature.

References

38500, I. I. (2008). *Corporate Governance of Information Technology*. International Organization for Standardization.

(CO), C. O. (2009). *PRINCE2: Managing Successful Projects with PRINCE2* (5th ed.). Norwich, UK: TSO (The Stationery Office).

Armour, P. G. (2014). Estimation is not evil. *Commun. ACM, 57*(1), 42-43. doi: 10.1145/2542505

Baddoo, N., & Hall, T. (2003). De-motivators for software process improvement: an analysis of practitioners' views. *The Journal of Systems and Software, 66*, 23-33.

Beck, K., & Andres, C. (2005). *Extreme Programming Explained: Embrace Change* (2nd ed.): Addison-Wesley.

Beedle, M., Devos, M., Sharon, Y., Schwaber, K., & Sutherland, J. (1999). SCRUM: An Extension Pattern Language for Hyperproductive Software Development, in Pattern Languages of Program Design.

Bloch, M., Blumberg, S., & Laartz, J. (October 2012). Delivering large-scale IT projects on time, on budget, and on value. Retrieved 2 January, 2014, from http://www.mckinsey.com/insights/business_technology/delivering_large-scale_it_projects_on_time_on_budget_and_on_value

Boehm, B. (1986). A spiral model of software development and enhancement. *SIGSOFT Softw. Eng. Notes, 11*(4), 14-24. doi: 10.1145/12944.12948

Boehm, B. W. (1988). A spiral model of software development and enhancement. *Computer, 21*(5), 61-72. doi: 10.1109/2.59

Boehm, B. W., Penedo, M. H., Stuckle, E. D., Williams, R. D., & Pyster, A. B. (1984). A Software Development Environment for Improving Productivity. *Computer, 17*(6), 30-44. doi: 10.1109/MC.1984.1659160

Brown, J. T. (2008). *The Handbook of Program Management*. New York: McGraw-Hill.

Charvat, J. (2003). *Project Management Methodologies: Selecting, Implementing, and Supporting Methodologies and Processes for Projects*: John Wiley & Sons.

Connolly, B. (2014). Top 10 Enterprise IT Disasters. Retrieved from http://www.cio.com.au/article/542245/top_10_enterprise_it_disasters/?pp=2

Coplien, J. O. (1994). *Borland Software Craftsmanship: A New Look at Process, Quality and Productivity*. Paper presented at the 5th Annual Borland International Conference, Orlando, Florida.

Cusumano, M. A., & Smith, S. (1995). Beyond the Waterfall: Software Development at Microsoft. *MIT Sloan School of Management*.

Dorling, A. (1993). SPICE: Software Process Improvement and Capability Determination. *Software Quality Journal, 2*(4), 209-224. doi: 10.1007/BF00403764

Florentine, S. (2013). Why Are So Many IT Projects Failing?

Garland, R. (2009). *Project Governance: A practical guide to effective project decision making*. UK and USA: Kogan Page Limited.

Group, T. S. (2013). Chaos Manifesto 2013.

Hepworth, P. (2014). CEO of AXELOS, owner of ITIL and PRINCE2 praises the strength and passion of the Global Best Practice community (pp. 2): AXELOS.

Herbsleb, J., Carleton, A., Rozum, J., Siegel, J., & Zubrow, D. (1994). Benefits of CMM-Based Software Process Improvement: Initial Results. *Carnegie Mellon University*.

Jeffries, R., Hendrickson, C., & Anderson, A. (2001). *Extreme Programming Installed*. Boston: Addison-Wesley.

Jurison, J. (1999). Software Project Management: The Manager's View. *Communications of the Association for Information Systems, Volume 2*(Article 17).

Koh, A., & Crawford, L. (2012). Portfolio Management: The Australian Experience. *Project Management Journal, 43*(6), 33-42. doi: 10.1002/pmj.21300

Kumar, S. (2011). What is Agile Scrum? Retrieved from http://satheespractice.blogspot.ca/2011/11/what-is-agile-scrum.html

Larman, C. (2004). *Agile and Iterative Development: A Manager's Guide*: Addison-Wesley Professional.

Larman, C., & Basili, V. R. (2003). Iterative and Incremental Development: A Brief History. *Computer, 36*(6), 47-56. doi: 10.1109/mc.2003.1204375

Liew, R. (2013). Myer website crashes during Boxing Day sale. Retrieved 21 January, 2014, from http://www.smh.com.au/business/retail/myer-website-crashes-during-boxing-day-sale-20131226-2zx6j.html

Lindstrom, L., & Jeffries, R. (2004). Extreme Programming and Agile Software Development Methodologies. *Information Systems Management, 21*(3), 41-52.

Martin, R. C. (2003). *Agile software development: principles, patterns, and practices* Upper Saddle River, N.J: Prentice Hall.

Murray, A., Bennett, N., Edmonds, J., Patterson, B., Taylor, S., & Williams, G. (2009). *Managing Successful Projects with PRINCE2*: The Stationery Office.

Nerur, S., Mahapatra, R., & Mangalaraj, G. (2005). Challenges of migrating to agile methodologies. *Commun. ACM, 48*(5), 72-78. doi: 10.1145/1060710.1060712

O'Sullivan, M. (2013). Virgin Australia flight delays after check-in system crashes worldwide. Retrieved 21 January, 2014, from http://www.smh.com.au/travel/travel-news/virgin-australia-flight-delays-after-checkin-system-crashes-worldwide-20130806-2rck8.html

OGC. (2010). Portfolio, Programme and Project Management Maturity Model. UK: Office of Government Commerce in the United Kingdom.

Paulk, M. C., Konrad, M. D., & Garcia, S. M. (1995). CMM Verus SPICE Architectures. *IEEE Software Process Newsletter, 3*, 7-11.

PMI. (2013). *A Guide to the Project Management Body of Knowledge* (5th ed.). Newtown Square, Pennsylvania 19073-3299 USA: Project Management Institute, Inc.

Rankins, G. J. (2009). Comparing PMBoK and PRINCE2 in 2009.

Rehacek, P. (2014). Standards ISO 21500 and PMBoK Guide for Project Management. *International Journal of Engineering Science and Innovative Technology (IJESIT), 3*(1).

Rico, D. F. (2002). Software Process Improvement.

Rowen, R. B. (1990). Software project management under incomplete and ambiguous specifications *IEEE Transactions on Engineering Management, Volume 37*(Issue 1), Pages 10 - 21 doi: 10.1109/17.45260

Royce, W. W. (1970). Managing the Development of Large Software Systems. *IEEE*, Pages 1-9.

Sargeant, R., Hatcher, C., Trigunarsyah, B., Coffey, V., & Kraatz, J. A. (2010). Creating value in project management using PRINCE2.

Satalkar, B. (2011). Waterfall Model Vs. V Model. 2014, from http://www.buzzle.com/articles/waterfall-model-vs-v-model.html

Standard, A. N. Z. (2013). AS/NZS 8016:2013 Governance of IT enabled projects: SAI Global Limited under licence from Standards Australia Limited and Standards New Zealand.

Takeuchi, H., & Nonaka, I. (1986). The New New Product Development Game. *Harvard Business Review*.

UK, O. o. G. C. (2010). P2MM – PRINCE2 Project Management Self-Assessment. UK: Office of Government Commerce.

Williams, G. (2010). PRINCE2 - Maturity Model (P2MM). In G. C. Limited (Ed.): Office of Government Commerce.

Wood, W. A., & Kleb, W. L. (2003). Exploring XP for scientific research. *Software, IEEE, 20*(3). doi: 10.1109/MS.2003.1196317

Zealand, S. A. L. S. N. (2013). Governance of IT enabled projects. Australia: Standards Australia Limited.