

Agile Software Development as a Response to Complexity

by
Wayne Austen Pringle-Wood

*Thesis presented in fulfilment of the requirements for the degree of
Master of Philosophy (Information and Knowledge Management)
at the University of Stellenbosch*



Supervisor: Prof. Bruce W. Watson

April 2014

Abstract

Organisations are faced with ever increasing complexity. While there are many responses to complexity just as there are many definitions of complexity this thesis highlights the use of agile software development as a useful method. The case of agile software development is influenced by its people first approach and minimal process implementation to implement constraints in which phenomena of complexity can be understood.

Chapter 1 introduces why complexity is relevant in the organisation today and the issues associated with complexity. A basic introduction to agile software development and why it is a response to complexity is tabled.

Chapter 2 investigates the concepts of complexity. To highlight the difference between linear and non-linear systems and reductionistic thinking the ideas that not all things are complex is explored. Key concepts of complex systems are described to come to a better understanding of what entails a complex system. Information theory and dynamic systems are discussed including system attributes such as attractors and bifurcations. Finally the theory of complex adaptive systems is presented and a classification of all complexity theories is tabled.

Chapter 3 is centred on agile software development, presenting practices and processes as a understanding of how agile software development is applicable to complexity. Three agile methods are identified.

Chapter 4 presents the ideas of modelling and limits to understanding. Models of complex systems are useful but are limited, due to the properties of complex systems. The concept of mental models leads to models of organisations and how leaders need to help with transformation of these models towards models that are more aligned to agile type thinking.

Chapter 5 looks at the core concepts, and practices of agile applied to complexity and why these are relevant in responding to complexity. Towards the end of chapter the role of narrative is explored in the terms of problem definition and solutioning in complex systems.

Chapter 6 concludes the thesis with insights where agile software development is an appropriate response to complexity and the conditions in which it is not.

Opsomming

Organisasies staar toenemende kompleksiteit in die gesig en alhoewel daar verskeie maniere bestaan om kompleksiteit teen te werk, asook verskeie definisies van wat kompleksiteit is, fokus die tesis op aanpasbare ("agile") sagteware ontwikkeling as 'n bruikbare metode. Met eienskappe soos "mense/verbruiker/gebruiker – eerste" benadering asook minimale proses implementering, verskaf aanpasbare sagteware ontwikkeling die raamwerk waar binne die konsep kompleksiteit verstaan kan word.

Hoofstuk 1 bespreek die kwessies rondom kompleksiteit en die relevansie daarvan in organisasies vandag. Verder word aanpasbare sagteware ontwikkeling kortliks bespreek en hoe dit gebruik kan word om kompleksiteit te verstaan, word getoon.

Hoofstuk 2 ondersoek die verskillende konsepte rondom kompleksiteit. Daar word gekyk na linêre en nie-linêre stelsels asook die oortuiging dat nie alle dinge kompleks is nie word geondersoek. Sleutel kenmerke van komplekse stelsels word beskryf om sodoende 'n better begrip te kry van wat 'n komplekse stelsel behels. Informasie teorie en dinamiese stelsels word bespreek, insluitend kenmerke soos "attractors" en "bifurkasies". Laastens, word die teorie rondom komplekse aanpasbare stelsels bespreek en 'n klassifikasie van alle komplekse teorieë word uiteengesit.

Hoofstuk 3 fokus op "agile software development" en hoe sulke praktyke en prosesse toepaslik is op kompleksiteit. Daaropvolgend word drie aanpasbare metodes ge-identifiseer en individueel bespreek.

Hoofstuk 4 verduidelik die idees rondom modellering en grense van begrip. Modelle van komplekse stelsels kan nuttig wees, maar weens die eienskappe van komplekse stelsels, is die nuttigheid beperk. Die konsep van mentale modelle lei tot modelle van organisasies en hoe leiers hierdie modelle moet transformeer na modelle wat meer in-lyn is met agile denkwyses.

Hoofstuk 5 kyk na die kern konsepte en praktyke waar "agile" toegepas word tot kompleksiteit en waarom dit relevant is om kompleksiteit teen te werk. In die tweede helfde van die hoofstuk word die rol van narratiewe ondersoek in terme van probleem-identifisering en die soek na oplossings in komplekse sisteme.

Hoofstuk 6 sluit af met insigte tot aanpasbare sagteware ontwikkelingsmetodes as toepaslike oplossing tot kompleksiteit asook die toestande waaronder dit geld.

Declaration

By submitting this thesis/dissertation electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the sole author thereof (save to the extent explicitly otherwise stated), that reproduction and publication thereof by Stellenbosch University will not infringe any third party rights and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

December 2013

Acknowledgements

This is dedicated to my mother; thank you for all you all you for everything. I would like to thank my friends and family for their support and patience during the completion of this thesis.

Sincere thanks goes to my supervisor Prof Bruce Watson for the guidance and insight throughout the process.

Table Of Contents

Abstract.....	I
Declaration.....	III
Acknowledgements.....	IV
Table Of Contents.....	V
List of Figures.....	X
List of Tables.....	XI
1 INTRODUCTION.....	1
1.1 Introduction.....	1
1.2 What is Complexity.....	2
1.3 Issues associated with Complexity.....	3
1.3.1 Complexity is not easy to identify.....	3
1.3.2 Approaches to thinking about complex systems.....	4
1.3.3 Effects of planning on complex systems.....	4
1.3.4 Influences of control in complex systems.....	5
1.3.5 Modelling Complexity.....	5
1.3.6 Structure in complex systems.....	6
1.4 Managing complexity in organisations.....	6
1.5 Agile Software development.....	8
1.6 Models of software development.....	9
1.7 Agile software development as a response to complexity.....	10
1.8 Research Problem statement.....	12
1.8.1 Thesis Argument.....	12
1.8.2 Research Approach.....	12
1.9 Thesis Outline.....	13

2 COMPLEXITY	15
2.1 Introduction.....	15
2.1.1 Newtonian thinking and reductionism	16
2.1.2 Non-linear systems.....	18
2.2 A definition of complexity	18
2.2.1 Difference between systems	19
2.2.2 Key Components of Complexity	20
2.2.3 Additional Properties of Complex systems.....	22
2.3 Information and Complexity.....	22
2.3.1 Thermodynamics and Entropy.....	23
2.3.2 Information Entropy.....	24
2.3.3 Algorithmic Information Content	25
2.3.4 Effective Complexity	26
2.3.5 Conclusion	26
2.4 Dynamic Systems and Chaos.....	27
2.4.1 Chaos and Complexity.....	27
2.4.2 Logistic Map	28
2.4.3 Attractors.....	31
2.4.4 Bifurcations.....	34
2.4.5 Sensitivity to start conditions.....	34
2.5 Complex Adaptive System	34
2.5.1 Tornados versus People	35
2.5.2 Organisations as complex adaptive systems.....	36
2.5.3 Feedback	36
2.5.4 Self-Organisation	37
2.5.5 Emergence.....	40

2.6 Wicked Problems	40
2.7 Classifying complexity theories.....	42
2.7.1 Limits to understanding	44
2.8 Conclusion	45
3 AGILE SOFTWARE DEVELOPMENT	47
3.1 Introduction.....	47
3.1.1 History of Software development	48
3.2 Process control models	51
3.2.1 Defined process control model	51
3.2.2 Empirical process control	52
3.3 Agile software process tools	52
3.3.1 Common aspects of agile process tools	53
3.4 Agile manifesto	54
3.4.1 Principles of the Agile Manifesto	54
3.5 Agile methodologies	58
3.5.1 XP (Extreme Programming)	59
3.5.2 Scrum	62
3.5.3 Lean Software Development.....	72
3.6 Agile Culture.....	77
3.6.1 Principles of Agile	77
3.7 Conclusion	79
4 MODELLING COMPLEXITY AND LIMITS TO UNDERSTANDING	80
4.1 Introduction.....	80
4.2 Modelling Complex systems.....	81
4.2.1 Mental modelling of complexity.....	82
4.2.2 Organisational models	83

4.3 Models for Complex problems	89
4.3.1 Cynefin Framework	89
4.4 Building an agile organisation	94
4.4.1 Decision making	96
4.4.2 Transitioning to an Agile Organisation.....	97
4.5 Leadership in Complex Adaptive systems.....	99
4.6 Leadership from a military perspective.....	99
4.6.1 Military solutions to uncertainty	100
4.7 Conclusion	101
5 AGILE APPLIED TO COMPLEX SYSTEMS.....	103
5.1 Introduction.....	103
5.2 Concepts of agile and complexity.....	104
5.2.1 Core Concepts of Agile.....	105
5.3 Problem definition	114
5.3.1 User Story	117
5.3.2 Format of a user story	118
5.3.3 The Role of Narrative	119
5.4 Conclusion	119
6 CONCLUSION.....	121
6.1 Introduction.....	121
6.2 Why Agile.....	122
6.3 Agile: No silver bullet.....	124
6.4 The case against "Best practices".....	125
6.4.1 Measuring non-linear systems	126
6.5 Instilling Resilience	127
6.6 Further research	127

7 BIBILOGRAPHY 129

List of Figures

Figure: 1 The Reductionistic duck	17
Figure: 2 Graphical Representation of stability	29
Figure 3 Graphical Representation of Chaos	31
Figure: 4 Visual representation of a point attractor	31
Figure: 5 Visual representation of a periodic attractor	32
Figure: 6 Visual representation of an Lorenz attractor	33
Figure: 7 Basic Patterns Seen in Cellular automata still lifes,	39
Figure: 8 High level abstract of waterfall based process	50
Figure: 9 Abstract of defined process model	51
Figure: 10 Agile process frameworks	53
Figure: 11 Practises of XP	62
Figure: 12 A high level view of Scrum Framework	63
Figure: 13 Sprints in Scrum	65
Figure: 14 Retrospective Structure	68
Figure:15 Scrum product backlog	69
Figure: 16 Scrum task board	70
Figure: 17 Principles of Agile Software development	78
Figure: 18 Boisot's I-space Model	84
Figure: 19 Cynefin model	90
Figure 20 Concepts of Mission Command	101
Figure: 21 Linear problem solving.	113
Figure 22 Iterative problem solving	114
Figure: 23 Example of a user story	118

List of Tables

Table: 1 Inputs for stability.....	29
Table: 2 Iterations for stability.....	29
Table: 3 Inputs for Chaos.....	30
Table: 4 Iterations of Chaos.....	30
Table: 5 Complexity Theories and associated disciplines	43
Table: 6 Six Steps of Social Learning.....	85
Table: 7 Decisions in multiple contexts: A leaders guide to decision making	94

1

INTRODUCTION

1.1 Introduction

The world has become more interconnected, leading to an increase in intertwined events. Daily, we encounter new surprises in our world. Inconceivable events seem to have become daily occurrences, shaking our cognitive expectations. Many events are associated with complexity, from financial market collapses, hurricanes appearing out of typical seasons, to organisations that have bigger influences than many governments. Nobody can dispute the world is a complex place.

Complexity is a fast and emerging topic of research that challenges the boundaries of traditional understanding patterns that the world presents to us, and epistemology in general¹. Technology and innovation have outpaced our understanding of complex systems. Humans now build things and construct systems that make sense in design because of the isolated nature of creation and human design. Once the systems' connections grow, the interactions and interdependencies deepen, and the complexity of the system increases to unforeseen levels².

Economic change is also driving the way we interpret and store information, where service-based industries comprising of knowledge workers are becoming more economically significant than those of manufacturing industries³.

¹ Peter Allen, Steve Maguire, and Bill McKelvey, *The SAGE Handbook of Complexity and*

² Sidney Dekker, "Drift Into Failure: From Hunting Broken Components to Understanding Complex Systems" (2011): 1–236.

³ Boris Groysberg and Michael Slind, "Leadership Is a Conversation," *Harvard Business Review* 90, no. 6 (2012): 75–84.

The interesting issue at hand is how this complexity theory influences the way organisations operate in today's world, and the understanding of these organisations. For example, the effects of complexity on an organisation can be seen from the hierarchy and design of the organisation structure to management and control of its system ⁴.

Further questions arise when organisations from a multitude of industries are identified as having adapted to absorbing and dealing with complexity. Clear examples of such can be found at nuclear power plants, air traffic control centres, emergency rooms and, large Internet companies, in “Managing the Unexpected”, Weick and Sutcliffe⁵ identify high reliability organisations, and differentiate these organisations from those which cannot deal with complexity in the system.

The lack of ability to apply best-practises for the management of complexity becomes noticeable and self-evident within the nature of organisation systems. Approaches may be similar, however, the practices and processes that are put in place are specific to each situation. This highlights the importance of shared context in complex systems, especially in relation to knowledge and learning.

1.2 What is Complexity

A complex system is constituted of many individual agents. Each of these agents has a regulation to their behaviour which is controlled by a simple set of rules.

According to McCarthy⁶, and common consensus in literature, the complexity theory states that systems which are deemed to be complex have three main properties:

1. The system is made up of a large number of elements
2. There are significant interactions between these elements
3. A level of organisation is displayed in the system.

⁴ P Cilliers, “What Can We Learn From a Theory of Complexity?,” *Emergence* (2000).

⁵ Karl E Weick and Kathleen M Sutcliffe, *Managing the Unexpected*, (Jossey-Bass, 2011).

⁶ Ian P McCarthy, “Technology Management—a Complex Adaptive Systems Approach,” *International Journal of Technology Management* 25, no. 8 (2003): 728–745.

Complex systems exhibit the phenomenon of emergence as their natural behaviour and patterns produced by the system are unforeseeable. Emergence is the outcome of self-organisation in the system where individual agents are not under any central control. The unpredictability of the system cannot be attributed to the individual agents as it is the very existence of their inexplicit relationships that produces the complexity.

Favoured examples in literature when articulating complexity systems include descriptions of ant colonies, stock-markets, and the internet.

The term that differentiates complex biological systems and systems based on organisations is '*adaptive*'. In complex adaptive systems, agents have decision making capacity and the capacity to act intelligently based on their epistemological patterns⁷.

In chapter two, a more detailed understanding of complexity and its associated concepts is presented. The nature of complexity theory is that it encompasses many areas of thought that are all relevant to understanding how to view complex systems.

1.3 Issues associated with Complexity in Organisations

Some common problems face individuals managing complexity in organisations. When presented with complexity, individuals are often overwhelmed, and the task of managing the system would seem to be out of their grasp⁸.

1.3.1 Complexity is not easy to identify

Individuals in the organisation have to be mindful of the environment they are in, as complexity often goes unnoticed. Small signals in complex systems can lead to big changes in the future. If these signals are not identified early and understanding of the situation is not generated, then unexpected shocks can add even more uncertainty and chaos to the system.

⁷ McCarthy, "Technology Management—a Complex Adaptive Systems Approach."

⁸ Gareth Morgan, *Images of Organization*, (SAGE Publications, Incorporated, 2006).

1.3.2 Approaches to thinking about complex systems

An analytical thinking approach to complex problems does not produce understanding that is meaningful. When a problem is too large to understand, the traditional approach to solving the problem is to become reductionistic in thinking. By breaking the problem into smaller parts, these parts can be analysed in their simplest forms. When the understanding of individual parts is reconstructed again, the thinking is that there is understanding of the whole⁹. Complexity does not allow for this reductionistic type of thinking; there are too many interconnected agents, with many relationships to consider to come to understanding.

1.3.3 Effects of planning on complex systems

The notion of a person or group of people being in control of an organisation is very prevalent in management literature. It is often observed that planning and long term forecasting is done continuously in organisations to pave the way for the future by setting vision, strategy, and goals, or in times of crisis, to implement turnaround plans for an ailing organisation. Under certain conditions individuals may have the ability to predict short term events in complex systems. However, this is not true for long term planning and outcomes.

When dealing with complexity, a mind shift needs to take place with agents in the system in so far as they are oblivious to the nature of complexity in the system and have expectant deterministic outcomes. Cilliers states that in dynamic systems, knowledge is limited to the individual agents, or in some cases, needs to be limited. He then emphasises that we are not able to control destiny in any way. This is an imperative warning to organisations where the leadership is intently attached to a plan, as they fail to foresee the signals that the complex system is exhibiting and leverage change as required by the system.

The fear of individuals having limited control to change the course of the ship and let the system emerge is a very real tension, especially where a person has perceived power over the changing course of events¹⁰.

Individuals need to be open to changing their views on their interactions with the system, in addition to moulding the system with their predisposed beliefs and expectations. This type of

⁹ Paul Cilliers, *Complexity and Postmodernism*, (Psychology Press, 1998).

¹⁰ Morgan, *Images of Organization*.

behaviour is best described by Weick¹¹ who states that sense-making is an on-going process. Individuals are shaped by the environment they interact with and conversely the environment is shaped by the individual¹².

For an organisation to adopt thinking in terms of complexity, it has to wrestle with some important and critical issues. Questions of understanding, values, culture and processes need to be explored deeply, possibly reviewed and come to terms with.

1.3.4 Influences of control in complex systems

Interestingly, in complex systems, the amount of control from a central point does not have an influence on the quelling of complexity that is seen in the system. Indeed, the amount of control exhibited on the system can affect how the system is able to respond¹³. Globalisation has introduced changes in the value chain. This coupled with evolving technologies and a control-based approach to management has resulted in inefficiency and become non-viable as a method of application¹⁴. The application of control does not lead to order. It could in fact lead to agents rebelling and the system entering a state of chaos.

1.3.5 Modelling Complexity

The notion of modelling a system to provide understanding is aligned with the thinking that humans are able to see a complete picture of the environment that they partake in. Complexity, by its nature, is not compressible. This concept is also discussed further by exploring the concepts put forward by Kolmagarov and Shannon in Chapter 2, Information and complexity.

While models are useful in understanding, attempting to model complexity introduces some nuances that affect the validity of the model and the resultant cognition of the problem. For example, boundaries can be implied in places that they do not actually exist. This is partially

¹¹ Karl E Weick, *Sensemaking in Organizations*, (SAGE Publications, Incorporated, 1995).

¹² Weick, *Sensemaking in Organizations*.

¹³ Cilliers, "What Can We Learn From a Theory of Complexity?."

¹⁴ Groyberg and Slind, "Leadership Is a Conversation."

problematic for complex systems, as they are inherently open. Meaning and context of information from the system is lost the more the model is distilled, and therefore fidelity of reality is lost.

To build a meaningful model of a complex system, one would have to model all non-linear historical relationships, as well as those happening in real time¹⁵.

1.3.6 Structure in complex systems

Complex systems are normally seen as unstructured, agile and adaptable. However, complex systems do not operate in a state of instability made popular with understanding of chaos and the chaos theory. Cilliers¹⁶ argues that even though the above is important to the complex systems, the system itself needs some form of identity. In having this identity it creates a tension against change and its inherent understood properties.

1.4 Managing complexity in organisations

In order to embrace complexity and develop a greater understanding of how to better manage it in an organisation, traditional approaches to organisational management have been scrutinised. Based on the work of Taylor, traditional management that exhibits top down control from a central location is seen to be dysfunctional in dealing with the complexities encountered in organisations of today.¹⁷

Classically, organisations have dealt with work that is defined by a set process, and based on set rules and criteria for task completion while finding best-fit workers for the job. Organisations now find that the focus of work is changing. Work is based on creativity and innovation with strong dependencies on relationships between individuals.

Individuals have made contributions to management thought that has revolutionised thinking about modern organisations. Morgan, in his book 'Images of Organisation', presents topics of

¹⁵ P Cilliers, *Complexity and Postmodernism: Understanding Complex Systems*, 2002.

¹⁶ Paul Cilliers, "On the Importance of a Certain Slowness," *Emergence, a Journal of Complexity Issues in Organizations and Management* 8, no. 3 (2006): 105.

¹⁷ Allen, Maguire, and McKelvey, *The SAGE Handbook of Complexity and Management*.

self-organisation and complexity theories. He presents five key ideas to managing complexity in organisations¹⁸, including:

1. Rethinking organisation, with regards to hierarchy and control
2. Managing the changing of contexts
3. Small changes leading to large effects
4. Transformation is constant and order is emergent
5. Metaphors can facilitate the process of self-organisation

Alternately, Nonaka provides us with a view on management based on Japanese culture and practice. He describes the only certainty as discontinuity in organisations today, and the reasons that organisations are successful is due to their ability to handle this discontinuity and gain competitive advantage from it¹⁹. Being aware of complexity and chaos in organisations and its effect on knowledge creation, Nonaka argues:

“The main job of managers is to orient this chaos towards purposeful knowledge creation”

Nonaka is popular for his SECI model of knowledge conversion, wherein he theorises the tacit-to-explicit knowledge transformations in organisations²⁰. An earlier publication by Nonaka and Takeuchi, “The New New Product Development Game”²¹, provides insights of how they suggest to manage complexity.

Nonaka and Takeuchi list six items that are requisites for an organisation to innovate and gain competitive advantage²²:

1. Built-in instability
2. Self-organising project teams

¹⁸ Morgan, *Images of Organization*.

¹⁹ I Nonaka, “Ikujiro, ‘the Knowledge-Creating Company,’” *Harvard Business Review* (1991).

²⁰ Nonaka, “Ikujiro, ‘the Knowledge-Creating Company,’”

²¹ Hirotaka Takeuchi and Ikujiro Nonaka, “The New New Product Development Game,” *Harvard Business Review* 64, no. 1 (1986): 137–146.

²² Takeuchi and Nonaka, “The New New Product Development Game.”

3. Overlapping development phases
4. Multi learning
5. Subtle control
6. Organisational transfer of learning

Many of these concepts of controlling chaos and dealing with complexity came from a hard science background, based on a clinical information processing approach to organisations. Interestingly, attention to soft science of organisations was also inferred. The concepts of sense-making, context, and interpreting of meaning was deemed to be important²³.

Complex systems and their dynamic nature allow fundamentally new contexts to emerge. These new contexts constantly shift the way theories are applied to managing²⁴. Complexity thinking limits our understanding of organisations, shifting thinking to learning-based approaches to management and aligning experiences into a framework to interpret situations being faced.

1.5 Agile Software development

Information technology projects have a notorious reputation for unpredictability and late delivery. Managing delivery in IT projects is a difficult endeavour. Each project has multiple levels of complexity associated with it and a number of assumptions that are made about the project.

Many of the issues attributed to project failure are due to these surprises or events that occur that have not been planned for. By their very nature, IT projects are the epitome of complexity.

Some examples of complexity that are experienced within an IT project include:

1. Social complexity between the individuals and teams involved with delivering the product.

²³ Allen, Maguire, and McKelvey, *The SAGE Handbook of Complexity and Management*.

²⁴ Allen, Maguire, and McKelvey, *The SAGE Handbook of Complexity and Management*, 372.

2. Task complexity, as developers are unable to predict how they are going to solve the problem in front of them.
3. Time complexity and pressure to deliver.

1.6 Models of software development

What has been observed in software development is the application of a multitude of models to the development process. Initially, models used in software development were based on a defined process model²⁵. Examples of this, such as those used in the waterfall method²⁶ which were very specific and structured, proved to be problematic to software development. The issues experienced with the application of waterfall to IT projects have been covered in other literature²⁷. The major factor in with dealing with complexity is that the prescriptive nature of the model limits the system's ability to adapt for changes by limiting requisite variety in the system²⁸.

²⁵ Jay Xiong, *New Software Engineering Paradigm Based on Complexity Science*, (Springer Science+Business Media, 2011), 35.

²⁶ Winston W Royce, "Managing the Development of Large Software Systems" 26, no. 8 (1970).

²⁷ Andreas Brennecke and Reinhard Keil-Slawik, eds., "History of Software Engineering" (October 17, 2008): 11.

²⁸ The Law of Requisite Variety, developed by Ashby. It says that any control system must be capable of a number of possible states that is greater than or equal to the number of states in the system under its control.

Variety can be dealt with by either:

- 1) Reducing variety by standardising inputs and controlling the environment as much as possible.
- 2) Designing a system that is capable of absorbing more variety.

Conversely, loosely coupled models that provide little structure to the process of software development and allow agents to operate unconstrained are equally inefficient²⁹. If the requisite variety is unlimited, then the output could never be achieved. Some level of compromise between the level of coupling of the model being applied to the system has to be achieved. In theory, a methodology that allows the system to control its requisite variety to achieve its goal is necessary.

As a response to failing IT projects, a small community of individuals involved in successes and failures while delivering large IT projects started to propose new ways of working. As a response to heavy weighted defined process based approaches, these individuals proposed innovative ways of development based on the empirical process model that enabled and embraced change. This allowed organisations to adapt to changing circumstances in their environment. Agile software development was coined after a meeting in Snowbird, Utah, in 2001, where the thought leaders came up with the agile manifesto.³⁰

The term Agile software development is used to describe many light weight development frameworks which shared common principles and practices.

1.7 Agile software development as a response to complexity

Agile software development has made its mark in delivering high value IT solutions. Agile software development frameworks provide organisations with a very effective platform for handling complexity.

By changing cognitive patterns and processors, people are able to exercise creativity, let designs emerge, and partake fully in the knowledge economy.

As the complexity increases and organisations need to attain competitive advantage, provide customer value and retain stakeholders, the tools required to manage people, teams and organisations need to be reassessed.

²⁹ B Wehmeyer, “Complexity Theory as a Model for the Delivery of High Value IT Solutions” (July 3, 2007): 1–142.

³⁰ The agile manifesto is considered the foundations of all agile processes. A full description of the agile manifesto, its principles and practices is provided in chapter 3 “Agile software development”.

Complexity theory is a shift in thinking in terms of classical science upon which modern science has been built³¹. So too is an Agile based approach to managing in an organisation. Agile thinking and frameworks, implemented initially in IT, but later applied to other facets of the organisation, allow companies to absorb complexity.

Agile frameworks provide the practices and processes that empower individuals and teams to adapt to uncertainty and ambiguity.

1. Iterations of work to constantly validate learning
2. Shortened feedback cycles
3. Retrospectives
4. Space for self-organisation
5. Flat hierarchical structures
6. Cross functional teams

Agile methodologies also contain soft-systems approaches to management. Important facets of an agile approach to management incorporate Continuous improvement, team work communication patterns, shared context, sense-making, problem solving and knowledge transfer. Reinforcing this view point, Cockburn and Highsmith argue³²:

“The most important implication to managers working in the agile manner is that it places more emphasis on people factors in the project: amicability, talent, skill, and communication.”

At a high level of maturity and adoption in the organisation, Agile methodologies that have transcended the Information technology space can provide the platform for the capacity to react quickly to changes in the system. This leveraging of possibilities and options can allow the organisation to alleviate or avoid catastrophic failure while generating useful information for next step decision making, or gain competitive advantage in the market.

³¹ Allen, Maguire, and McKelvey, *The SAGE Handbook of Complexity and Management*.

³² Alistair Cockburn and Jim Highsmith, “Agile Software Development, the People Factor,” *Computer* 34, no. 11 (2001): 131–133.

1.8 Research Problem statement

The umbrella of Agile software development incorporates many methodologies and frameworks. These frameworks focus on to two main areas of the software development problems faced in organisations today.

1. The first is process based improvements in software development.

How do processes and process management help or hinder the software development cycle, the ability for individuals and teams within organisations to deliver software which caters for problems that the organisation is faced with

2. The second is the soft systems approach to management

Where a people first approach is recognised as a driving force behind solving problems. This includes individual autonomy and empowerment. This drives phenomena such as self-organisation, emergence and evolutionary outcomes.

1.8.1 Thesis Argument

This research will inspect how agile software development methodologies can be applied as a method to incorporate and provide tools that help with the response to complexity which is encountered in organisations by individuals and teams.

1.8.2 Research Approach

Initial research was conducted on complexity theory. As complexity theory is broad and incorporates many facets of understanding of the term complexity, certain key areas were identified as being necessary to understand and demonstrate the meaning of complexity.

After a literature overview, the following subject matter was highlighted as being relevant to complexity theory and investigated in further detail: the understanding of linear and non-linear systems and how they differ in aspects of causality.

Next an understanding of common properties of complexity agreed by the mainstream thinkers was undertaken.

An important subset to complexity is dynamic systems and the theory of chaos, including attractors and bifurcations that influence states of dynamic systems causing a move towards

new emergent behaviour. To demonstrate, chaos logistics map algorithm was further explored as a simplistic way to show how a system can move from stability into chaos.

To understand the attempts to measure complexity, literature from Gell-Mann and Shannon proposed ways to measure complexity based on an understanding of Information theory and entropy. In order to differentiate mathematical complexity, the complexity observed in biological systems, such as ant colonies and organisations, the concept of a complex adaptive system and how its agents are constructed needed to be defined.

Discovery of the properties of self-organisation and emergence in complex systems and specific properties that are required for the occurrence of the phenomena, and the need to visually represent these properties lead to the research of the demonstration of emergence in cellular automata.³³

Research into agile software development methodologies started with a retrospective analysis on how the path to agile software development from early process models and definitions of defined process and empirical process control began. From this, further research was conducted on the most popularly used methodologies in agile software development; Extreme programming, Scrum, Lean.

The next chapter introduced failure in complex systems. The research leads to how organisations manage complexity and how thinking is applied to complexity. Newtonian analysis to problems was researched further, as it formed the basis of initial management thought which still has strong influences on how organisations are currently managed

The remaining research is on how Organisations are successful at managing complexity. The application and comparison of processors and tools in agile methods begins here against enlightened thought and practices on managing complexity. Investigations into mindfulness, sense-making, and knowledge assets, as well as abstract models of complexity, such as Cynefin, were conducted.

1.9 Thesis Outline

Chapter 2 identifies what complexity is and outlines the current epistemology of complexity theory and thinking. Included in this chapter is an exploration of dynamic systems, including

³³ Cellular automata is described in Chapter 2 A definition of complexity

chaos theory and types of attractors in systems. The measurement or challenges to measuring complexity is discussed, which include applications of entropy and information systems. After the theories of complexity science have been introduced, the term complex adaptive system is tabled and discussed, as it is relevant to organisations and managing

Chapter 3: Agile software Development, explores the origin of software development and how it has evolved over time and the changing models that have been used to enable effective delivery of complex solutions in organisations. The various frameworks that have been bundled under the umbrella of agile are explored. This includes, Extreme Programming, Scrum, and Lean software development. The author shows the shift from defined process models to empirical process models. This is an important concept for understanding how this effects organisations responding to complexity.

Chapter 4 introduces the concept of modelling complex systems and various tools that can be used to help with the understanding of complex systems. At the end of the chapter, a look at leadership and how this affects shared understanding and organisational adaptability for dealing with complexity is explored. Models of leadership, which have worked in situations from the battlefield to the organisation, are identified and presented, as they have similarity with the principles behind agile software development.

Chapter 5 integrates the concept of Agile software development methodologies with managing complexity in the organisation. By applying agile practises and principles from frameworks, insight is gained in how to better manage complex systems. While agile will not cure complexity, it helps organisation absorb it. Certain key aspects of agile software development are identified as being key modulators for complexity.

2

COMPLEXITY

2.1 Introduction

In the following chapter, the concepts of complexity and complex adaptive systems are introduced. This chapter is the foundation for understanding complexity with basic concepts that have led to the better understanding and current views of complexity science presented.

A new way of thinking called non-linear dynamics or complexity theory is becoming more widely accepted. Based on this thinking, research being undertaken into complex systems tries to explain how large numbers of simple agents in a system are able to organise themselves whilst under no central control creating emergent patterns, and still learning and continuously adapting to the environment³⁴.

A basic roadmap of the chapter is as follows: Initially, the stage is set explaining Newtonian thinking which is based on linear thinking and reductionism. Expanding on this is the notion of problem domains and the understanding of the relationship of simple to complicated to complex. Basic well used examples of complex systems are presented to help the reader to apply a complexity based mindset to the rest of the chapter.

Next, the concept of information and how it relates to complexity is explored. The laws of thermodynamics set the stage for the exploration of information theory based on the work of Shannon, Kolmagrov and Gell-Mann.

Next, dynamic systems are presented. The topic includes the role of chaos in dynamic systems as well as a demonstration of a dynamic system with the use of the logistic map algorithm. This leads to the discussion of how attractors affect dynamic systems and the types of attractors that are experienced in systems. The ideas of start conditions and bifurcation of a system is also presented, as it is an important part of how emergent behaviour is observed.

³⁴ Melanie Mitchell, *Complexity: a Guided Tour*, (Oxford University Press, USA, 2009).

Next, the notion of complex adaptive systems (CAS) and how they differ from a complex system in their makeup of agents and direction is examined. The exploration of organisations as complex adaptive systems is tabled in this section, along with the properties of self-organisation, emergence, and feedback in systems.

A basic approach when dealing with complexity is for individuals to create a model of the system that is trying to be understood. The ideas of modelling and understanding are presented with ideas sourced from Boisot, Cilliers and Snowden.

The notion of complex problems, understood as wicked³⁵ problems, is investigated as a starting point for understanding how problem spaces are relevant to complexity in systems. This very much ties back up to the simple/complicated/complex dilemma highlighted in this chapter. In addition to this, the attempt to create a meta-overview of complexity from the problems that are encountered at the work level to the system level is presented.

The material that was chosen to help describe complexity is encompassing; in fact it takes on the attributes of the meaning of complexity which can be traced back to the Greek word Plexus, meaning entwined. Complexity is associated with interconnectivity of elements within the system. Laying the foundations from where the concepts of complexity theory have evolved, the basis of human thinking and theory has to be understood. This starts with the origins of Newtonian thinking and the concept of reductionism.

2.1.1 Newtonian thinking and reductionism

Reductionism has been dominant in scientific theory since the 1600s. Descartes, an early reductionist, describes his scientific method as “to divide all the difficulties under examination into as many parts as possible, and as many as were required to solve them in the best way and to conduct my thoughts³⁶.”

This worldview underlying traditional science may be called "mechanistic" or "Newtonian". It is based in reductionism, determinism, materialism, and a reflection-correspondence view of knowledge. Although it is simple, coherent and intuitive, it ignores or denies human agency, values, creativity and evolution.

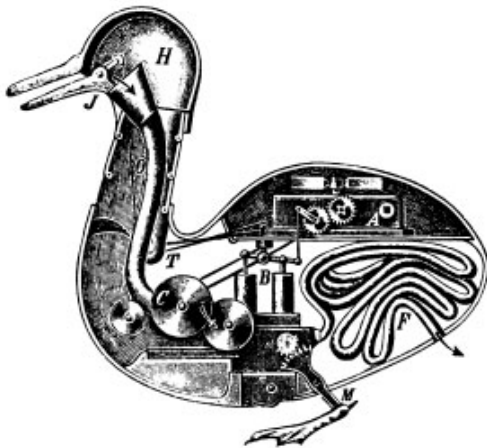
³⁵ Wicked problems are further described in section 2.6

³⁶ Mitchell, *Complexity: a Guided Tour*.

Nothing demonstrates this better than understanding the application of Newton's three laws. Newton's three laws of motion state:

1. Constant motion: Any object not subject to a force moves with unchanging speed.
2. Inertial mass: When an object is subject to a force, the resulting change in its motion is inversely proportional to its mass.
3. Equal and opposite forces: If object A exerts a force on object B, then object B must exert an equal and opposite force on object A.

Newton applied these laws to objects that were encountered on earth from where he understood and defined the laws of gravity. This was evident by the use of apples to help articulate his theories and also expanding this to describe the movement of planets in the solar system. This type of thinking provided a platform which would equate to a mechanical understanding of the world, thus providing the ability to predict everything.



*Figure: 1 The Reductionistic duck*³⁷

In the modern era, reductionism has come under pressure. Although it showed that this thinking was able to explain large and small problems, physics and thinking based on Newton's discoveries was unable to explain complexity on an everyday level. This is no more evident than in predicting next week's weather pattern³⁸. The theories of Quantum mechanics have shed new light and challenged the concepts proposed by Newton.

³⁷ <http://en.wikipedia.org/wiki/Reductionism>

³⁸ Mitchell, *Complexity: a Guided Tour*.

Stemming from Quantum mechanics comes the Uncertainty principle, as defined by Werner Heisenberg, providing evidence that prediction of future states is not possible. Heisenberg theorised that measuring the values of the position of a particle and measuring its momentum at the same time was impossible. It is possible to have information that is certain on one element, e.g. position, but little information on momentum.

2.1.2 Non-linear systems

Traditionally, scientists have described behaviour of systems in linear terms. Simple systems exhibit behaviour that is linear in nature, a world where cause and effect are simple to understand, as a given action only has one outcome. In addition to this linear approach, systems can be reduced to their individual components. Each component can be described completely, and reassembly of all individual components together leads to the same whole.

Non-linear systems are more than the sum of their parts; you cannot describe a non-linear system by reductionist methods. Holistic thinking has to be employed to identify patterns or behaviour in the system.

Non-linear systems are highly sensitive to initial conditions, so much so that tiny error states or noise can escalate into major qualitative changes in the system. The links of cause and effect disappear in the complexity of interactions and the long term future of the system is unpredictable.

The observation is that most relationships are non-linear in nature, making them impossible to describe simply. The key attribute in a non-linear relationship is that cause and effect are not correlated and can lead to many different outcomes.

2.2 A definition of complexity

Missing from literature is a concise definition of complexity that is agreed upon and shared by leading thinkers in the field of complexity. Complexity is not easy to define, with complexity having different meanings based on the individual's understanding. So different is this understanding of complexity that Mitchell, M. (2009) in 'Complexity: a guided tour', relates the following when trying to find the definition of complexity:

"If the faculty of the Santa Fe Institute—the most famous institution in the world devoted to research on complex systems—could not agree on what was meant by complexity, then how can there even begin to be a science of complexity?"

Traditionally, a notion has been formed of what a complex system is and how it interacts with the use of real world examples. It is what connects these real world examples that helps us describe complex behaviour and identify problems that are resultant of this complexity that need to be dealt with on a daily basis.

2.2.1 Difference between systems

It is important to note that not all situations or problems that organisations face fall into the domain of complexity. While it is easy to be preoccupied with complexity because of its nature and allure, it is important to be able to categorise systems as complex, simple, complicated, or in a state of chaos. This is important when responding to problems, as wrongly identifying the domain in which the problem lies can induce the wrong response.

Cilliers argues that before complexity can be understood, the distinction between simple and complex has to be appreciated, as well as the distinction between complex and complicated³⁹. This is also important for the application of dealing with and responding to complexity, which is stated later in this text with model based approaches to decision making in complex systems.

2.2.1.1 Simple vs Complex

Systems observed and characterised as simple or complex from the point of view of the observer may have direct proportionally effects on the distance from the system in which the observer is located⁴⁰. A system observed from afar may appear simple from an initial macro level, although when an alternative vantage point is taken with more detail, then levels of complexity can be revealed which were hidden from the initial view point. Complexity

³⁹ Cilliers, *Complexity and Postmodernism*, 2-3.

⁴⁰ P Cilliers, "Boundaries, Hierarchies and Networks in Complex Systems," *International Journal of Innovation Management* (2001).

cannot be identified as originating at a certain point in the system, due the dynamic nature of interactions and relationships in the system that lead to it.

2.2.1.2 Complicated vs Complex

The line between complex and complicated is also troubling for many to come to terms with. While systems with many agents performing advanced tasks lead towards complexity, this is not the case. If those agents and tasks can be analysed fully, then the system is complicated. Snowden⁴¹ applies the theory that complicated domains are the place for expert knowledge and analysis. Complex systems are normally associated with living entities, e.g. the brain, a social system, while metaphors for complicated and simple systems drift trend towards mechanical objects. Cilliers highlights this with the statement that a Boeing 747 is complicated, while mayonnaise is complex⁴².

2.2.2 Key Components of Complexity

Many people would debate the characteristics of a complex system, but for a description that resonates as being as concise as possible, Cilliers' definition provides a base for this paper. Cilliers⁴³ describes complex systems as having the following characteristics, which most complexity researchers would agree that a complexity system should have most or all of:

1. Complex systems consist of a large number of elements.
 - If the number of agents in the system is relatively small, they can be described with relative ease. As the number of agents grows to a large scale, conventional means of description becomes impractical.
2. These elements interact dynamically.
 - The agents within the system must interact with each other. This interaction may be on a physical level or by transferring information. The condition to

⁴¹ D Snowden, "Multi-Ontology Sense Making Making: a New Simplicity in Decision Making" (2004).

⁴² Cilliers, *Complexity and Postmodernism*.

⁴³ Cilliers, *Complexity and Postmodernism*.

interaction is that it is of a dynamic nature, and the strength of the interactions varies over time.

3. Interactions are rich; any element in the system can influence or be influenced by any other.
 - Agents in the system influence and are influenced by other agents in the system. The number of connections does not affect the agents' capabilities in the system. Behaviour of an agent with many connections can be performed by multiple agents with few connections. There are also multiple routes between agents.
4. Interactions are non-linear.
 - If interactions are linear in the system, the interactions could be compressed into a smaller describable state. Non-linear relationships are a prerequisite for complexity as small changes in interactions cause large changes.
5. Interactions are typically short range.
6. There are positive and negative feedback loops of interactions.
 - Feedback loops are built into the system; positive feedback has a stimulating effect and negative feedback has a damping effect.
7. Complex systems are open systems.
 - Complex systems are open in nature, with borders being difficult to define. The only borders that are imposed on complex systems are those of the observer who has to frame the system to understand it. Framing is dependent on the position from where the system is observed. This framing could have an effect on the perceived complexity of the system.
8. Complex systems operate under conditions far from equilibrium.
 - Stability in a complex system is seen as a state of death of the system. The system uses constant energy and flow to maintain some sort of organisation.
9. Complex systems have histories.
 - A complex system will evolve through time. The system's current state is dependent on past behaviour.

10. Individual elements are typically ignorant of the behaviour of the whole system in which they are embedded.
 - Individual agents are inherently simple. They cannot understand the behaviour of the whole, due to the information and interactions presented to them.

2.2.3 Additional Properties of Complex systems

In addition to the general properties of complexity present in a system, it is important to note the additional phenomena associated with the system. The main phenomena that a complex system demonstrates are self-organisation⁴⁴ of agents and behaviour and emergence of design, requisite variety, non-linearity and chaotic behaviour. While these are important, singly or jointly they demonstrate complexity at theoretical level.

Complex systems demonstrate behaviour that is not forced or directed by internal or external control. Complex systems also display emergent behaviours. In describing emergence, Cilliers states that: "Complex systems display behaviour that results from the interaction between components and not from characteristics inherent to the components themselves."

2.3 Information and Complexity

One of the key issues with complexity is defining quantitatively how complex a system is, or comparing one system to another to say that system A is more complex than system B. Many complex systems scientists use the concept of information to characterise and measure order and disorder, complexity and simplicity⁴⁵. Because complex systems all produce and use information, the first step to understanding complexity is to understand how theories of information have evolved. To reiterate Mitchell⁴⁶, there is no single science of complexity or a theory that encompasses what complexity is. There is not one accepted way to measure it even though many have been put forward.

⁴⁴ Cilliers, *Complexity and Postmodernism*, 90.

⁴⁵ Mitchell, *Complexity: a Guided Tour*, 12.

⁴⁶ Mitchell, *Complexity: a Guided Tour*, 13.

The term information is used very broadly to describe many things that individuals encounter on a daily basis. More generally, it is referred to the medium that is used when communication happens between two parties. The most important understanding is that information takes many forms and cannot be characterised as being only words or numbers but many states. Although the basis of thought is towards computer systems, transmitting and computing information, this happens equally in living systems. This is especially true of complex systems, where entities in the systems are concerned with the communication and processing of information in various forms.

2.3.1 Thermodynamics and Entropy

To start to understand the attempts to deal with and quantify complexity and the study of information, a basic appreciation of the first two laws of thermodynamics is needed. Thermodynamics describes energy and its interaction with matter. Energy can be attributed with the system's potential to do work⁴⁷.

A physical example of the application of energy can be seen in the cooling of a refrigerator. Electricity is used to power the compressor, which cools the interior of the refrigerator. In the process of cooling it also creates heat. The amount of energy transformed comprises of the amount of work done (in this case cooling the temperature of the refrigerator) plus any lost energy that was converted to heat (by the compressor). This loss of energy, which is unable to be converted into additional work due to the transformation of energy, is known as Entropy.

Thus, the first law of thermodynamics applies: Energy is conserved. The total amount of energy in the universe is constant. Energy can never be destroyed.

The Second law of thermodynamics is especially relevant. The law deals with the direction of time and its irreversibility. It is the only law of nature that distinguishes between past and future. This law of thermodynamics also contributes to one of most major problems in physics, the problem of why time flows in a certain direction, with the associated understanding that it is a forward type of motion. Why cannot time flow in other directions? The second law states that Entropy of a system can only increase until it reaches its maximum value.

⁴⁷ Mitchell, *Complexity: a Guided Tour*.

As a system transforms energy, less of it remains in a usable form as more is taken up as Entropy and the disorder increases⁴⁸.

2.3.2 Information Entropy

During the 1940s while working at Bell Labs Claude Shannon was working on the problems of transmitting signals faster and more reliably over physical telephone infrastructure⁴⁹. Shannon defined information as having a source which sends messages to a receiver. The information would be transmitted over a channel which would allow for the maximum transmission rate based on channel capacity.

Based on Shannon's paper, "A Mathematical Theory of Communication", he defined modern information theory. His theory replaced energy with information in the laws of thermodynamics. In doing so, his theory equated that the more disorder (entropy) associated with the message, the more information it contained⁵⁰

Shannon's definition of information content has also been labelled as the average amount of surprise, which is experienced by the recipient of the message⁵¹. This is also the amount of uncertainty on the part the receiver as to what message would be sent next. Simply put, the amount of entropy associated with the message multiplied by the message length gives us the information content.

Examples of Shannon information Content:

A message consists of a highly structured pattern of repeating digits (101010) only transmitted one digit at a time. The amount of information is low, and so is the associated entropy. In another example if the message was random and with no structure then information content would be higher and so would associated entropy.

⁴⁸ Cilliers, *Complexity and Postmodernism*, 8.

⁴⁹ Mitchell, *Complexity: a Guided Tour*, 52.

⁵⁰ Cilliers, *Complexity and Postmodernism*, 8.

⁵¹ Mitchell, *Complexity: a Guided Tour*, 54.

2.3.2.1 Issues with Shannon entropy

The object or process in question has to be put in the form of messages of some kind. Sometimes this is not easy. For example, how do you measure the entropy of the human brain? The most complex entities are not the most ordered or random ones, but lie somewhere in between. Simple Shannon entropy does not capture our intuitive concept of complexity.

2.3.3 Algorithmic Information Content

Alternatives to measures of complexity based on information besides content entropy exist. Andre Kolmogorov was one person who proposed that algorithmic information content could be used to describe complexity. In the proposal, he theorised that the shorter the computer algorithm needed to produce a string, the less complex the information is.

The following examples demonstrate Algorithmic complexity:

String 1 contains a repeating pattern of two letters A and B and is 64 characters long

ab...

The algorithm required to produce this string would be very simple “Print ab *32”.⁵² Algorithmic complexity is low.

String 2 contains 64 characters of information with no repeating patterns:

Sdfgudbvakdfvoewjnwelkspdosdkbcwqqwedwhww...

The algorithm required to create this would not be able to be simplified to anything shorter. String 2 is non-compressible.

Just like information theory, higher information content is associated with randomised information and so higher complexity⁵³.

⁵² While arguments may be heard about the comparison of programming languages to write this line of code, with some being more efficient than others, it has been found that the optimisation provided is negligible in demonstrating the algorithmic complexity of the output.

⁵³ Mitchell, *Complexity: a Guided Tour*.

2.3.4 Effective Complexity

Building on Kolomgorov's theory, but understanding that complete randomness in the real world does not produce quantities of meaningful information and in order to subvert the notion that complete randomness is extremely complex, Gell-mann proposed a measure of effective complexity. This measures of the combination regularity and randomness that is produced.

The first step to describing effective complexity of an item is to describe the regularities that exist. To take the above example again, String 1 has low algorithmic complexity as to print the reoccurring pattern is trivial. In the case of String 2 the algorithmic complexity is low as well. The string is completely random and therefore there are no regularities to describe.

2.3.5 Conclusion

As identified above, complexity can be measured in a variety of ways. As Mitchell, M. (2009) 110 has identified, each of the ways of measuring complexity captures something about our current understanding of a complex system. She goes on to identify that each measurement has some practical and theoretical limitations, so much so that there has been little success in characterising a real-world complex system. Without a single definition of what complexity is, it would be very difficult to come to an agreement on how to measure it.

Algorithmic and Information based approaches to complexity prove to be problematic in fully describing complexity. However the theories re-emphasise a key point, namely that complexity is incompressible⁵⁴, a world where reductionism based thinking has no place.

Further questions could be asked. If humans are the most complex systems in existence, are they more complex than the reverent ant colony or the earth as a whole? When does this rating of complexity matter and when does it not?

Further on in the dissertation, we look at ways of identifying and communicating complex systems and behaviours. Snowden makes the observation that you cannot reduce complexity, but only absorb it. With this in mind, is it truly beneficial to know the size of complexity or to just identify systems operating in complexity versus those which are not?

⁵⁴ Cilliers, *Complexity and Postmodernism*, 24.

2.4 Dynamic Systems and Chaos

2.4.1 Chaos and Complexity

Chaos theory has been hyped as the basis of explaining all unknown and unpredictable phenomena that are experienced. The meaning of chaos is attributed with occurrences of events filled with mayhem and randomness. Fortunately, this is not the case when chaos theory is understood. Parker and Stacey⁵⁵ describe chaos as an intricate mixture of order and disorder, regularity and irregularity, patterns of behaviour, which are irregular but are nevertheless recognisable as broad categories or archetypes, of which there is endless individual variety.

While Cilliers⁵⁶ dismisses the importance of investigating chaos as a part of complexity, he argues that chaos is limited in the application to understanding complexity as the robust nature of complex systems mitigates some of the basics of chaos theories. For example; chaotic system's inherent sensitivity to start conditions. In disagreement with Cilliers, the author believes there are some important topics in complexity that have applications in thinking towards organisational studies. By considering chaos, it leads to reinforcing the understanding that in the short term random nature of chaos in a system reinforces the ideas that the next behavioural pattern or macroscopic state of the system cannot be predicted⁵⁷. To be able to identify this and act in this situation is important when applying management practices to complexity.

The ideas of chaos originally from natural sciences have moved towards application and understanding in social sciences⁵⁸. The study of chaos highlights, on a system level, concepts such as sensitivity dependence, bifurcation, attractors, and irreversibility⁵⁹. Chaos also shows

⁵⁵ D Parker and R Stacey, "Chaos, Management and Economics: the Implications of Nonlinear Thinking" (1994).

⁵⁶ Cilliers, *Complexity and Postmodernism* p.ix.

⁵⁷ Mitchell, *Complexity: a Guided Tour*.

⁵⁸ Allen, Maguire, and McKelvey, *The SAGE Handbook of Complexity and Management*, 55.

⁵⁹ Allen, Maguire, and McKelvey, *The SAGE Handbook of Complexity and Management*, 55.

the transition from a position of stability in the system to chaos, with the occurrence of a move from negative feedback to positive feedback in the system.

2.4.2 Logistic Map

The simplicity of the logistic map equation provides an easy way to demonstrate the concept of chaos. The expression shows how sensitivity to initial conditions produces behaviour ranging from ordered to chaotic. The following text provides a basic understanding on two states of logistics map⁶⁰.

The logistic map equation is written mathematically as:

$$X_{n+1} = RX_n(1 - X_n)$$

The equation demonstrates a sensitivity to start conditions when R is a number between 3.57 and 4. The graph of the equation then produces behaviour that is repeatedly chaotic in the space in which it has been defined.

As a number between zero and one, representing the ration of existing population to the maximum possible population at year n, $X = 0$ represents the initial ratio of population to the max population at time zero⁶¹.

R is a positive number and represents the combined birth and death rate of the population

The non-linear equation captures two effects:

1. Reproduction rate of the population which increases in proportion to the current population
2. Death rate which is proportional to the carrying capacity of the environment less the current population

⁶⁰ For a more complete definition and outputs please see <https://www.wolframalpha.com/input/?i=logistics+map> and http://en.wikipedia.org/wiki/Logistic_map

⁶¹ Mitchell, *Complexity: a Guided Tour*, 27.

2.4.2.1 Logistics stability

To demonstrate the progressive nature of the logistic map algorithm, a number of iterations with different values are demonstrated. Initially a state of stability is demonstrated over four iterations with the input values of:

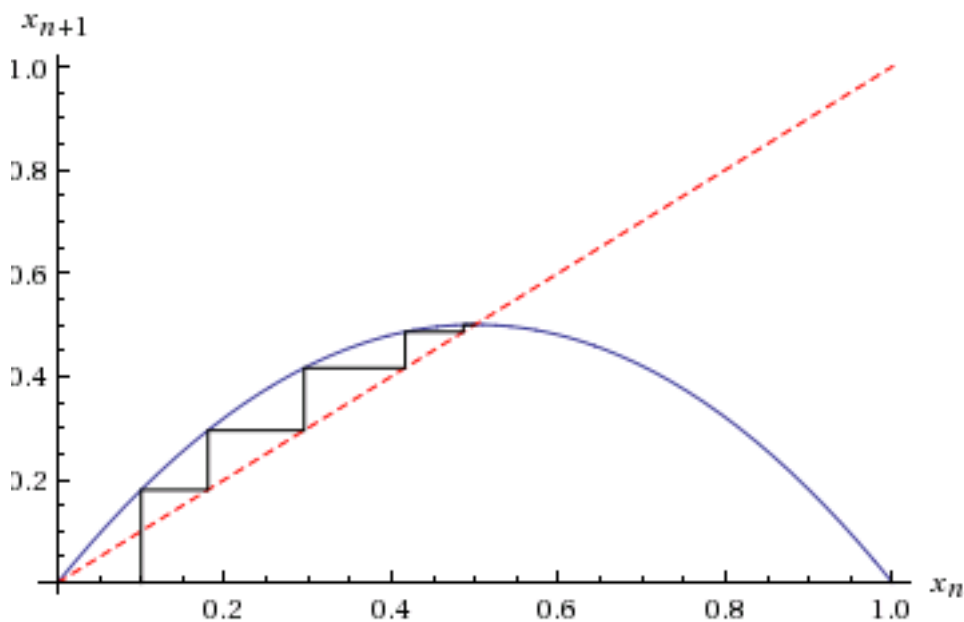
Let R be 2 and x equal 0.1

logistic map	
parameter r	2
initial condition x_0	0.1

Table: 1 Inputs for stability

N	0	1	2	3	4
x_n	0.10000	0.18000	0.29520	0.41611	0.48593

Table: 2 Iterations for stability



(lines successively connect the first 50 iterates and the dashed line $y = x$)

Figure: 2 Graphical Representation of stability⁶²

⁶² Graph generated by wolfram alpha <https://www.wolframalpha.com/input/?i=logistics+map>

2.4.2.2 Logistics chaos

To demonstrate the algorithm in a state of chaos input values are changed and four iterations are run again

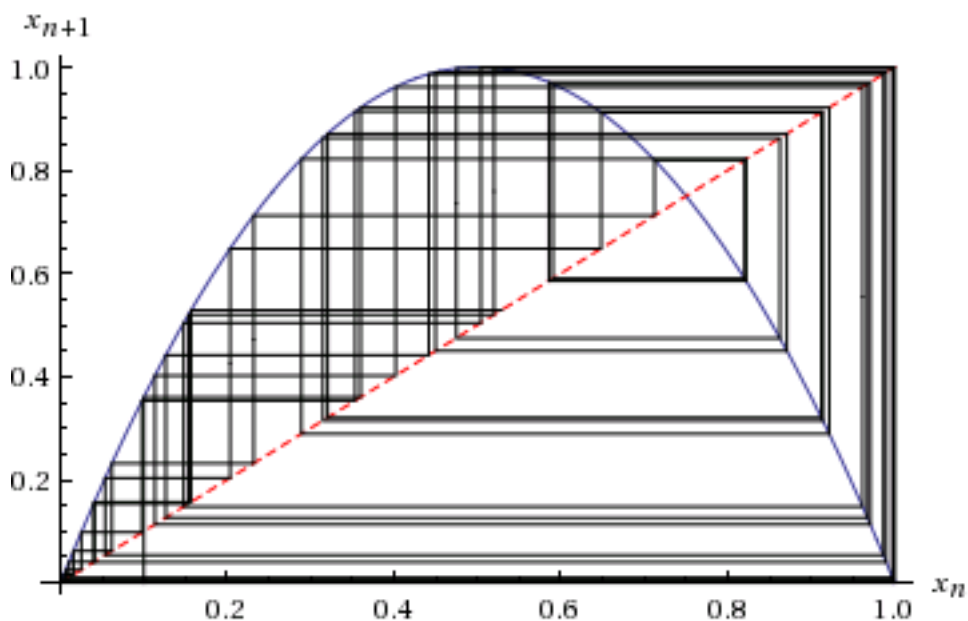
Let R be 4 and x equal 0.1

logistic map	
parameter r	4
initial condition x_0	0.1

Table: 3 Inputs for Chaos

N	0	1	2	3	4
x_n	0.10000	0.36000	0.92160	0.28901	0.82194

Table: 4 Iterations of Chaos



(lines successively connect the first 50 iterates and the dashed line $y = x$)

*Figure 3 Graphical Representation of Chaos*⁶³

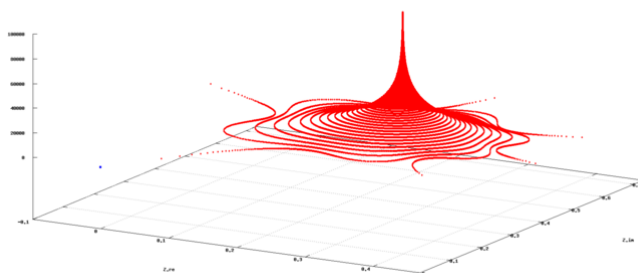
2.4.3 Attractors

Attractors are phenomena, which are found in systems that are at equilibrium and in chaos, influencing systems toward specific trajectories or points of stability. In social systems, attractors are used as metaphors to describe behaviour or identify boundaries and events to which dynamic systems respond.

Although the attractor induces some deterministic behaviour, the ability to predict future states in the long term is impossible. In the short term, because of the attractor properties containing the system in a cage like pattern, small variations are not fully understood, so the system appears to be at a point of stability.

2.4.3.1 Point Attractor

A fixed point attractor brings the system to a state of stability. A marble settling at the bottom of a bowl is an example of fixed point attractor; the condition is that the point displays stable equilibrium for the system. Even after the system is altered, it will have a natural tendency to return to this state.



*Figure: 4 Visual representation of a point attractor*⁶⁴

⁶³ Graph generated by wolfram alpha <https://www.wolframalpha.com/input/?i=logistics+map>

⁶⁴ http://en.wikipedia.org/wiki/File:Critical_orbit_3d.png

2.4.3.2 Periodic Attractor

For a periodic attractor to be evident the system is in constant movement which is regular, observable and repeatable.

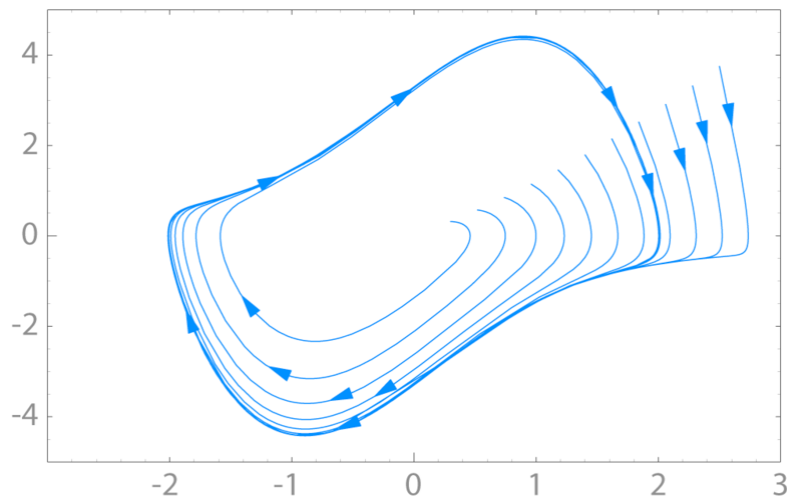


Figure: 5 Visual representation of a periodic attractor⁶⁵

2.4.3.3 Strange attractors

Strange attractors are indicators of deterministic chaos⁶⁶. The system lacks the attractors above, but seems to be attracted to pattern sin the perceived chaos, even though movement in the system never reaches a stable equilibrium. This can also be identified as a move from chaotic behaviour to self-organisation⁶⁷

⁶⁵ <http://en.wikipedia.org/wiki/File:VanDerPolPhaseSpace.png>

⁶⁶ Steven M Manson, "Simplifying Complexity: a Review of Complexity Theory," *Geoforum* 32, no. 3 (2001): 405–414.

⁶⁷ Parker and Stacey, "Chaos, Management and Economics: the Implications of Nonlinear Thinking," 36.

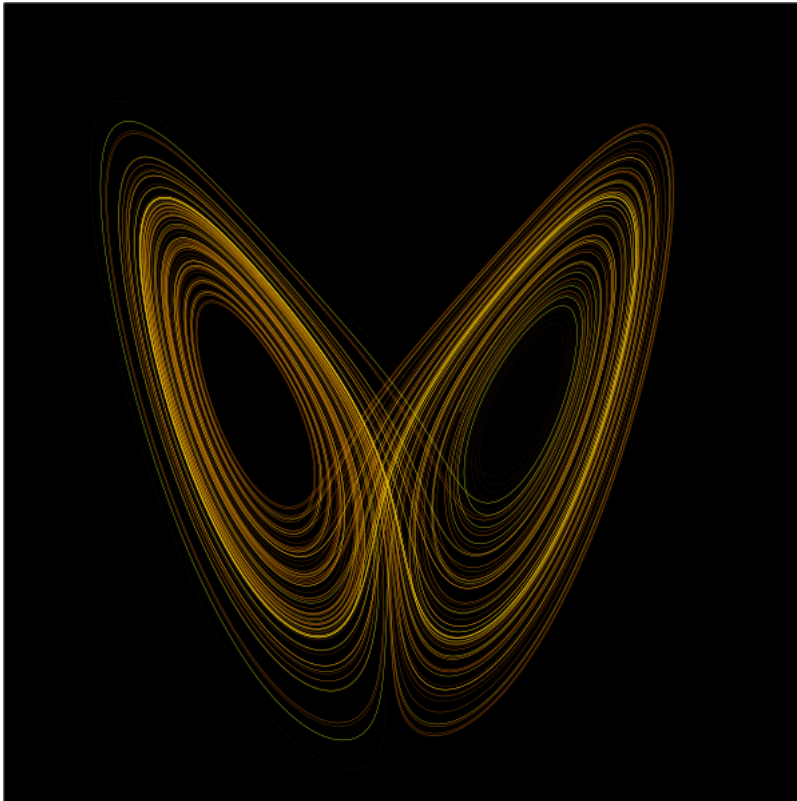


Figure: 6 Visual representation of an Lorenz attractor⁶⁸

2.4.3.4 Lorenz Attractor

The Lorenz attractor is a non-linear dynamic system that demonstrates chaos theory. The highlighting feature of the Lorenz attractor is its sensitivity to start conditions. A minute change to the initial start condition leads to a massive difference in the path of the solution. Edward Lorenz first discovered this sensitivity to initial conditions when working on the problem of weather prediction⁶⁹. On entering a decimal number which had been rounded to 3 places instead of six, he found the computer model that he was using gave vastly different results

⁶⁸ http://en.wikipedia.org/wiki/File:Lorenz_attractor_yb.svg

⁶⁹ Mitchell, *Complexity: a Guided Tour*, 22.

2.4.4 Bifurcations

Bifurcations are seen as a point of change for dynamic system. This change will move the system to a new future than what was previously dominant in the system's path dependence. Occurrences of Bifurcation in a system are one way only, with reversibility not possible. Large changes in systems are seen as jumps to a new attractor. The point of the system making a shift from its current behavioural pattern to the new one is known as a Bifurcation.

2.4.5 Sensitivity to start conditions

Under certain circumstances systems are highly sensitive to initial conditions. This term is used to describe a situation where small changes in the initial system may lead to highly unpredictable long term outcomes⁷⁰. A tiny error in the system such as a decimal place or additional noise being present can induce major changes to the behaviour of the system. The butterfly effect, a term used to describe how a butterfly flapping its wings can create instability in weather patterns on the other side of the world is an attempt to demonstrate the concept of sensitivity to initial conditions⁷¹.

The mathematician Poincare described the first examples of chaotic systems. Whilst trying to solve the three body problem, he invented algebraic topology, discovering dependence in initial conditions.

2.5 Complex Adaptive System

Origins of Complex adaptive systems are mainly influenced from the natural sciences. Initially, complexity theories were based on mathematical models of systems in nature at a macro level⁷², as described previously in the sections on measuring complexity and dynamic systems. Complex adaptive systems seek to model systems with common understanding

⁷⁰ Mitchell, *Complexity: a Guided Tour*, 20.

⁷¹ Manson, "Simplifying Complexity: a Review of Complexity Theory."

⁷² Ralph Stacey, *Complex Responsive Processes in Organizations*, (Routledge, 2003).

coming from agent based models based on simple rules for agents, which demonstrates the emergence of complex behaviour⁷³.

The study of complex adaptive systems focuses on the relationships of agents in the system. There is normally a strong network that binds the interactions of agents in a complex adaptive system. Agents have decision making capabilities with diverse agendas. Which are still governed by rules applicable to the system. The agents do not act randomly⁷⁴

2.5.1 Tornados versus People

It is hard to dispute that a tornado is not a complex system. Tornados are known very sensitive to start conditions based on the certain atmospheric conditions and weather patterns. Tornados are also the result of emergence in large cell storms. One thing that tornados are not is adaptive. They do not change course or gather data from the environment with the ability to adapt. On the other hand, the towns, which lie directly in the path of the approaching tornado, are very adaptive. The people who make up the population of the town are able to react to changing weather conditions to ensure survivability of themselves and the town as much as possible. They have learnt to adapt to approaching danger.

While sharing the concepts of complexity, there are unique aspects that differentiate systems that display some form of adaptability compared to those which are non-adaptive. Complex adaptive systems have the ability to consciously alter configurations adapting and influencing their current and long-term survival. This principle of self-organisation by agents in the system without a central controller is a key property of a complex adaptive system. The collective behaviour from the agents is deemed to be emergent.

Complex adaptive systems display high amounts of entropy, and the system is a constant state of dynamic stability, with the ability to have unpredictable shifts in direction. This changes the view of an organisation from stability-seeking, linear dependent.

⁷³ D Snowden and P Stanbridge, "The Landscape of Management: Creating the Context for Understanding Social Complexity," *Emergence-Mahwah*- ... (2004).

⁷⁴ John Cleveland, "Basic Concepts and Application to Systems Thinking" (June 11, 1994): 1–28.

2.5.2 Organisations as complex adaptive systems

Organisations are a demonstration of complex adaptive systems. By their nature, organisations are adaptive by virtue of the social aspect of the system and the need to survive. Highsmith⁷⁵ makes the following observation about organisations:

“Organisations are chaordic. Every organisation exhibits properties of chaos and order that defy management of the phenomena through the use of linear predicative planning and execution practices”

Organisations are viewed as comprising multiple actors or agents with diverse agendas, internally and externally, who seek to coordinate their actions so as to exchange information, act, and interact in a non-linear and dynamic fashion. The agents in the system also follow operating rules or schema-data as in any complex system. In organisation, this schema-data is known more as strategies, plans, culture and vision⁷⁶. This schema-data is evident in the organisations' role in the market and outward appearance.

Just like any other complex system, there are challenges that organisations face, dealing with emerging trends, information generation that is beyond expected, knowledge generation with accelerated diffusion⁷⁷.

2.5.3 Feedback

Non-linear adaptive systems are driven by either a positive and or negative feedback loop. The key difference is controlling of entropy of the system.

Understanding the short term future of an adaptive system is more predictable. In a state of instability, it takes time for the system to react to changing conditions and to extrapolate these conditions to changes having an effect on the systems outcomes.

Thus, individuals are able to plan for the next desired short-term state based on current information of the system. The more frequent the feedback, the better the forecast of the next state of the system will be. It is important to note that the long-term outcomes based on fixed

⁷⁵ J Highsmith, “Highsmith: Agility in Software Development - Google Scholar,” *Agile Software Development Ecosystems* (2002),

⁷⁶ McCarthy, “Technology Management—a Complex Adaptive Systems Approach.”

⁷⁷ McCarthy, “Technology Management—a Complex Adaptive Systems Approach.”

points of reference do not have the same effect. Learning and feedback from the system in order to understand the emergent nature of the system and influence short range behaviour in iterations.

Positive Feedback (amplifying) encourages the system to do more of what it was doing before the feedback was given. Positive feedback is seen to have amplifying effect on the system, again the increase of entropy leads to increases in outputs (information), and moving it towards chaotic behaviour. Positive feedback is instrumental in change in a system.

Negative feedback has a stabilising effect on complex systems. It conveys the message to stop the system from doing what it was doing before the message was received.

Feedbacks are iterations in the system cycle. A feedback loop is often used as an observation of output from one cycle applied as control mechanism for the next cycle

2.5.4 Self-Organisation

Self-organisation can best be described as a process of structuring or restructuring without outside influence. This kind of behaviour is evident in and the default of many systems.

Complex systems are described as being self-organising situations where order is created out of randomness, in effect changing the concept of order decaying into disorder⁷⁸. Spontaneous emergence of order without the involvement of a designer is a hard concept to grasp with dominant thinking of influences of control. The truth is that self-organisation is not mystical or random. Self-organisation occurs at states where the system is far from equilibrium, and constant flux enables the dynamic exchange of energy in the system⁷⁹. The structure of the system is not dependant on outside design or control. Structure results from the interaction of the system and its environment.

Cilliers⁸⁰ defines self-organisation as:

“The capacity for self-organisation is a property of complex systems which enables them to develop or change internal structure spontaneously and adaptively in order to cope with, or manipulate, their environment.”

⁷⁸ Mitchell, *Complexity: a Guided Tour*, 40.

⁷⁹ Cleveland, “Basic Concepts and Application to Systems Thinking.”

⁸⁰ Cilliers, *Complexity and Postmodernism*, 90.

“Dissipative structures⁸¹” is the term used by Ilya Prigogine for describing self-organising systems, observing that the system consumes energy and then dissipates it into the environment. Prigogine describes dissipative structures as having a global structure, with constant internal changes to patterns

2.5.4.1 Cellular Automata

Complexity theories have used Cellular automata, first developed John von Neumann, as an instrumental way of visualising and modelling self-organisation and emergence⁸². Cellular describes the division of the system into individual sites or cells and automata is the computational component of the system⁸³.

Physical cellular automata can be described as a grid based system of cells or sites⁸⁴. Each site has a set of simple rules that governs its behaviour. Rules are time dependent and are computed at each time step of the system based on the current state of the system⁸⁵.

Cellular automata mimic complex systems as they contain large numbers of simple agents. The system as a whole does not have a central control mechanism⁸⁶. The consequence of this is that the system exhibits complex behaviour that cannot be planned⁸⁷.

2.5.4.1.1 Game of Life

Conway's game of life is a simplified version of cellular automata. The game of life has only two states for each site or agent in the grid, being either on or off. Further understanding of the meaning of the game of life, Conway described sites that are off as dead and sites that are

⁸¹ Nicoletta Orsucci Franco Sala, “Chaos and Complexity Research Compendium” (October 30, 2011): 9.

⁸² Allen, Maguire, and McKelvey, *The SAGE Handbook of Complexity and Management*.

⁸³ Allen B Downey, *Think Complexity*, (O'Reilly Media, 2012).

⁸⁴ Mitchell, *Complexity: a Guided Tour*. 147.

⁸⁵ Downey, *Think Complexity*.

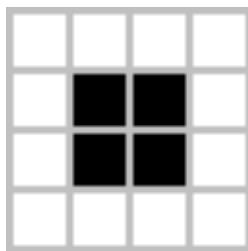
⁸⁶ Mitchell, *Complexity: a Guided Tour*.

⁸⁷ Allen, Maguire, and McKelvey, *The SAGE Handbook of Complexity and Management*.

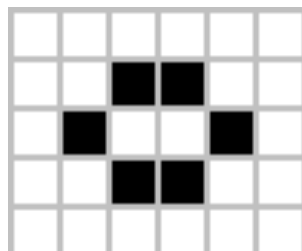
on as alive. The game of life has four basic rules that govern the state of the site. Rules of sites in a cellular automata system depends on the value of a site, and the values of its two nearest neighbours⁸⁸.

1. **Birth:** a site with three neighbouring sites is set to on.
2. **Survival:** a site that is current on which two or three neighbours will stay on
3. **Loneliness:** a site with less than two neighbouring sites that are alive will die or be set to off.
4. **Death:** a site with less than three neighbours in the on state will stay off.
5. **Overcrowding:** a site with more than 3 neighbouring sites that are on will die

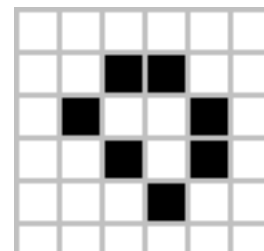
Many interesting patterns emerge from Conway's game of life. The patterns are broken into 3 categories being still lifes, oscillators and patterns with movement.



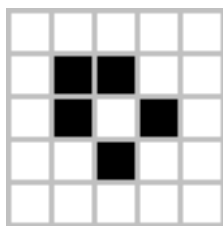
Block



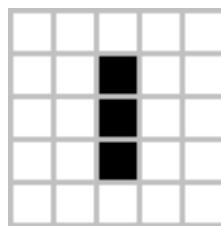
Beehive



Loaf



Boat



Line

Figure: 7 Basic Patterns Seen in Cellular automata still lifes,

⁸⁸ Stephen Wolfram, "Complex Systems Theory (1984)," *Stephenwolfram.com*, April 1, 1984, <http://www.stephenwolfram.com/publications/articles/general/84-complex/2/text.html>.

2.5.5 Emergence

The concept of emergence covers varied types of behaviour over many sets of disciplines. A common set of characteristics can be understood that are part of emergent behaviour can be identified as the following;

1. A period of disequilibrium in which spontaneous fluctuations emerge forming the seeds of new emergent order;
2. Positive feedbacks which amplify the fluctuations of #1;
3. Re-combinations and new correlations of existing resources, capabilities, symbols, language, and work patterns;
4. Coordinating mechanisms that stabilise the new emergent order.

As a summary to encapsulate these four statements, Emergence leads to a level of unpredictability in the future state of the system based on new macro level behaviour based on the interactions of agents in the system⁸⁹.

All complex adaptive systems natural and social, display emergence. Emergence is especially visible in organisations that are in states of constant change. This is an alternative view how organisations develop structure, strategies with little outside control. As most of the change is not driven by imposition of control, the observation is that the solution is creative and has commitment on the individual level due to the feeling of empowerment⁹⁰. It is important to note that self-organisation and emergence are linked, but not necessarily dependent.

2.6 Wicked Problems

Complexity, as described above, can be abstract from problem domains in which individuals are faced with daily. In general conversation, problems are understood to be complex or unknowable. It is these problems which are complex in nature which are a more tangible way for most people of identifying complexity than the abstract metaphors which have been placed in describing up to now. The key point to highlight is that the complexity associated with the system drives the associated problem space. To represent the problem space the term

⁸⁹ Allen, Maguire, and McKelvey, *The SAGE Handbook of Complexity and Management*.

⁹⁰ Allen, Maguire, and McKelvey, *The SAGE Handbook of Complexity and Management*.

wicked problem was devised by Horst Rittel⁹¹. Rittel's view was that of social planning space, stating that wicked problems have ten characteristics:

1. There is no definitive formulation of a wicked problem.
2. Wicked problems have no stopping rule.
3. Solutions to wicked problems are not true-or-false, but good or bad.
4. There is no immediate and no ultimate test of a solution to a wicked problem.
5. Every solution to a wicked problem is a "one-shot operation"; because there is no opportunity to learn by trial and error, every attempt counts significantly.
6. Wicked problems do not have an enumerable (or an exhaustively describable) set of potential solutions, nor is there a well-described set of permissible operations that may be incorporated into the plan.
7. Every wicked problem is essentially unique.
8. Every wicked problem can be considered to be a symptom of another problem.
9. The existence of a discrepancy representing a wicked problem can be explained in numerous ways. The choice of explanation determines the nature of the problem's resolution.
10. The planner has no right to be wrong (i.e., Planners are liable for the consequences of the actions they generate).

Conklin later generalised the concept of problem wickedness to areas other than planning and policy. The defining characteristics are:⁹²

1. The problem is not understood until after the formulation of a solution.
2. Wicked problems have no stopping rule.
3. Solutions to wicked problems are not right or wrong.
4. Every wicked problem is essentially novel and unique.
5. Every solution to a wicked problem is a 'one shot operation.'

⁹¹ Jeff Conklin, "Wicked Problems & Social Complexity" (2006).

⁹² Conklin, "Wicked Problems & Social Complexity."

6. Wicked problems have no given alternative solutions.

The wicked problem description does not cover complexity explicitly, but the description of the nature of the problem is where Rittel tries to evoke awareness that these problems are different than normal problems that could be encountered, without the associated complexity. It is these problems which are seen in organisations that create an increase in the amount of information which has to be processed.

2.7 Classifying complexity theories

The chapter has touched on the issues of representation of an interpretation of complexity. This is especially relevant to the social aspect of complexity. The epistemological issues of understanding complexity are very real. To fully appreciate how each of the theories applies to the understanding of complexity, the following table is summary of complexity from the paper ‘Simplifying Complexity’ by Mason⁹³. The table tries to identify aspects of each level of complexity in order to build a clearer picture of attributes which can be understood as relevant in complex systems.

Over-arching Area	Associated Disciplines	Level observed at	Epistemology	Focused on
Algorithmic complexity	Information theory Mathematical complexity Limits of knowledge, limited to problem solving	Agent	Rules	Given properties of systems parts

⁹³ Manson, “Simplifying Complexity: a Review of Complexity Theory.”

Deterministic complexity	Chaos Mathematical attractors Feedback Sensitivity to initial conditions and bifurcations Strange attractors	Agent/System	Rules / Heuristics	Properties of systems parts and influences on state
Aggregate complexity	Relationships between agents Structure and environment, Learning and Emergent behaviour, System change.	Organisation	Heuristics	Properties of relationships between systems parts, past behaviours, systems states

Table: 5 Complexity Theories and associated disciplines

Any understanding of complexity is based on the perspective of the observer that has been applied to the system⁹⁴. Each discipline can be applied to complexity to gain further understanding of the system. Logical grouping of disciplines into three categories or overarching areas helps with coherence with regards to the complexity theories⁹⁵. Complexity theories can be classified into three major divisions according to Manson⁹⁶

1. Algorithmic complexity defines that complexity is based on the difficulty in describing the system characteristics. Concepts such as information theory and mathematical complexity are key components in describing this approach.
2. Deterministic complexity describes the chaos theory. The basis of interaction of key elements can cause sudden changes in the system.

⁹⁴ Manson, "Simplifying Complexity: a Review of Complexity Theory."

⁹⁵ Manson, "Simplifying Complexity: a Review of Complexity Theory."

⁹⁶ Manson, "Simplifying Complexity: a Review of Complexity Theory."

3. Aggregate complexity describes complex behaviour of systems based on the interactions of individual elements in the system.

In addition to this, complexity can be categorised into two properties⁹⁷; objective complexity describes the complexity in the world we live in or the complexity of structures of the systems, whereas subjective complexity is the personal response that is needed to understand the world. From this we deduce that complexity is paradoxically a property of a system or a manifestation of the observer.

2.7.1 Limits to understanding

An observer-based view on both subjective and objective complexity is the amount of energy that is needed to make sense of the world we live in given the amount of information available. Complexity forces individuals to confront the limits of knowledge, which include objectivity and subjectivity, towards the system being observed⁹⁸.

The challenge is providing the necessary tools and models to individuals to deal with complexity on the structural level or system based, and also the subjective space. While the subject space may render wicked type problems that are evident from the complexity of the system, the ability to change the system to absorb the complexity may not be as transparent. Thus complexity can be used as a metaphor to challenge conceived ideas of a system. It can also be applied literarily

As Allen states:⁹⁹

“A complexity perspective thus provides a scientifically grounded basis for accepting two paradoxical forms of wisdom. Individuals can change their worlds through their interventions, but their agency must be reflexive and respectful of the complexity of the system in which they are embedded.”

⁹⁷ M H Boisot, I C MacMillan, and K S Han, “Explorations in Information Space: Knowledge, Agents, and Organization - Max H. Boisot, Ian C. MacMillan, Kyeong Seok Han - Google Books” (2008).

⁹⁸ Cilliers, *Complexity and Postmodernism*.

⁹⁹ Allen, Maguire, and McKelvey, *The SAGE Handbook of Complexity and Management*, 3.

2.8 Conclusion

Natural science is to thank for the birth of Complexity theories, with the attempts to apply mathematical models to natural systems. This led to additional theories associated with complexity, such as dynamic systems and chaos and complexity adaptive systems. Regardless of whether the system is based in the natural or social sciences, the same theories are applicable, with interaction of individual agents are non-linear in nature and there are elements of self-organisation.

Gell-Mann¹⁰⁰ highlights that the amount of different concepts needed to capture all understandings of complexity would be great. In identifying this dilemma, they also state that there is no agreement on how to define or measure complexity. Any theory applied to defining complexity also runs into ontological and epistemological issues that are intertwined.

"Is complexity an objective property of a system; or does it characterise an observer's subjective efforts of represent and make predictions about the system?" While Cilliers¹⁰¹ argues complexity is both and, neither, he elaborates that the concept of complexity forces us to confront the limits, and hence meaning, of other concepts and their relations, including objectivity and subjectivity, among others.

Complexity theories provide a framework to approach thinking and seeing the world¹⁰². The strength of complexity is that it bridges the natural sciences and social sciences. Complexity is studied in various fields outside of organisational science, changing the way that organisations are understood¹⁰³.

Complexity can be applied metaphorically challenging our ideas of organisations, or it can be applied literally. The concepts behind the study of complex systems seem to be at a level of

100 Murray Gell-Mann, "What Is Complexity? Remarks on Simplicity and Complexity by the Nobel Prize-Winning Author of *The Quark and the Jaguar*," *Complexity* 1, no. 1 (May 16, 2013): 16–19, doi:10.1002/cplx.6130010105.

101 Cilliers, *Complexity and Postmodernism*.

102 Eve Mitleton-Kelly, "Organisations as Complex Evolving Systems" (1998): 4–5.

103 Mitleton-Kelly, "Organisations as Complex Evolving Systems."

abstraction that can be applied to almost any phenomena in which agents have rich interactions¹⁰⁴.

This chapter introduced some of the basic concepts and principles behind complexity. The application of a variety of concepts from complexity enables the greater understanding of organisations as complex adaptive systems. The stretching of the ontological and epistemological boundaries that are accepted and applied to organisational thinking is important. The confrontation of the limits and meaning of knowledge objectivity and subjectivity is the essence of the concept of complexity¹⁰⁵.

¹⁰⁴ Allen, Maguire, and McKelvey, *The SAGE Handbook of Complexity and Management*, 2.

¹⁰⁵ Mitchell, *Complexity: a Guided Tour*.

3

AGILE SOFTWARE DEVELOPMENT

3.1 Introduction

Building software is different to any other type of engineering that humans have done in the past. Historically, software developers were trained by engineers. This dictated how individuals approached creating software for nearly a generation¹⁰⁶. Traditional software development methodologies implemented disciplined processes based on concepts from physical engineering. The aim was to make software development predictable and efficient, and the thought was that with upfront planning and design this was achievable¹⁰⁷.

The first conference about software engineering was held in Berlin Germany in 1967 by NATO. Some attendees proposed an iterative model to software development with a test before you build mentality that we are only seeing mature software teams implement in the last decade¹⁰⁸. However the majority of the group decided to back defined process models to software development, sparking off the adoption of what is known now as waterfall based software development.

¹⁰⁶ “Agile Software Development: a Gentle Introduction.” *Agile-Process.org*, accessed October 27, 2013, <http://www.agile-process.org/>.

¹⁰⁷ III James A Highsmith, *Adaptive Software Development*, (Dorset House, 2000).

¹⁰⁸ Brennecke and Keil-Slawik, “History of Software Engineering.”

3.1.1 History of Software development

Software development is a central discipline in computer science, and it encompasses many factors that can be observed within the realm of computing and creation of new functionality in technology¹⁰⁹.

1. Software development is highly innovative, with rapid change associated with the field there is no central core of knowledge that an individual must know¹¹⁰
2. Few results are supported by comparative studies
3. There is a three to four year lifespan of work in the field in which it remains relevant
4. Old problems are given new names, with the need to resolve these problems instead of applying past solutions
5. Software development is driven by economic and societal demands
6. The ability for individuals to work with interdisciplinary fields is required, e.g. business, mathematics, and psychology,

3.1.1.1 The software Crisis

The software crisis was identified in the early days of computing. As problem complexity increased, coupled with the expectations of the users and the high rate of change that was being experienced, it became problematic to create software that would be correct and understandable.

The some of the following key aspects made the crisis visible:

1. Over budget projects
2. Late delivery of projects
3. Inefficient software
4. Low levels of software quality
5. Requirements gaps on delivery

¹⁰⁹ Brennecke and Keil-Slawik, “History of Software Engineering.”

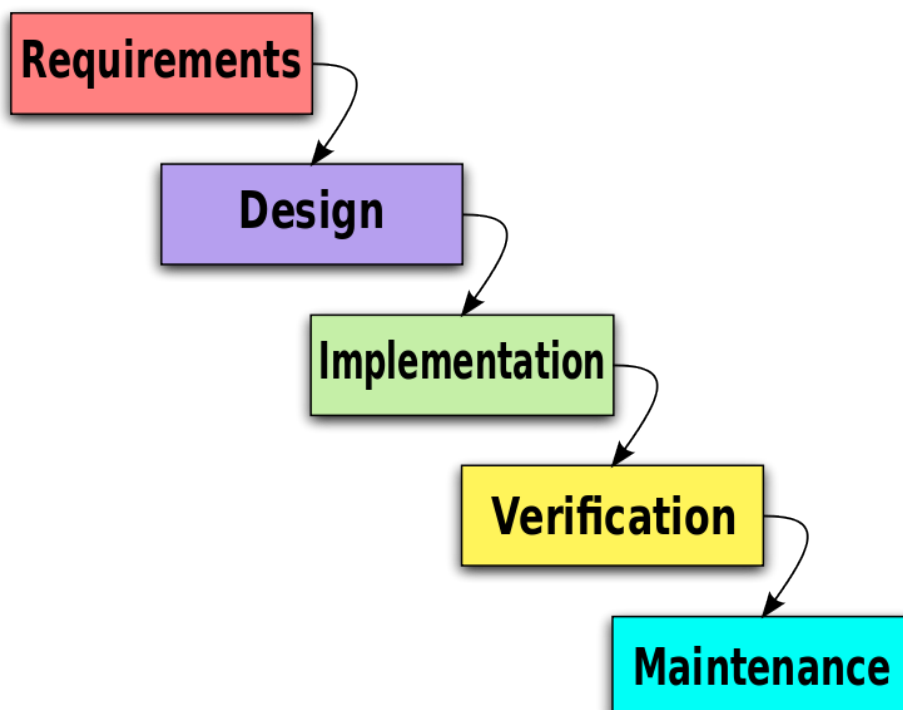
¹¹⁰ Software development spans various fields of knowledge in which programmers must become knowledgeable about to deliver working software

6. Projects where unmanageable
7. Code was difficult to maintain
8. Software was never delivered.

3.1.1.2 Waterfall and predictive processes in software development

Waterfall consists of predefined steps executed in sequential order as per the figure below. Requirements were initially identified and agreed on. A design was created for the software based on these requirements and in the implementation phase the software was coded with verification of the software code happening at the end ensuring the code matched requirements specified in step one. Winston Royce is believed to be the creator of the waterfall model for software development¹¹¹. However his article discourages the use of the model with large software systems, so much so that Royce describes the concept of waterfall in the paper in the following terms

“I believe in this concept, but the implementation described above is risky and invites failure.”



¹¹¹ Royce, “Managing the Development of Large Software Systems.”

Figure: 8 High level abstract of waterfall based process

3.1.1.3 Agile Processes for Software development

A new style of software development, known as Agile software development, has provided organisations with the ability to adapt to the environment they are faced with and enable people to be put first.

Agile processes were developed by people who were and still are deeply involved in creating software. Schwaber states¹¹²:

“Agile processes for software development came into being during the 1990’s. We constructed them based on experience, trial-and-error, knowledge of what didn’t work, and best practices.”

Agile methodologies encompass many process tools that can be applied to software development, and the shared aspect of each is a common set of principles. These principles have a definite contrast to the some of the methodologies used in traditional software development¹¹³. Whilst differences are observable in the frameworks or process applied to development space, there are a certain core set of values and principles that are essential for individuals to understand as a foundation for how to approach work or tasks. There is no magic formula, no silver bullet¹¹⁴. There are, however, some solid theories about which approaches foster high performance and which are likely to hinder it.

Agile software development can be characterised by a method that is incremental and iterative. Agile methods also prescribe the breaking down of tasks into small units of work that are delivered in a time boxed iteration by a cross functional team to produce a complete piece of software.

¹¹² Mary Poppendieck and Tom Poppendieck, *Lean Software Development*, (Addison-Wesley Professional, 2003).

¹¹³ M Fowler, “The New Methodology,” *Martinfowler.com*, accessed October 30, 2013, <http://martinfowler.com/articles/newMethodology.html>.

¹¹⁴ Frederick Brooks, “The Mythical Man-Month - Essays on Software Engineering - . Brooks (Addison-Wesley, 1995) WW” (January 14, 2013).

This umbrella encompasses values and principles such as self-organisation, empirical measurement through feedback loops, and responding to change. The agile manifesto is a starting point to understand these values and principles.

3.2 Process control models

To understand the difference between agile methods to software development and traditional models like Royce's waterfall method, a basic understanding of the two types of process control models is necessary. Traditionally, management models and approaches are steeped in the defined process control model. In contrast to this Agile frameworks are empirical by nature. The approach in which way each model operates is significantly different¹¹⁵.

There are two major approaches to controlling any process:

3.2.1 Defined process control model

When applying a defined control model to a process, it is imperative that every piece of work and its associated activity chain be completely understood. Given a well-defined set of inputs, the same outputs are generated every time. A defined process can be started and allowed to run until completion, with the same results every time¹¹⁶.

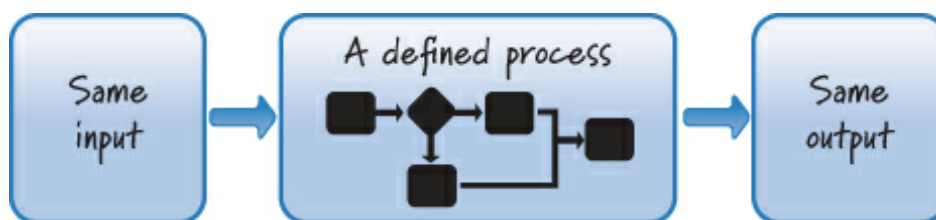


Figure: 9 Abstract of defined process model¹¹⁷

¹¹⁵ Andrea Tomasini and Martin Kearns, “Agile Transition-What You Need to Know Before Starting” (n.d.).

¹¹⁶ Tomasini and Kearns, “Agile Transition-What You Need to Know Before Starting.”

¹¹⁷ Kenneth S Rubin, *Essential Scrum*, (Addison-Wesley Professional, 2012).

Control over the activity is time-based, measuring from the commencement of the activity to its completion. What is observed is that roles in an organisation have a specific set of skills which are then assigned to a single activity¹¹⁸.

Individuals are drawn to Defined Process Models, as they provide simplicity and the initial feeling of safety imposed by the predefinition of the steps required to an end state, even though the end state will probably not be attained.

3.2.2 Empirical process control

Empirical process control differs from defined process control by measuring the outcome of the process which is kept to a defined time interval.

The empirical model of process control provides and exercises control through frequent inspection and adaptation for processes that are imperfectly defined and generate unpredictable and unrepeatable outputs¹¹⁹. The gauntlet of feedback loops associated with an empirical model allows individuals and teams to correct processors based on real-time data that is being received from the system

Measurement of the process happens incrementally, in which productivity is not measured, and the focus shifts to outcomes of the process such as qualitative factors. Changes to the output of the process are affected by changing constraints in the environment that the process is part of. Empirical process control uses observation to deduct meaningful events needed for process change.

3.3 Agile software process frameworks

Agile software development has a number of frameworks that can be applied to the process of development. The framework enables shared understanding and a common way to approach problems and scenarios in software development. Like any tools, Scrum, XP are neither perfect nor complete. The tools do not prescribe a complete solution to achieve agility

¹¹⁸ Tomasini and Kearns, “Agile Transition-What You Need to Know Before Starting.”

¹¹⁹ Tomasini and Kearns, “Agile Transition-What You Need to Know Before Starting.”

or deliver software that will solve business problems. Rather they provide certain constraints and guidelines¹²⁰.

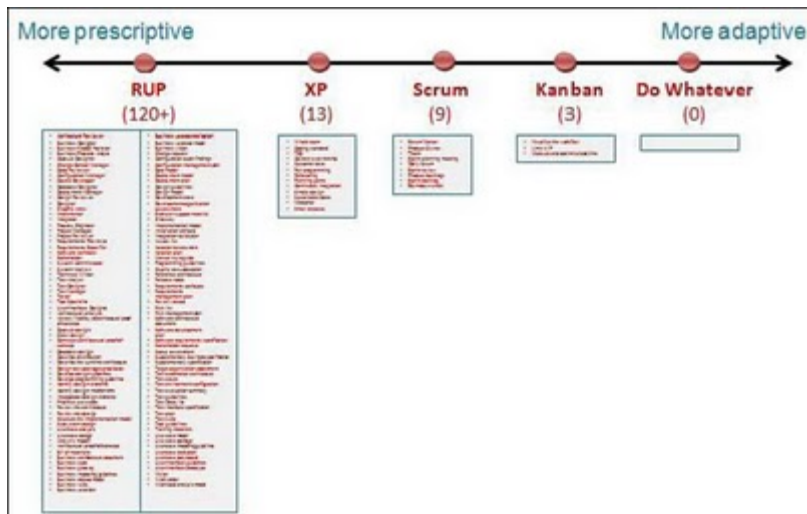


Figure: 10 Agile process frameworks¹²¹

3.3.1 Common aspects of agile process tools

1. Self-organisation – people doing complex work are much more effective organising themselves and the work than someone who is not doing the work.
2. Bottom-up intelligence – figuring out how to do work is a management activity best performed by the people doing the work, since the work is unpredictable, with many twists and turns.
3. Empiricism – it is hard to plan what you do not know, so we instead see what has been accomplished, and then figure out what to do next. We do this frequently to control risk and determine the best path to our goal.
4. Transparency – we periodically have to know what is actually happening to make effective empirical decisions.

¹²⁰ Henrik Kniberg, “Kanban and Scrum-Making the Most of Both” (2010).

¹²¹ Henrik Kniberg, “Kanban vs Scrum,” *Crisp AB. Viitattu* 1 (2009): 2011.

3.4 Agile manifesto

In 2001 leaders from the various lightweight software development frameworks gathered in the ski resort of Snowbird in Utah to discuss the principles behind the frameworks they had created and make this explicit as alternatives to heavyweight documented based development practices. Together they created the agile manifesto, which states:

We are uncovering better ways of developing software by doing it and helping others do it.

Through this work we have come to value:

Individuals and interactions over processes and tools

Working software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

3.4.1 Principles of the Agile Manifesto

In addition to the core manifesto, they also created a set of principles for agile software development. The principles explain the thinking behind creating the agile manifesto from experiences of the group.

3.4.1.1 Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

Projects are driven around producing customer value. Unlike traditional projects, where project success is measured by adherence to an overall plan, agile reviews and evaluates customer value on a frequent basis. The initial plan of a project may not contribute to the success of the project at the end¹²².

¹²² M Fowler and J Highsmith, “The Agile Manifesto,” *Software Development* (2001).

3.4.1.2 Welcome changing requirements, even late in development Agile processes harness change for the customer's competitive advantage.

Responding to change is the best-known item of the agile manifesto and something that most people attribute to being "agile". Change is ever present in a software project, instead of resisting change, agile accommodates it as efficiently as possible. The importance of accommodating change is to communicate the consequences of change¹²³.

Change is normally seen as something external that the team has to absorb, normally characterised by things like changes in requirements, business environments and problem domain. The team should also be able to deal with internal change when creating software, things like changes to the codebase, individuals and technology stacks could be identified as internal change.

Therefore, teams only plan for short iterations and long term plans are malleable. Feedback results in change. Due to the high levels of feedback built into agile processes, the need to facilitate change is important¹²⁴.

3.4.1.3 Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

As a primary measure of an agile teams progress, there is nothing greater than the demonstration of working software after a team has completed iteration. Not only does it provide a tangible marker, but also a verification of customer requirements and instant feedback loop for the team. Delivering working software on a regular basis enables functionality to grow; the shorter the delivery cycle, the better for agile projects¹²⁵.

3.4.1.4 Business people and developers work together daily throughout the project.

Frequent interactions between business people, the purchasers of the software and developers are vital for the success of an agile project. To highlight how important this interaction is in

¹²³ Fowler and Highsmith, "The Agile Manifesto."

¹²⁴ Fowler and Highsmith, "The Agile Manifesto."

¹²⁵ Fowler and Highsmith, "The Agile Manifesto."

an agile project, the creators of the manifesto used the word daily to ensure that there was continual commitment and shared responsibilities between all parties¹²⁶.

3.4.1.5 Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

Agile is a people-first approach to software development. Individuals and teams must be trusted to get the job done. Decision making is deferred to people who have the most knowledge and information about the situation¹²⁷. While process is necessary for projects, it is of second importance to people.

3.4.1.6 The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Agile methods are thought to be lacking in documentation. While documentation is important, it is not as important as understanding. Face-to-face communication conveys better understanding and context than codified information.

3.4.1.7 Working software is the primary measure of progress.

Working software is an accurate measure of the progress of a project. It takes in to account all steps of the software lifecycle to ensure that everything is covered. This iterative approach ensures that milestones cannot be misconstrued. Risks are made visible not by documentation, but rather by conversations about physical products.

3.4.1.8 Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

Agile software development is a mindset encompassing creativity and innovative thinking. Without this mindset, agility cannot be achieved. Sustainable pace ensures that the team can

¹²⁶ Fowler and Highsmith, “The Agile Manifesto.”

¹²⁷ Fowler and Highsmith, “The Agile Manifesto.”

stay alert and creative for the full length of the project. The concept of longer hours leading to more productivity is incorrect.

3.4.1.9 Continuous attention to technical excellence and good design enhances agility.

Agile may provide more speed and flexibility to the project to achieve and maintain agility the correct design principles and technical practices have to be implemented. Agile processes allow and encourage the changing of requirements while code is being created. This forces design to be continuous and iterative.

3.4.1.10 Simplicity, "the art of maximising the amount of work not done", is essential.

Simple approaches to work are better placed when dealing with unknowns, and the ability to add to simplicity is easier than to remove from something that is complicated. Agile embraces minimalism in thinking; this is achieved by building the smallest possible thing that is required by all users of the system, and not catering for individual needs of each user¹²⁸. By giving people a simple set of rules, creativity is stimulated.

3.4.1.11 The best architectures, requirements, and designs emerge from self-organising teams.

The systems that are built in are normally complex. The ability to predict or plan systems in advance is very rare, in understanding this agile embraces that the design and architecture of the system will emerge from understanding created during iterations, not from early planning. To facilitate this emergence of design, teams must be allowed to self-organise around problems¹²⁹.

¹²⁸ Fowler and Highsmith, "The Agile Manifesto."

¹²⁹ Fowler and Highsmith, "The Agile Manifesto."

3.4.1.12 At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly.

Agile methodologies are not implemented to be followed blindly. The premise around agility is that it invokes a learning based approach to work, and this should also apply to the process. The philosophy of inspect and adapt applies to process applications. The processes implemented must be refined by the team using it¹³⁰. By reflecting constantly the team can implement constant improvement.

3.5 Agile methodologies

Agile software development is a broad term that is core to the thinking of many approaches to software development. Each approach has unique ideas and practices that make each of them unique. Some are more prescriptive in the role of process e.g. scrum and others are more geared towards technical practices of the team e.g. XP. Principles are largely interchangeable between approaches, with the practices and ideas being shared between communities. This is the key understanding of agile methodologies, the ability to change for the context they are implemented in, but also the adoption of practices that work enabling teams and individuals to learn from the problems they face and improve.

There are numerous methodologies that can be categorised under the umbrella of agile¹³¹. In the next section, three agile methodologies are described as a base understanding of agile software development appreciation. Extreme programming has been chosen because of the combination of principles and practices, while scrum is described as its focus on iterative process and identification of learning opportunities in the associated ceremonies. Finally, Lean software development is introduced as its seven principles can be applied beyond a software team and influence the mindset of the organisation¹³².

¹³⁰ Fowler and Highsmith, “The Agile Manifesto.”

¹³¹ Kniberg, “Kanban and Scrum-Making the Most of Both.”

¹³² Poppendieck and Poppendieck, *Lean Software Development*.

3.5.1 XP (Extreme Programming)

Extreme programming (XP) is described as a lightweight methodology for teams of software developers whose environment consists of vague and continuously changing requirements or problem statements.

XP takes some of the good practices that have been proven and turn them up to extreme levels. Some of these include:

- Code reviews help with finding out errors by having many people look at the code. In XP, code is reviewed constantly by pair programming.
- Testing of code is essential. This should be done all the time, with unit tests and the customer performing the functional tests.
- Design is lead by the team. To ensure the design is the best the team will constantly refactor code
- If simplicity is an enabler, having this as a key goal means the system will be left with the simplest design to meet functionality
- The ability to integrate with other systems is important, Continuous integration is used in a project.
- The shorter the iteration the better. In XP iterations happen in minutes and hours not weeks and months or years.
- XP helps all parties in the software development lifecycle. Programmers are empowered to work on things that matter on a daily basis. Customers and managers will get value out of every iteration, with visible progress.

Beck¹³³ describes XP as being distinguishable by the following attributes

1. Feedback is early and concrete derived from short cycles
2. Planning is incremental, which lets the plan evolve through the project lifecycle
3. Functionality is able to be flexibly scheduled to be implemented there by responding to business needs
4. Defects in software are caught early due to automated testing

¹³³ Kent Beck, "Extreme Programming Explained," *Embrace Change* (2006).

5. Communication via oral, tests and source code
6. The design of the system is evolutionary, with a process lasting as long as the system.
7. Programmers with ordinary skills closely collaborate to get the tasks completed
8. Practices that cater for short term instincts of the programmer and longer term interests of the project.

3.5.1.1 Values of XP

XP has four core values, and Beck describes the values of XP as being essential for keeping the team on track to a long-term goal. This is important as short-term individual goals can be contradictory compared to the goals of the social group.

1. Communication is identified as the cause of most errors in software development. Communication between team members and also between customers is vitally important.
2. Simplicity in XP is summed up by the statement “Do the simplest thing that could possibly work”. A simple system is of higher quality and easier to add to and change in the future.
3. Feedback is the key measure of the state of the product being developed. XP sees feedback coming from contact with the customer and automated software tests which are used in the development of the system. Paradoxically feedback is important part of communication, but feedback spurs on communication
4. Courage of individuals and the team allows for the team to be autonomous. Courage is the last value in XP because without the previous three having courage would not lead to anything positive for the team. Courage allows a team to try something new, change the way the project is going and experiment.

3.5.1.2 Practices of XP

Originally XP was defined to have 12 practices that were part of the teams responsibility. These practices inform people how to do XP.

- **The Planning Game:** Quickly determine the scope of the next release by combining business priorities and technical estimates. As reality overtakes the plan, update the plan.
- **Small releases:** Put a simple system into production quickly, then release new versions on a very short cycle.
- **Metaphor:** A simple shared story of how the whole system works is the cornerstone of development.
- **Simple design:** The system should be designed as simply as possible. Extra complexity is removed as soon as it is discovered.
- **Testing:** All code is covered by unit test, which must pass for development to continue. Customers write tests demonstrating that features are finished.
- **Refactoring:** Restructure the system without changing behaviour to remove duplication, improve communication, simplify, or add flexibility.
- **Pair programming:** Programming is done by two people at one machine.
- **Collective ownership:** Anyone can change any code anywhere in the system at any time.
- **Continuous integration:** Integrate and build the system many times a day
- **40 hour week:** Never work more than 40 hours in a week. Never work overtime a second week in a row.
- **On-site customer:** A person who is from the business is on the team, available full-time to answer questions.
- **Coding standards:** All code is written in accordance with coding rules. This is to ensure communication through the code.

Beck makes the point that practices support each other and cannot survive alone. All 12 practices provide balance. Figure: 11 re-enforces how the practices support each other



Figure: 11 Practises of XP¹³⁴

3.5.2 Scrum

Scrum is described as a framework for developing and sustaining complex products. It provides the tools necessary for individuals to address complex adaptive problems while productively and creatively delivering high value products¹³⁵. Originally developed by Jeff Sutherland and Ken Schwaber¹³⁶ scrum has its roots in Japanese management thinking and Complex Adaptive Systems theory. Specifically Hirotaka Takeuchi and Ikujiro Nonaka first described scrum in the paper "New New Product Development Game"¹³⁷. The authors explain how a cross functional team of people participate in new approach to product

¹³⁴ Beck, "Extreme Programming Explained."

¹³⁵ Ken schwaber and Jeff Sutherland, "The Scrum Guide" (December 1, 2011): 1–16.

¹³⁶ Kevin Vlaanderen et al., "The Agile Requirements Refinery: Applying SCRUM Principles to Software Product Management," *Information and Software Technology* 53, no. 1 (January 1, 2011): 59, doi:10.1016/j.infsof.2010.08.004.

¹³⁷ Takeuchi and Nonaka, "The New New Product Development Game."

development. They liken the approach similar to a team advancing the ball during a rugby game.

Scrum is a lightweight framework based on empirical process control, measuring the output of the system at the end of each iteration. The framework is simple for anyone to understand consisting of a few prescribed elements; a time-boxed iterative process, roles of individuals, ceremonies and artefacts¹³⁸ used by a team practicing scrum. This framework is elegantly simple¹³⁹, however the framework is difficult to master in practice. Scrum's core functions revolve around providing a platform for the implementation of various processes and techniques that facilitate improvement of the team and highlight impediments in the organisations.

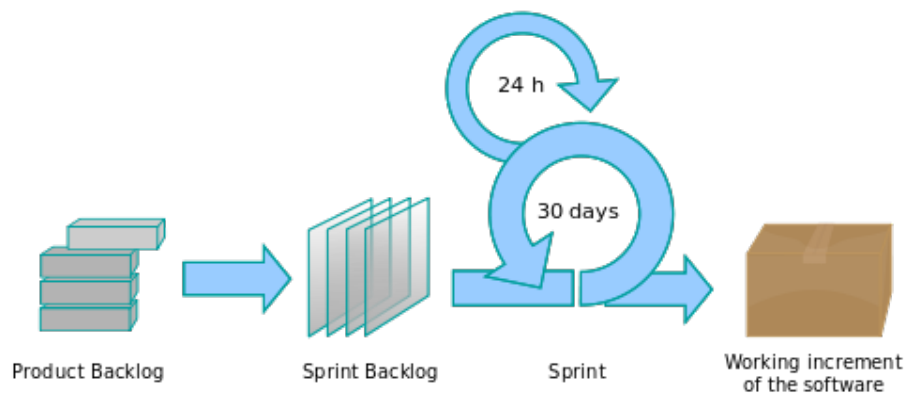


Figure: 12 A high level view of Scrum Framework

¹³⁸ Artefacts in scrum are used to communicate progress and future work. Artefacts and their purpose are described later.

¹³⁹ Rubin, *Essential Scrum*.

3.5.2.1 Ceremonies

Scrum ceremonies are time-boxed events to ensure the correct amount of time is spent planning, with the least amount of waste possible. Other than the ceremonies prescribed by the framework other meetings should be kept to a minimum¹⁴⁰.

Each event in scrum allows for the team to inspect and adapt to a way of working to optimise the process or learn.

3.5.2.1.1 Sprint

Scrum is built around the concept of a sprint. Sprints are time-boxed iteration at the end of which the team is able to deliver a useable product¹⁴¹. According to The Scrum Guide, a recommended sprint or development iteration lasts 30 days or less, with most teams having shortened this cycle to two weeks or even a week. It is important to note that all sprints have the same cadence for the entire project and sprints run sequentially.

Sprints ensure that nothing for the development team changes in the time-box. This includes¹⁴²:

- No changes to sprint goals
- Individuals in the team do not change during the sprint
- Quality goals for the sprint do not change

During the sprint, as knowledge is created, the development team and the product owner clarify and re-negotiate scope of the work in the iteration.

Sprints provide short term horizons for work which are achievable and limit the risk in the changing environment.

¹⁴⁰ K Schwaber and J Sutherland, “The Scrum Guide—the Definitive Guide to Scrum: the Rules of the Game (2011),” URL <Http://Www.Scrum.Org/Storage/Scrumguides/Scrum%20Guide> (2012): 5.

¹⁴¹ Schwaber and Sutherland, “The Scrum Guide—the Definitive Guide to Scrum: the Rules of the Game (2011),” 6.

¹⁴² Schwaber and Sutherland, “The Scrum Guide—the Definitive Guide to Scrum: the Rules of the Game (2011),” 6.

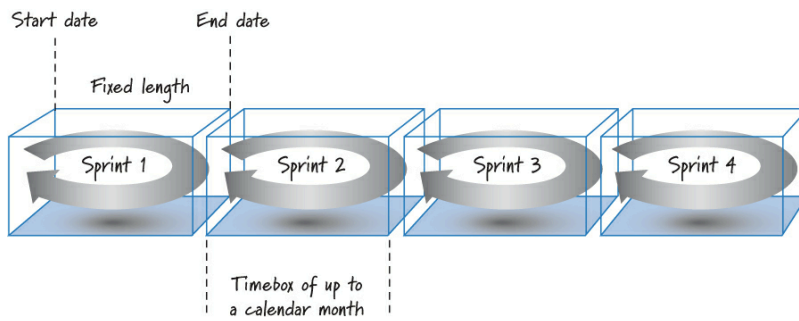


Figure: 13 Sprints in Scrum¹⁴³

3.5.2.1.2 Sprint planning

At the start of a sprint, work which is prioritised for the upcoming iteration is planned collaboratively by the team in Sprint Planning. Sprint planning consists of two parts¹⁴⁴.

In first half of Sprint Planning, known as Sprint Planning 1, the team identifies what will be delivered in the product as a result from the forthcoming sprint. The Product Owner will identify stories and features in the Product Backlog to work on in the upcoming sprint. By using past performance¹⁴⁵ the development team commits to items from the product backlog.

The Product Owner and the team agree on a sprint goal as a guideline to the team on building the product increment¹⁴⁶.

In the second half of Sprint Planning, the development team designs a solution to deliver the requirements of the functionality they have committed to for the sprint. The work is broken down to a task level, with each task lasting no longer than a day¹⁴⁷. The team can self-organise around the tasks to complete the work

¹⁴³ Rubin, *Essential Scrum*.

¹⁴⁴ Schwaber and Sutherland, “The Scrum Guide—the Definitive Guide to Scrum: the Rules of the Game (2011),” 8.

¹⁴⁵ Team velocity is normally used as a standard metric for a team to commit to future work items. By sizing work items based on perceived complexity, the team is able to attribute them to the amount of work they are committing to in a sprint

¹⁴⁶ Rubin, *Essential Scrum*.

¹⁴⁷ Rubin, *Essential Scrum*.

3.5.2.1.3 Daily scrum

The development Team gathers together for a daily stand-up meeting which is time-boxed to 15 minutes. This meeting is held in the same place and at the same time. This ceremony gives the development team the chance to inspect work that has been done in the last 24 hours and plan for the next 24 hours.

During the meeting, each person shares status of the work they have been doing and what they are going to do for the upcoming day. This meeting is not used as a status update session to external stakeholders and the scrum master ensures that only the development team members participate in this.

To ensure this three simple questions are answered by each of the team members¹⁴⁸:

1. What I did yesterday,
2. What I am going to do today,
3. What Impediments I have.

3.5.2.1.4 Sprint review

At the end of a sprint, a review of all finished work that was committed to by the team is demonstrated. This provides the opportunity to inspect and adapt the product backlog to incorporate learning about the product from the last sprint¹⁴⁹. The presentation of working software is used to capture feedback and conversation and enable collaboration in building a better product.

Import aspect of the sprint review include¹⁵⁰:

- The product owner identified which work has met the done criteria and which has not.
- The development team discuss what went well during the sprint, what problems they encountered and how they solved the problems

¹⁴⁸ Rubin, *Essential Scrum*.

¹⁴⁹ schwaber and Sutherland, "The Scrum Guide," 11.

¹⁵⁰ Rubin, *Essential Scrum*.

- The Product Owner discusses the current product backlog and using empirical evidence from the teams progress to date the product owner highlights estimated completion dates for the product.
- The group collaborates in the next steps and actions to take into Sprint planning.

The sprint review provides the platform to revise and adjust the product backlog to take into account new opportunities discovered. This is achieved by the sharing of concept by a diverse set of stakeholders with different perspectives of the product and environment¹⁵¹.

3.5.2.1.5 Sprint retrospectives

At the end of a sprint and after the sprint review, the team and the Scrum Master will hold a retrospective that looks back the previous sprint. The retrospective allows the team to inspect and adapt based on learning.

Retrospectives can be used for addressing a multitude of issues. One of the main uses for retrospectives is that they provide the opportunity for teams to inspect interpersonal issues and relationships within the team or process based issues¹⁵².

Rubin¹⁵³ Highlights the concept of a retrospective by quoting Keith¹⁵⁴ from the text Project Retrospectives

“Here is Edward Bear, coming downstairs now, bump, bump, bump, bump, on the back of his head, behind Christopher Robin. It is, as far as he knows, the only way of coming downstairs, but sometimes he feels that there is another way, if only he could stop bumping for a moment and think of it.”

The major premise of the retrospective is for the team to identify what went well and what can be improved, prioritise the issues and create a plan of action to implement improvements in the way the team is working¹⁵⁵.

¹⁵¹ Rubin, *Essential Scrum*.

¹⁵² Rubin, *Essential Scrum*.

¹⁵³ Rubin, *Essential Scrum*.

¹⁵⁴ Kerth, Norm. 2001. *Project Retrospectives: A Handbook for Team Reviews*. Dorset House.

¹⁵⁵ Rubin, *Essential Scrum*.

The Scrum Master's responsibility in facilitating the retrospective is to create a safe space from where meaningful data and insights can be generated by the team. The Scrum Master is also responsible for ensuring that the retrospective has meaningful outcomes and prepares for the ceremony around the current sprint¹⁵⁶

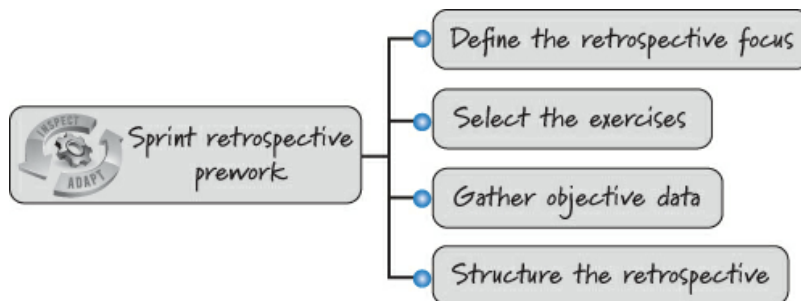


Figure: 14 Retrospective Structure¹⁵⁷

3.5.2.2 Artefacts

Scrum has four main artefacts associated with the process, a product backlog, sprint backlog, Burn down chart and a product increment. Artefacts help the team to be transparent and communicate among themselves and stakeholders who have an interest in what the team is delivering.

3.5.2.2.1 Product backlog

The product backlog is a list of requirements for the product, which is maintained by the product owner¹⁵⁸. In most implementations of Scrum requirements are written in format known as a user story. The product backlog is prioritised based on the most important features for the product to be built next. Items towards the top of the backlog are estimated by size, refined and understood. These items drive the next iterations development efforts

¹⁵⁶ Rubin, *Essential Scrum*.

¹⁵⁷ Rubin, *Essential Scrum*.

¹⁵⁸ Vlaanderen et al., "The Agile Requirements Refinery: Applying SCRUM Principles to Software Product Management," 59.

Items that are lower on the product backlog are left at a level of understanding that is broad. Until the item has migrated its way to the top of the backlog when more in depth investigation will happen¹⁵⁹.

Often backlog also holds information about requirements, such as risk and dependencies. Product backlogs grow and are elaborated with feedback from end users of the product. This creates a living document that is in a constant state of change due to changing markets, business requirements and technology¹⁶⁰.

The Product Owner and team will collaborate and groom the product backlog as necessary during the sprint. Having the domain knowledge about the product lets the team understand product backlog items. Part of the grooming process is to estimate the size of items in the backlog. Items are normally sized in points by giving each item a value from the Fibonacci sequence¹⁶¹. Sizing is subjective to each team, but the greater the number of points the greater the perceived complexity of the item.

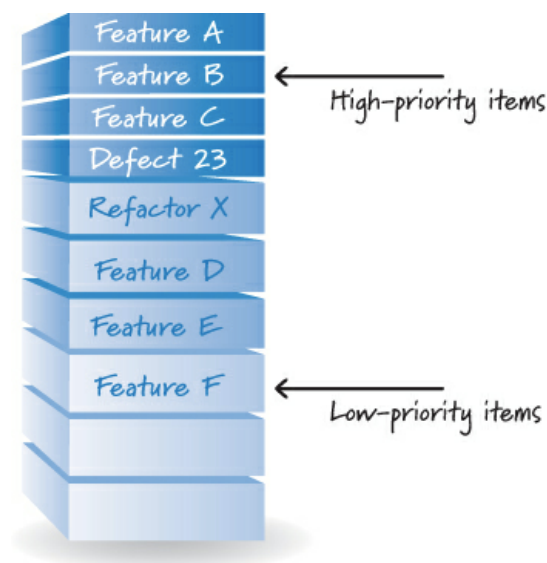


Figure: 15 Scrum product backlog¹⁶²

¹⁵⁹ Schwaber and Sutherland, “The Scrum Guide—the Definitive Guide to Scrum: the Rules of the Game (2011).”

¹⁶⁰ Rubin, *Essential Scrum*.

¹⁶¹ Schwaber and Sutherland, “The Scrum Guide—the Definitive Guide to Scrum: the Rules of the Game (2011).”

¹⁶² Rubin, *Essential Scrum*.

3.5.2.2.2 Sprint backlog

During Sprint Planning one, the development team will commit to work that will fill a sprint from the product backlog. The development team owns the sprint backlog and once they have committed to the work for the sprint there cannot be any addition or subtraction of work from the sprint by any other parties outside of the development team.

Sprint backlogs are normally transposed on to task boards with the columns of to do, doing and done, illustrated in figure: 16

Story	To Do	In Process	To Verify	Done
As a user, I... 8 points	Code the... 9 Code the... 2 Test the... 8	Test the... 8 Code the... 8 Test the... 4	Code the... DC 4 Test the... SC 8	Test the... SC 6 Code the... SC 8 Test the... SC 8 Test the... SC 6
As a user, I... 5 points	Code the... 8 Code the... 4	Test the... 8 Code the... 6	Code the... DC 8	Test the... SC 6 Test the... SC 8 Test the... SC 6

Figure: 16 Scrum task board¹⁶³

3.5.2.2.3 Product Increment

The product increment is the resultant software created by a team once they have finished a sprint. The product increment must be working software that is ready to ship to a production environment.

3.5.2.3 Roles and Responsibilities

Scrum, as defined by the Scrum Alliance, has three main roles that are part of the process. In other literature up to three additional roles are described, however they are not intricate to the Scrum process.

¹⁶³ <http://www.mountaingoatsoftware.com/system/asset/file/29/MockedTaskBoard.jpg>

3.5.2.3.1 Scrum Master

The Scrum Master acts as a coach to the team. In doing this, they ensure that the team is able to meet their sprint commitment. This could include protecting the team from outside influences, removing impediments¹⁶⁴, and ensuring collaboration and communication between team members. In addition, the Scrum Master is tasked with facilitating the Scrum process. This includes running the daily scrum, sprint planning meetings and sprint retrospectives. One thing the Scrum Master is not is a manager. The Scrum Master has a servant leadership position within the team and is often used as a voice of reason or consciousness¹⁶⁵.

3.5.2.3.2 Product Owner

The Product Owner is the single point of responsibility for the product¹⁶⁶. Daily tasks for the Product Owner would include making decisions about what is going to be built by the Scrum Team in the upcoming sprints and in what priority to present the work to the team. Doing this, the Product Owner will be responsible for what is known as managing the Product Backlog.

The product owner does not instruct the Scrum Team on how to design, develop and deliver stories and tasks within the sprint. As the Scrum Team is a self-managed entity the responsibility resides with the team. However, the Product Owner should be not more than 10 minutes away from the team to validate designs and answer the Scrum Team's questions.

¹⁶⁴ Schwaber and Sutherland, “The Scrum Guide—the Definitive Guide to Scrum: the Rules of the Game (2011),” 6.

¹⁶⁵ Schwaber and Sutherland, “The Scrum Guide—the Definitive Guide to Scrum: the Rules of the Game (2011),” 6.

¹⁶⁶ Schwaber and Sutherland, “The Scrum Guide—the Definitive Guide to Scrum: the Rules of the Game (2011).”

At the Sprint Review Ceremony, the Product Owner will accept or reject work done in a sprint that the Scrum Team demonstrates¹⁶⁷.

The Product Owner will also set the vision for the product. This is used to guide the team to building the best possible product without having a long-term defined plan.

3.5.2.3.3 Team

The Scrum Team is constituted of people with different skills ensuring that the team is cross functional¹⁶⁸ and has the ability to create a product increment at the end of the sprint. Team size depends on the product being built; it is recommended that the Scrum team have no more than 9 people¹⁶⁹ as members. The scrum team does not have a leader as an individual within the team as the team is a self-organising unit¹⁷⁰.

3.5.3 Lean Software Development

3.5.3.1 History

In the 1940s, Toyota started motor vehicle manufacturing for the Japanese market. A knock-on effect of the Second World War was that people did not have disposable income for highly priced cars. Although mass production was the cheapest way to manufacture, the market was not large enough to make this viable for Toyota. This caused Toyota to find a way to produce cars in small quantities at the same cost of mass-produced cars of the American motor

¹⁶⁷ Schwaber and Sutherland, “The Scrum Guide—the Definitive Guide to Scrum: the Rules of the Game (2011).”

¹⁶⁸ Schwaber and Sutherland, “The Scrum Guide—the Definitive Guide to Scrum: the Rules of the Game (2011),” 5.

¹⁶⁹ Schwaber and Sutherland, “The Scrum Guide—the Definitive Guide to Scrum: the Rules of the Game (2011),” 6.

¹⁷⁰ Schwaber and Sutherland, “The Scrum Guide—the Definitive Guide to Scrum: the Rules of the Game (2011),” 5.

industry. Taiichi Ohno is credited with creating Toyota Production System (TPS) as a response to this challenge.

Ohno¹⁷¹ describes the Toyota Production System "a system for the absolute elimination of waste." He explains that the system rests on two pillars: Just-in-Time flow and automation (also called Jidoka).

TPS influenced lean manufacturing. The Poppendiecks applied well known industrial lean practices to software development. These lean principles strengthen the understanding, credibility and use of agile practices and principles¹⁷².

3.5.3.2 Principles of lean software development

Lean software development according to Poppendieck comprised of Seven principles;

1. Eliminate waste
2. Amplify learning
3. Decide as late as possible
4. Deliver as fast as possible
5. Empower the team
6. Build integrity in
7. See the whole

3.5.3.2.1 Eliminate waste

Poppendieck¹⁷³ identifies Waste in software development as anything that does not add value to the product, while stating value is driven by customer perception. Waste is often seen in the software development cycle, as completed software that is not in production, requirement gathering for the future. Very much like lean manufacturing, any surplus inventory on hand is considered waste. In software development, creating software, which is not put into

¹⁷¹ Taiichi Ohno and Taiichi Ōno, *Toyota Production System*, (Productivity Press, 1988).

¹⁷² Poppendieck and Poppendieck, *Lean Software Development*.

¹⁷³ Poppendieck and Poppendieck, *Lean Software Development*.

production, is seen as waste. Additionally, waste is also seen as handover between teams where software is handed off for completion.

Poppendieck¹⁷⁴ states there are 7 forms of waste in software development;

1. Partially Done Work
2. Extra Processes
3. Extra Features
4. Task Switching
5. Waiting
6. Motion
7. Defects

3.5.3.2.2 Create Knowledge Amplify Learning

Software development is a discovery and knowledge creating process¹⁷⁵. The premise of software development is not to create a cookie cutter model to generating the same result but to solve unique problems that a customer has. This very much different from lean production where the goal is reduction in variation.

In the text "Lean software development"¹⁷⁶, a comparison is made between chefs who create recipes and production lines working off the recipe to make a dish. This is used to explain two import concepts, firstly the ability of chefs being able to adapt to the environment around them with changing ingredients and occasions, and secondly, the concept of iterations, in which variations of the end product are created as a learning process. This enables refinement and feedback. This is metaphor for how a software development process happens. As the complexity is increased so should the amplification of learning

3.5.3.2.3 Defer commitment (decide as late as possible)

¹⁷⁴ Poppendieck and Poppendieck, *Lean Software Development*.

¹⁷⁵ Poppendieck and Poppendieck, *Lean Software Development*.

¹⁷⁶ Poppendieck and Poppendieck, *Lean Software Development*.

Uncertainty is prevalent in software development. To help with dealing with this, practices that provide for making decisions as late as possible are key to provide an options based approach to decision making. Decisions are based on fact and informed by data versus speculation of the future provide to be of more value. Additionally, Poppendieck argues that when dealing with complex systems,

“a key strategy for delaying commitments is to build the capacity of change in the system.”

3.5.3.2.4 Deliver fast

Traditionally, the view in software development was that a slow pace of delivery, which was pedantic, produced high quality. However, with long release cycles and feedback loops, this method does not provide an organisation the competitive advantage of amplifying learning and enabling decisions to be delayed.

Delivering fast enables companies to have fewer resources tied up with work in process (partially done software) and more time analysing feedback cycles.

There is a difference between delivering software fast or as fast as possible and just delivering fast. One of the most important principles achieving fast delivery is the ability to build quality into the product.

Poppendiek states that

“There are two ways to achieve high quality. You can slow down and be careful, or you can develop people who continually improve their processes, build quality into their product, and develop the capability to repeatedly and reliably respond to their customers many times faster than their competitors”

3.5.3.2.5 Empower the team

Traditional management theory, as developed by Taylor, prescribes how managers tell workers how to complete tasks. In lean the opposite is true; the thinking is the opposite one of finding good people and leaving them to do their job. This leads to great execution which is driven by getting the details right. The people who know the specifics of the work being done are the people doing the work.

Delaying of decisions and fast execution means that there is automatic inability for central authority to manage activities of workers. Pull techniques with local signalling for the management of work are used.

Management takes on another role, one of leading and enabling. Most importantly encouraging individuals in task progress. In addition, important functions such as catching errors and removing impediments from individuals and teams are now management tasks.

3.5.3.2.6 Build integrity within

Poppendeck identifies two types of integrity which lean software development encourages in products.

Perceived integrity: A Customer's whole experience of a system. Poppendeck describes this as the customer is delighted by the product and thinks: "*this is exactly what I want.*" A measurement of perceived integrity is marketshare of the product, as it is a measurement of perception of the customer over a time period.

Conceptual integrity: To create perceived integrity, the system should work smoothly as a whole. Software integrity is also driven by the need to adapt over time. Some of these attributes of conceptual integrity include coherent architecture, good usability and it is fit for purpose. This type of integrity comes from leadership, expertise in the problem domain, good communication and a discipline applied to software development.

3.5.3.2.7 Optimise the whole

To insure integrity in a complexity system, deep understanding is required across diverse areas of the system. Typically, in product development, problems arise because experts of an area¹⁷⁷ will only perform optimisation of area they represent, rather than the whole. Thus the overall system will suffer as sub-optimisation is the result of measurement of contribution of people with a penchant to specialisation.

The problem of sub-optimisation is extrapolated when two entities with competing views are part of the same system. Poppendeck uses the example of two organisations coming together

¹⁷⁷ Products are normally divided into areas such as database, application logic and GUI

to fulfil work. The goal of maximising performance for the each of the entities is greater the goal of whole system performance.

3.6 Agile Culture

Agile is about a shift in thinking. Although there are many process tools that are applied to agile software development, this does not differentiate it from any other approach to management. The true underlying essence of agile is described by the agile manifesto. Agile is an idea, supported by a set of values and beliefs. Agile is not all about process, but the culture it instils from the implementation of the process combined with values beliefs and principles.

The process is adaptive; instead of resisting change like a defined process, the process is built around embracing change. Often the process itself is in a constant state of change, in order to adapt to the system they are in.

Agile methodologies put people before the process. They recognise that the process is there to support the people doing the work, and the skills of the team of individuals cannot be replaced by a process.

3.6.1 Principles of Agile

Rubin identifies some of the principles of Agile software development. In Figure 17, the collection of practises and explanations allow for understanding how the application of agile

as a response to complexity in the forthcoming chapters is a coherent, relevant argument.

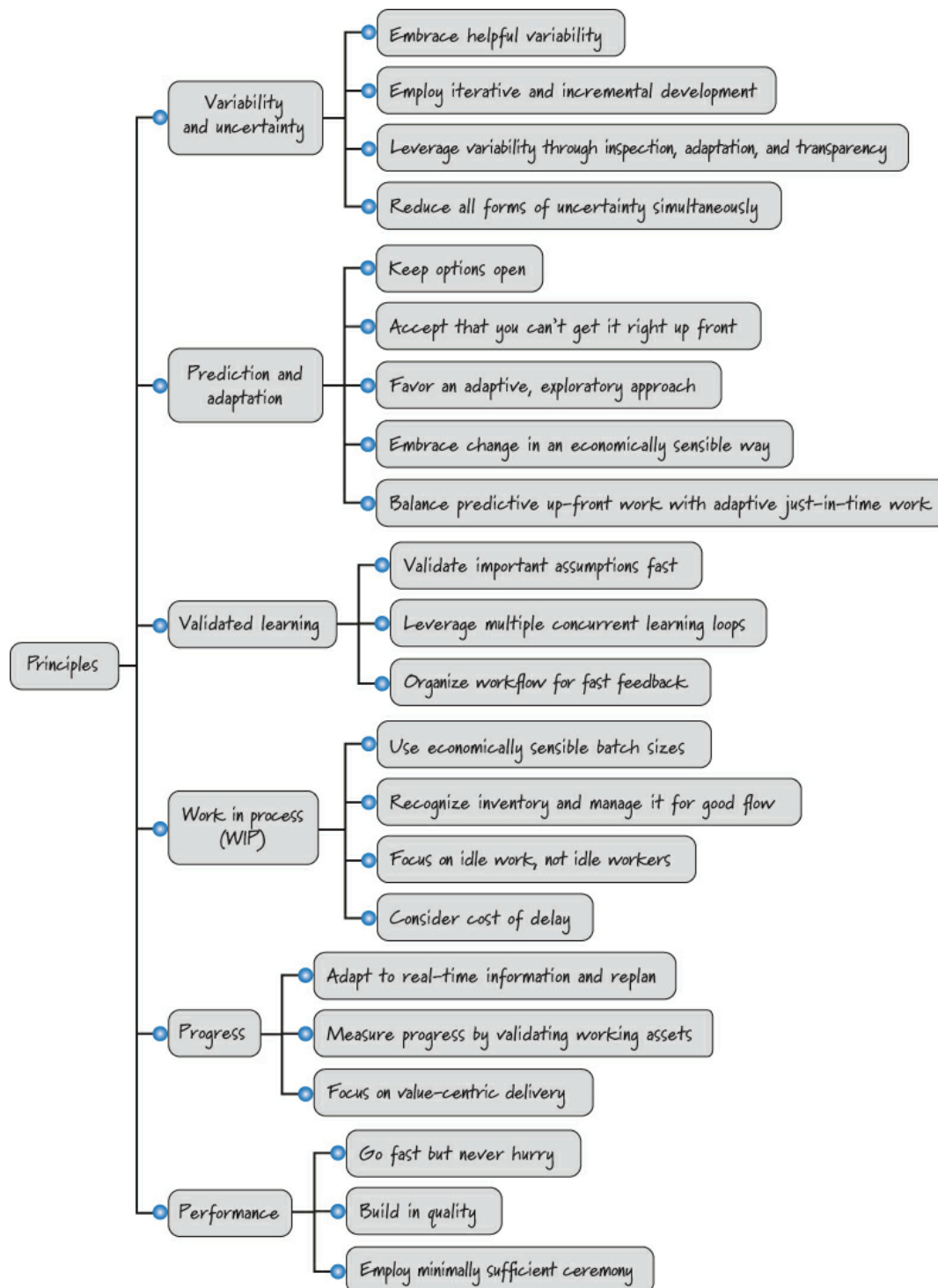


Figure: 17 Principles of Agile Software development¹⁷⁸

¹⁷⁸ Rubin, *Essential Scrum*.

3.7 Conclusion

This chapter has provided a brief outline of the history of software development and how the transition to agile software development has changed the ideas of individuals from following a defined process model to allowing for application of empirical process control.

The base understanding of agile was investigated from the concepts of the agile manifesto, which supports the various agile methodologies that are used in software development. In addition to this, three popular methodologies were researched to provide an understanding into the variety concepts and principles which influence software development.

This thesis is connected with the concept of agile software development on two levels. The first is as a whole; What concepts does agile provide that changes thinking in organisations at a mental model level¹⁷⁹ for individuals and organisations by application of agile principles and practices that stem from the agile manifesto and are entrenched in the various agile frameworks.

Secondly, the application people-first methods¹⁸⁰ that encourage learning and processes¹⁸¹ that agile has adopted and packaged in various methodologies to embrace complexity and produce results.

¹⁷⁹ Models of complexity are presented in chapter three with various models presented and theories of agile adoption presented to embrace these models.

¹⁸⁰ Examples of these methods include retrospectives, built in feedback loops and short term planning

¹⁸¹ Processes in this case refer to the concept of empirical process control, which includes iterations and estimating work size.

4

MODELLING COMPLEXITY AND LIMITS TO UNDERSTANDING

4.1 Introduction

The following chapter takes on the aspect of modelling complex systems.

Initially the concept of mental models and understanding of the system is presented. This encapsulates the work of Cilliers, ‘Complexity and Postmodernism and Why We Cannot Know Complex Things Completely’, with the application of Weick’s sense-making theory to enforce the concepts.

Next the models of Boisot are presented with application of the I-space to complexity and the models of dealing with complexity in an organisational perspective, in which organisations defer to either the strategy of reducing complexity or absorbing complexity. This is fundamental in understanding the application of agile to responding to complexity based on how the organisation views itself.

As an extension of Boisot’s work, Snowden created the Cynefin framework for categorising and responding to complexity. This model is explored to further gain understanding on how to attack complexity on a situational level.

The understanding that human systems can be differentiated from other systems occurring in nature is now being realised. Human complex adaptive systems share the same concepts such as mathematical complexity, emergence, self-organisation, and un-order, and the system also has some unique properties such as language, communication, and knowledge¹⁸².

¹⁸² Snowden and Stanbridge, “The Landscape of Management: Creating the Context for Understanding Social Complexity.”

It is important to note that models have drawbacks, as highlighted by Cilliers¹⁸³. Complex systems by their nature have properties that are unable to be comprehended fully. This leads to the problematic nature of models. Applying a defined model to a complex system invokes the question of whether or not the system can or should be modelled on a formal or computational basis. In addition to this, the claim that complex systems have a history also adds to the problem of modelling. Where to model cannot be conceived of without taking their context into account¹⁸⁴.

Concepts such as subjectivism and objectivism of the models that are created for the system are explored. How this affects individuals and groups in how mental models of the system are built and verified.

As complexity theory has emerged as a field of study, the common perception is to label organisations as complex adaptive systems¹⁸⁵. As the field of complexity is influenced by thinkers from a variety of disciplines, this achieves new insight in organisations by application of different ideas.

4.2 Modelling Complex systems

Modelling complex systems can be separated in two distinct parts. The first is the human and social model which is constructed by individuals to apply meaning to their environment. For this, literature from Cilliers, Weick and Boisot is used to create a clearer picture on the limits of cognition, knowledge transfer and shared understanding by individuals in the system. From this understanding, the scene is set for the core principle of this text of how agile principles and practices can help an organisation through a people-first approach to best support this mental model creation or adaption to respond to ambiguity and uncertainty that is introduced in complex system.

The second is an abstract level model, which a person can use to identify the problem and understand how to react to the situation they face. This type of model will not help build a clearer picture of the system or attach new meaning in which the person can act on. Sources

¹⁸³ Paul Cilliers, "Knowledge, Complexity, and Understanding," *Emergence, a Journal of Complexity Issues in Organizations and Management* 2, no. 4 (2000): 7–13.

¹⁸⁴ Cilliers, "Knowledge, Complexity, and Understanding."

¹⁸⁵ Sally Lansdell, "Issue 2-4" (March 10, 2001): 1–167.

include Snowden and Stacey in understanding models that help with the understanding of the system and its properties from the models that are built and the behaviour of the system. These models enable the categorisation of problems in the organisation and the ability to provide direction on how to react to ensure the best results.

4.2.1 Mental modelling of complexity

Senge describes a mental model as a combination of a set of assumptions, perspectives stories and beliefs that individuals attach to, to provide context when applying cognition to a problem. These models held across a collective group define the culture of the organisation¹⁸⁶. Cilliers reinforces this thinking with the statement "The argument is just that, as far as complex systems are concerned, our knowledge will always be contextually and historically framed¹⁸⁷".

Caution should be taken with regard to the claims made about the "knowledge" we gain from many of these models. The models are often as complex as that being modelled, and thus do not always lead to deeper understanding of the systems at stake. In order to gain "knowledge" from complex models, they have to be interpreted, and these interpretations will always involve a reduction in complexity¹⁸⁸. This thinking comes from the non-linear nature of complex systems. Cilliers¹⁸⁹ puts forth the argument that most would agree with that complexity is incompressible.

This does not mean that constructing models that represent the system on a level that is deemed to be reasonable is negative. The important note is that creating a more accurate representation of the system cannot be simpler than the system by default¹⁹⁰. To create the representation, omissions have to be made from the model because interactions are unseen and

¹⁸⁶ P Senge, "The Fifth Discipline," *The Art & Practice of Learning Organization* ... (1990).

¹⁸⁷ Paul Cilliers, "Why We Cannot Know Complex Things Completely," *F Capra* 4, no. 1 (2007): 77–84.

¹⁸⁸ Cilliers, "Why We Cannot Know Complex Things Completely."

¹⁸⁹ Paul Cilliers, "Knowledge, Limits and Boundaries," *Futures* 37, no. 7 (September 2005): 605–613, doi:10.1016/j.futures.2004.11.001.

¹⁹⁰ Cilliers, "Knowledge, Limits and Boundaries."

non-linear, and the impact of leaving out what as inconsequential can have unforeseen consequences.

Complex systems also have a level of codetermination that the system and the knowledge are tightly coupled and in transformation. The system cannot exist without the knowledge and vice versa¹⁹¹.

Cilliers argues that complexity is an epistemological matter. He states that humans are unable to understand the millions of non-linear interactions that are needed to describe complexity, and that it is just a lack of knowledge¹⁹², rather than an ontological understanding of the way the world is.

The emphasis is that the controller has all knowledge of the system. With complex systems, the ability to construct a model is beyond the capacity of understanding. This leads to the problem of information created by a complex system, where it is too much and varied to understand or the information is no longer relevant because the system is displaying other properties for a model to be built¹⁹³.

However, there are no claims that there is something wrong with modelling complex systems. Computational and mathematical models of different kinds are doing wonderful things, and new avenues should be pursued all the time.

4.2.2 Organisational models

The I-space is a knowledge management model created by Boisot which highlights the learning patterns. This is more a knowledge management model, with the steps stated below in the Social learning cycle. The importance of this model is the association of organisation type with information handling, and this is how Boisot differentiates organisations and the ability to deal with complexity.

¹⁹¹ Cilliers, "Knowledge, Complexity, and Understanding," 9.

¹⁹² Cilliers, "Why We Cannot Know Complex Things Completely," 78.

¹⁹³ Jürgen Appelo, *Management 3.0*, (Addison-Wesley Professional, 2010).

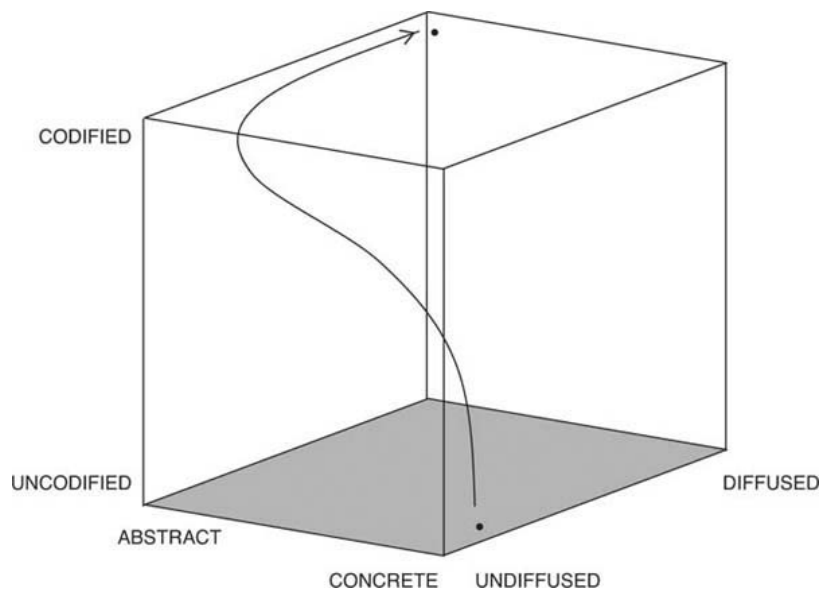


Figure: 18 Boisot's I-space Model

The Social learning cycle happens around the I-space cube. The following six steps describe the SLC.

Activity	Description
Scanning	A leftward movement in the I-Space that converts data available to others – threats, opportunities, etc. into novel and insightful patterns that are relatively un-diffused
Codification	A gradual articulation of initially fuzzy patterns into clear, compact, and robust representations that can be manipulated and stored. This is an exercise in data compression that moves it up the I-Space
Abstraction	Extracting from patterns their invariant and generalizable features. This results in further data compression and moves data from the front of the I-Space to the back.
Diffusion	A rightward shift of data compressed or otherwise across a population of agents located along the diffusion dimension of the I-Space

Absorption	The internalisation and contextualisation of data through acts of assimilation to existing schema. This entails a downward movement in the I-Space, one in which data gets interpreted
Impacting	A movement from the back of the I-Space towards the front in which data at varying levels of codification and abstraction are applied to concrete situation

*Table: 6 Six Steps of Social Learning*¹⁹⁴

In addition to the I-space, Boisot identifies four organisational types which operate in different areas. These organisations could be more or less adapted to dealing with complexity via agile type thinking than others, due to the ability to process information, structure relationships, and levels of command and control.

Fiefs. These thrive in situations with concrete, un-diffused, un-codified information. Characteristics:

- Information diffusion limited by lack of codification to face-to-face relationship.
- Relationships personal and hierarchical
- Submission to superordinate goals
- Hierarchical coordination
- Necessity to share values and beliefs

Bureaucracies. These thrive in situations with abstract, un-diffused, codified information. Characteristics:

- Information diffusion limited and under central control
- Relationships impersonal and hierarchical
- Submission to superordinate goals
- Hierarchical coordination

¹⁹⁴ Boisot, MacMillan, and Han, "Explorations in Information Space: Knowledge, Agents, and Organization - Max H. Boisot, Ian C. MacMillan, Kyeong Seok Han - Google Books."

- No necessity to share values and beliefs

Markets. These thrive in situations with abstract, diffused, codified information.

Characteristics:

- Information widely diffused, no control
- Relationships impersonal and competitive
- No superordinate goals—each one for himself
- Horizontal coordination through self-regulation
- No necessity to share values and beliefs

Clans. These thrive in situations with concrete, diffused, un-codified information.

Characteristics:

- Information is diffused but still limited by lack of codification to face-to-face relationship.
- Relationships personal but non-hierarchical
- Goals are shared through a process of negotiation
- Horizontal coordination through negotiation
- Necessity to share values and beliefs

4.2.2.1 Organisational Strategies for Complexity

Boisot¹⁹⁵ cites (Holland 1995) in the statement “The observation is that the strategies of complexity reduction and complexity absorption represent distinct cultural strategies adopted by adaptive systems.”

Boisot states that the process of learning in an organisation is located between chaos and order¹⁹⁶. This also proves problematic, as complexity can stand in the way of knowing. Two approaches to maintaining the state in which learning happens Boisot defines are complexity

¹⁹⁵ Max Boisot and John Child, “Organizations as Adaptive Systems in Complex Environments: the Case of China,” *Organization Science* 10, no. 3 (1999): 237–252.

¹⁹⁶ Max Boisot, *Knowledge Assets*, (Oxford University Press on Demand, 1999).

reduction and complexity absorption¹⁹⁷. Reducing complexity is associated with activity deploying cognitive strategies that give it structure. Absorbing complexity is achieved by deploying appropriate behavioural strategies, often in coordination with others¹⁹⁸.

The emergence that is associated with chaos and complexity is driving organisation's ability the building variety with the capacity for the response to the environment. This chaos generates anxiety in humans who have the need for order and stability in organisations due to the associated rules and routines and major sources of stability they inherently provide¹⁹⁹. The successful organisations will be seen in those that are able to change the rules to be transient and open to reinterpretation and negotiation.

A drive for organisations to interact on a customer level will force information to the right of the I-space, to a zone where it is less codified and less abstracted. Boisot highlights that this increases complexity, and the associated management activity will also change. Command and control approaches are replaced by principles of self-organisation or agents bounded by regulations or boundary conditions.

Organisations need to be capable of managing complexity, by reducing it by building effective cognition strategies favoured by markets and bureaucracies which requires a move up the I-space. Or the alternative is the approach of absorbing complexity through the process of implementing relational strategies as preferred by fiefs or clans which is typically a downward movement in the I-space.

4.2.2.1.1 Absorbing Complexity

The strategy of building organisational capacity based on the idea absorption as a response to complexity is fundamentally different from that of a strategy of reducing complexity. Absorption requires the investment in un-codified knowledge and the fostering of exploratory

¹⁹⁷ Allen, Maguire, and McKelvey, *The SAGE Handbook of Complexity and Management*.

¹⁹⁸ Boisot and Child, "Organizations as Adaptive Systems in Complex Environments: the Case of China."

¹⁹⁹ Boisot, MacMillan, and Han, "Explorations in Information Space: Knowledge, Agents, and Organization - Max H. Boisot, Ian C. MacMillan, Kyeong Seok Han - Google Books."

and exploitative learning²⁰⁰. While not tangible from the start, the observation is that this benefits people in the organisation, enhancing effectiveness in order to make sense of new experiences. This can be connected to the theories by Cilliers discussed in this chapter. Where individuals hold models of the system, support their subjective interpretations of complexity. Enabling the holding of multiple identities enables individuals and organisations to have conflicting representations of variety in the environment²⁰¹. Conversely this provides “behavioural plasticity”²⁰², where responses are less of good fit but generalisation provides the variety needed to deal adequately with many problems encountered.

A high level understanding of absorbing complexity through Boisot’s eyes would be that complexity can be absorbed through the creation of options and risk-hedging strategies though co-operating closely with other entities in the environment.

4.2.2.1.2 Reducing complexity

Traditional thinking in organisations is to implement of reductionistic-based strategies for dealing with complexity and moving to points of stability²⁰³. This method involved the focus on the articulation of knowledge by codifying and abstracting to move I-space. This method reduces the variability of phenomena being processed and the number of categories in which the phenomena can be categorised into²⁰⁴. This also leads a specialisation of individuals and organisations with the capability to only respond to problems that fit the patterns in which have been defined²⁰⁵.

²⁰⁰ Boisot, *Knowledge Assets*, 264.

²⁰¹ Boisot and Child, “Organizations as Adaptive Systems in Complex Environments: the Case of China.”

²⁰² Boisot and Child, “Organizations as Adaptive Systems in Complex Environments: the Case of China.”

²⁰³ Boisot, *Knowledge Assets*, 264.

²⁰⁴ Boisot, *Knowledge Assets*, 265.

²⁰⁵ Boisot and Child, “Organizations as Adaptive Systems in Complex Environments: the Case of China.”

Organisations that have an aversion for complexity will naturally follow the route of reductionist strategies in dealing with complexity. Boisot then goes on to elaborate that systems that are attracted to order are associated with bureaucratic and market based institutions, they focus on stability, repeatability and the ability to control the diffusion of information to a select population²⁰⁶.

4.3 Models for Complex problems

4.3.1 Cynefin Framework

The word complex is often misused. Due to a lack of understanding of what complexity actually is, situations are often referred to as being very "complex" where there is little evidence that this is in fact the case. It is very easy to confuse the complex with the complicated and the simple, as highlighted in Chapter 2.

More importantly, when dealing with situations in organisations, it is very important to prepare the correct response to the domain encountered. Mistakes and unintended consequences can be made if complexity is dealt with by relying on the tools and a mindset geared to solving complicated problems²⁰⁷.

The Cynefin Framework, developed by Dave Snowden, is a visual representation of states of a system and the effects of each state. Cynefin's strength is that it enables individuals to create a shared understanding of the context of the system state they are dealing with, and identify the appropriate actions to approach the decision making process for the problem at hand, thus enabling the correct response to a variety of situations. Cynefin will not give the person using the framework the answers to the problem; rather it is used as an interpretive model which is used as a sense-making tool.

²⁰⁶ Boisot, MacMillan, and Han, "Explorations in Information Space: Knowledge, Agents, and Organization - Max H. Boisot, Ian C. MacMillan, Kyeong Seok Han - Google Books."

²⁰⁷ Gökçe Sargut and Rita Gunther McGrath, "Learning to Live with Complexity.," *Harvard Business Review* 89, no. 9 (2011): 68.

4.3.1.1 Domains in Cynefin

Cynefin describes 5 domains in which a problem could exist. The domains are connected to current thinking of properties of complexity:



Figure: 19 Cynefin model²⁰⁸

Cynefin, although useful in situations from software development to organisational problems where complexity is evident, may not provide one logical answer or direction. Instead, problems may span multiple domains or oscillate between many domains. While most problems will fall into complex or complicated, not everything is complex. To ignore the other domains of simple and chaotic would be short sighted.

²⁰⁸ Rubin, *Essential Scrum*.

4.3.1.2 Complex domain

In the complex domain, cause and effect are not correlated; therefore the level of predictability is very low. Solutions to problems in the Complex domain are only apparent in retrospect because of the emergent nature of the domain. Working in this domain requires creativity and innovation backed by an inspect and adapt mindset by implementing learning based approaches. Repeatable solutions that have worked in the past are no longer valid and solutions need to be designed not to avoid failure at all costs but to be safe to fail with built in recovery. This solutioning is supported by collaboration and communication between individuals.

4.3.1.3 Simple Domain

In the Simple domain, cause and effect are directly connected and predictable, Snowden²⁰⁹ states that best practices apply, with existing mental models playing important parts of problem solving. In dealing with simple, the quest for efficiency and repeatability is key, normally by using a linear process with repeatable steps.

4.3.1.4 Complicated Domain

In the Complicated domain, the relationship between cause and effect can be identified by analysis or expert knowledge, and good practices need to be applied to the domain. There are several different approaches to solving problems, but an expert is required to provide these solutions. This domain benefits from the concept of flow for systems performance.

4.3.1.5 Chaotic Domain

Chaos provides problems that require a novel practice to be applied to the situation to find a solution. Action should be swift, as this is a state of crisis, and some form of order is the first goal in problem solving. Normally, a leader will take charge and act.

²⁰⁹ David J Snowden, “Multi-Ontology Sense Making: a New Simplicity in Decision Making” (2005).

4.3.1.6 Disorder Domain

In the domain of Disorder, the individual cannot identify which domain they are in. Very little sense-making of the situation at hand is possible, with individuals acting on personal preference for action. This could lead to falling into a linear based problem solving pattern as described in Chapter 5.

4.3.1.7 Decision making in Cynefin

As an introduction to decision making process, associated with the domains identified previously, the following table, taken from *A leader's framework for decision making*²¹⁰, highlights the steps in Cynefin that are applicable to each domain.

	THE CONTEXT'S CHARACTERISTICS	THE LEADER'S JOB	DANGER SIGNALS	RESPONSE TO DANGER SIGNALS
Simple	<p>Repeating patterns and consistent events</p> <p>Clear cause-and-effect relationships evident to every- one; right answer exists</p> <p>Known knows Fact-based management</p>	<p>Sense, categorize, respond</p> <p>Ensure that proper processes are in place</p> <p>Delegate</p> <p>Use best practices</p> <p>Communicate in clear, direct ways</p> <p>Understand that extensive interactive communication may not be necessary</p>	<p>Complacency and comfort</p> <p>Desire to make complex problems simple</p> <p>Entrained thinking</p> <p>No challenge of received wisdom</p> <p>Over-reliance on best practice if context shifts</p>	<p>Create communication channels to challenge orthodoxy</p> <p>Stay connected without micromanaging</p> <p>Don't assume things are simple</p> <p>Recognise both the value and the limitations of best practice</p>

²¹⁰ David J Snowden and Mary E Boone, "A Leader's Framework for Decision Making," *Harvard Business Review* 85, no. 11 (2007): 68.

Complicated	<p>Expert diagnosis required</p> <p>Cause-and-effect relationships discoverable but not immediately apparent to everyone; more than one right answer possible</p> <p>Known unknowns Fact-based management</p>	<p>Sense, analyse, respond</p> <p>Create panels of experts</p> <p>Listen to conflicting advice</p>	<p>Experts overconfident in their own solutions or in the efficacy of past solutions</p> <p>Analysis paralysis Expert panels</p> <p>Viewpoints of non-experts excluded</p>	<p>Encourage external and internal stakeholders to challenge expert opinions to combat entrained thinking</p> <p>Use experiments and games to force people to think outside the familiar</p>
Complex	<p>Flux and unpredictability</p> <p>No right answers; emergent instructive patterns</p> <p>Unknown unknowns Many competing ideas</p> <p>A need for creative and innovative approaches</p> <p>Pattern-based leadership</p>	<p>Probe, sense, respond</p> <p>Create environments and experiments that allow patterns to emerge</p> <p>Increase levels of interaction and communication</p> <p>Use methods that can help generate ideas: Open up discussion (as through large group methods); set barriers; stimulate attractors; encourage dissent and diversity; and manage starting conditions and monitor for emergence</p>	<p>Temptation to fall back into habitual, command-and-control mode</p> <p>Temptation to look for facts rather than allowing patterns to emerge</p> <p>Desire for accelerated resolution of problems or exploitation of opportunities</p>	<p>Be patient and allow time for reflection</p> <p>Use approaches that encourage interaction so patterns can emerge</p>

Chaos	<p>High turbulence</p> <p>No clear cause-and-effect relationships, so no point in looking for right answers</p> <p>Unknowables</p> <p>Many decisions to make and no time to think</p> <p>High tension Pattern-based leadership</p>	<p>Act, sense, respond</p> <p>Look for what works instead of seeking right answers</p> <p>Take immediate action to re-establish order (command and control)</p> <p>Provide clear, direct communication</p>	<p>Applying a command-and-control approach longer than needed</p> <p>“Cult of the leader”</p> <p>Missed opportunity for innovation</p> <p>Chaos unabated</p>	<p>Set up mechanisms (such as parallel teams) to take advantage of opportunities afforded by a chaotic environment</p> <p>Encourage advisers to challenge your point of view once the crisis has abated</p> <p>Work to shift the context from chaotic to complex</p>
Disorder				

Table: 7 Decisions in multiple contexts: A leaders guide to decision making²¹¹

4.4 Building an agile organisation

Agile software development has shown that process plays an important part of the solution to dealing with complexity. Process alone, however, will not be enough to the organisation faced with complexity. Agile software development has always been a people-first approach to problems. This is no more evident than in the change process when moving towards organisational agility.

Agile software development is seen to do away with the hierarchy associated with organisations. While the structure does flatten, there is still a role for a leader in the organisation. This is especially true for when an organisation is undertaking a transition to agile. Important parts of the transition still need leaders to drive implementation. This is needed to instil the correct vision and values based on the culture of the organisation in teams and individuals. If done correctly, this will enable innovative teams to have the correct

²¹¹ Snowden and Boone, “A Leader's Framework for Decision Making.”

models with the ability to adapt to complexity and its associated uncertainty and ambiguity²¹².

Transforming an organisation towards agile will have an effect on all forms of management. Initially, as work and process is redesigned, the need arises to redesign the responsibilities of managers and the decision making process. In addition to this, new roles in the organisation would be needed to support the shift to agile. Agile frameworks will cause a shock to mental models associated with traditional management and its associated functions, such as project management. Managers in agile environments will see the traditional principles of management change. The new role of the managers will be that of goal setting, ensuring vision is formulated and understood through a culture of support for individuals, grounded in learning-based collaboration and the upholding organisational values²¹³. This shift from command and control forms of management to motivating and supportive leadership is characterised by trust. People need to thrive on change and uncertainty, rather than simply coping with them.

Building a culture of acceptance to change is important when dealing with complex systems. Individuals in the organisations need to become comfortable with the concept of constant change and uncertainty, and change their mental models of resistance and coping to acceptance and growth. Boisot²¹⁴ differentiates organisations and the way they deal with complexity. The first organisation he identifies is the human productive-based organisation, which is categorised the culture of capital labour trade-offs. These organisations are geared towards complexity reduction. The organisation is contained within excessive order, shying away from chaos or complex behaviour. Conversely, an organisation which is comfortable with absorbing complexity is more culturally adept at dealing with complexity which has yet

²¹² Nirmal Pal and Daniel C Pantaleo, "Introduction the Agile Enterprise," (New York: Springer-Verlag, 2005), 1–10, doi:10.1007/0-387-25078-6_1.

²¹³ R Stacey, "Responding to Complexity and Uncertainty: the Agile Organisation | Complexity & Management Centre," *Complexityandmanagement.Wordpress.com*, accessed April 7, 2013, <http://complexityandmanagement.wordpress.com/2012/08/29/responding-to-complexity-and-uncertainty-the-agile-organisation/>.

²¹⁴ Boisot, *Knowledge Assets*, 39.

to been identified. This is also echoed by Pal (2005)²¹⁵ in where vision and values drive teams which are adaptive and innovative in a corresponding infrastructure, influenced by leaders.

4.4.1 Decision making

Decision making in complex systems is fraught with problems. For a single person to make a decision, they would need a complete view of the system. Due to the nature of the system and its relationships of agents and interactions, this is impossible from one single static view of the system²¹⁶. In addition to this, individual ability to recognise factors of cause and correlation of other agents in the system is limited.

Decision making in the described context is seen as an extension of Herbert Simon's work on bounded rationality. Reality dictates that agent behaviour is not rational. Agents are governed by what is interpreted by the agent as sufficient information to make a decision. This, as Simon describes, leads to behaviour that is "satisficing" or best for the agent at the time, and not maximising²¹⁷.

The natural instinct of individuals when making decisions is to revert back to models based on past experiences. Snowden describes this as first fit pattern matching²¹⁸. This is because, when observing the world, individuals extract cues that are salient to the situation²¹⁹ and fill the gaps with past experience. However, the results could be surprisingly far from expected, especially when dealing with a new context.

Decision making in organisations that have adopted agile methodologies is naturally deferred to the lowest possible level. Teams are empowered to control the direction and design of the task they are working on. Decisions are made in the context of the individual and team. This

²¹⁵ Pal and Pantaleo, "Introduction the Agile Enterprise."

²¹⁶ Sargut and McGrath, "Learning to Live with Complexity.."

²¹⁷ H A Simon, "Models of Bounded Rationality: Empirically Grounded Economic Reason - Herbert Alexander Simon - Google Books" (1982).

²¹⁸ Snowden and Stanbridge, "The Landscape of Management: Creating the Context for Understanding Social Complexity," 145.

²¹⁹ Weick, *Sensemaking in Organizations*.

is important, as it provides the ability of the correct decisions to be made based on the correct information. Delaying decision making to the last possible moment, results in greater value in building and maintaining options in the event of change. The nature of this relates back to the premise that in a complex system, it is impossible to know all agents and interactions.

Leaders need to understand that shifting contexts are encountered in the organisation. Managers have to build skills that bridge contexts from simple to chaotic. Classical leadership traits such as charisma, intuition and intellect are no longer the only attributes that make a leader successful when dealing with complexity.

4.4.2 Transitioning to an Agile Organisation

Stacey²²⁰ identifies two ways in which agile transformation is achieved. He proposes two constraining views on the transition process with top down approaches at one end, and through distributed approaches to reliance on the informal organisation at the other end.

4.4.2.1 Top down prescriptions

The transformation to the agile organisation must come from the top leaders who should:

- Prepare new vision statements the right vision statement.
- Redesign organisational structures so that hierarchical states are replaced with agile states.
- Develop breakthrough business models which lead to actions that disrupt industry ecosystems.

Create a break-through culture in which people are open-minded about change, focus the customer, collaborate within and across organisational boundaries, and focus intensively on goals and execution.

1. Design new metrics to incentivise desired behaviour.
2. Form teams which are adaptive and innovative.

²²⁰ Stacey, “Responding to Complexity and Uncertainty: the Agile Organisation | Complexity & Management Centre.”

3. Cross functional teams that self organise around problems

4.4.2.2 Distributed action

Another approach calls for organising to master change and uncertainty and leveraging the impact of people and information, so repeatedly reinventing the organisation. The prescriptions are:

1. Formulate a clear vision, set bold goals, avoid micro-management, and work to win universal buy-in.
2. Give up control and shift from command and control to motivating and supportive leadership characterised by trust. Design management hierarchies that are flat
3. Distribute decision-making authority to operational employees and design internal information flows that are open to all rather than being confined to privileged managers.
4. Do not deliberately choose organisational structures but allow the flexibility that enables rapid reconfiguration of cross functional teams and resources required to meet customer requirements
5. Create a culture that supports people, values thinking, learning and cooperation to solve problems People need to thrive on change and uncertainty rather than simply coping with them and they need to be aggressively change-embracing.
6. Optimise opportunism and maximise the positive impact of local decision making.
7. Use teams as the standard form of organisation. Develop people with strong social and communication skills that enable them to function in intensely cooperative and team-based activities.
8. Relying on simple rules and the informal organisation

One view stresses simple rules where managers should:

Identify whether they are at the edge of chaos or trapped in either stability or chaos.

Move to the edge of chaos by avoiding: too much structure which leads to stability and too little which produces chaos.

Work out what to structure and what not to structure. Keep activity loosely structured but at the same time rely on targets and deadlines.

Develop a culture which fosters frequent change in the context of a few strict behavioural rules.

Create channels for real-time, fact-based communication within and across groups.

Allow a semi-coherent strategy to emerge, one that is not too fixed nor one that is too fluid.

4.5 Leadership in Complex Adaptive systems

Denning²²¹ puts forward the case for creating new mental models for leadership. He argues that mental models need to cater for discipline and freedom. This is to ensure there is space for growth and creativity.

Agile provides a framework for leaders to manage complexity in their environments. While the aspect of direct control that neo-classical management models are based on is removed, aspects of agile processes and principles allow for the management of flow and patterns. Managers can no longer be concerned about the detail, as this is counter-productive to the understanding the system. Rather, the notion of being in change is shifted to being connected, and leadership is based on co-creation and collaboration as an enabler in solving problems.

The significant advantage is that Agile processes allows for the managing of constraints in a complex systems. These constraints are what allow for the manifestation of self-organisation. Cilliers²²² highlights that boundaries are still required if we want to talk about complex systems in a meaningful way—they are in fact necessary to bring some type of structure to the system. However, the constraints must be capable of being amended, either by reducing them or increasing to ensure they are providing positive influences on agents in the system.

4.6 Leadership from a military perceptive

War and terrorism are seen as the most disturbing complex adaptive system that humans can relate to. This is no more evident than in the current age of the war on terror from an American standpoint.

²²¹ Denning, *The Leader's Guide to Radical Management*, 2010, <http://>.

²²² Cilliers, "Knowledge, Limits and Boundaries."

In order to deal with the threats that are encountered in this new age, a different view on leadership has been developed. Until recently, battlefield command through the ages has been associated with the concept of detailed command of the environment. Commanders are able to achieve this by focusing on identifying accurate information about troops in both enemy and friendly forces. The size and complexity of the battles commanded, normally lasting less than a day, allowed for this²²³.

The theory and application of Leadership has changed in the United States army. The understanding is that war is unpredictable, and steeped in disorder and uncertainty. The application of mission command as a way of managing has changed the leadership styles of the army to ensure they are able to adapt to the environment that is faced.

Mission command is described in the²²⁴ Mission command handbook as the ability to conduct military operations by decentralised agents based on orders for effective mission accomplishment. This is enabled by individuals exercising disciplined initiative within the commander's intent, based on an environment of trust and mutual understanding.

Through the use of this management approach, they have the ability to deal with the problems they face in varying contexts by making acceptable decisions faster. Enabling this is the implementation of concepts such as decentralisation, certainty to act through freedom of action, and initiative.

4.6.1 Military solutions to uncertainty

The mission control manual identifies uncertainty as a result of two problems. The first problem is information based and the second is action focused.

By applying solutions to reduced information based uncertainty, information processing capabilities at the top of the organisation (The author has inter-changed the term force for organisation as this text is not related to wartime scenarios) are increased by collecting more and better data²²⁵. However, uncertainty is still evident in lower structures of the organisation, which are closer to daily operations, due to information delay.

²²³ *Mission Command*, 2003, 14.

²²⁴ *Mission Command*.

²²⁵ *Mission Command*, 11.

Action based approaches to uncertainty are inclusive among the organisation. Authority is delegated to lower levels for decision making to deal with the uncertainty encountered with daily operations. This approach enables agility for the organisation. The action based approach requires more decisions to be made with less information. This short term decision making is at conflict with the information based approach wherein individuals at the top of the organisation delay decision making because of the need for better or more information.


		
<ul style="list-style-type: none"> • Probabilistic • Unpredictable 	Assumes war is	<ul style="list-style-type: none"> • Deterministic • Predictable
<ul style="list-style-type: none"> • Disorder • Uncertainty 	Accepts	<ul style="list-style-type: none"> • Order • Certainty
<ul style="list-style-type: none"> • Decentralization • Spontaneity • Informality • Loose rein • Self-discipline • Initiative • Cooperation • Acceptable decisions faster • Ability all echelons • Higher tempo 	Tends to lead to	<ul style="list-style-type: none"> • Centralization • Coercion • Formality • Tight rein • Imposed discipline • Obedience • Compliance • Optimal decisions, but later • Ability focused at the top
<ul style="list-style-type: none"> • Implicit • Vertical and horizontal • Interactive 	Communication types used	<ul style="list-style-type: none"> • Explicit • Vertical • Linear
<ul style="list-style-type: none"> • Organic • Ad hoc 	Organization types fostered	<ul style="list-style-type: none"> • Hierarchic • Bureaucratic
<ul style="list-style-type: none"> • Delegating • Transformational 	Leadership styles encouraged	<ul style="list-style-type: none"> • Directing • Transactional
<ul style="list-style-type: none"> • Art of war • Conduct of operations 	Appropriate to	<ul style="list-style-type: none"> • Science of war • Technical/procedural tasks

Figure 1-4. Concepts of Command and Control

Figure 20 Concepts of Mission Command

4.7 Conclusion

This chapter began with the concept of modelling, with the application of knowledge and information to personal models of complex systems, to the models of organisations that deal with complexity with different strategies. Boisot's statement resonates, a world in which richness and reachability of knowledge, if not properly managed, will make us informationally obese, slowing us down at the very moment that the complexity we encounter requires us to be lean and agile.

The overriding assumption by most authors in the field in modelling complexity is that models are beneficial for providing insight to individuals and social constructs, however

these same models have limitations in use. These limitations are bought on by the natural properties of complex systems.

Complex systems are interactive, constantly changing processes in which emergent behaviour creates surprises and adaptability. The environment that complexity is associated with needs a different application of thinking to provide results. Be it from armed conflict in terrorism to conflict in organisations or the creation of software. Decision making is changing based on the structure and amount of information received in the system. Conversely, the models of leadership have to also adapt, becoming enablers for individuals to respond to complexity daily.

5

AGILE PRACTICES APPLIED TO COMPLEX SYSTEMS

5.1 Introduction

The aim of this chapter is to understand how the application of the concepts, frameworks, and principles of agile software development to organisations affects the facets of complexity identified in the previous chapters, also describing how it provides the organisation with a framework and understanding across social structures to deal with complexity on a system level. In addition to this, it also provides individuals who are part of the system with the necessary toolset to effectively deal with the complexity they experience at an agent level. Examples of this could be from the communication patterns to the problem solving processes.

The correlation between complexity and software development has been explored before by the likes of Highsmith and Schwaber, who agree that a software development team displays definite properties of a complex adaptive system. In addition to this, Jim Highsmith has referred to the principles of complex adaptive systems and complexity theory to explain, by analogy, the reasons why agile processes work²²⁶.

In this chapter, the agile principles and processes are compared to properties of complexity, and the manner in which agile processes are able to leverage or negate these properties for outcomes when dealing with complexity is identified.

Initially, the concept of iterating through work is discussed with the principles behind iterating when dealing with complex problems

Generalisation over specialisation is investigated, and concepts of cross functional teams are tabled.

²²⁶ Poppendieck and Poppendieck, *Lean Software Development*.

Cross functional abilities of the team, including the role of diversity of individuals compared to traditional approaches of specialisation, are compared. This leads to the topic of self-organisation, where teams have the autonomy to do work without outside control.

The idea of retrospective coherence is discussed and with it the provision of agile to provide regular cadences for retrospectives to ensure that there is a learning pattern instilled in the system, and the adapting or validation of shared and individual mental models.

The communication of problems between parties in the organisation is applied to the concept of a user story from agile software development. This helps describe the minimal amount of codification of knowledge as being effective in dealing with complexity and opening communication channels to achieve coherence in the system between agents.

5.2 Concepts of agile and complexity

Agile software development, when applied to complexity, changes management practices to enabled outcomes. The application of soft thinking and empirical process-based control to allow disciplined execution within a space fosters creativity and innovation. This includes the embracing of transparency, iterative design principles, and putting people first.

Stacey²²⁷ conceptualised strategy as 'order emerging from chaos' and counselled managers to abandon stability and harmony as objectives; rather, successful organisations welcomed uncertainty, actively promoted instability of a sort and channelled resulting tensions and conflicts in beneficial ways²²⁸. The argument is that agile software development is the mechanism for achieving what Stacey stated, as agile software development comes from an industry typically not attributed with management ideas. The failures experienced in the Information technology industry by applying traditional management theory to developing complex software products have lead to a different solution to be applied to management.

In addition to this, in seeing disorder, uncertainty and crisis in a positive light, this work questioned not only the merits but also the feasibility of centralised management of organisational processes; top-down control was proving to be, in the long term, dysfunctional

²²⁷ Ralph Stacey, "Strategy as Order Emerging From Chaos," *Long Range Planning* 26, no. 1 (February 1993): 10–17.

²²⁸ Allen, Maguire, and McKelvey, *The SAGE Handbook of Complexity and Management*.

to organisations²²⁹. Furthermore, the logic of complexity suggests that learning and the generation and sharing of knowledge need to be facilitated by providing the appropriate socio-cultural and technical conditions to support connectivity and interdependence, and to facilitate emergence and self-organisation²³⁰.

Snowden²³¹ refers to this as social network stimulation, increasing the levels of interaction between technical staff and users. The premise is that the users have needs which are unknown or are not able to be articulated, whilst the technical staff have the capabilities for the problem solving. The argument of why this is appropriate to complexity is that it has three heuristics:

1. Finely grained problems are attacked by small teams providing parallelism in problem solving
2. Self-organisation is bounded within constraints, a cross functional team ensures that the diversity of cognitive capabilities increases models attributed to problem solving.
3. The results of the effort are visible to all in the organisation, without the need of reinterpretation or explanation of meaning, this is especially applicable to senior decision makers

5.2.1 Core Concepts of Agile

Notable among the approaches have been the principles of Scrum which include:

1. Work is organised in short iterative cycles;
2. The management does not interrupt the team during a work cycle;
3. The team reports to the client, not the manager;
4. The team estimates how much time work will take;
5. The team decides how much work it can do in an iteration;

²²⁹ Allen, Maguire, and McKelvey, *The SAGE Handbook of Complexity and Management*.

²³⁰ Mitleton-Kelly, "Organisations as Complex Evolving Systems."

²³¹ D Snowden, ed., ... *Was So Skinny and Thin*, accessed October 12, 2013, <http://cognitive-edge.com/>.

6. The team decides how to do the work in the iteration;
7. The team measures its own performance;
8. Work goals are defined before each cycle starts;
9. Work goals are defined through user stories;
10. Impediments are systematically removed.

5.2.1.1 Work is done in iterations

The application of iterations to work reflects a fundamental difference in thinking than the application of a linear traditional process.²³² The main goal of iterative process is twofold. Firstly, the attempt to deal with unpredictability by creating short term constraints, in this case, a time boxed interval and amount of work able to be completed. Secondly, frequent feedback loops are present in the iteration to allow for leaning from information created by the iteration.

Dealing with complex problems does not correlate to time-based tasks, as it is dynamic and unpredictable. While a complex problem may take a minute to come to a solution, another problem that initially looks very similar may take a year to solve. The ability to set the system a constraint around a problem is important. Constraints do not have a negative effect on the system. Rather, by reducing or removing possibilities of outcomes, constraints can be found to have a positive effect on the system²³³.

The ability to create a plan for work over the long term to meet future desired states is extremely difficult in all systems, and when dealing with complex systems, due to their nature, impossible. In the case of software development, iterations provide important boundaries for the team. The setting of short term goals and limiting of work for an iteration enables a team to practice disciplined delivery, while not compromising innovative behaviour and accelerated learning. Iterations are based on previous empirical work evidence to decide how much work can be done.

²³² Such as waterfall in software development

²³³ Cilliers, "Boundaries, Hierarchies and Networks in Complex Systems."

The concept of working in iterations is central to agile software development, providing a space for work and tasks of varying size. Stability is found in short term plans being executed in the current iteration. As work shifts from linear based production style to complexity based, this proves to be problematic for push²³⁴ based planning approaches.

Working software containing new information is seen as the output of each iteration in agile software development. This makes for a better understanding of the problem domain. While the product may not have all the functionality that is needed for the system to be fully functional. Rather, it reflects work done in the current and any previous iterations and progress in the problem space.

It is widely accepted that in software development iterations may fail. When this does occur, the iteration will not produce a working example of software that can be used for progress updates. In this case, the knowledge generated from the accelerated learning from failure in the iteration should be used to plan the next. In complex systems, the feedback generated from each iteration on the emerging behaviour of the system can be used to shape patterns for future states.

From this understanding, iterations are a risk mitigation strategy, due to the short time domains, and a failure of an iteration does not correlate to total failure for the project.

Positive outcomes are amplified allowing for rapid gains. Negative outcomes have a damping effect and iterations are serial in nature, with one iteration following on from another. To increase learning potential of iterations, breaking down work into small chunks or sizes can allow for parallelism in the iteration. This is useful for testing multiple hypotheses that have been devised for a complex problem.

Iterations should be as short as possible; the ability to gather frequent feedback is a key concept and the only way to know the true status of the project. Iterations allow for quick changes between production cycles, reducing barriers to change, with high quality

Iterations provide a reduction in planning time spent upfront. This concept is important in dealing with complexity, where the predictability of the system is impossible over the long term. Short term planning for the iteration is a more feasible approach to dealing with complex problems. With a fixed time scale, and just in time iterative planning, the feedback from previous iterations with the knowledge gathered by individuals doing the problem

²³⁴ Push based approach vs pull based

solving, sets favourable start conditions for the iteration that helps the emergence of a design that best fits the problem domain.

5.2.1.2 The capacity of iterations is dynamic

Work is not measured on process efficiency; the concern is not about getting as much done as quickly as possible, rather the focus on quality and getting the correct thing done within the iteration. Because of the nature of empirical process control, the measurement of progress is done on the increment of a product being delivered. Questions such as which problems were solved and how they are represented are as important as the physical product at the end of the process. Due to the complexity of the problems that are faced with work in the iteration, it is left up to the individuals doing the problem solving to decide how much time is necessary to apply to the problem space. Just as complexity is non-linear, solving complex problems has the same facets.

5.2.1.3 Cross functional teams

Teams in agile environments are traditionally composed of individuals from various backgrounds. With the goal of the team to build a working iteration of a software product, the team would typically consist of software developers, quality assurance, and various other individuals from diverse backgrounds, such as marketing and sales. This enables a team to complete a complex problem from start to finish. The shared understanding and learning from the team is important for the next iteration and feedback cycle.

This type of thinking is very different to a traditional approach to organisation. Typically, people are separated across organisational silos based on differences in work.

With a generalist approach to cognitive capacity over a specialised role based approach, this approach leads to the generation of variety within the team. This variety is an important aspect of complex problem solving as Ashby is famous for stating "Only Variety can destroy variety". While Boisot²³⁵ highlights a team as having cognitive and behavioural diversity, leading to a large variety of perceptions and interests in the team, this enables the team to identify and act on threats and opportunities. Stacey reinforces Boisot with the observation

²³⁵ Boisot, *Knowledge Assets*, 227.

that diversity adds to the survivability of a system in adverse environments, by increasing flexibility and enabling innovations²³⁶.

Weick²³⁷ states "that teams are composed of at least some individuals with different expertise are better to grasp variations in their environment and see specific changes that need to be made." He goes on to highlight the benefit of generalists in teams as being better at recombining knowledge, skills and abilities into combinations for solutions to the problem faced. This is due to a broad range of experiences that have been encountered.

Having individuals that are not expert in one role allows the ability to transition between mental models of understanding, and it has been found that specialists are locked into a mental model which is harder to discard and rebuild based on the environment they are faced with.

Before the age of agile software, Rittel proposed the need for rational dialogue among diverse stakeholders when faced with trying to solve wicked problems²³⁸. While this may have been ahead of its time, it is now accepted as an approach in organisations, especially those which have implemented agile methodologies to work.

In summary, teams that encounter complexity, both on a personal level and system level, must be composed of individuals with diverse skill sets and generalist type capabilities. This will enable the team to identify problems within their environment and come to solutions based on diverse thinking and competing views of individuals which challenge predefined mental models.

Ordinary people working together become extraordinary. Teams work together on problems. Individuals put aside their own interests for the good of the team. People with different mental models, interpretations, perspectives and problem solving abilities who work together are able to solve problems that would be impossible to solve in isolation. In collaborating with others individuals are being subjected to alternative mental models, enabling shared understanding.

²³⁶ Stacey, *Complex Responsive Processes in Organizations*.

²³⁷ Weick and Sutcliffe, *Managing the Unexpected*, 56.

²³⁸ Conklin, "Wicked Problems & Social Complexity."

Agile encourages teams to be cross functional. When the team is a cohesive group of people with varying skill sets working as a single unit, they are able to deliver a working iteration of a product or solve a problem.

Cognitive diversity does not work in linear problems, or where collaboration is low in the environment. Then experts are better suited to the problem domain.

5.2.1.4 Self-organisation

A self-organising team can be identified by how much authority the team has as a whole. Attributes of a self-organising team are the ability to decide how work is organised between team members, and in what order it will be completed. Self-organising teams are also able to define team structure in terms of leaders and members of the team. The team is fully responsible for the work that is done; there is no management or external person questioning the work of the team.

A team is a field that is used for individuals to build trust by interacting. This is best done by personal physical interactions and dialogue²³⁹.

Self-organising teams facilitate the building of requisite variety that is needed for knowledge creation and problem solving²⁴⁰. While Nonaka's thoughts might be a challenge, his theory is that management sets a level of instability in the system by setting challenging goals and enabling autonomy²⁴¹.

²³⁹ Ikujiro NonakaTokyo Hirotaka Takeuchi both Professors of Management at the Institute of Business Research both at Hitotsubashi University, *The Knowledge-Creating Company : How Japanese Companies Create the Dynamics of Innovation*, (Oxford University Press, USA, 1995), 85.

²⁴⁰ NonakaTokyo Hirotaka Takeuchi both Professors of Management at the Institute of Business Research both at Hitotsubashi University, *The Knowledge-Creating Company : How Japanese Companies Create the Dynamics of Innovation*, 85.

²⁴¹ NonakaTokyo Hirotaka Takeuchi both Professors of Management at the Institute of Business Research both at Hitotsubashi University, *The Knowledge-Creating Company : How Japanese Companies Create the Dynamics of Innovation*, 85.

Nonaka's first discovery of the concept of self-organising teams manifested itself when management came to the understanding that the problem is not able to be solved without the application of an inclusive approach to problem solving by individuals actually performing the work. Typically, these teams were constituted in the form of a task force. Once the problem was solved, the team was dissolved. This can be observed as controlled self-organisation, in which the team that is put together has some bounded constraints to work within, being people and problem.

The concept of a boundary around agents may be against the ideas of open systems and free interactions. Cilliers²⁴² states that constraints provide positive effects to the system. This boundary around self-organisation provides a direction for self-organisation towards value.

5.2.1.5 Agile constraints in complex systems

That complex systems cannot be fully understood, and therefore never managed in completeness, has been argued previously in this thesis. The application of constraints is very applicable in dealing with complexity.

Constraints are the modulators of complex systems. They form the boundary to which self-organisation and agents can operate within. Snowden states the complex system can only work with some form of constraint. In fact, a complex system needs a more centralised and disciplined control than an ordered one. While direct control over individuals on a micro level is absurd, the boundaries around agents are controllable to steer positive outcomes and emergent behaviour.

Constraints do not have a negative effect on the system. By reducing or removing possibilities of outcomes, constraints can be found to have a positive effect on the system²⁴³.

Self-Organisation is the system's ability to structure or organise without explicit control or constraints from outside the system. But the internal constraints, based on local interactions, feed this self-organisation²⁴⁴.

²⁴² Cilliers, "Knowledge, Limits and Boundaries."

²⁴³ Cilliers, "Boundaries, Hierarchies and Networks in Complex Systems."

Agile frameworks provide the internal constraints in the lightweight process that enable this self-organisation. Be it scrum, lean or any other tool, the constraints imposed are easily adaptable for the agents, and have built in opportunities, such as retrospectives, especially geared to implementing process change.

To summarise the role of constraints: Good fences make good neighbours. Be mindful of constraints for control, role, and tasks. Relax constraints for inquiry and dialogue, tighten constraints when deciding and executing. Manage constraints to create the kind of space a team needs to accomplish solving problems that are encountered.

5.2.1.6 Retrospection is built into the process

Agile methodologies are adaptive by nature. The ability to respond to change in the product space is not the only space in which change happens. Agile methodologies allow for the inspection and adaption of process in each iteration. Teams normally start with a vanilla process and change it to something that best suits them. Processes are unique to teams. Due to this, process of change and organisational wide process adherence between teams is not usually observed.

As stated in chapter 2, complex systems display emergent behaviour. This proves problematic if repeatability is being sought in the system. Cause and effect are only evident in retrospect. Once causality has been identified, it will not ensure repeatable outcomes, but more understanding of the system. The key understanding is that in complex systems, causality is retrospectively coherent.

The use of set cadences to build retrospective coherence among individuals and teams is a very powerful tool. Setting this coherence allows individuals to become mindful of the situations they have encountered in the last iteration. Weick and Sutcliff describe this property of mindfulness in high reliability organisations as a commitment to longevity and resilience²⁴⁵. In addition to this, one of the properties of sense-making is retrospect, where Weick describes the sense-making process as only applicable to events that have occurred²⁴⁶.

²⁴⁴ Giovanna Marzo Serugendo, Marie-Pierre Gleizes, and Anthony Karageorgos, “Self-Organising Systems,” in *Natural Computing Series*, Natural Computing Series, (Berlin, Heidelberg: Springer Berlin Heidelberg, 2011), 8, doi:10.1007/978-3-642-17348-6_2.

²⁴⁵ Weick and Sutcliffe, *Managing the Unexpected*.

5.2.1.7 Insights into problem solving

While understanding wicked problems is hard because of their nature, working on the problem with thinking, tools and methods which have been used on simple problems causes frustration which is detrimental to individuals and the organisation²⁴⁷. This is mostly caused by the inability to categorise the problem space, forcing the application of inappropriate methods.

Social complexity also contributes to increasing ambiguity around a wicked problem. While wicked problems lie in the domain of complex adaptive systems, the more diverse parties involved in the problem space, the harder the problem domain becomes to understand²⁴⁸. This can be related back to communication and loose coupling of agents and interactions in the complex system.

People are unaware of the basic attributes of complex problems. A linear based approach as a attempt to effective to problem solving is one of the key misunderstandings.

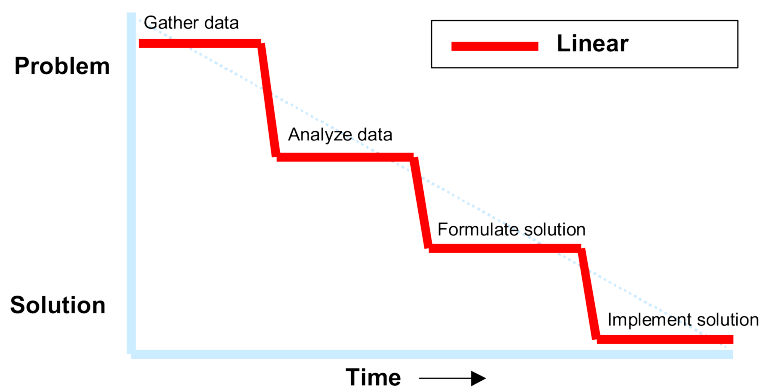


Figure: 21 Linear problem solving²⁴⁹.

²⁴⁶ Weick, *Sensemaking in Organizations*.

²⁴⁷ Conklin, "Wicked Problems & Social Complexity," 2.

²⁴⁸ Conklin, "Wicked Problems & Social Complexity," 3.

²⁴⁹ Conklin, "Wicked Problems & Social Complexity," 5.

This linear approach to problem solving is ingrained in organisations. The thinking is that the more complexity displayed in the problem domain, the more orderly the problem solving process must be. So established is this belief that it has risen to a rule based epistemology as noted by Snowden²⁵⁰

What was observed by Conklin in an experiment in which a group of individuals were presented a problem was that they did not follow this waterfall process. Once the problem domain was presented, they initially gained understanding of the problem and then started problem solving. At the end of a short problem solving period, the group would then move back to problem understanding from the information generated from the proposed solutions. While Conklin identifies this as opportunity driven problem solving, this process can be identified with the concept of iterating.

This process was then mapped against the original waterfall diagram.

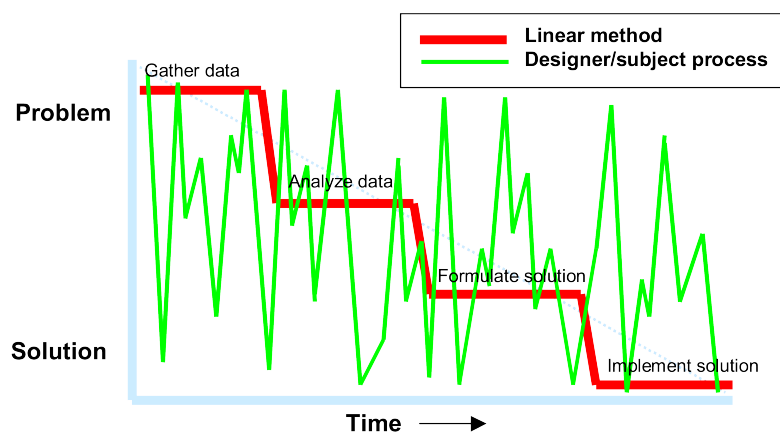


Figure 22 Iterative problem solving²⁵¹

5.3 Problem definition

Problem definition in software development has always been an issue. Firstly, in software development, there is a multitude of different ways²⁵² with no accepted standard of defining

²⁵⁰ Snowden and Boone, "A Leader's Framework for Decision Making."

²⁵¹ Conklin, "Wicked Problems & Social Complexity," 6.

problems or requirements of users. So much so that there have been numerous standards published on how to gather requirements from users. One such standard is IEEE Standard 830, from the Computer Society of the Institute of Electrical and Electronics Engineers (IEEE). This standard recommends approaches to organising the requirements specification document, how to do prototyping, and the characteristics of good requirements.

Documentation of this kind generates vast amount of codified information in an effort to make all knowledge explicit. This approach normally generates documents of systems spanning hundreds of pages. The documentation of requirements also assumes the ability of identifying the complete problem space upfront. This is impossible when dealing with changing contexts and the uncertainty and ambiguity which is associated with the problem space

Just as much as the design of the solution is unknown because of its emergent nature, so is the problem space that is being explored. The problem space will only be understood in more detail when the feedback loop in an iteration is used to identify with what was built, (created knowledge) and understand the problem space more generating new requirements.

The following problems can be attributed to the process of requirements gathering in IT systems development²⁵³.

1. In general, users do not know what they want until they get it, then they want something different
2. This in part because the interview process can only really explore what they do not like about the current state of affairs, a sort of need defined by negation of the present
3. Systems analysts like any interviewer start to form subconscious hypotheses after a fairly small number of interviews and then only pay attention to things that match those hypotheses in subsequent interviews
4. Outliers, or odd demands, are often ignored, while these may present some of the best opportunities of radical innovation and improvement

²⁵² Documentation methods can range from no documentation to full requirement documents with process flows and state diagrams. Normally this type of documentation is specific to a methodology or an organisation.

²⁵³ Snowden, ... *Was So Skinny and Thin*.

5. Most radical new uses of technology are discovered through use, not through requests and more often than not accidentally (think facebook, twitter etc.)
6. People only know what they know when they need to know it; it requires a contextual trigger which cannot be achieved in an interview
7. Early stage problems in roll out are easily ignored, or more frequently not reported, as they seem minor but then they build and result in major setbacks.

Snowden also goes on to theorise on how two key factors influence the process in identifying what people know in general²⁵⁴.

1. If you ask people what they know they tell you what they think you ought to know. This is normally explicit knowledge, the things that can be written down but will not contain what really should be known.
2. More valuable is the tacit knowledge, and is only known when it is needed to be known, as it is triggered by context by the knowledge holder

The observation from the above statements is the inability to actually transfer understanding that is required to solve complex problems, as well as information limiting in complex systems.

Boisot²⁵⁵ refers to this concept in different terms. He describes information or knowledge that is un-codified, but is complex and concrete as well as rich. The argument is that that rich data in large quantities is not helpful as it has too much associated information contained. When dealing with complexity, the ability to selectively process data is the key to providing agility and leanness to respond. The ability to absorb and process data without losing the ability to achieve coherence is critical to the success in dealing with complexity.

Individuals make sense of the world by integrating and combining different types of knowledge, including embodied, symbolic, and narrative knowledge to achieve coherence.

²⁵⁴ David Snowden, "Story Telling: an Old Skill in a New Context," *Business Information Review* 16, no. 1 (1999): 5.

²⁵⁵ Allen, Maguire, and McKelvey, *The SAGE Handbook of Complexity and Management*, 450.

This coherence affects how the mental models held by individuals are reinforced or modified towards a meaningful end goal²⁵⁶.

The challenge for agile software development dealing with vague or unknown problem domains is the ability to enhance the capacities to absorb and process data without compromising our ability to achieve coherence²⁵⁷.

5.3.1 User Story

Agile software development has developed a practice for describing the problem around which software is built, called a user story. The story is written from the person who is actually interacting with the system perceptively²⁵⁸. The user story is not meant as a complete specification and is more commonly referred to as a place holder for a conversation. During this conversation, the people who are involved in solving the problem that the user story represents have a discussion which leads to understanding on how to plan, construct, and test this piece of software.

The goal of a user story is to create shared understanding between the individuals with the problem and the group solving the problem. The format of the story restricts codification of all knowledge relating to the problem which is often seen in traditional requirement documentation. It focuses on the verbal interpersonal communication which is driven by the story. Snowden²⁵⁹ reiterates the common understanding of that what individuals know cannot be measured or made explicit with the statement;

²⁵⁶ Allen, Maguire, and McKelvey, *The SAGE Handbook of Complexity and Management*, 450.

²⁵⁷ Allen, Maguire, and McKelvey, *The SAGE Handbook of Complexity and Management*, 450.

²⁵⁸ Mike Cohn, "Advantages of the 'as a User, I Want' User Story Template | Mountain Goat Software," [Mountaingoatsoftware.com](http://www.mountaingoatsoftware.com/blog/advantages-of-the-as-a-user-i-want-user-story-template), accessed October 13, 2013, <http://www.mountaingoatsoftware.com/blog/advantages-of-the-as-a-user-i-want-user-story-template>.

²⁵⁹ Snowden, "Multi-Ontology Sense Making Making: a New Simplicity in Decision Making," 6.

"we always know more than we can say, we will always say more than we can write down"

The phrase "you can always say more than you write down", is a key driving point behind these principles of how user stories are used to convey meaning and context in a complex system.

5.3.2 Format of a user story

When user stories are written, the end users and the software developers sit together and talk about the problem domain. Once there is understanding of the problem, user stories are written around a generic format by everyone involved. The most widely accepted format for user stories is as follows.

User Story Title	Find Reviews Near Address
As a <user role> I want to <goal> so that <benefit>.	As a typical user I want to see unbiased reviews of a restaurant near an address so that I can decide where to go for dinner.

Figure: 23 Example of a user story

Mike Cohn²⁶⁰ identifies three main factors why this format is successful;

1. Requirements are put in the first person, giving the people working on the problem the ability to identify with the person and the context who is using the system.
2. Stories give more meaning to the person who is going to prioritise which item is going to be built first. It gives meaning through the communication of what the problem is and who it is going to benefit
3. Information is limited in the story by the nature of the format, but meaning is still high because of the conversations which happened around the creation of the story

²⁶⁰ Cohn, "Advantages of the 'as a User, I Want' User Story Template | Mountain Goat Software."

5.3.3 The Role of Narrative

Narrative has long been overlooked as a useful tool in organisations. The opportunities provided by storytelling in the context of communication²⁶¹ reveal patterns of culture and behaviour and understanding that are more effective than traditional approaches, such as questionnaires and problem analysis.

Boisot²⁶² makes the differentiation of complexity reduction compared to complexity absorption. Complexity absorption leads to accumulation of tacit experiential knowledge inside the organisation, while reduction focuses on articulation of knowledge for the coordination of dispersal.

The user story forces the premise of complexity absorption, with a short half-life associated with the user story. The need to transfer tacit knowledge is of importance. After the interaction of problem solving is done, the tacit knowledge is represented, not by the user story, but by the piece of working software, with the source code depicting the model that was used in the problem space.

Being at home with complexity will allow an organisation to absorb complexity more than if it feels threatened by it²⁶³. Stories are a way to enforce this understanding in complex environments, eliciting new levels of customer understanding and insight into customers and products²⁶⁴.

5.4 Conclusion

This chapter compares the specifics of agile software development with the application to complexity in organisations. Certain key tools are specifically relevant to dealing with complexity.

²⁶¹ David J Snowden, "Narrative Patterns: the Perils and Possibilities of Using a Story in Organisations" (2003): 12.

²⁶² Boisot, *Knowledge Assets*.

²⁶³ Boisot, *Knowledge Assets*.

²⁶⁴ Snowden, "Story Telling: an Old Skill in a New Context."

Iterations help with learning in environments with high uncertainty and ambiguity. The concept of an iteration also provides a boundary in which something can be delivered in a short period of time.

Teams are cross functional to leverage cognitive diversity. This also allows a team to be self-sufficient in problem solving without having to call on experts with heavy cognitive bias to one solution.

Retrospective understanding is important in the application of understanding. As a sense-making exercise, stopping and reflecting on the past is important. This enables individuals to build or rebuild mental models used in the future. The importance of this is paramount as individuals fall back to best fit pattern matching based on past experience when confronted by novel circumstances.

Lastly, the role of user stories and narrative in complex systems and how software development has moved away from codifying all knowledge possible and now uses stories and storytelling to convey context and meaning.

6

CONCLUSION

6.1 Introduction

Organisations are facing the challenge of shifting from industrial economy based on hierarchal control of standardised processes to decentralised information driven economy²⁶⁵. This forces organisations to implement different ways of thinking and doing in order to compete. To enable this, a different approach to thinking needs to be in place; a world where flexibility is more important than hierarchy. Connections are on a global scale, with network based relationships reinforced by dynamic roles and multiple identities. This change is evident in the social processes that are accepted as current.

This is a shift from the past where modernism has been on the forefront of thinking about organisations since the start of the 20th century. This influence was related to expansion of cities driven by the industrial revolution. While this provides an efficient worldview based on control and backed by wealth creation, this linear and causality based thinking has come under scrutiny due to the levels of complexity spurned by the revolution of worlds economy towards that supported by service based economy and the push to do more with less.

The challenges in the world today faced by organisations are multifaceted. Not only do they have to be able to respond to complexity in the market they choose to participate in, the characteristics of the organisation are benign, identified as complex systems with a large amount of diverse agents. No longer can the organisation shield itself from complexity, drive linear processes and traditional management thinking. The argument of this thesis is that the core concepts of agile gives the ability for an organisation to transition towards a sustainable organisation paradigm, where unpredictability and uncertainty are encountered due to the dynamic nature of the non-linear environment.

²⁶⁵ Snowden and Stanbridge, “The Landscape of Management: Creating the Context for Understanding Social Complexity.”

6.2 Why Agile

The software development industry has been on the forefront of developing new methodologies for organisations to apply to work. Because in the software development space, work is so different and exceedingly complex to that from the factory floor where modernistic thinking and methods were born, the realisation that these methods would not work for knowledge intensive activity has driven the creation of agile software development and transition organisations to have the possibilities to be flexible by replacing modern based theories.

These organisations are repeatedly delivering value to their customers by responding to the complexity and giving the customer what they really want, not just the understanding from the first day when they knew the least about the problem.

Agile software development has provided two types of benefits that are applicable to complexity. The first is the understanding that implementing a people's first approach to dealing with complexity is the first step forward. By implementing its values and principles, this encompasses the humanising factor of agile. Its structure of small cross functionality teams to bring diversity is also an import aspect. This engages individuals on another level bringing collaboration around problems to the forefront. The change in the way that individuals and teams think about problems is also critical, as the concept of mental models as discussed in chapter four is very important when faced with complexity.

Secondly, agile provides a minimal process necessary to deal with complexity where they must quickly adapt based on the interconnected actions of competitors, customers, legislation, and stakeholders and leverage new emergent behaviour. The tools that agile methodologies provide the organisation with at all levels, innovative ways of dealing with complexity, understanding the problem domain and enabling a solution to emerge. Agile does this by creating the correct culture and mindset needed in knowledge based organisations, a mindset that allows for people to self-organise around problem domains, but with the constraints needed for the outcomes to emerge and be useful for the organisation

Agile frameworks provide a platform for learning in complex environments. Tools and thinking such as small experiments allow for new and unique outcomes to be discovered, giving the organisation the ability to change direction to take competitive advantage of these learnings. Smaller iterative learning cycles enable individuals and teams to attain knowledge without the need for full initial understanding in problem domains that are uncertain. This

concept is extremely important, as the extent of explicit knowledge in the world is virtually unlimited. Achieving understanding in small cycles increases the efficiency and effectiveness of individuals. Iterations reduced the complexity of the work by limiting exposure to the work that is being done.

Emergent properties of systems are seen by the amount of physical outputs from the design process. Agile approaches to controlling emergence based around iterative models provide the platform for amplifying or damping emergence.

Scrum uses basic artefacts and ceremonies which provides a powerful way to convey understanding between individuals. A simple tool, such as a task board, creates transparency of work being doing, changing the approach and perceptions and problem solving via high levels of collaboration and trust in the organisation. The empirical nature of the process to accommodate constant evaluation in the organisation is powerful, but it also leads to the insight that agile methods cannot be replicated by standardisation across organisations. Each implementation is unique.

The application of the Cynefin model to the act of programming by an individual results in the understanding that lies in the complicated domain. The code can be understood and broken apart by a specialist. The power that agile software development provides is in the human interactions in the project. These interactions are infinitely complex. This complexity recognised and absorbed, and, as Boisot²⁶⁶ has stated, it creates potential for giant leaps forward in innovation. The key factor behind complexity is that it is visible in everyday life. If the illusion is discarded that you can reduce any problem down in order to explain causality, then there is an appreciation for complexity.

Lessons learnt from agile software development and the practices implemented, both in process and the people first approach, cannot be called a new management fad. People are non-linear and have to be put first. People are not predictable. To have a process that is predictable and then populate the process with unpredictable agents will negate the process. Agile software development understands this, with empirical processes providing a boundary for self-organisation around problem domains and emergence to occur, and it provides the space for work to be completed on an autonomous level.

²⁶⁶ Boisot, MacMillan, and Han, "Explorations in Information Space: Knowledge, Agents, and Organization - Max H. Boisot, Ian C. MacMillan, Kyeong Seok Han - Google Books."

These core concepts have been around for an extended period. Some used successfully independently in various degrees.

6.3 Agile: No silver bullet

While agile might sound like the solution for all problems, with the promises of self-organisation, emergence and collaboration, more often than not it is not. Agile does not fit all situations; if levels of complexity are not high and linear based, processes that are working at implementing agile type thinking would be detrimental.

Agile provides a platform combining practices and principles in a very powerful way as an approach to the organisation. This shift in thinking and people-first based approach and adaptive processes combined with the correct leadership organisations are able to succeed when dealing with complexity. The caveat applies that change towards agile thinking has to be implemented properly, whether from top down approaches and bottom up, to instil the culture of mindfulness.

It is important to acknowledge that there are also negative connotations and perceptions about agile software development. With implementation failure rates of more than 70%, this leads to the assumption from traditional management thinkers that agile is a fad and it has a defined shelf life.

With the popularisation of using agile principles and practices as a management concept, more failure will be seen and more criticism of the methods and frameworks will be heard. Denning²⁶⁷ categorises agile and scrum as a subset of what he terms as radical management. The reasoning behind that is that scrum and agile are specific to software development environments, citing the agile manifesto's statement of "working software" to back up his claim. He takes the concepts and relabels them as client driven iterations, which encompasses the philosophy behind agile software development. Organisations that do not use agile software development have the same type of values and principles ingrained in the culture which enables similar results when responding to complexity. Agile software development represents a weaker attractor to repetitive behaviour due to the iterative nature of process and the associated potential information generated by the iteration.

²⁶⁷ Denning, *The Leader's Guide to Radical Management*.

An agile process implemented as a recipe will not provide hyper-productivity, rapid delivery or self-organisation. Failures in agile implementation are often seen around organisations that are chasing these or implementing short-term fixes to productivity without understanding the deep principles and practices as identified in the agile manifesto.

As Max Pucher stated²⁶⁸:

"Agility is not an intrinsic property of software or methodology and cannot be supplied through these. Only agile-thinking management and employees will make a business more agile."

Agility is a way of thinking, is the mindset which recognises that the world we live in is full of uncertainty, ambiguity and probability. This mindset is created when the illusion that individuals are still able control the outcomes of systems, which is a forgone idea in most management and systems and thinking. Agile methodologies are not a recipe that needs to be followed to ensure success; more importantly they are tools which help people change mental models that are held of the environment around them.

6.4 The case against "Best practices"

In the face of complexity, decision maker's retreat to places of safety. This is normally associated with choosing naive linear based solutions to solve organisational problems. Often manifested in the implementation of best practice. The theory of best practise may apply to simple linear problems, however, the inverse is true with complex problems; application of best practices will not lead to a result that will be suitable. Snowden reinforces the thinking of the application of agile as a response to complexity with comparison of how ordered ontology and rule based epistemology²⁶⁹ is applied to complexity without success²⁷⁰.

²⁶⁸ Why SOA does not deliver. Article in Output Magazine 2007 <http://www.adaptive-process.com/>

²⁶⁹ Thinking based on Taylorism and Business process reengineering, or mechanical based approaches to management of organisations

²⁷⁰ Snowden, "Multi-Ontology Sense Making Making: a New Simplicity in Decision Making."

Problems start to become evident when the management models that have grown from manufacturing and are typically linear based are applied to less structured parts of the organisation where high levels of complexity are encountered.²⁷¹ What is a shared concept among these approaches is the focus on efficiency, repeatability and consistency, with a strong quantitative analysis of the system based on measurement. This can be described as the application of a best practice or management template. When applied to systems of uncertainty and ambiguity (complex adaptive systems) difficulties are encountered as significant elements that are unknown or impossible to anticipate can be present. If there is no inefficiency in the system, Snowden argues, there is no adaptive capacity.

Therefore, the acknowledgment of complexity is better than application of a band-aid to the problem with the imposition of planning models. Because of the uncertainty of most of the environments, the only outcome that can be used with any reliance is the ability to learn in a space of constant change.

6.4.1 Measuring non-linear systems

Measurements in systems are normally based on a defined process model. In addition to this, measurements of most things in a system imply that causality can be determined. Measurement of a non-linear system with contexts that are in flux will not help predict future states in the system under measurement. To measure, one must manage and understand the system in its entirety, and this is impossible. The only measurements that can be applied to situations of complexity are the focus on the outcomes of the system, not the measurement of the agents in the system. Embracing complexity means nothing more than to be aware of this unpredictability and the fallacy of repeatability.

As stated in chapter three, with the comparison of defined process models against empirical process models, this proves problematic for non-linear systems. Responding to complexity means working with outcomes not measures.

²⁷¹ Snowden, "Multi-Ontology Sense Making Making: a New Simplicity in Decision Making," 5.

6.5 Instilling Resilience

When dealing with chaotic effects in the system, no amount of agility from a process will enable the organisation to uncover and respond to catastrophic events. Events which are too small to be detected will always be a shock to the system when they manifest themselves into large outcomes. The ability to deal with these shocks and facilitate the recovery of the system to become productive is what is important when dealing with resilience.

Any approach applied to dealing with complexity, be it an agile process or any other chosen tool, will not help understand the true nature of the problem without the appreciation of the human construct, being a collection of feeling expectations and sensations taken into account.

Times of organisational expansion are often associated with a transition away from flexibility, and it is often observed that application of additional governance in organisations is usually to ensure predictability and repeatability of work, with the goal not impeding creativity. This is normally seen when an organisation is at a level of growth that exceeds its initial expectations and *ad hoc* processors that have worked well in the past are no longer seen as successful because of increased social complexity encountered by a management level and not the worker level.

The reality is that the implementation of this governance leads to a loss of autonomy of individuals. With empirical based processes, where the individuals who are constrained by the process but also have the power to change it, the balance of power shifts to being an enabler of bottom-up change to process. This creates a sense of flexibility and motivation for individuals.

6.6 Further research

In the research process for this thesis the uncovering of mental modelling and limits to knowing were an important discovery, especially when applied to developing software in complex systems. While software development still insists on the concept of intern mediation between end users and problem solvers, this can be problematic due to some concepts of model interpretation between individuals at the abstract level in a software development cycle. This interpretation of a model by the receiver will cause more abstraction, as cues that are context specific for the person who created the model can be seen as surplus data in

receiver's eyes. The outcome could be a well-known cause of pain in software development "my requirements were never met by the software development department."

7

BIBLIOGRAPHY

Allen, Peter, Steve Maguire, and Bill McKelvey. *The SAGE Handbook of Complexity and Management*, SAGE Publications Limited, 2011.

Appelo, Jürgen. *Management 3.0*, Addison-Wesley Professional, 2010.

Beck, Kent. “Extreme Programming Explained.” *Embrace Change* (2006).

Boisot, M H, I C MacMillan, and K S Han. “Explorations in Information Space: Knowledge, Agents, and Organization - Max H. Boisot, Ian C. MacMillan, Kyeong Seok Han - Google Books” (2008).

Boisot, Max. *Knowledge Assets*, Oxford University Press on Demand, 1999.

Boisot, Max, and John Child. “Organizations as Adaptive Systems in Complex Environments: the Case of China.” *Organization Science* 10, no. 3 (1999): 237–252.

“History of Software Engineering” (October 17, 2008): 1–61.

Brooks, Frederick. “The Mythical Man-Month - Essays on Software Engineering - . Brooks (Addison-Wesley, 1995) WW” (January 14, 2013).

Cilliers, P. “Boundaries, Hierarchies and Networks in Complex Systems.” *International Journal of Innovation Management* (2001).

Cilliers, P. *Complexity and Postmodernism: Understanding Complex Systems*, 2002.

Cilliers, P. “What Can We Learn From a Theory of Complexity?.” *Emergence* (2000).

Cilliers, Paul. *Complexity and Postmodernism*, Psychology Press, 1998.

Cilliers, Paul. “Knowledge, Complexity, and Understanding.” *Emergence, a Journal of Complexity Issues in Organizations and Management* 2, no. 4 (2000): 7–13.

Cilliers, Paul. “Knowledge, Limits and Boundaries.” *Futures* 37, no. 7 (September 2005): 605–613. doi:10.1016/j.futures.2004.11.001.

Cilliers, Paul. “On the Importance of a Certain Slowness.” *Emergence, a Journal of Complexity Issues in Organizations and Management* 8, no. 3 (2006): 105.

- Cilliers, Paul. "Why We Cannot Know Complex Things Completely." *F Capra* 4, no. 1 (2007): 77–84.
- Cleveland, John. "Basic Concepts and Application to Systems Thinking" (June 11, 1994): 1–28.
- Cockburn, Alistair, and Jim Highsmith. "Agile Software Development, the People Factor." *Computer* 34, no. 11 (2001): 131–133.
- Cohn, Mike. "Advantages of the 'as a User, I Want' User Story Template | Mountain Goat Software." *Mountaingoatsoftware.com*. Accessed October 13, 2013. <http://www.mountaingoatsoftware.com/blog/advantages-of-the-as-a-user-i-want-user-story-template>.
- Conklin, Jeff. "Wicked Problems & Social Complexity" (2006).
- Dekker, Sidney. "Drift Into Failure: From Hunting Broken Components to Understanding Complex Systems" (2011): 1–236.
- Denning. *The Leader's Guide to Radical Management*, 2010. <http://>.
- Downey, Allen B. *Think Complexity*, O'Reilly Media, 2012.
- Fowler, M. "The New Methodology." *Martinfowler.com*. Accessed October 30, 2013. <http://martinfowler.com/articles/newMethodology.html>.
- Fowler, M, and J Highsmith. "The Agile Manifesto." *Software Development* (2001).
- Gell-Mann, Murray. "What Is Complexity? Remarks on Simplicity and Complexity by the Nobel Prize-Winning Author of The Quark and the Jaguar." *Complexity* 1, no. 1 (May 16, 2013): 16–19. doi:10.1002/cplx.6130010105.
- Groysberg, Boris, and Michael Slind. "Leadership Is a Conversation." *Harvard Business Review* 90, no. 6 (2012): 75–84.
- Highsmith, J. "Highsmith: Agility in Software Development - Google Scholar." *Agile Software Development Ecosystems* (2002).
- James A Highsmith, III. *Adaptive Software Development*, Dorset House, 2000.
- Kniberg, Henrik. "Kanban and Scrum-Making the Most of Both" (2010).
- Lansdell, Sally. "Issue 2-4" (March 10, 2001): 1–167.

- Manson, Steven M. "Simplifying Complexity: a Review of Complexity Theory." *Geoforum* 32, no. 3 (2001): 405–414.
- Marzo Serugendo, Giovanna, Marie-Pierre Gleizes, and Anthony Karageorgos. "Self-Organising Systems." In *Natural Computing Series*, 7–32. Natural Computing Series, Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. doi:10.1007/978-3-642-17348-6_2.
- McCarthy, Ian P. "Technology Management—a Complex Adaptive Systems Approach." *International Journal of Technology Management* 25, no. 8 (2003): 728–745.
- Mission Command*, 2003.
- Mitchell, Melanie. *Complexity: a Guided Tour*, Oxford University Press, USA, 2009.
- Mitleton-Kelly, Eve. "Organisations as Complex Evolving Systems" (1998): 4–5.
- Morgan, Gareth. *Images of Organization*, SAGE Publications, Incorporated, 2006.
- Nonaka, I. "Ikujiro, 'the Knowledge-Creating Company,'." *Harvard Business Review* (1991).
- Nonaka, Ikujiro, Tokyo Hirotaka Takeuchi both Professors of Management at the Institute of Business Research both at Hitotsubashi University. *The Knowledge-Creating Company : How Japanese Companies Create the Dynamics of Innovation*, Oxford University Press, USA, 1995.
- Ohno, Taiichi, and Taiichi Ōno. *Toyota Production System*, Productivity Press, 1988.
- Pal, Nirmal, and Daniel C Pantaleo. "Introduction the Agile Enterprise." 1–10, New York: Springer-Verlag, 2005. doi:10.1007/0-387-25078-6_1.
- Parker, D, and R Stacey. "Chaos, Management and Economics: the Implications of Nonlinear Thinking" (1994).
- Poppendieck, Mary, and Tom Poppendieck. *Lean Software Development*, Addison-Wesley Professional, 2003.
- Royce, Winston W. "Managing the Development of Large Software Systems" 26, no. 8 (1970).
- Rubin, Kenneth S. *Essential Scrum*, Addison-Wesley Professional, 2012.
- Sargut, Gökçe, and Rita Gunther McGrath. "Learning to Live with Complexity.." *Harvard Business Review* 89, no. 9 (2011): 68.

Schwaber, K, and J Sutherland. "The Scrum Guide—the Definitive Guide to Scrum: the Rules of the Game (2011)." URL [Http://Www.Scrum.Org/Storage/Scrumguides/Scrum% 20Guide](Http://Www.Scrum.Org/Storage/Scrumguides/Scrum%20Guide) (2012).

schwaber, Ken, and Jeff Sutherland. "The Scrum Guide" (December 1, 2011): 1–16.

Senge, P. "The Fifth Discipline." *The Art & Practice of Learning Organization* ... (1990).

Simon, H A. "Models of Bounded Rationality: Empirically Grounded Economic Reason - Herbert Alexander Simon - Google Books" (1982).

Snowden, D. "Multi-Ontology Sense Making Making: a New Simplicity in Decision Making" (2004).

... *Was So Skinny and Thin*. Accessed October 12, 2013. <http://cognitive-edge.com/>.

Snowden, D, and P Stanbridge. "The Landscape of Management: Creating the Context for Understanding Social Complexity." *Emergence-Mahwah-* ... (2004).

Snowden, David. "Story Telling: an Old Skill in a New Context." *Business Information Review* 16, no. 1 (1999): 30–37.

Snowden, David J. "Multi-Ontology Sense Making: a New Simplicity in Decision Making" (2005).

Snowden, David J. "Narrative Patterns: the Perils and Possibilities of Using a Story in Organisations" (2003): 12.

Snowden, David J, and Mary E Boone. "A Leader's Framework for Decision Making." *Harvard Business Review* 85, no. 11 (2007): 68.

Stacey, R. "Responding to Complexity and Uncertainty: the Agile Organisation | Complexity & Management Centre." *Complexityandmanagement.Wordpress.com*. Accessed April 7, 2013. <http://complexityandmanagement.wordpress.com/2012/08/29/responding-to-complexity-and-uncertainty-the-agile-organisation/>.

Stacey, Ralph. *Complex Responsive Processes in Organizations*, Routledge, 2003.

Stacey, Ralph. "Strategy as Order Emerging From Chaos." *Long Range Planning* 26, no. 1 (February 1993): 10–17. doi:10.1016/0024-6301(93)90228-8.

Takeuchi, Hirotaka, and Ikujiro Nonaka. "The New New Product Development Game." *Harvard Business Review* 64, no. 1 (1986): 137–146.

Tomasini, Andrea, and Martin Kearns. “Agile Transition-What You Need to Know Before Starting” (n.d.).

Vlaanderen, Kevin, Slinger Jansen, Sjaak Brinkkemper, and Erik Jaspers. “The Agile Requirements Refinery: Applying SCRUM Principles to Software Product Management.” *Information and Software Technology* 53, no. 1 (January 1, 2011): 58–70.
doi:10.1016/j.infsof.2010.08.004.

Wehmeyer, B. “Complexity Theory as a Model for the Delivery of High Value IT Solutions” (July 3, 2007): 1–142.

Weick, Karl E. *Sensemaking in Organizations*, SAGE Publications, Incorporated, 1995.

Weick, Karl E, and Kathleen M Sutcliffe. *Managing the Unexpected*, Jossey-Bass, 2011.

Wolfram, Stephen. “Complex Systems Theory (1984).” *Stephenwolfram.com*, April 1, 1984.
<http://www.stephenwolfram.com/publications/articles/general/84-complex/2/text.html>.

Xiong, Jay. *New Software Engineering Paradigm Based on Complexity Science*, Springer Science+Business Media, 2011.

“Agile Software Development: a Gentle Introduction..” *Agile-Process.org*. Accessed October 27, 2013. <http://www.agile-process.org/>.