Creative Approach to Software Development:

Rethinking the Software Development Life Cycle

A Research Report

presented to

THE GRADUATE SCHOOL OF BUSINESS
UNIVERSITY OF CAPE TOWN

in partial fulfilment

of the requirements for the

Masters of Business Administration Degree

by

Puleng Molahlehi

December 2008

Supervisor: Professor Jon Foster-Pedley
Acknowledgments

This report is not confidential. It may be used freely by the Graduate School of Business.

To the Lord God, the giver of strength and wisdom, I would not have done this without you and I thank you from the bottom of my heart.

I wish to pass my heart-felt gratitude to my Supervisor, Professor Jon Foster-Pedley for all the support and material as I did the research. I wish to extend avery warm, heartfelt gratitude to my fiancé, ThekoMakhoalibe for always being there to strengthen me and encourage me in this journey. To my family and friends for their love and support in everything, I appreciate it and know your love means everything to me.

I certify that except as noted above the report is my own work and all references used are accurately reported in footnotes.

Signed:

               Puleng Molahlehi
ABSTRACT

Purpose - The purpose of this paper is to contribute to the Information Technology field by integrating the creative problem solving methods and creativity principles to the software development process. This is done by first establishing the need to integrate creativity for products in the 21st Century, then studying creativity and its application and how it can be integrated in the Software Development Life Cycle, which has been used for decades as a software development tool. The end product is a Software Development Life Cycle that integrates creativity and is believed to be of value to software development success in the 21st century.

Design/methodology/approach – The papers begins with the background from authors of the 21st century revolution; Daniel Pink, Ken Robinson, and Eddie Obeng. Their views of the emerging 21st Century markets and changes that should be expected in the 21st century are studied. Then a thorough study of creativity and creative problem solving is made and how it can be applicable in the Information Technology field. Two workshops were held at a Tenet Conference that was held for Information Technology staff at various Universities in South Africa to get their views on the subject of creativity in software development. A survey was made to establish the impact of the emerging market trends on software development, which proves software to be one of the most utilised products in the 21st Century with changing needs. Having established this, the paper builds on the application of these theories in a software development process using system-thinking methodology to come up with a new model for software development that embraces creativity.

Findings - The paper maintains that traditional software development life cycle is no longer sufficient as a tool for Software Development in the 21st century. Furthermore, it established a new methodology, which integrates creativity in the Software Development Life Cycle.

Originality/value – This is still a new research area, the contribution challenges the traditional software development process and suggests new ways of looking at the process to best cater for the emerging market trends of the 21st century.

Keywords - Creativity, Software Development Life Cycle (SDLC), Information Technology (IT), Innovation

Paper type - Research paper
# Table of Contents

Acknowledgments........................................................................................................................... 1  
ABSTRACT .................................................................................................................................... 2  
Table of Contents ............................................................................................................................ 3  
1. Area of study............................................................................................................................ 6  
2. Literature Review..................................................................................................................... 8  
   2.1 Background ............................................................................................................................ 8  
      2.1.1. The 21st Century according to Eddie Obeng.............................................................. 8  
      2.1.2. The 21st Century according to Ken Robinson.......................................................... 10  
      2.1.3. The 21st Century according to Daniel Pink............................................................ 12  
      2.1.4. Software Development in the 21st Century.............................................................. 14  
   2.2. What is creativity ............................................................................................................. 17  
   2.3. Leadership in Creativity ................................................................................................. 18  
   2.4. Teamwork in Creativity ................................................................................................. 21  
   2.5. Environment that stimulates creativity ......................................................................... 23  
   2.6. Models for creative processes......................................................................................... 25  
   2.7. Creative Project Management......................................................................................... 31  
3. Software Development Life Cycle...................................................................................... 33  
4. Learning Objectives ............................................................................................................... 41  
5. Research Question ................................................................................................................. 43  
   5.1. Primary Question ............................................................................................................ 43  
   5.2. Secondary Questions....................................................................................................... 44  
   5.3. Learning Objectives ....................................................................................................... 44  
6. Research Methodology .......................................................................................................... 45
Figure 12: Integration and Test Phase ................................................................. 38
Figure 13: Installation and Acceptance............................................................... 39
Figure 14: Widening Development Phases to Consider More Alternatives (Source: Couger, 1996) ..................................................................................................................................... 42
Figure 15: Triangulation, Jon Foster-Pedley .............................................. Error! Bookmark not defined.
Figure 16: Graph of user requirements from software........................................... 55
Figure 17: Affinity Diagram for good software....................................................... 62
Figure 18: Interrelationship Diagram................................................................. 65
Figure 19: Interrelationship Diagram................................................................. 66
Figure 20: Causal loop diagram.......................................................................... 67
Figure 21: Three requirements for creativity...................................................... 68
Figure 22: Affinity Diagram for Bad Software..................................................... 72
Figure 23: Themes from bad software................................................................... 73
Figure 24: Delaying Convergence to use creativity techniques to consider more alternatives ............................................................ 77

List of Tables:
Table 1: Differences between the old and new world (Source: Obeng, 1998, P.19) ............... 9
Table 2: A checklist of important traits found in a creative manager (Source: Thierauf, 1993, P42-43).................................................................................................................. 20
1. Area of study

“The labour markets of the 21st century are changing beyond recognition. This is not a revolution in a figurative sense, but a real one comparable in scale and impact to the massive upheavals of the Industrial Revolution. This revolution is being driven like the last one by the developments in technologies, and this one has hardly begun. Change is a constant factor in human history.

New forms of work rely increasingly on high levels of specialist knowledge and on creativity and innovation particularly in the uses of new technologies. These require wholly different capabilities from those required by the industrial economy.”

(Robinson, 2001:5-6).

The author agrees with Robinson that the 21st Century market requires increasingly creative and innovative solutions to solve problems with high complexity in an ever-changing environment. The increasing dependency on technology and the exponential rate of change in the technology industry requires strict integration of creativity in development and management of information systems. As a major player in the exponential change that is prominent now, the Information Technology field is subjected to more pressure in this century than ever before. Information System personnel are expected to be more inventive and innovative not only to gain a competitive advantage but to simply sustain existence. Couger (1996) emphasizes this by mentioning the importance of enabling the existing information systems workforce to become more creative to meet the growing demands of creative solutions in the increasingly complex world.

In this document, the author focuses on creativity in the software development process. Robinson (2001) says there must be a systematic strategy to facilitate and reward creative output. Hence in order for a software development entity to tap into creativity, it should use a holistic approach
that involves making the current processes and procedures to tie in practices that encourage creativity in order to meet the demands of the 21st century market.

Glass (1995) compared two aspects in software development, creativity and discipline, and pointed out that software construction was primarily a complex problem-solving activity, which required creativity. Ultimately software design has been acquiring more importance as the complexity level of software increases. Complexity in software design is increasing rapidly, forcing development teams to be more efficient and more ingenious in their solutions (Gomes, 2001). Software designers must therefore find new design methodologies, in trying to optimize development time, processing time, required memory, and other resources.

Little research has been done in this area (Couger 1996:12) and conventional processes that were developed in the last century are still being used in Information Systems Development. Among many models used in Information Technology is the Software Development Life Cycle (SDCL). This constitutes a series of steps adhered to in a software development process. From the early sixties, Software Developers have adopted this model for software development processes and it is still being used to date. SDLC, in its simplest format consists of four steps namely planning, building, testing and implementing.

While the SDLC is a very powerful tool in software development that can be adapted in various software development projects of various magnitude and allows for agility and flexibility, this paper specifically looks at the creativity practices and models that can enhance or upgrade the SDLC to make it adaptable in the 21st century software projects that are believed to be far more complex and abstract in nature. The approach in this research will be to use literature studies, surveys, interviews and system thinking methodology to come up with a new model that will incorporate the element of creativity in the SDLC.
2. Literature Review

2.1 Background

The call to reconsider the software development processes comes from the assumption that the 21st century comes with totally different market expectations and experience as stipulated by the three writers EddingObeng, Ken Robinson and Daniel Pink.

2.1.1. The 21st Century according to Eddie Obeng

“Every few hundred years throughout Western history, a sharp transformation has occurred. In a matter of a few decades, society altogether rearranges itself, its world views, its social and political structure, its arts, its key institutions. Fifty years later a new world exists. And the people born into that world cannot even imagine the world in which their grandparents lived and into which their own parents were born.” Peter Drucker

Eddie Obeng in his book “New rules for the new world” unfolds the tremendous change in the world that has taken place. He speaks of the new world and the old world where the old world was the period where organisations were learning faster than the rate at which the world was changing. However the new world is the period where the rate of learning in organisations is much lower than the rate of change.

Figure 1: The old and the new world (Source: Obeng, 1998, P.6)
Obeng (1997) further shows that the challenges of the new world are not only that organisation cannot learn as fast as the world is changing, but products are increasingly depending on a wider range of knowledge, skills, values, technologies, and competences implying learning about greater range of subject at a faster speed.

Obeng also mentions the greater levels of uncertainty that comes with this change. He says the natural response to change by the human species is that of fear especially when others bring the change; it tends to pose a threat and a feeling of insecurity.

He summarises the differences between the new and the old world in the table below:

<table>
<thead>
<tr>
<th>Old world</th>
<th>New world</th>
</tr>
</thead>
<tbody>
<tr>
<td>It is possible to learn faster than the world changes</td>
<td>World changes much faster than the ability to learn</td>
</tr>
<tr>
<td>Constraints if cyberspace</td>
<td>Constraints in meatspace/ touchspace</td>
</tr>
<tr>
<td>Change is usually an event. Linear and first order</td>
<td>Change is often non-linear and not in first order</td>
</tr>
<tr>
<td>Change is often perceived as a “step change” and a clear discontinuity</td>
<td>Change is often alterations and may show emergent effects</td>
</tr>
<tr>
<td>Ways of working are valid until it is obvious that they are invalid</td>
<td>Everything should have a sell by date</td>
</tr>
<tr>
<td>Local is defined by movement of atoms</td>
<td>Local is defined by interactions and movement of information</td>
</tr>
<tr>
<td>Human beings exchange goods and services in order to survive themselves</td>
<td>Human beings exchange goods and services in order to survive themselves.</td>
</tr>
</tbody>
</table>

Table 1: Differences between the old and new world (Source: Obeng, 1998, P.19)

Obeng clearly shows the challenges that come with the shift from the world before midnight and the world after midnight. He says that the two worlds bring different realities, which will in turn give rise to different assumptions about how it all works, the actions people take, and finally people’s mindsets and how they interpret any additional incoming data.
2.1.2. The 21st Century according to Ken Robinson

“Organisations face three challenges in making the most of their creative potential and human resources. The first is to understand the real nature of creativity. The second is to implement systematic strategy for developing individual creative capacities. Third, there must be a systematic strategy to facilitate and reward creative output.”

(Robinson, 2001, P.3)

In his book, Out of our Minds, Robinson talks about the tremendous revolution that comes with the 21st Century where labour markets are changing beyond all recognition. He suggests that the main drivers of this revolution are the developments of technologies. Although change is a constant factor in human history, the rate and scale of change in the 21st century is beyond what men’s minds can fathom as it impacts directly on the nature of work, transforming the how, who, when and for how long. These impacts even on the social and cultural fabric of the societies as seen clearly in the knowledge-based industries which are massively replacing labour-intensive work to machinery that comes with the new technologies.

Robinson mentions that these new forms of work rely increasingly on high levels of specialist knowledge and on creativity and innovation particularly on the use of new technologies. These require wholly different capabilities from those of the industrial economy.

He mentions three key areas of development necessary for survival on this fast changing world: Firstly, generation of ideas for new products and services is essential in order to maintain the competitive edge. Secondly, education and training that will allow people the necessary flexibility and adaptability for businesses to respond to the changing markets. Thirdly, secure lifelong employment in a single job is obsolete.

“To move forward we need a fresh understanding of intelligence, of human capacity and of the nature of Creativity.” (Robinson, 2001, P.9)
Robinson’s argument to education being the answer to the 21st century is that there’s must be more to intelligence than academic ability and much more to education than developing it. He points out that if there is more to intelligence than this, most of human culture with its complex fabric of scientific, technological, artistic, economic and social enterprises would never have happened. He believes that human intelligence is much more richer and dynamic than we have been led to believe by the formal academic education. He also points out that the scientific studies of the brain are confirming that human intelligence is complex and multi-faceted.

“Brain-scanning techniques show that even simple actions draw simultaneously on different functions and regions of the brain.” (Robinson, 2001, P.9)

Robinson goes further to clarify on the essential elements of the culture of creativity by emphasising the link between culture and creativity. He says cultural conditions can kill or kindle creativity. Individual creativity is stimulated by work, ideas and achievements of other people in all fields. “Human intelligence is creative in a profound sense” (Robinson, 2001:11). Creative insights often occur by making connections between ideas or experiences that were previously unconnected. Just as intelligence in a single mind is interactive, creativity is often interdisciplinary. Robinson mentions that the best creative teams often contain specialists in different fields, and he says this has serious implications for the culture of organisations that want to promote creative development.

“We all have creative abilities and we all have them differently” (Robinson, 2001, P.12)

It is therefore necessary to unlock the creative ability in each individual. He says creativity is a dynamic process that draws on many different areas of the person’s experiences and intelligence. Creativity and innovation must be harnessed not just released. Creativity prospers under certain conditions:

- Where there is flow of ideas between people who have different sorts of expertise.
- An atmosphere where risk taking and experimentation are encouraged rather than stifled.
- It draws from skills and expertise across an organisation.
- Creativity flourishes where there is a systematic strategy that promotes it.
Robinson also mentions the key ingredients of the creative process:

- The importance of the medium. Creativity involves taking an action, finding one’s own medium is a key element in tapping into creativity.
- The need to be in control of the medium. Technical control is necessary for creative work; the command and the confidence are necessary for the output.
- The need to play and take risks. Creativity is not only a matter of control; it is about speculating, exploring new horizons and using imagination. Playing with new ideas and trying out new possibilities.
- The need for Critical judgement. Creative outcomes must be of value. Critical evaluation involves a shift in the focus of attention and mode of thinking as we attend to what is working or not working.

“The most extraordinary development may yet come from the merging of information technologies and human intelligence” (Robinson, 2001, P. 17)

2.1.3. The 21st Century according to Daniel Pink

“We are moving from an economy and a society built in the logical, linear, computer like capabilities of the information age to an economy and a society built on the inventive empathetic, big picture capabilities of what’s rising in its place, the conceptual age.” (Pink, 2005, P2)

Pink contributes to this school of thought in his book “A Whole New Mind”. He demonstrated the transformation from the information age to the conceptual age. Pink shows the transition from the agricultural age to the conceptual age and the impact of the transition and the necessary human strength or discipline to sustain each phase.

Pink depicts this progression in the figure below, which shows the transition from the agricultural age to the conceptual age in terms of the three factors; affluence, technological
progress and globalisation. He stipulated that as the three forces gather enough collective momentum they have led to a new era.

ATG – Affluence, technology, globalisation

Pink further demonstrates the kind of expertise needed in the eras stating that there has been a significant progress from a society of farmers to that of knowledge workers, but the conceptual age requires creators and empathisers, pattern recognisers and meaning makers. This requires drawing more from the right side of the brain, which he calls, R-directed thinking.

Pink goes further to mention R-directed aptitudes, which he believes are the pillars on which tomorrow’s fortunes and individual fulfillment will be build: Design. Story. Symphony. Empathy. Play. Meaning (Pink, 2005: 65).

**Design**

Pink mentions that product offering is no longer about the functionality but design has become an essential element. Consumers are now looking for much more than functionality, but an experience, an emotional connection, beauty and engagement.
Story
Post the information age; information is no longer of value as it is free and abundant. Hence consumers are no longer concerned with the argument but an essence of persuasion, communication, self-understanding, the ability to fashion a compelling story is key in product development.

Symphony
Seeing the bigger picture, crossing boundaries, being able to combine pieces together into an interesting whole more than just focus and specialization.

Empathy
We are now in the world of ubiquitous information and advanced analytical tools, logic is common and it can’t do anything alone. What is essential in this age is the ability to make people tick, forge relationships and care for others.

Play
People are becoming increasingly conscious of health and general well being and are finding ways to integrate play in their work and life.

Meaning
As material wealth has increased greatly, people are now searching for meaning, a sense of purpose, spiritual fulfilment and transcendence as they are liberated from their daily struggles for survival.

2.1.4. Software Development in the 21st Century

In agreement with the above writers, Edward De Bono, after 23 years of promoting and teaching creative-thinking techniques in 45 countries feels that creativity has taken a back seat in the information age. Thierauf (1993) expresses this sentiment and goes further to say “there is need
to rethink the policy of not fixing it until it’s broken”, focusing away from “what is” to “what can be”.

“Software is an industry of vital importance. The industry is large, with over $200 billion in revenues in the United States alone. The software industry in the U.S. employs about 1.5 million people directly, and counting those employed in other types of firms it is estimated that some 2.5 million or more people work as software developers or in software production in some capacity (U.S. DOC 2000). Indeed, software now employs more people than the automobile, computer, semiconductor, and steel industries.”

(Florida, 2003)

Software like other products needs to be adapted to the 21st century market and customised to fit the needs of the emerging market. The industry is under compelling pressure, as people are increasingly depending on technology for day to day activities in they work and personal capacity.

Software is ubiquitous in nature. It affects virtually every sector of the economy, from farming to business services, entertainment, to medical care, software workers work across many sectors of the economy. “In fact, recent research indicates that as many as three-quarters of software developers — the people who do the actual creating of software, as distinct from administrative and support people — work outside “the software industry” per se. Furthermore, other studies indicate that only a minority of software workers actually has degrees in computer science, software engineering, or related disciplines. Thus software appears to be a field in which creativity matters more than credentials.” (Florida, 2003)

Thus to improve the industry and align it with the changing markets of the 21st century, there is need to research on creativity and how it can be used to produce novel and high quality software products that are sustainable and attractive to the emerging market.
In software, however, competitiveness revolves principally around design. Software can be thought of as an “industry of the mind”—where knowledge, intelligence and human creativity are the crucial inputs, and people are the crucial asset. (Florida et al, 2003)

Glass (1995) points out that software construction was primarily a complex problem-solving activity, which required creativity ultimately. Winograd (1996) further mentions that software development is more like art creation and human characteristics, including creativity, should be taken into consideration. Taking these views into consideration, this paper looks at ways to improve the software development process and ensure that it stimulates creativity of those that are involved in the process. It is important however to recognise the budget and time constraints in software development that may hinder creative flow. On the contrary, emphasis on the structure and control is likely to stifle creativity and innovation. This research therefore investigates the cost effective way of integrating creativity with the software development process.
2.2. What is creativity?

The term creativity is used in different ways and the definition is formulated in different respects. Creativity has many synonyms, such as productive thinking, divergent thinking, originality, imagination, and brainstorming. Some definitions of creativity have served as a starting point for theoretical and empirical investigations, but often they are ambiguous or non-operational (Ackoff & Vergara, 1981).

Edmonds and Candy (2002) define creativity as a process toward achieving an outcome recognised as innovative, and describe creative work as involving the generation and evaluation of new ideas, solutions, and artefacts.

Thierauf (1993) says creativity is often defined by synonyms including: productive thinking, divergent thinking, originality, imagination and lateral thinking. Parnes, Noller, and Biondi (2005) define creativity as “the association of thoughts, facts and ideas into a new and relevant configuration, other than has meaning beyond the sum of its part that provides a synergistic effect”.

In the context of Software Development, Couger (1996) has listed creativity definitions that are directly applicable to Information Systems field from various authors. These definitions include “effective surprise”, “the birth of imaginative new ideas”, “the imaginatively gifted recombination of known elements into something new”. Talbot (1997) defines creativity in organizations as: “Making a change that sticks (for a while)”. These definitions emphasise the extent to which creativity is an essential element in software development in the 21st century. In fact, any software that does not offer surprise or an imaginative new idea or both may not sell in the 21st century given the competitive market and the human intelligence (Robinson, 2001).
2.3. Leadership in Creativity

“Today into the 21st Century and beyond, creative agility is essential for managers since many businesses are at crossroads point. Businesses can either go forward or backward.”

(Thierauf, 1993, P.4)

“Executives are not creative enough to meet the challenges facing their companies today or in the future.” (Edward De Bono)

Leadership plays a key role in creativity. Clapham (2000) suggests that leadership may play a key role in encouraging employees to be creative. “There are a variety of leadership styles and while the style preferred by most subordinates is known as transformational leadership, many managers continue to use management by exception, laissez-faire, or contingent reward management styles. The behaviours of these managers may have significant effects on the ability of their software developers to be creative.” (Connelly, 2001)

Howell and Avolio (1993) distinguish the leadership styles into active and passive management, which are differentiated by the timing of the leader’s intervention. “In the more active form of management by exception, the leader continuously monitors followers’ performance to anticipate mistakes before they become a problem and immediately takes corrective action when required. The leader actively searches for problems or any deviations from what is expected. In active management by exception, the leader clarifies the standards at the outset that he or she is using to monitor deviations. In passive management by exception, the leader intervenes with criticism and reproof only after mistakes are made and standards are not met. The leader waits until the task is completed before determining that a problem exists and then brings the problem to the awareness of followers. The leader only clarifies standards after a mistake has occurred” (Howell and Avolio, 1993, p.891).

In software development it may be costly to have a passive leader, but to loosen inhibitions and allow for creativity in a software development company it is important to allow software developers to make mistakes and not only learn valuable lessons but to open developers to possibilities and experimentation that can birth innovative solutions.
Thierauf (1993) defines a creative manager as one who has distinct characteristics that set him of her apart from the less creative colleagues. He or she tends to have more energy, is more inquisitive, and is more responsive to emotions and feelings than the less creative one. Thierauf further explains that they have “flexible repression”, which means that the creative person can bring a lot of material buried in the subconscious to conscious awareness.

Thierauf (1993) outlines a checklist for traits found in a creative manager

<table>
<thead>
<tr>
<th>Has the ability to examine his or her own ideas objectively</th>
</tr>
</thead>
<tbody>
<tr>
<td>Has the constant sensitivity to problems and their solutions</td>
</tr>
<tr>
<td>Has a great initiative</td>
</tr>
<tr>
<td>Has the ability to examine cause and effect relationship</td>
</tr>
<tr>
<td>Has the capability to generate a wide range of ideas</td>
</tr>
<tr>
<td>Can open up to new experiences and abandon traditional defences</td>
</tr>
<tr>
<td>Is not afraid to ask questions that allow for better understanding of the problem</td>
</tr>
<tr>
<td>Likes ventures involving calculated risks</td>
</tr>
<tr>
<td>Believes, even after repeated failures, that the problems can be solved</td>
</tr>
<tr>
<td>Has the confidence to explore new problems and find out new approaches to problem solving</td>
</tr>
<tr>
<td>Is willing to stand when integrity demands it</td>
</tr>
<tr>
<td>Does not blame other or make excuses for mistakes or failures</td>
</tr>
<tr>
<td>Has neither fear or resentment toward authority at all levels of the organisation</td>
</tr>
<tr>
<td>Knows how to use creative memory</td>
</tr>
<tr>
<td>Is willing to allow sufficient time for the incubation period</td>
</tr>
<tr>
<td>Is capable of asserting his or her self confidence</td>
</tr>
<tr>
<td>Has the capability to tolerate ambiguity and complexity</td>
</tr>
<tr>
<td>Is capable of originality of thought</td>
</tr>
<tr>
<td>Has great motivation to succeed</td>
</tr>
<tr>
<td>Is determined to give to finish work even under conditions of frustration</td>
</tr>
<tr>
<td>Trait</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Is willing to give up immediate gain or comfort to reach long range</td>
</tr>
<tr>
<td>goals</td>
</tr>
<tr>
<td>Has a great amount of energy that can be channelled unto productive</td>
</tr>
<tr>
<td>effort</td>
</tr>
<tr>
<td>Perseveres besides obstacles and opposition</td>
</tr>
<tr>
<td>Is open and direct with company personnel and respects their rights</td>
</tr>
<tr>
<td>Wants to examine things from a new viewpoint</td>
</tr>
<tr>
<td>Knows how to inspire and encourage employees</td>
</tr>
<tr>
<td>Is governed by the inner stimuli and has a rising level of aspiration</td>
</tr>
<tr>
<td>Believes that fantasy and day dreaming can be an important part of</td>
</tr>
<tr>
<td>the creative process</td>
</tr>
<tr>
<td>Has an inherent desire and respect for perfection</td>
</tr>
<tr>
<td>Moves toward solutions using intuition</td>
</tr>
<tr>
<td>Knows that getting stuck on a problem is usually caused by asking the</td>
</tr>
<tr>
<td>wrong questions</td>
</tr>
<tr>
<td>Is alert to new perspectives and knows that much depends on the angle</td>
</tr>
<tr>
<td>is seen</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Is willing to listen to suggestions but makes the final judgement</td>
</tr>
<tr>
<td>Tries to think in terms of analogies and metaphors</td>
</tr>
<tr>
<td>Makes use of dreams to separate good ideas from far out ideas</td>
</tr>
<tr>
<td>Makes use of a computer’s power to enhance creativity</td>
</tr>
</tbody>
</table>

Table 2: A checklist of important traits found in a creative manager (Source: Thierauf, 1993, P42-43)
2.4. Teamwork in Creativity

“Teams are now the primary force of organisations. They are worth cultivating at their core. Their core is the mind of each team member” (Kline, 1999, P. 81)

Fischer and Nakakoji (1997) state that though creative individuals are often thought as working in isolation, the role of interaction and collaboration with other individuals is also critical. Software development process usually consists of teams of people such as those who write requirements, developers, testers, installation teams. Teamwork is of crucial importance in this field. Similarly, cognitive scientists and system scientists observe that for human creativity it is important to create a stress-free, stimulating working environment with ample facilities for collaboration and exchange of ideas, and where making mistakes is allowed or even encouraged (Weatherall, 1998; Candy & Edmonds, 2000; Greene, 2002). Collaboration is of key importance as there is co-dependency between the various parties involved and idea must be continuously shared and exchanged at each stage of the software development life cycle.

Candy and Edmonds (2000) say that even though some creative work may be successfully completed individually, collaboration with other parties may provide valuable insights. To get the best out of creative projects, as seen earlier it is important to draw from a diversity of skills and experiences hence teamwork is crucial. To support this, Sheremata (2000) says groups that are integrated to varying degrees, and whose members are from various groups perform most new product development.

According to Austin, (1997), “cognitive processing theory predicts an increase in individual awareness and creativity in moderately novel situations. The different individual perspectives operating within a diverse group may increase the novelty of a situation. This framework suggests that there may be an optimal level of diversity that will stimulate creative thinking within a group, and the relationship between group diversity and creativity may be curvilinear.” (p. 342). In fact, the radicalness, magnitude, and novelty of ideas produced by a team are mostly contingent on the composition of the team (West and Anderson, 1996).
Aside from team heterogeneity, there are a number of other factors that are important for producing creative teams. Group processes are a significant determinant of the creativity of a team (Amabile et al, 1996; West and Anderson, 1996). Team members need to foster many of the behaviours that are also required of leaders. They must encourage their peers to contribute, avoid dictating the manner in which ideas should be presented, and allow their peers to contribute without fear of ridicule. Sheremata (2000) recommends that organizations reduce status and power differentials among organizational players, and hold regular meetings, in order to increase information flow among problem solvers.
2.5. Environment that stimulates creativity

“Thinking time and learning time are both critical to creativity and innovation.”

(Hearthfield, 2000)

“Gates has gone into seclusion for two, one-week “Think Weeks” a year. Family, friends and Microsoft employees are banned from his retreat.” (Hearthfield, 2000)

“By mastering the theory and skill of a thinking environment people do enrich their work, their life and their relationships. Organisations do produce better ideas in less time with better business outcomes. They also increase the motivation and commitment of their work force”

(Kline, 1999)

Environment plays a major role in the creative energy. Even the leader of one of the world leading creative Information Technology companies takes time away to think. In her book “Time to think”, Kline speaks about the ten components of the think environment.

The first component is attention, which Kline defines as “listening with palpable respect and genuine interest, and without interruption”. The quality of other people’s thinking is greatly affected by the attention that is given to them. The answers lie in a person’s own mind, their solutions are hidden in there; all that is needed is the necessary attention and environment to surface those answers.

Encouragement is the second component that she defines as “giving courage to go to the cutting edge of ideas by moving beyond internal competition.” In searching for good ideas, competition has very detrimental effects as it only distracts one from discovering how good they might be by themselves without measuring it against others, which only ensures comparative success. Kline says a Thinking Environment prevents internal competition by using particular processes to set up a wholehearted, unthreatened, selfless search for good ideas.

Appreciation. “Reality is not just the bad; it is also the good. The mind requires an awareness of both in order to work at its best.” (Kline, 1999) A Thinking Environment requires positivity and clear demonstration of appreciation. Kline talks about developing an appreciation ratio of 5:1 of appreciation and criticism.
Equality. "To know you will get your turn to speak makes your attention more genuine and relaxed". (Kline, 1999) It is essential for people in the same environment to treat each other as thinking equals even if there is a hierarchy. All must be valued equally as thinkers hence attention must be given to all. It also makes your speaking more succinct.

Diversity. “The greater the diversity of the group and the greater the welcoming of diverse points of view, the greater the chance of accurate, cutting-edge thinking.” (Kline, 1999) Diversity must be embraced in a thinking environment. Welcoming diverse ideas from a diverse group that brings forth multiple realities facilitates a rich context.

Place.“When the physical environment affirms our importance, we think more clearly and boldly.” (Kline, 1999) She goes on to say that “when our bodies are cared for and respected, our thinking improves.” People attach meaning to the environment; they think better in a place that reflect their personal values.

Feelings. “Fear constricts everything, especially thinking. Crying can make you smarter. After laughter thinking improves.” (Kline, 1999) Although the work environment is usually the last place everyone shows their true feeling, Kline insists that feelings are main ingredients of a thinking environment as they allow sufficient emotional release to restore thinking.

Incisive questions. “A wellspring of good ideas lies just beneath a limiting assumption. An Incisive Question will remove it, freeing the mind to think afresh.” (Kline, 1999) Incisive questioning removes limiting assumptions so that people can think clearly, creatively and dynamically.

Information. “Withholding or denying information results in intellectual vandalism. Facing what you have been denying leads to better thinking.” (Kline, 1999) In a thinking environment, information is very important and must be accurate so that no one has to work with assumptions. It is important to make information readily available.
2.6. Models for creative processes

There are many interpretations and models of a creative process. It is important to understand the most basic creative process to lay a foundation on the subject. Amabile (1983) mentions five stages of creative problem solving depicted on the picture below. First there is the problem presentation stage where a problem or a task is defined, followed by the preparation stage where all the information and solution approaches which are relevant to the problems or task are gathered. The next stage, the solutions is produced and it is referred to as the response generation phase, which is followed by the intense valuation and finally the outcome stage where the best solutions is chosen. She specifically indicates that no acceptable solution is determined, or backtracking to a previous stage occurs.

According to Petty (1997), the Creative Process has six phases; Inspiration where one makes a research and come up with as many ideas as they can, followed by clarification which focuses one on the goals. Then the evaluation phase where one reviews the work and evaluated and learns from it, followed by the distillation phase where one decided on the ideas to take forward, and takes them through the incubation phase where the work is just left alone. The last phase is the perspiration phase where one now determinedly works on the best ideas.

The creative process according to Baer (2006), consists of five stages: Foraging which is about gathering as much information about a problem followed by reflecting which ushers one into
generating a lot of ideas, then the adoption phase which is about selecting the best idea. In the nurturing phase the idea is improved on and one should be determined not to give up, this Baer refers to as Knuckling up.

Shneiderman (2002) mentions four activities involved in a creative process, namely: the collection activity, which consists of gathering information about a subject and learning more about it, then the relation activity, which entails consultation with peers and relevant people. The creation activity is about exploring, composing and evaluating the solutions and is followed by the donation activity, which is about disseminating the results and contributing to different sources. He also mentions that these activities are not necessarily linear and one may always revert back to the earlier stage when required.

Santanen et al. (2002) describe the Cognitive Network Model of creativity. He says Creative ideas emerge from novel juxtapositions in working memory of concepts that had not previously been considered at the same time. He indicates that with the wider spread of ideas people get to think about a problem in a given amount of time, the more likely they are to have a creative idea. The cognitive network model is based on the assumption that knowledge can be represented as highly associated, complex bundles of information. By traversing the links that connect some activated frame to other frames within the network, activation of successive frames spreads through memory causing yet other frames to become primed for subsequent activation. Santanen et al. (2002) state that creativity emerges when two or more frames from areas not typically associated with one another are brought together in the context of the problem at hand.

The benefit of increased variety of concepts is offset by the cognitive cost of switching ideas.
Brainstorming for idea generation is a technique that was developed by Alex Obson in the 1960’s. He suggests that brainstorming attempts to storm a problem quickly by allowing the participants to suggest off the top of their heads, ideas concerning the problem’s solution without allowing criticism.

Graham Wallas (1926) formulated a descriptive model of the creative process for an individual thinker. The process consists of four phases: Preparation, which entails gathering the facts, knowledge, and information relevant to the problem. Then the Incubation phase, where a thorough search and identifying possible ideas or solutions is done, followed by a conscious awareness of new ideas and solutions to the problem, which is called illumination. And finally, the verification stage, which consists of testing for viable or acceptable ideas of solutions to a problem.
Creative Approach to Software Development Life Cycle

Creativity Problem Solving (CPS) Model

The last model that will also be used as a framework for understanding application of creativity in Information Systems is Creativity Problem Solving Model (CPS). This model was derived from the scientific method (Couger, 1996) which included rules for concept formulation, conduct of observation and experiments, and validation of hypothesis by observation and experiments. The unique feature of this model is divergence-convergence activities to the problem solving process. This model consist of five stages from the definition of the problem at hand, compiling all the relevant information about the problem, generating ideas, evaluating and prioritizing them and developing an implementation plan. The diamond behind the stages represents the continuous activity of divergence-convergence in each stage (Couger, 1996). This model will be used as a framework since it allows problems and opportunities to be examined from a solid analytical perspective (Couger, 1996).

![CPS Model](image)

Creativity Web

One of the models studied, is the creativity web (Foster-Pedley (2005) adapted from Stacey (1996)). Stacey (1992) in his book “Creativity and Complexity in an Organization” says organisations are only creative when their individual members learn and interact creatively with each other in groups. He mentions five control parameters that will determine if an organization is at the edge or not, that will push the organization from the stable zone, through the edge of chaos, into the unstable zone.
The first one is the rate of information flow, which if increased will push the organization away from stability. Stacey proposes the use of a shadow system, the “grapevine”, which can retain faster flows of information because it is informal and information is acted upon at local levels. The degree of diversity is the next factor; Stacey proposes that the organization must have diversity enough to provoke learning but not enough to cause anarchy.

The degree of connectivity is also crucial as a few connections tend to bring stability but many bring instability, but Stacey suggests in between there is a critical point where the connections are rich enough but not too rich and can result in endless variety of behavior. Another controlled parameter that Stacey mentions is the level of contained anxiety. He states that when anxiety is avoided altogether by strict adherence to hierarchy or bureaucracy the organization’s shadow system will strictly operate in the stable zone and they will avoid change is every manner.

The last controlled parameter is the degree of power differentials, if it is extreme and permanent in an organization, the members operate under fear, and the organisation’s shadow system is driven deep underground. Yet also on the extreme end, where power is equally distributed and a few people exert what power they have, there will be a power vacuum and it tends to raise anxiety levels and plunges an organisation’s shadow system into a disintegrative state.

Stacey maintains that the old maps are no good because “we are sailing through uncharted waters.” Foster-Pedley (2005) has adopted these five key control parameters and added more parameters that are necessary to push organizations from the stable zone through the edge of chaos, and into the stable zone. He mentions lack of inhibitors, good boundaries, intentionality and watchful anticipation. This is depicted in the figure below that shows the key ingredients of creativity and will be studied closely to determine their application to a Software Development project.
Figure 5: Creativity web to be used as a basis for line of enquiry (Foster-Pedley, 2005; adapted from Stacey (1996))
2.7. Creative Project Management

Eddie Obeng’s book “New Rules for the New World” clearly demonstrated the intense change in projects of the 21st century. In his study, Obeng, (1997) defines the idea of four project types. He clearly demonstrates the importance of identifying the project type that according to the “what and the “how” and use the framework below identify the project type and deal with it accordingly using appropriate skills, techniques, and knowing what to expect. This framework is demonstrated in the diagram below:

<table>
<thead>
<tr>
<th>Specific Objectives Defined</th>
<th>Loosely Defined Objectives</th>
</tr>
</thead>
<tbody>
<tr>
<td>“How” is not known</td>
<td>Quest</td>
</tr>
<tr>
<td>“How” is well known</td>
<td>Painting by numbers</td>
</tr>
<tr>
<td></td>
<td>Movie</td>
</tr>
</tbody>
</table>

The most popular project type, which has been prominent pre the 21st Century is the Painting by numbers project. Which occurs when both the specific objective and the how to do it are clear and well known. Formal project management techniques concentrate on this area. (Obeng, 1997)

A quest occurs when the objective is known but how to go about is not. Many research projects can fit this description especially in the Information Technology field where clients will clearly point out their requirements and the developer has to find the right tools to use to program

The next project is a movies, which occurs when the how is known, but the end-result is not yet clear. “Many arts projects start out this way.” (Obeng, 1997)

A Foggy project occurs when neither the objective nor the solution is clear. This kind of project can be expressed as ‘something’s wrong, but I don’t know what.’ Now this is the most prevalent of all projects in the rapidly changing world, that Obeng defines as the new world.
Project management required for each type of project

Painting by numbers is the classic project management - a highly deterministic approach with clear objectives, milestones and deliverables. In this kind of projects, people are directed by the clearly stated requirements. The project management is purely task based and does not usually require creativity.

On the contrary, although with a Quest, objectives are clearly stated, creativity is required for finding the optimal methodology to use as the “how” is not known. It’s people that determine what to do next and make intuitive leaps to discover the way forward. This is a continuous improvement project.

Movies rely more on individuals’ creativity as it unveils and forms a shape as it continues. It depends on coordination, ambiguity, and making it up as it continues. The plan is the controlling force for providing resources but people provide the finishing and unique touches.

A foggy project is completely different from any project mentioned above. In this case, the “what” and the “how” are unknown and the key for the success of this kind of project is the sticking together, and progressing bit by bit. This is also a radical change projects and it is usually a high-risk project. One needs corporate courage to do it.

These days “Foggy projects” as opposed to Painting by Numbers projects are becoming more common. Because they are hard to handle and they involve engaging one’s emotions, they require leadership with skills and capabilities to manage all change, however open and uncertain. It specifically needs techniques and tools invented for the New World.
3. Software Development Life Cycle

“SDLC, the Systems Development Life Cycle relates to models or methodologies that people use to develop systems, generally computer systems.” (http://en.wikipedia.org/wiki/Systems_Development_Life_Cycle)

The software development lifecycle has been used for decades and has successfully been adapted to different scenarios and project types to allow flexibility and agility in Software Development.

**Stages in the Software Development Life Cycle**

Software life cycle models describe phases of the software cycle and the order in which those phases are executed. It has been adopted and modified to accommodate different scenarios in Software development, but the basic model is shown below:

**THE SDLC WATERFALL**

Small to medium database software projects are generally broken down into six stages:
The waterfall model is the most commonly used methodology for software development. It consists of closely related stages where an output of one stage is an initial input of the next phase. Each stage consists of processes and procedures that end up producing a stage’s deliverable. Below is an elaboration of each phase, in order to clearly understand the current software development process. It is important to note that there is not much variation between the stages of a waterfall model and other Software Development Life Cycles.

**PLANNING STAGE**

The planning stage establishes the intentions on the system, the project definition and the basic project structure. Establishes also the feasibility, and risks of the project and describes the appropriate management and technical approaches.
The management team and the client are key players in this stage. The planning phase establishes clearly defined high-level requirements and the plans forward. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these requirements, also referred to as goals.

The outputs of the project planning stage are the configuration management plan, the quality assurance plan, and the project plan and schedule, with a detailed listing of scheduled activities for the upcoming Requirements stage, and high-level estimates of effort for the out stages.

**REQUIREMENTS DEFINITION STAGE**

The requirements gathering process takes as its input the goals identified in the previous stage in the project plan. Each goal is further refined into a set of one or more requirements.

This is the definition stage for each of the goals and results in the major functions of the system being defined, the operational data areas and reference data areas, and the initial data entities.

Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual description.
Figure 9: Requirements Definition Phase

The output of this stage is the Requirements Document and the Requirements Traceability Matrix (RTM) and the project plan is updated with the finer details of the requirements.

DESIGN STAGE

The design stage takes as its initial input the requirements identified in the stage above and then broken down into design elements. These elaborate on the desired software features in detail, and generally. It is important to note that these design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.
The outputs of the design stage are the design document, an updated RTM, and updated project plan with the new findings.

DEVELOPMENT STAGE

Flowing from above, the design document is now used by the developer to design a prototype. For each design element, a set of one or more software artifacts will be produced. Software artifacts include but are not limited to menus, dialogs, data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artifacts, and an online help system will be developed to guide users in their interactions with the software.

The outputs of the development stage are the working software, the implementation plan, online help, test plan, updated project plan and schedule and an updated RTC.
INTEGRATION & TEST STAGE

Migration from the development environment to a separate test environment is done and the software is ready for thorough testing using the test plan from the previous stage. At this point, all test cases are run to verify the correctness and completeness of the software.

The outputs of this stage include are production initiation plan, that describes reference data and production users, an acceptance plan which contains the final suite of test cases, and once more the project plan is updated.

INSTALLATION & ACCEPTANCE STAGE

This is the final stage where all the testing has been done, and correctness of the system and
completeness confirmed. The software is now installed at the client’s side for further testing by client and acceptance and sign off.

![Diagram of installation and acceptance process]

**Figure 13: Installation and Acceptance**

The outputs of this stage include production software, a completed acceptance test suite, and a customer acceptance memorandum of the software. The Software artifacts and project plan and schedule are archived for the future upgrade.

**Other SDLC Models**

The waterfall model is one of the three most commonly cited lifecycle models. Others include the Spiral model and the Rapid Application Development (RAD) model, often referred to as the Prototyping model, which are not discussed in this paper.

**The limitations of the software development life cycle**

There are several models for developing software that reduce planning risks. All of these models contain the underlying four stages mentioned above, requirements definition, design phase,
implementation and testing. Many Software Development Life Cycles are linear and require the documented completion of a one phase before going on to the next phase and are directed at the satisfaction of the customer’s explicit requirements. This is limited to customer requirements and the developer is not involved early enough to add a creative touch and make recommendations. It is easy to re-invent the wheel where desperation of a customer overrides flexibility and brainstorming by all the stakeholders to come up with the best software solutions.

The SDLC, although it is a great tool for development it may limit the potential and the experimentation that is necessary to come up with a creative solution. Also the failure to be able to revert back to previous stages once one has moved to the next phase makes it stiff and inflexible to accommodate learning and improvement for future work.

The SDLC is a perfect tool for painting by number projects but certainly a weak tool for foggy projects which Obeng (1997), believes are increasingly becoming dominant in the 21st century, hence the SDLC will fail increasingly on emerging projects.
4. Learning Objectives

The need to understand the creative process and to merge it with the Software development process is essential in order to ensure creativity in Software development. Poincare describes the creative process as a fruitful combining which reveals to us unsuspected kinship between facts, long known but wrongly believed to be strangers to one another (Couger, 1996).

Newell, Simon and Shaw in their paper “The Process of Creative Thinking” mention that at least one of the following conditions that must be satisfied for a solution to be creative:

1. The product of the thinking has novelty or value (either for the thinker or for his culture)
2. The thinking is unconventional in the sense that it requires modification or rejection of previously accepted ideas.
3. The thinking requires high motivation and persistence, taking place either over considerable span of time (continuously or intermittently) or at high density.
4. The problem as initially posed was vague and undefined, so that part of the task was to formulate the problem itself.

Couger (1996) extrapolates this further by saying that the two characteristics that appear most in creativity are “uniqueness or newness” and “value or utility”, to which he adds “if the Information Systems profession uses a creative approach that includes those outcomes, substantive results can be expected”. This will be further researched in this topic and techniques that incorporate and tie in the creative process into Software Development will be explored.

Creativity is generally viewed to be abstract, but for creativity to makes sense in the Information Systems field, it has to be made practical as mentioned by Couger (1996). This paper will find practical steps to enrich the Software Development Life Cycle to encompass creativity. In order to do this, the author closely observes the models of creativity and uses the principles extracted from the design of these models in order to achieve this.
This paper looks critically at all the above models to enhance the traditional Software Development Life Cycle, which is believed to be obsolete and insufficient for Software projects in the 21st Century. Couger (1996) shows a very high level of creative development approach on Figure 14 below, where the traditional development approach is enhanced with the use of creativity techniques to come up with a creative development approach.

Figure 14: Widening Development Phases to Consider More Alternatives (Source: Couger, 1996)

This will be the framework for the research. The creativity techniques necessary to enhance the Traditional Development approach are explored using the theory from the creativity models mentioned above.
5. Research Question

“A hypothesis is an unproven statement or proposition about a factor or phenomenon that is of interest to the author”. (Malhorta & Birks, 2003)

The hypothesis is *Traditional Software Development Life Cycle can no longer suffice as a development process for Applications in the 21st Century.*

The author first establishes if there is necessity to merge the creative process with software development process. Then, answer the question of whether the principles about creativity can be applied in software development and what methods or processes can be used to stimulate the developers’ creativity to produce novel and high-quality software products that will be sustainable in the 21st century.

“Until now, most of the studies about creativity in software development are qualitative or prescriptive, which lack of practical evidence to support industries to make any decision to balance the two aspects in software development: creativity and discipline.” Tong (1992). In order to get the best of the research, the impact of creativity in software from both the perspective of the developers and the users was identified. Hence, the author developed questionnaires for both developers and user about the subject of creativity in software development, designed and carried a system thinking methodology to answer the main question.

The author is proposing an improvement on the Software Development Life Cycle what will enable creativity and flexibility to develop sustainable systems in the 21st Century.

5.1. Primary Question

How can the Software Development Life Cycle be enhanced to foster flexibility and creativity in Information Systems?
5.2. Secondary Questions

a. What is the current Software Development methodology and what are its limitations in facilitating development in the 21st century?
b. What creativity practices can be incorporated into the Software Development Life Cycle?
c. What model can best represent a Creative Software Development Process?

5.3. Learning Objectives

The author is passionate about Software Development and has been in the field for more than eight years, the question that intrigues the author is the integration of creativity models in software development as she believes the dynamic, unique, individual, creativity can allow imaginative new ideas to flow that will position software companies in a competitive advantage in the 21st century but may be hindered by the limitation of the Traditional Software Development Life Cycle.
6. Research Methodology

6.1. Research Philosophy and approach

Couger (1996) says little research has been done in this area hence the author has to use comprehensive research methods to incorporate all the elements of both fields of study.

The main objective of this study is to build a new model, which entails developing a theory and adapting it to improve an existing process. Yin (1994) recommends an inductive, qualitative research methodology in this case. The author believes it is very important to gather theory and understand critical parts of the theory about creative processes and the software development methodology. However, a qualitative research methodology is only limited by the imagination of the author (Morse, 1994). Hence a combination of the two methods is proposed in order to get more accurate results.

An approach that combines both quantitative and qualitative methods of data collection and analyses will be used as it takes advantage of the strengths of both quantitative and qualitative methods. (Foster-Pedley, 2005). This will draw from a variety of sources from Software Project Managers to Software Developers using the triangular approach. “Triangulation is an approach to data analysis that synthesizes data from multiple sources. Triangulation seeks to quickly examine existing data to strengthen interpretations and improve policy and programs based on the available evidence”. (http://www.igh.org/triangulation/). By obtaining information using many sources such as a questionnaire and interviews, a better perception and a richer context to work from was created.

A phenomenological study is a study that “attempts to understand an experience from the participant’s point of view” (Leedy & Ormond, 2005:144). “It discloses the essential meaning of human endeavours”. (Bishop & Scudder, 1991:5) Involving Information Technology Specialist, the Software Development Project Managers and software developers in a discussion gives the
author leeway to more meaningful information and experience. Furthermore, to enrich the findings, Information Technology specialists also fill in a survey and contribute to the findings. Using the phenomenological approach will allow the author to “look at multiple perspectives on the same situation” (Leedy & Ormond, 2005:144) and will certainly aid the author in answering the research question.

6.2 Research Design

Given that the primary aim of this research is to improve an existing model, contributions from various stakeholders is critical. Involvement of experienced Information Systems (IS) project managers, software developers and software users is vital for enrichment of the new model.

Leedy & Ormond (2005) say, “Potential sources of data are limited only by the researcher’s open mindedness and creativity”. The fieldwork has therefore been divided into two phases. The first phase aimed at increasing the learning and understanding of creative problem solving and its application in the software development field. The author held two workshops at a conference consisting of Information Technology (IT) specialists from various South African Universities. The aim of the first workshop was to establish the necessity of this study and gain input from these experts about how creativity can be incorporated in IT processes. The second workshop specifically focused on the changing market and its needs for software. It involved interaction with Information Technology Service Support staff at Universities in South Africa to establish their needs in Software as they use a variety of software daily. The Information and Communication Technology Services Staff at the University of Cape Town was also asked to fill in the same survey to establish the software need of the 21st century.

The second phase consists of a discussion with software developers at the workshops and also on a blog found at www.creativesdlc.wordpress.com to discuss how the Software development life cycle can improved to become more creative and accommodate the changing needs from the users.
6.3 Data Analysis

The research will use the System Thinking approach to develop the new model using the finding from the theory, questionnaire and interviews. Systems thinking approach is fundamentally different from traditional forms of analysis. Instead of focusing on the individual pieces of what is being studied, systems thinking focuses on the feedback relationships between the thing being studied and the other parts of the system. It creates a better understanding of the big picture and incorporates all the elements. “The methods of systems thinking provide us with tools for better understanding these difficult management problems. The methods have been used for over thirty years (Forrester 1961) and are now well established.

Systems Thinking as a Set of Tools

The field of systems thinking has generated a broad array of tools that let one to:
(1) Graphically depict one’s understanding of a particular system's structure and behavior,
(2) Communicate with others about this understandings, and
(3) Design high-leverage interventions for problematic system behavior.

These tools include causal loops, behavior over time graphs, stock and flow diagrams, and systems archetypes. These will allow one to depict their understanding of a system, and will help one to test the potential impact of their interventions at critical point.

Affinity Diagrams

An Affinity Diagram is a tool that gathers large amounts of language data such as ideas, opinions, issues into common themes or grouping based on their natural relationships as Haselden (2003) states. It is also referred to as the KJ method because it was developed by KawakitaJiro, a Japanese anthropologist. It allows synthesizing of large amounts of data by finding relationships between ideas. According to Winchip (2001), Kawakita developed the affinity diagram as a method to arrange huge amounts of data gathered from field observations. He further explains that the value of the affinity diagram lies in the fact that its intrinsic ordered method prevents innovative insights from going astray. This method is also believed to be highly
Affinity diagrams can be used in any situation where:

- The solution is not readily apparent.
- One wants to reach a consensus or decision and has a lot of variables to consider, concepts to discuss, ideas to connect, or opinions to incorporate.
- There is a large volume of information to sort through.

Kelly (2000) explains the three steps concerned with creating an affinity diagram, as a team brainstorming tool, as follows:

1. “Writing all things to be organized on sticky paper.”
2. “Silently sorting the sticky notes as a team into related groupings.”
3. “Creating ‘header’ cards by discussing and agreeing on category names.”

**Interrelationship Diagram**

An interrelationship diagram is an analysis tool that allows identification of the cause-and-effect relationships among critical issues. The analysis helps distinguish between issues that serve as drivers and those that are outcomes. This follows the affinity diagram and helps “displays all of the interrelated factors involved” (Anjard, 1995:36) in the affinity diagram. “The interrelationship diagram shows the logical relationships between the factors and in so doing provides for a deeper and more definitive analysis than the affinity diagram alone (Anjard, 1995). The interrelationship diagram provides the optimum base from which to complete the analysis, namely through the use of systems thinking and causal loops diagram.

**Causal Loop Diagram**

Causal loop diagrams (CLDs) provide a language for articulating our understanding of the dynamic, interconnected nature of our world. “CLDs describes systems thinking as wholes,
interrelationships and patterns, rather than static snapshots and in this sense is ideally suited to the research undertaken.” (Senge, 1990) Furthermore, the systems thinking approach is ideally complemented by the use of CLDs (Richardson & Pugh, 1981; Wolstenholme, 1990). “Creating causal loop diagrams is not an end unto itself, but part of a process of articulating and communicating deeper insights about complex issues. It is pointless to begin creating a causal loop diagram without having selected a theme or issue that you wish to understand better.” (Kim, 1992)
7. Synopsis, Analysis and Theory Building

Several methods were used to get data for this project. Two workshops were held by the author at South African Techie Events organised by TENET in Johannesburg. The first event was the annual event organised for the Information Technology specialists from all Universities in South Africa. About seventy candidates attended the thesis presentation. The second was the Information Technology service support event for service support representatives from all the Universities in South Africa. Also with about eighty people attending the presentation, a questionnaire (Appendix 1) was dished out to everyone after the presentation of the thesis. The aim of the questionnaire was to find out if the requirements for software in the 21st century have changed like demands for any other products as predicted by Pink (2003). The questionnaire was specifically sent to Information Technology Specialists who use computer software daily and depend on computers in their daily activities.

The other part of the data analysis entails a discussion with the software developers present at the conference where the author sought to establish the necessary ingredients to build processes that may allow software developers to cater for the changing market demands of the conceptual age and the challenges.

The first presentation at the National Techie Event in South Africa

A presentation was made to about seventy Information Technology Specialists enlightening them on creativity and Creative Problem Solving (CPS) Methods. In order to lay a foundation it was important to establish the relevance of the thesis topic and find out from other Information Technology Specialists if it is deemed necessary to study and incorporate creativity with Information Technology. Hence at the beginning of the presentation the audience was asked whether they are familiar with creativity and creative problem solving methods to which everyone said they are not. They were also asked if they felt creative in their jobs, to which only three out of seventy or 4% responded positively.
At the end of the presentation the questions in Appendix 4 were asked to the audience and statistics recorded:

—*How many feel creative in their jobs?*

3 out of 70 people or 4% of the audience said they felt creative in their jobs.

—*Do you think creativity can add value to your jobs?*

Everyone strongly believed creativity could add value to his or her jobs.

—*What management practices do you think are hindering creativity in your workplaces?*

The following responses were received:

- The majority felt that they worked for organizations that use the top-down approach where everything must come from above to be implemented.
- The Corporate Creativity killer phrases that exist within the culture of the organization. These were mentioned in the presentation adapted from Chic Thompson in his book “What a great idea” where he mentions the phrases that are common in workplaces that kill creativity.
- The management is closed to different perspectives
- The failure to involve all the stakeholders, including the clients or developers when solving problems

—*Do you feel free in your work place to make a mistake?*

59 out of 70 or 84% of the audience felt they did not have the freedom to make mistakes at work. They mostly believed they handle very crucial systems where the penalty of making mistakes was very high.

—*How do you deal with failure in your work environment?*

65 out of 70 or 93% of the audience felt they learn from failure and use it as a stepping-stone for future projects.

—*Is diversity embraced in your workplace? (in its full context)*

Only 10 out of 70 or 14% felt diversity was embraced in their workplaces. They mostly felt their leaders did not know how to embrace difference of opinion or another perspective.

—*Is relationships between various IT departments conducive for creative problem solving?*
They mostly felt the departments have a very negative relationship that did not do justice to the overall vision of the IT faculty. 3 out of 70 or 4% felt there was strong collaboration and good relations between the various departments but the majority felt there was an underlying tension that hindered creative problem solving.

—How is the relationship between IT and the faculties?

All the Universities’ IT departments have the one thing in common which is to serve faculties’ IT needs hence the various faculties on campuses are their major clients. When asked about this crucial relationship, they were mostly negative about this relationship and only 5 out of 70 felt there is smooth collaboration with faculties and the relationships with the various faculties were highly positive, while the rest were very negative.

—Do you have a shared vision that embraces creative solutions?

Only one University had a vision that incorporates creativity

—Do you individually feel your contribution is valuable to the department?

About 66 out of 70 or 94% felt their contribution was valuable to their departments.

—Is creativity a part of your performance goals?

None of them had creativity as a part of their performance goals.

Using the above observations as the basis and confirmation to go forward with the research topic, the author focused the research on the particular aspect of creativity in Software Development.

The first presentation at the Service Support Event in South Africa

This was held in Johannesburg on the 18th November 2008 at a service support event organised by TENET. This event was targeted at Information Technology staff at the various Universities in Southern Africa. About eighty people attended the conference and the author made an opening presentation on the subject of creativity and the changing markets of the 21st Century.

The presentation was focused on the changing markets of the 21st Century and creative problem solving methods that can be used to address the changing market demands.
A questionnaire, in Appendix 1, was sent out to establish whether the software requirements have also changed with the changing business environment. Pink’s view of the changing market demands was used as a basis of the questionnaire to establish how strongly the six pillars of the conceptual age, Play, Design, Symphony, Empathy, Meaning and Story (Pink, 2003) are affecting the software development industry. Over and above the candidates at the conference who filled the survey, also forty people from the Department of Information and Communications Technology Services at the University of Cape Town filled the survey. These were professionals in the IT industry who use computers daily in their work environment hence all the survey participants are well accustomed to the software industry and some are software developers.

The author defines the six pillars of the R-Directed thinking from Pink (2003) in the context of Computer Software and comes up with variables that can be used to best represent the six parameters and centers the questionnaire around these variable.

**Variables**

**Design** – Pink (2003) says that product offering is no longer about the functionality but design has become an essential element. Previously, computer software was all about the functionality of a system and the speed of execution. That is why an Information Technology environment would consist of “nerds” glued to black computer screens with “funny” characters on the screens. But the Graphical User Interface (GUI) emerged and changed the entire computer industry. Now, design of a software system is playing a major role in the software industry and even has its own phase in the software development life cycle. To find out if the market is increasingly demanding beauty in the design, according to Pink, the following variables and question were used in the questionnaire:

**Variables:**  
*Functionality, Design, Graphical/Appeal*

**The survey question:**  
* I would buy software that has all the functionality I need even if the design is not appealing.  
* Beauty in the user interface matters*
Play – Pink (2003) says it is not just seriousness but also play. Software vendors are increasingly integrating games and emoticons in the software systems in order to include an element of play in the software. To find out if this element is important to software users, the following variables were used in the questionnaire.

Variables:  
**Flexibility**

The survey question:  
*Integrate a little bit of play in the software more than just functionality*

Meaning – Pink (2003) says it is not just accumulation but also meaning. People are increasingly looking for meaning and a sense of purpose and emotional engagement. Software such as facebook, myspace and other social networking software have sky-rocketed in the past few years as they facilitate a sense of touch and purpose and an individual space. To find out how important this element is, a variable that is used in the survey is:

Variables:  
**Interactivity, Adaptability**

The survey question:  
*I like to feel intimate with my software as I interact with it*

Symphony - Pink (2003) says it is not longer about focus but also symphony. Focus is essential for software requirements but could users be looking for much more than the requirements, to solve their problems at a larger scale and also the ability to project in the future and see the bigger picture? The variables used to determine these were:

Variables:  
**Robustness, Specificity**

The survey question:  
*I like an element of surprise to my software, more than just sticking to my requirements*

Story - Pink (2003) says it is no longer about argument but also story. This was one of the hardest to integrate with Software Development as each user can define their own meaning of story in this context. Hence the variable used, although it may be perceived as ambiguous as each user can define their own meaning of it.

Variables:  
**User Experience**

The survey question:  
*I want software that gives me experience more than just functionality*
**Empathy** - Pink (2003) says it is not about giving logic but empathy. Software development used to be a very left-directed brain endeavor where it was all about logic and functionality, and it was used purely to perform a given task but over the years this has significantly changed and people interact with software more often than not. Whether it be on their cell phones, on their computers, ipods, DVD players and other digital gadgets, they interact with software significantly and the growing need for empathy may impact of users’ expectation of software systems.

**Variables:**  
*Ease of use, Logical, User-friendliness*

The survey question:  
*For me, user friendliness is as important as functionality*  
*I prefer software that I can customise for my own use more than just standard software*

**Results**

The results from the survey are summarised below. A graph showing the importance placed on each factor when people buy software is depicted below.

![Survey Results](image)

Figure 15: Graph of user requirements from software
From the graph it is clear that functionality of a system is the most important factor in a software system and the user-friendliness and ease of use are the next most important factors that people look at when purchasing a software system. Gone are the days where users just wanted to buy software that works and produces results, but coupled with functionality, they want design, which entails user-friendliness and ease of use. Another important factor in buying software is robustness where more that eighty percent of people feel is it important. With the diversifying market and the software industry that is becoming more complex, users are looking for solutions that are robust.

This is also emphasised by the results below where users had to select statements that are applicable. The majority of users would rather have software that is functional even if the design. But more users believe user friendliness is still just as important as functionality. The results also show increasing sophistication of users, as eighty percent prefer to have software that they can customise and personalise. It is also intriguing to note that more than half of the users want a software that gives them experiences more than just functionality.

<table>
<thead>
<tr>
<th>Survey Question</th>
<th>Yes (%)</th>
<th>No (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>I would buy software that has all the functionality I need even if the design is not appealing.</td>
<td>75% (63)</td>
<td>25% (21)</td>
</tr>
<tr>
<td>For me, user friendliness is as important as functionality</td>
<td>73% (61)</td>
<td>27% (23)</td>
</tr>
<tr>
<td>I prefer a software that I can customize for my own use more than just a standard software</td>
<td>80% (67)</td>
<td>20% (17)</td>
</tr>
<tr>
<td>I want a software that gives me experience more than just functionality</td>
<td>51% (43)</td>
<td>49% (41)</td>
</tr>
<tr>
<td>I like an element of surprise to my software, more than just sticking to my requirements</td>
<td>44% (37)</td>
<td>56% (47)</td>
</tr>
<tr>
<td>Beauty in the user interface matters</td>
<td>36% (30)</td>
<td>64% (54)</td>
</tr>
<tr>
<td>Over and above the functionality of the system, assure me sustainability</td>
<td>86% (71)</td>
<td>14% (12)</td>
</tr>
<tr>
<td>I like to feel intimate with my software as I interact with it</td>
<td>36% (30)</td>
<td>64% (54)</td>
</tr>
<tr>
<td>Integrate a little bit of play in the software more than just functionality</td>
<td>42% (35)</td>
<td>58% (48)</td>
</tr>
</tbody>
</table>

The results of variables used to represent the six- R-directed thinking:
Design –
Variables: *Functionality, Design, Graphical/Appeal*

The survey question: *I would buy software that has all the functionality I need even if the design is not appealing. *Beauty in the user interface matter

Results:
Functionality came out as the most important of user requirements with about 70% saying they consider it to be very important. While Design also has more than 70% saying is important and more than 50% saying graphical design is important. This certainly shows a trend and an inclination to design and beauty of the software. 75% of the people say they would buy software will all functionality even if is was not appealing. 65% say beauty in the user interface does not matter.

Play –
Variables: *Flexibility*

The survey question: *Integrate a little bit of play in the software more than just functionality*

Results:
Flexibility is highly esteemed by the participants and more than half of the participants consider flexibility to be of importance. 42% of the participants say they want play more than just the functionality. This confirms Pink’s view of the increasing need to play more than just being serious.

Meaning –
Variables: *Interactivity, Adaptability*

The survey question: *I like to feel intimate with my software as I interact with it*

Results:
Interactivity and Adaptability are important to more than 50% of the users and although the majority say there is no need for intimacy with the software while interacting, 36% of participants feel that need to be intimate with the software.

**Symphony -**

Variables: *Robustness, Specificity*

The survey question: *I like an element of surprise to my software, more than just sticking to my requirements*

Results:

More people feel that software must offer robustness more than specificity. It seems users are looking for the bigger picture more than just one-dimensional solution that might be obsolete within a short time. 44% say they like the element of surprise in their software, which demonstrate that the developer has really thought out of the box and come up with interesting ideas that the client did not necessarily demand.

**Story -**

Variables: *User Experience*

The survey question: *I want software that gives me experience more than just functionality*

Results:

About 60% consider user experience as a key factor. 51% say over and above functionality, user experience is necessary.

**Empathy -**

Variables: *Ease of use, Logical, User-friendliness*

The survey question: *For me, user friendliness is as important as functionality*

*I prefer software that I can customise for my own use more than just standard software*
Results:
User friendliness and ease of use are the second and third most important factors in choosing software. While logical is still important, it is has less impact proving Pink’s statements that clients are looking for empathy more than just logic. In fact, 80% of the users say they prefer software that they can customise more than just standard software.

Integration with Open-Ended questions

From the above results, the author has established the alignment of Pink’s 6 R-directed thinking with the software market. The six pillars were defined in the software development context and further analysis done with the open-ended questions to extract variables that emerged.

The participants of the survey are “sophisticated users”, as they represent a community of users in the IT industry, of which some are software developers, IT technicians. Based on this, the author was expecting one would expect their demands to be a lot more strict and result oriented but the survey shows that a significant number of participants would still like to see a bit of play in the software they purchase and others still want the emotional attachment to the software system and there is an emerging trend of R-directed demands.

To further define these needs and what they mean to the software industry. The open-ended questions were asked to get depth and meaning from the participant. The first question was for the user to state the software they like using the most and why they like the software. Further more, they still had to fill the information about the software they hate the most and what they hate about that software.

Using the data from the open-ended questions, in Appendix 2. The systems thinking methodology was used to come up with the high intervention points and come to a better understanding of the software user requirements of the 21st century.

The Affinity Diagram

Affinity diagrams can be used to:
* Draw out common themes from a large amount of information.
* Discover previously unseen connections between various ideas or information.
* Brainstorm root causes and solutions to a problem.

From the open-ended questions, the common themes among the responses were identified and categorized into nine key variables showed in the table below.

<table>
<thead>
<tr>
<th>Category - variable</th>
<th>Key Phrases from the questionnaire</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customizability</td>
<td>Customizable</td>
</tr>
<tr>
<td></td>
<td>Allows changes to meet your specific needs</td>
</tr>
<tr>
<td></td>
<td>Modified with different characters and their voices</td>
</tr>
<tr>
<td></td>
<td>Open to further development</td>
</tr>
<tr>
<td></td>
<td>Can add to the software</td>
</tr>
<tr>
<td></td>
<td>Allows choice of functions wanted without having to install everything</td>
</tr>
<tr>
<td>Design</td>
<td>Good design</td>
</tr>
<tr>
<td></td>
<td>Immense quality and scope</td>
</tr>
<tr>
<td></td>
<td>A lot of time spent polishing software</td>
</tr>
<tr>
<td></td>
<td>Great enhancements</td>
</tr>
<tr>
<td></td>
<td>Clean but sexy interface</td>
</tr>
<tr>
<td></td>
<td>Cool</td>
</tr>
<tr>
<td></td>
<td>Good information presentation</td>
</tr>
<tr>
<td></td>
<td>Best graphic interface</td>
</tr>
<tr>
<td></td>
<td>Love the look and feel</td>
</tr>
<tr>
<td></td>
<td>Everything at a click of a button</td>
</tr>
<tr>
<td></td>
<td>Allows changes to the look of the interface</td>
</tr>
<tr>
<td></td>
<td>Makes imaging very easy</td>
</tr>
<tr>
<td>Flexibility</td>
<td>Flexible</td>
</tr>
<tr>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td>GUI interface</td>
<td>Gives all the features of Linux on Windows desktop</td>
</tr>
<tr>
<td></td>
<td>Evolves with technology</td>
</tr>
<tr>
<td></td>
<td>Adaptable</td>
</tr>
<tr>
<td>Functionality</td>
<td>Functional</td>
</tr>
<tr>
<td></td>
<td>Meets business needs</td>
</tr>
<tr>
<td></td>
<td>Manages any business entity</td>
</tr>
<tr>
<td></td>
<td>Best CD/DVD-writing</td>
</tr>
<tr>
<td></td>
<td>Assists in all aspects of business</td>
</tr>
<tr>
<td></td>
<td>Specific</td>
</tr>
<tr>
<td></td>
<td>Does what is needed</td>
</tr>
<tr>
<td></td>
<td>Has all the tools a network analyst might need</td>
</tr>
<tr>
<td>Intelligence</td>
<td>Updates automatically</td>
</tr>
<tr>
<td></td>
<td>Cleans computer regularly</td>
</tr>
<tr>
<td></td>
<td>Measures performance indicators</td>
</tr>
<tr>
<td></td>
<td>Allows individuals to review a strategy against organizational performance</td>
</tr>
<tr>
<td></td>
<td>Ability to communicate and trace tasks</td>
</tr>
<tr>
<td>Interactivity</td>
<td>Doesn’t make assumptions.</td>
</tr>
<tr>
<td></td>
<td>Answers all questions</td>
</tr>
<tr>
<td></td>
<td>Good interactivity</td>
</tr>
<tr>
<td></td>
<td>Allows for playing</td>
</tr>
<tr>
<td></td>
<td>Does what you tell it to do</td>
</tr>
<tr>
<td></td>
<td>Helps the user become more creative</td>
</tr>
<tr>
<td>Reliability</td>
<td>Very fast</td>
</tr>
<tr>
<td></td>
<td>Reliable</td>
</tr>
<tr>
<td></td>
<td>Stable</td>
</tr>
<tr>
<td>Robust</td>
<td></td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Minimal downtime</td>
<td></td>
</tr>
<tr>
<td>Secure</td>
<td></td>
</tr>
<tr>
<td>Safe</td>
<td></td>
</tr>
<tr>
<td>Can be configured and adjusted for each user’s needs</td>
<td></td>
</tr>
<tr>
<td><strong>Synchronization</strong></td>
<td>Easy synchronization</td>
</tr>
<tr>
<td></td>
<td>Has become global standard</td>
</tr>
<tr>
<td></td>
<td>Allows sharing of local and network printers</td>
</tr>
<tr>
<td></td>
<td>Fits well with other applications</td>
</tr>
<tr>
<td><strong>User friendliness</strong></td>
<td>Makes things easier by allowing administrator to go through other’s computers from their own</td>
</tr>
<tr>
<td></td>
<td>Easy to use/ user friendly</td>
</tr>
<tr>
<td></td>
<td>Allows full control</td>
</tr>
<tr>
<td></td>
<td>Needs no sophisticated resources in order to use it</td>
</tr>
<tr>
<td></td>
<td>Straight to the point</td>
</tr>
<tr>
<td></td>
<td>No installation required</td>
</tr>
<tr>
<td></td>
<td>Usable</td>
</tr>
<tr>
<td></td>
<td>Assists users quickly without having to wait in a queue</td>
</tr>
</tbody>
</table>

**Figure 16: Affinity diagram for good software**

The key themes below surfaced from this process:

i. Design – This is one of Pink’s 6 R-Directed thinking pillars and no wonder is emerges again as it is observed that the conceptual age customer wants beauty more than just functionality.

ii. Flexibility – This allows the user the liberty play with the software. Gone are the days of delicate software, which is easily broken by one command from the user.
iii. Functionality – Functionality is a non-negotiable systems requirement. It has become a command and non-functional systems will not stand the test of time.

iv. Intelligence – users are intrigued by software that does much more for them that they asked, they want a software the does things automatically that they should have done manually. Some of the examples given here are automatic updates, automatic cleaning of the machine, ability to communicate and trace tasks.

v. Interactivity – The interaction with the system must be a great experience that the user enjoys. Some of the themes given include the ability to answer all questions, ability to play, and become more creative while one is using the software.

vi. Reliability – in a world that Obeng (1998), calls the world after midnight, where the rate of change is exponential, reliability becomes a key factor in selecting a software. This is due to the increasing dependency on the computer systems, and the uncertainties around technology.

vii. Synchronization – This is another key feature of software in the world of centralization and globalization. Any good software system must have interoperability and be easily connected to other systems.

viii. User friendliness – The ability to learn the system easily and make things easy for the user with readily available help menus and graphical interface is key in software development.

From the above affinity diagram, the interrelationship diagram was formed to determine the interrelationships between the main themes that emerged.

The Interrelationship Diagram
This tool is now going to be used to systematically identify, analyze, and classify the cause and effect relationships that exist among all the key variables above so that key drivers and outcomes can become the focal point of R-Directed software Development.

This tool helped to explore the cause and effect relationships between the variables and allowed the key issues to emerge through the process allowed the author to identify potential key drivers of software development.

The number of outgoing and incoming arrows was counted to identify the drivers and the outcomes. Then the analysis made to determine the number of arrows going in and out of it of each variable and demonstrate with the ratio [out:in].

* Outgoing arrows. A higher number of outgoing arrows indicate an item that is a potential cause or a driver.
* Incoming arrows. A higher number of incoming arrows indicate an item that is a key outcome.
* Neutral variables may have equal number of incoming and outgoing arrows.
Now that from the diagram, it is clear what the drivers are, and the outcomes, and the neutral variables, the author went further to specify the high leverage interventions to determine the main drivers of the software market and the diagram below clearly shows that the main driver is the functionality of the system.

This was done by looking for a variable with the greatest number of outgoing arrows, then placing it at the bottom of the diagram at the highest leverage point followed by the next greatest outgoing arrows, etc, then finding the variable with the greatest incoming arrow and placing it...
right the top of the diagram followed by the next in the opposite direction as shown in the diagram below.

Figure 18: High Leverage Intervention
The greatest driver was functionality. This is an underlying feature of a software system, even in the testing phase of the Software Development Life Cycle, the first test is on the functionality of the system. Also, the important observation is that the main outcome is the intelligent system. Hence a system that has all the mentioned variables and also allows users to customize and synchronize the system is intelligent and is the main requirement of systems in the 21st century.

The Causal loop diagram

The next step was to draw the causal loop diagram. This is going to help understand the interconnections between the variables and allow us to define the software expectations of users in loops that interconnect and target high leverage areas of the software users.
Thinking through the individual variables and determining the negative or positive relationships between them enabled the author to come up with the loops below. The closely related variables were grouped together in order to see the connection in the entire system.

![Causal loop diagram](image)

**Figure 19: Causal loop diagram**

Functionality of a system is affected by many factors from the design, customizability or synchronization. Functionality is not longer about the system executing the requirements of the users, but the minimum requirements of a functional system is customizability which will allow enough flexibility for the user to modify it, and run it on any platform.

A flexible system can also allow for intelligence of the system. Where certain tasks are not pre-empted by the user, but the system knows when and how to perform the tasks without user interaction. They can also be convenient systems, that are secure and self sufficient, e.g. when there is an error in the data entry, it should automatically discard the wrong data without crashing, it should know when to do backups automatically and many other functions that can be automated. Nowadays users want software that can run on all their electronic gadgets e.g. Cell
phones, PDA’s, Laptops and others electronic devices. A flexible system is built on the basis that it is not fixed to one platform or device but it is built for multiple devices and operating systems.

This impacts on the user friendliness of a system, which in turn impacts on reliability and the design of the system. A system may be flexible and intelligent but if the user interface is not appealing or user friendly, then it is inoperable.

This is in agreement with Van Langen (2000)’s three characteristics of a creative software system, which are:
1. Interact with its environment,
2. Learn, and
3. Self-organise (i.e., plan, execute, control, and change its process).

Figure 20: Three requirements for creativity.
When one considers the user requirements that emerge from the Causal Loop Diagram, and compare them with Van Langen’s requirements for a creative system, one can conclude that the nowadays software users are certainly looking for much more than functionality but an element of creativity on the software that they buy.

Van Langen’s three requirements are elaborated below:

**Interactivity:**
He says that a creative system must be able to interact with its environment, for that is the only way, in the long run, to acquire knowledge about the state of the art and current needs and, thus, about what would be new and valuable.
Van Langen (2000) defines an interactive system as a system that is able to interact with its environment when performing a task (such as diagnosis, designing, planning, or scheduling). He goes further to say that an interactive system is able to acquire knowledge through communication, and to act within its environment, for example by asking questions, looking around, visiting external information sources, making drawings or schemas, and discussing opinions and proposals with stakeholders.

**Learning systems**
Furthermore, a creative system must be able to learn, for that is the only way, in the long run, to adapt knowledge acquired through its interactions with the environment and through its own experiences.

A learning system is a system that is able to acquire new knowledge and to ‘forget’ obsolete knowledge. Such a system is able to use its experiences, so that in similar future situations good practices are repeated and poor practices avoided.

**Self-organising systems**
Finally, a creative system must be able to self-organise, for that is the only way, in the long run, to be unpredictable in its operations and, thus, to deliver unexpected results.

A self-organising system is a system that is able to organise and re-organise its own process, when needed, in order to achieve its task and produce good results. Such a system is able to make changes to its internal operations, in response to a changing environment or a problem encountered when trying to achieve its task.

A system’s ability to be creative is essential to finding solutions to problems it may encounter when trying to perform a given task. “A problem is defined as a lack of knowledge to transform a system’s specific state to some desired state” (Thomas et al., 2002). Through communication, observation, and sub-processes such as recall of earlier experiences, analysis, and association (Gabora, 2002; Gero & Kannengiesser, 2002; Santanen et al., 2002), new knowledge can be
generated that enables the mismatch to be solved. To explain why a creative system must satisfy each of these three requirements, consider the following three cases.

First, suppose there is a system that interacts with its environment but that does not learn or self-organise. As a consequence, given a similar task, this system will produce a similar result in the same way. Although this system can solve problems in a conventional sense, this is insufficient for creativity. Even if the result is assessed to be creative the first time round, it won’t necessarily be assessed to be creative in a similar situation a second time round.

Second, suppose there is a system that interacts and learns but that does not self-organise. As a consequence, given a similar task, this system may produce a different result but always in the same way. Although this system can explore, this is insufficient for creativity. Even if a number of results are assessed to be creative, over time the assessor may gradually acquire an understanding of the way that the system works, hence be able to predict the outcome, and thus no longer be surprised by the results.

Third, suppose there is a system that interacts and self-organises, but that does not learn. As a consequence, given a similar task, this system may produce a different result but according to a fixed procedure depending on the organisation of the system. Although this system can adapt its behaviour to a certain extent, this is generally insufficient for creativity. Even if a number of results are assessed to be creative, the system will (tend to) organise itself in a similar way in a similar situation, and thus produce similar results, and hence be predictable.

Among the three loops in the causal loop diagram is the design loop, which according to Pink (2003) is one of the key factors that influence buyers in the conceptual age. The design of a system is of critical importance just as the functionality is, as users prefer to have systems that are easy to use and are appealing. The increasing dependency on computers results in software being the most utilised commodity hence user demands are influenced by the shift in mindset and experience of a given era.
In order to strengthen the findings the research also explores bad software practices. Though an open-ended question in the survey about what software users hate and what are their dislikes about it. The responses received in appendix 2 were categorised and grouped in the affinity diagram below, demonstrating the key variables that came from the phrases used in the responses.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Key Phrases</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design</td>
<td>Badly designed websites</td>
</tr>
<tr>
<td></td>
<td>Lack of polish</td>
</tr>
<tr>
<td></td>
<td>Not written well</td>
</tr>
<tr>
<td></td>
<td>Slow</td>
</tr>
<tr>
<td></td>
<td>Poorly designed</td>
</tr>
<tr>
<td></td>
<td>Old technology</td>
</tr>
<tr>
<td>Interface</td>
<td>Terrible interfaces</td>
</tr>
<tr>
<td></td>
<td>Boring screen layout</td>
</tr>
<tr>
<td>Cost</td>
<td>Costly to operate on a large scale</td>
</tr>
<tr>
<td></td>
<td>Expensive to customize</td>
</tr>
<tr>
<td>Functionality</td>
<td>Functions that fail</td>
</tr>
<tr>
<td></td>
<td>Redundancy</td>
</tr>
<tr>
<td></td>
<td>Uses a lot of memory and CPU</td>
</tr>
<tr>
<td></td>
<td>When database breaks, it’s hard to bring it back online</td>
</tr>
<tr>
<td></td>
<td>Beautiful interfaces but lack required functionality</td>
</tr>
<tr>
<td></td>
<td>Does not remove a lot of viruses</td>
</tr>
<tr>
<td></td>
<td>Limited functionality</td>
</tr>
<tr>
<td></td>
<td>Not SQL-based</td>
</tr>
<tr>
<td></td>
<td>Inconsistent behavior</td>
</tr>
<tr>
<td></td>
<td>Breaks down a lot when used</td>
</tr>
<tr>
<td></td>
<td>Rarely works as it should</td>
</tr>
<tr>
<td></td>
<td>Freezes</td>
</tr>
<tr>
<td></td>
<td>Loops</td>
</tr>
<tr>
<td></td>
<td>Applications that have a beautiful interface but that</td>
</tr>
</tbody>
</table>
The seven variables obtained above namely, functionality, design, interface, interoperability, user-friendliness, cost and restriction match with the variables mentioned in the good software design. The user requirements have surely changed in the twenty-first century, gone are the days when users were just after a working system, they are now better informed and using technology intensely hence they have greater demands on software packages. To further investigate the characteristics of bad software, the interrelationship diagram below was developed from the variables above.

| Figure 21: Affinity diagram for Bad Software |
Figure 22: Interrelationship Diagram for Bad Software

The above themes match with the variables from the best software, and indicate the themes of bad design that cannot be sustained in this day. The key driver is design, which will make the high leverage intervention point in the high leverage intervention diagram below. This implied that design plays key role in defining software system. If the design is bad, even though the functionality is there, the software will still be regarded as bad software and not able to meet the Twenty-First century market.
In order to continue looking in depth at the bad software comments, from the variables obtained, causal loop diagram was below in Figure 24. This is going to help understand the interconnections between the variables and allow us to define the users’ perception of bad software in loops that interconnect.

Looking closely at the variables and the negative or positive relationships between them, the loops below were generated. The closely related variables were grouped together in order to see the connection in the entire system and two main loops emerged which are software design and software robustness.
Software Design:
Software design has been acquiring more importance as the complexity level of software increases. This also drives development teams to be more efficient and more creative in their solutions. Software designers must find new design methodologies, trying to optimise development time, processing time, required memory, and other resources. (Gomes, 2001). The author is in agreements with Gomes as the results of the survey a depicted in the Affinity Diagram shows the increased sophistication of users and their demands hence putting pressure on software developers to optimize resources in order to offer more than just functionality in the software but appealing design that will enhance user experience.

Software Robustness:
“The open source experiment is both new and old. New, in the sense that it is remarkable that large, robust, innovative, complex software systems can be build by geographically disperse teams from across the planet.” (Menzies, 2008) The twenty-first century market requires robust software that can run on any platform and also be dynamic and intelligent. Software that allows even geographically dispersed users to work together on the same project. This is the new trend that is not only adapted by open source but by giants in the software market such as Microsoft and Google. The market today names any software that is not multifaceted and robust as bad.
software.

In conclusion from the survey results above the feedback and analysis from the daily users of software systems and information technology, it has been established that user requirements for software has certainly changed and become more sophisticated in the twenty first century. Pink (2003)’s view of the changing market of the 21st century that requires R-directed aptitudes and are build on Design. Story. Symphony. Empathy. Play. Meaning hold true for the software development industry. Hence software development is challenged to engage processes that will enable R-directed thinking and build on Pink’s six pillars.

The software Development Life Cycle

The software development life cycle is a linear process used in software development, which was founded in the sixties and has over the decades been used as an effective tool for developing software of differing magnitude from a small scale to robust high tech software systems. In order for the author to address the challenge mentioned earlier of the need to engage processes that will encourage R-directed thinking in software development, the main phases of the software development life cycles mentioned in Section 3 are studied closely and ways of integrating creativity in each of these phases using the creativity models and techniques studied in Section 2.

The software development life cycle represents a linear model that cannot be used to cater for the changing needs of the 21st century, the question is how it can be modified to integrate creativity and embrace the creative problem-solving model in order to enrich it and make it an effective tool for developing software that meets the changing market demands:

The foundation of the creative model is based on the model suggested by Couger (1996) as a creative development approach:
In this creative development approach, Couger suggests that in order to integrate creativity in software development, each stage in the software development life cycle needs creativity techniques that will enforce creativity in the entire software development model. The suggested creativity techniques in each phase are mentioned below.

a. Creativity Techniques for Requirements Definition:

The requirements definition stage is a stage where goals are clearly defined and refined further into one or more requirements. Courger (1996) says a creativity model that has proven useful is the Interrogatories technique where the designer find more information by asking the “who?, What?, Where?, When?, Why?, How?” questions to gain more information on the user requirements and also opening up to more alternatives to be considered.

In software development, it is critically important to understand user’s requirements and expectations of the software system. Kline (1999) suggest the methods of listening that engage the client at a deeper level, in order to fully grasp the users’ needs the designer has to develop a ways of communication and environment that will allow both the user and the designer to explore every alternative such that by the time they part, the requirements specification does not only reflect what the user needs, but the best solution for the problem at hand and cater for possible future needs.

The author proposes Amabile’s five stages of creative problem solving depicted on the picture below for this phase. First there is the problem presentation stage where a problem or a task is defined. This is after the interrogatory technique above where questions were asked to clarify
user requirements and now there is clarity on the problem and it may now be defined. This stage is followed by the preparation stage where all the information and solution approaches, which are relevant to the problem or task, are gathered. The next stage, the solutions is produced and it is referred to as the response generation phase, which is followed by the intense valuation and finally the outcome stage where the best solution is chosen.

It is also suggested at this phase to involve all the stakeholders, including the user/s and the programmer/s and leaders to get multiple perspectives that will lead to a creative solution.

*The outcome will be a requirements specification that clarifies the user requirements and a well-thought, creative solution.*

**Creative Technique for Logical Design**

The interrogatories technique may once more be useful in this phase especially the “why?” question, which can be used to find the logic of the system and explore greater alternatives for dividing the logic among modules.

This phase is crucial as seen in the finding above; design and functionality are critical factors of software development process. Hence a lot of thought and attention must be placed on this phase as sustainable and creative outcome of this phase will ensure sustainability and creativity of the final output. The proposed creative problem solving technique is Petty’s Creative Process, which has six phases; Inspiration where one makes a research and come up with as many ideas as they can, followed by clarification, which focuses one on the goals. Then the evaluation phase where one reviews the work, evaluates and learns from it, followed by the distillation phase where one decided on the ideas to take forward, and takes them through the incubation phase where the
work is just left alone. The last phase is the perspiration phase where one now determinedly works on the best ideas.

In the logical design phase, a lot of ideas need to be explored before any implementation can be done. Hence the 6-stage creative process will facilitate a process needed to come up with a lot of ideas from multiple perspectives in order to get to a creative logical design.

**Creative Techniques for physical design**

Courger (1996) proposes a Manipulative Verb technique suggested by Koberg and Bagnall. They have devised verbs to manipulate the problem in order to come up with new perspectives, some of the verbs that would be useful are:

Multiply, distort, divide, rotatem by-pass, eliminate, flatten, add, protect, squeeze, dissect, stretch and many more words. (Courger, 1996)

These words are used in a sentence relevant to the physical design on the system and cause people to think out of the box and come up with phrases about the system using these words. This gives birth to many ideas and the process of converging may be used to select the best ideas.

The Cognitive Network Model of creativity is suggested at this phase, which is based on an assumption that knowledge can be represented as highly associated, complex bundles of information. By traversing the links that connect some activated frame to other frames within the network, activation of successive frames spreads through memory causing yet other frames to become primed for subsequent activation. Santanen et al. state that creativity emerges when two or more frames from areas not typically associated with one another are brought together in the context of the problem at hand. The physical design on software development life cycle integrate two or more different disciplines already, in that the design is a computer program but in most
cases must reflect a particular subject that the computer system is being designed for. An example, software for the field of engineering has to reflect engineering even though designed by IT specialists. The Cognitive Network Model facilitates problem solving of two un-associated frames within the same framework to bring out a creative solution.

**Creativity Techniques for program design**

Courger (1996) says a creativity model that has proven useful is the Interrogatories technique centred around the “why” question to get the underlying reasons for using various programming methods and languages in the development phase. This will allow ideas that will open the programmers to many different solutions that may be used to provide the solution.

The suggested idea for this phase is the Creative Problem Solving technique, which has five stages from the definition of the problem at hand, compiling all the relevant information about the problem, generating ideas, evaluating and prioritizing them and developing an implementation plan incorporating the creative processes of convergence and divergence. This model allows problems and opportunities to be examined from a solid analytical perspective (Couger, 1996).

Although the above creative techniques add value to the four main stages of the Software Development Life Cycle, and facility creative output of each phase, the author is cognisant that even these processes are not enough to ensure creative output. But as noted in this paper there are other crucial elements that enhance creativity in a work place such as leadership, environment, teamwork and corporate culture. But the models above may allow the software development process to produce more creative output.
8. Conclusion

This research attempted to surface the need for creativity in Software Development and as thus build software development model that facilitate and tie in creativity techniques. The papers started with the theories about the 21st century and the challenges it brings in relation to the changing markets with writers such Daniel Pink, Ken Robinson, and Eddie Obeng.

The research question was:

*How can the Software Development Life Cycle be enhanced to foster flexibility and creativity in Information Systems?*

The creativity techniques and models were also discussed in order to understand the creative process. The author first establishes the meaning of these changing markets in the software field by facilitating workshops on the subject and using quantitative methods to find the market needs in software. The paper maintains that traditional software development life cycle is no longer sufficient as a tool for Software Development in the 21st century and established creativity models and techniques that can enhance creativity in software development allowing the methodology that caters for the 21st century markets.

The study unveiled a number of creativity models and techniques that can be adapted in each phase of the software development life cycle in order to achieve the above-mentioned goal. The research was an eye opener to software development disciplinary to draw from and integrate its processes with creativity to better address the increased sophistication of the 21st Century.
9. Limitations of the Study

“One should always try to generalize the research, but within stated limits”. (Easterby-Smith, horpe, Jackson, 2008)

The major limitation of this research is the time factor. It would have been great to test the new model in action and closely investigate if it will enable creativity and make notable difference to the approach to Software Development. But this is the beginning of the process, and the author believes the model can further be examined and improved with more experience and research.
10. Implication of further research

This research focuses on a process. But creativity is more than just a process, it is about people, leadership, work environment, systems. I would suggest a further research on other elements of creativity that can be used in software development.

Further research in this area will entail a stronger integration of creativity in Information Technology as a whole. Creativity tools are crucial in any field and can be used and adopted to improve systems and procedures. Further studies can also be done in creative leadership in Information Technology.
11. Bibliography

Books


Ceserani, J. 2003, “Big Ideas : Putting the Zest into Creativity and Innovation at Work”. Kogan Page


Couger, J.D. 1996, “Creativity and Innovation in Information Systems Organisations” Boyd & Fraser Publishing Co, Danvers, MA


**Journals**


Couger, J.D. 1990, “Ensuring creative approaches in information system design, Managerial and Decision Economics”, Vol 11, p281-295;


Couger J.D. 1998, “Creativity & Innovation Management”, Vol. 7 Issue 1, p51;


Iezzi, T. 2008, “CREATIVITY”, Vol. 16 Issue 6, p3;
Hales, D; Malmborg, L 2008 “Digital Creativity”, Vol. 19 Issue 2, p139-139;


Electronic References


http://digitella.co.uk/brochures/Digitella%20SDLC.pdf

http://www.igh.org/triangulation/

http://humanresources.about.com/od/motivationsucces3/a/learn_read.htm

http://www.greenfields.u-net.com/docs/home.html

http://creativequotations.com/fq-intro.htm

http://www.thesystemsthinker.com/systemsthinkinglearn.html
Appendix 1: Survey Questionnaire

Appendix 2: Survey Results

1. Overall results:

Please specify the rate of importance of the following factors in your software.

<table>
<thead>
<tr>
<th></th>
<th>Very Important</th>
<th>Important</th>
<th>Neutral</th>
<th>Unimportant</th>
<th>Very Unimportant</th>
<th>Response Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functionality</td>
<td>86% (72)</td>
<td>13% (11)</td>
<td>0% (0)</td>
<td>1% (1)</td>
<td>0% (0)</td>
<td>84</td>
</tr>
<tr>
<td>Design</td>
<td>26% (22)</td>
<td>55% (46)</td>
<td>18% (15)</td>
<td>1% (1)</td>
<td>0% (0)</td>
<td>84</td>
</tr>
<tr>
<td>User-friendliness</td>
<td>62% (52)</td>
<td>35% (29)</td>
<td>16% (13)</td>
<td>2% (2)</td>
<td>0% (0)</td>
<td>84</td>
</tr>
<tr>
<td>Robustness</td>
<td>51% (42)</td>
<td>30% (25)</td>
<td>14% (12)</td>
<td>2% (2)</td>
<td>0% (0)</td>
<td>82</td>
</tr>
<tr>
<td>Specificity</td>
<td>30% (25)</td>
<td>37% (31)</td>
<td>29% (24)</td>
<td>4% (3)</td>
<td>0% (0)</td>
<td>83</td>
</tr>
<tr>
<td>Adaptability</td>
<td>33% (28)</td>
<td>50% (42)</td>
<td>14% (12)</td>
<td>2% (2)</td>
<td>0% (0)</td>
<td>84</td>
</tr>
<tr>
<td>Logical</td>
<td>46% (39)</td>
<td>43% (36)</td>
<td>10% (8)</td>
<td>0% (0)</td>
<td>1% (1)</td>
<td>84</td>
</tr>
<tr>
<td>Ease of use</td>
<td>55% (46)</td>
<td>36% (30)</td>
<td>7% (6)</td>
<td>1% (1)</td>
<td>1% (1)</td>
<td>84</td>
</tr>
<tr>
<td>Graphical/App</td>
<td>22% (18)</td>
<td>41% (34)</td>
<td>27% (22)</td>
<td>10% (8)</td>
<td>1% (1)</td>
<td>83</td>
</tr>
<tr>
<td>Interactivity</td>
<td>30% (25)</td>
<td>45% (37)</td>
<td>22% (18)</td>
<td>4% (3)</td>
<td>0% (0)</td>
<td>83</td>
</tr>
<tr>
<td>User Experience</td>
<td>35% (29)</td>
<td>38% (32)</td>
<td>23% (19)</td>
<td>4% (3)</td>
<td>1% (1)</td>
<td>84</td>
</tr>
<tr>
<td>Flexibility</td>
<td>35% (29)</td>
<td>46% (38)</td>
<td>18% (15)</td>
<td>0% (0)</td>
<td>1% (1)</td>
<td>83</td>
</tr>
</tbody>
</table>

2. Please state if you agree with the statements below
<table>
<thead>
<tr>
<th>Statement</th>
<th>Yes</th>
<th>No</th>
<th>Response Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>I would buy software that has all the functionality I need even if the design is not appealing.</td>
<td>75% (63)</td>
<td>25% (21)</td>
<td>84</td>
</tr>
<tr>
<td>For me, user friendliness is as important as functionality</td>
<td>73% (61)</td>
<td>27% (23)</td>
<td>84</td>
</tr>
<tr>
<td>I prefer a software that I can customize for my own use more than just a standard software</td>
<td>80% (67)</td>
<td>20% (17)</td>
<td>84</td>
</tr>
<tr>
<td>I want a software that gives me experience more than just functionality</td>
<td>51% (43)</td>
<td>49% (41)</td>
<td>84</td>
</tr>
<tr>
<td>I like an element of surprise to my software, more than just sticking to my requirements</td>
<td>44% (37)</td>
<td>56% (47)</td>
<td>84</td>
</tr>
<tr>
<td>Beauty in the user interface matter</td>
<td>36% (30)</td>
<td>64% (54)</td>
<td>84</td>
</tr>
<tr>
<td>Over and above the functionality of the system, assure me sustainability</td>
<td>86% (71)</td>
<td>14% (12)</td>
<td>83</td>
</tr>
<tr>
<td>I like to feel intimate with my software as I interact with it</td>
<td>36% (30)</td>
<td>64% (54)</td>
<td>84</td>
</tr>
<tr>
<td>Integrate a little bit of play in the software more than just functionality</td>
<td>42% (35)</td>
<td>58% (48)</td>
<td>83</td>
</tr>
</tbody>
</table>
### Comments from the survey about characteristics of software that users like

Please tell me about the best software you've ever encountered and what you liked about it. (NB. It may be an online system, cellphone software etc). Please elaborate.

**Full Response**

1. Web-based services e.g. "online banking", it's effective, it's efficient most of them are user-friendly and I like them because it saves you time, run your errands at the convenience of your home.

2. Mind Mapping Software. Functional and Flexible

3. HEAT

   Good functionality, meet business needs, scalable, customisable and can manage almost any business entity.

4. Cisco IOS

5. Applications that can be customised by the user and that allow you to make changes to the application to suit your specific requirements. (OpenSource)

6. Cyawin. It gives me all the features of a Linux OS on a Windows desktop. I.e. I am able to have both worlds at the same time.

7. DameWare Software remote is the best for me because I can install/connect software whilst I'm in my office, and I can see the problem without going to the user's office even if he/she is not in the office but the computer should be on so that I can log on using my admin rights.

8. Dameware remote control software. It makes things easier for me because with the permission of the users I can go through their computers and install whatever software they want.

9. Skype. Easy to use, good design, lots of functionality (talking, chatting, ability to transfer files).

10. WordPerfect. It allows you full control, hides nothing.

Skype. Easy to use, small, needs no sophisticated resources.

12. Downloads.com. Very user friendly and gives you a lot of options. You usually find what you are looking for and a few other alternatives.

K3b: best CD/DVD-writing
Firefox: stable and secure.

Spore(Game): Immense/Immersive quality and scope.
Deep Freeze: Just cool.

15. DSTV software is well designed and I like the information presentation and interactive nature of the software.

16. freebsd the OS itself – being able to customise every last option that I wanted

17. GPS software. I cannot remember the exact name but it can be modified with different characters and their voices. This makes a trip so much fun while this software directs you to your destination.

18. old mail system called pegasus mail - straight to the point


21. MS Office Package: assists in all aspects of business.

22. Office 2007

23. Corel WordPerfect: best graphic interface

24. Joomla Content Management System. It comes fully functional but its open to further development. I can customise to what I want. There is a lot of other developers out there to share information with.

25. Techfinity - software I used in 2005 for logging calls and keeping clientele information.

26. Apache, nedi. I love the look and feel. From the first moment you get the idea that the program was written with functionality I mind but a lot of time was spent polishing it.

27. Total commander. Allows tabbed browsing of my file system, easy

29. JRButils, command line utilities to manage Novell edir. Very fast in comparison to other applications and can be configured and adjusted for each user’s needs.

30. Facebook. No installing. Multiple applications. Big integration, everyone is on it which makes software like it usable.

31. Umware: allows for playing.

32. DVDShrink: always works easy; minimal waffling with settings that I may have no idea about.

33. Sun’s VirtualBox: virtual machine software allowing me to run my favourite applications in any environment, be it Windows, Linux or the Mac. Best of all it’s free!!

34. OB2: enjoy working with DBMS.

35. SQL Server: it is the best.

36. CyberlinkPowerDVD: Everything at a click on a button.

37. Microsoft OneNote: It provides a nice look and feel with functionality which mimics doing projects manually. e.g. Notebooks, pages, pencils, etc.

38. Firefox: customisable without losing functionality.

39. I find Accounting software like Pastel, AccPac quite robust though not so user friendly. Functionality and end results are easy.

40. DJMix: It allowed me to change the look of the interface which gave me more exposure to the equipments which are used in entertainment.

41. Windows: Does what is needed and has become global standard.

42. Heat Call Logging System, you can add to the software.

43. None

44. Panition Magic

45. Backtrack (Linux): It has all the tools that a network analyst might need.

46. Nimblex-Linux: no installation necessary just use CD/USB full program usage.

47. Although many of the Vista programs are visually appealing this becomes less important if you use and know previous versions. Relearning is a killer. Humans are
amazingly adaptable they can learn a horrible interface but oddly struggle to learn a reworked one.

48. HP OpenView Service center.

49. SQL Reporting Services

50. VNC

51. PrintShare Software: I like the fact that it allows local and network printers to be shared, but via internet printing which is much safer than general printer sharing. Layout and design, with a combination of great functionality makes this one of the best packages around.

52. McAfee Antivirus: it updates automatically, scanning, cleaning my computer regularly. If I download anything from the Net, it notifies me if there is danger.

53. VNC Remote Control Software: Can assist users quickly without having to wait in a queue.

E-learning Software: Microsoft-based, to equip oneself with the necessary tools.

54. not sure

55. Time-Up Utilities: It has a lot of functionality and you can perform many things on it. It can customize your PC totally and at the same time recommend upgrade while optimising the system and clean-up.

56. OpenSource-GUI-Desktop-ease of use.

57. Dreamweaver and Adobe Pagemaker, WordPerfect, QuattroPro.

58. Balance Scorecard: measuring performance indicators. I like it because it is an ongoing process that helps any individual to review the strategy against organisational performance.

59. Any of the online banking systems. Functional and always offering new features. Minimal downtime.

60. Macrium Reflect Imaging Software. It makes imaging very easy, even when your network is down, you can image the PC without opening the PC case using the flash disk or CD-ROM.

61. The Heat Software program works very well. Can specify when calls put on hold, why calls can't be resolved or closed; type in the reason, also give feedback via email when call is close.
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>62.</td>
<td>The Heat Logging software. The ability to communicate between the technician and customer (user) and trace of tasks.</td>
</tr>
<tr>
<td>63.</td>
<td>Yesterday I went to Audiologes. The software I want to setup and to bring heavy audio is just incredible.</td>
</tr>
<tr>
<td>64.</td>
<td>Outlook 2007: Calendar/Appointments on right side.</td>
</tr>
<tr>
<td>65.</td>
<td>Unicentre Service Wesh: computer associates.</td>
</tr>
<tr>
<td>66.</td>
<td>Joomla: Very customisable, user friendly, GUI interface, very functional and I can choose which functions I want without having to install all the others I don't want.</td>
</tr>
<tr>
<td>67.</td>
<td>SPSS: doesn't make assumptions. Does what you tell it to do.</td>
</tr>
<tr>
<td>68.</td>
<td>Open Source</td>
</tr>
<tr>
<td>69.</td>
<td>DVD Shrink: User friendly, easy to use</td>
</tr>
<tr>
<td>70.</td>
<td>Google Earth</td>
</tr>
<tr>
<td>71.</td>
<td>CS3: it is everything I need. Simple, functional, adaptable, clean interface but still sexy</td>
</tr>
<tr>
<td>72.</td>
<td>CS3 web premium because it is an program to help you to be more creative in order to set up a website.</td>
</tr>
<tr>
<td>73.</td>
<td>Photoshop. Definitely: it's functionality and reliability is unbelievable and you can configure it to meet your design needs specifically.</td>
</tr>
<tr>
<td>74.</td>
<td>Open Office, you can customise it to meet your need.</td>
</tr>
<tr>
<td>75.</td>
<td>Windows 95: It's easily customised desktop and its compatibility.</td>
</tr>
</tbody>
</table>
Comments about software that users hate:

Please tell me about the most annoying or frustrating software you have ever encountered. Please elaborate.

Full Response

1. vista operating system,- it's not user friendly , or maybe the stiff person is me, i think i should familiarize myself with it
2. Complicated or Difficult to configure or buggy software
3. MS-Exchange
   When it works its a excellent, but when the database breaks, its a hell to bring it back on-line. Cannot just use one box without redundance around it which proves to be costly to operate on a large scale.
4. MS Works
5. Applications that have a beautiful interface but that lacks required functionality and is not customizable.
6. RMS(Residence Management System), no automated installation and configuration, software is buggy and cannot be easily migrated, upgraded or maintained. It has a very clunky web interface.
7. SPSS tha is used by the academics is not user friendly.
8. SPSS Software which is used by the academics is not easy to install unless you use a trial version which takes 15 days after that you must reinstall it.
9. Our antivirus software. Very irritating as it does not remove a lot of the viruses. It is like a virus itself.
10. Microsoft Office. Likes to control, does not allow the user to configure documents as they would like.
12. Jaws
13. ITS
14. Novel
15. Novel
14. Nokia PC suite. I had a 9300i Nokia and used the software to backup my entire content. I lost my phone and had to buy a newer version of that phone, E90. The backup created using the previous model cannot be restored on the newer model. I found that annoying as Nokia did not cater for backward compatibility when designing the new replacement software and phone.

15. Usually badly designed websites and websites not being able to accommodate all browsers and all OSs.

16. Don't really have an example, but the most frustrating for me is not knowing how an application works, and when consulting the help file it does not help much. I would then download a similar application with the same functionality.

17. groupwise! functionality - eg you cannot send an email from an archive - you have to un-archive it first and then go look for it in your inbox before you can even start to work on it?

18. ITS at NMMU. Old technology, terrible interfaces, limited functionality.


21. Solutions 6

22. Symantec Antivirus. Comes very restricted. It's all about money.

23. None to mention.

24. HEAT. There are so many do's and don'ts and the software is not user friendly. I mean you are not even allowed to use the X button to close. Buggy and half-assed code is apparent through the entire program.

25. DSRAZOR. The interface and inconsistent behaviour.

26. Netscape Messenger-email product. UCT used it before Groupwise. The product broke down a lot when in use. We had to use in-house scriopsts and fixes to hold it together.

27. Snort IDS, and intrusion detection system that works very well but the installation process and additional requirements makes it very difficult to get a fully working system.

28. Most freeware applications. The lack of polish and the knowledge of the user
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>29.</td>
<td>Heat: not written well.</td>
</tr>
<tr>
<td>30.</td>
<td>Vista: tries to be too clever.</td>
</tr>
<tr>
<td>31.</td>
<td>Heat: Easy to make a balls up-software is klunky.</td>
</tr>
<tr>
<td>32.</td>
<td>Console One or Java for that matter: Not happy with the 5,000,000,000 lines of error code it gives that does not help me troubleshoot in any way.</td>
</tr>
<tr>
<td>33.</td>
<td>Sintrex Monitoring Software: far too complex, not the least bit user friendly and rarely works the way it should.</td>
</tr>
<tr>
<td>34.</td>
<td>Tivoli: Found it very problematic when validating the data.</td>
</tr>
<tr>
<td>35.</td>
<td>IBM Pcomms: Need to remember commands on a GUI, DOS based.</td>
</tr>
<tr>
<td>36.</td>
<td>Linux: after you have worked on Windows, it is not very easy to adapt to the shell.</td>
</tr>
<tr>
<td>37.</td>
<td>Yahoo website: too cluttered and busy.</td>
</tr>
<tr>
<td>38.</td>
<td>Share Point Portal Software. A user does not know what the hell is going on. Even to Net Developers, it is a nightmare.</td>
</tr>
<tr>
<td>39.</td>
<td>Macintosh OS: slow. also don’t like idea of creators claiming other people copy them, but they copied Xerox.</td>
</tr>
<tr>
<td>40.</td>
<td>Software that is released with too many bugs in it.</td>
</tr>
<tr>
<td>41.</td>
<td>Too many to list.</td>
</tr>
<tr>
<td>42.</td>
<td>Alert Monitor in Heat: Keeps popping up.</td>
</tr>
<tr>
<td>43.</td>
<td>Microsoft Outlook.</td>
</tr>
<tr>
<td>44.</td>
<td>Anything from Allock!!</td>
</tr>
<tr>
<td>45.</td>
<td>SAP: still not a real Windows interface even within the software standard buttons vary in size, shape, position and look.</td>
</tr>
<tr>
<td>46.</td>
<td>Crystal Reports.</td>
</tr>
<tr>
<td>47.</td>
<td>Microsoft Great Plains.</td>
</tr>
<tr>
<td>48.</td>
<td>Groupwise</td>
</tr>
</tbody>
</table>
| 49. | Norton Antivirus: you must be connected to Internet via landline to update. It is
only functioning to certain places and depends which connections you have of the site.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>50.</td>
<td>Vista. Hate the idea that it commands you to perform/continue with a duty.</td>
</tr>
<tr>
<td>51.</td>
<td>ITS: not user friendly. Screen layout is very boring. Training is required.</td>
</tr>
<tr>
<td>52.</td>
<td>not sure</td>
</tr>
<tr>
<td>53.</td>
<td>I haven't found anything annoying as I can make any software work for me.</td>
</tr>
<tr>
<td>54.</td>
<td>N/A</td>
</tr>
<tr>
<td>55.</td>
<td>Microsoft Excel.</td>
</tr>
<tr>
<td>56.</td>
<td>Initial software (telephone enquiry system) of Ster-Kinekor. About 5 years ago, did not last long.</td>
</tr>
<tr>
<td>57.</td>
<td>Not yet.</td>
</tr>
<tr>
<td>58.</td>
<td>Vista.</td>
</tr>
<tr>
<td>59.</td>
<td>Not come around such.</td>
</tr>
<tr>
<td>60.</td>
<td>Windows from V3.0 to Vista.</td>
</tr>
<tr>
<td>61.</td>
<td>Freeze, loop.</td>
</tr>
<tr>
<td>62.</td>
<td>F-secure antivirus: doesn't pick up all viruses, has a large footprint i.e. uses a lot of memory, CPU, and Add resources.</td>
</tr>
<tr>
<td>63.</td>
<td>New versions of Office: auto completion, auto formatting, etc. Everything automatic that I didn't ask for.</td>
</tr>
<tr>
<td>64.</td>
<td>Poorly designed, with less functionality</td>
</tr>
<tr>
<td>65.</td>
<td>Not user friendly.</td>
</tr>
<tr>
<td>66.</td>
<td>Webmail and Mind mappers</td>
</tr>
<tr>
<td>67.</td>
<td>Windows Vista. Basically a change that breaks. Functions tend to fail as their basic system requirements can't handle the interface requirements.</td>
</tr>
<tr>
<td>68.</td>
<td>Statistica and Mathematica. Probably because I'm struggling to figure it all out.</td>
</tr>
<tr>
<td>69.</td>
<td>May I mention Phoneman.</td>
</tr>
<tr>
<td>70.</td>
<td>O Exam</td>
</tr>
<tr>
<td>71.</td>
<td>Alert Monitor in Heat.</td>
</tr>
<tr>
<td>72.</td>
<td>Windows Vista</td>
</tr>
</tbody>
</table>
Appendix 4: Questions asked to the audience after the presentation

—How many feel creative in their jobs?
—Do you think creativity can add value to your jobs?
—What management practices do you think are hindering creativity in your workplaces?
—Do you feel free in your work place to make a mistake?
—How do you deal with failure in your work environment?
—Is diversity embraced in your workplace? (in its full context)
—Is relationships between various IT departments conducive for creative problem solving?
—How is the relationship between IT and the faculties?
—Do you have a shared vision that embraces creative solutions?
—Do you individually feel your contribution is valuable to the department?
—Is creativity measured in your performance goals?