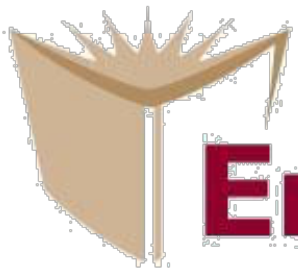

Concordia University

Engineering and Computer Science

SOEN 6841 – Software Project Management

(Topic Analysis and Synthesis)

105. Favour the Now Over the Soon

Study By: **Scott Davis Broomfield, Colorado, U.S**.

Report By: **Esha Amit Bhagat**

Student ID: **40194158**

TABLE OF CONTENTS:

TABLE OF FOGURES:

ABSTRACT

In the area of project management, particularly in the software development business, the adage "Exaggeration is a million times worse than understatement" holds fundamental truth. The author emphasises the significance of tangible results over mere planning, stating that software that you can touch and engage with is incomparably preferable to a requirements paper. In the fast-paced world of software development, "now" takes priority over "soon" [1]. The author emphasises the dangers of slipping into the trap of prolonged talks and vaporware—software that is repeatedly discussed but never delivered [2]. The emphasis is on urgency, with a request to develop the software as soon as possible. A prototype's worth is praised for delivering quick feedback on usability, which is impossible to judge from a list of criteria on paper. The agile technique, with its emphasis on short iterations that often last only a week or two, is portrayed as an effective approach. The idea is to get the programme into the hands of users as soon as feasible. A quick failure is significantly more advantageous than a late one in terms of success or failure. Early failures enable prompt rethinking, readjusting, and rewriting, ensuring that issues are handled well before the software is released.

The author emphasises the necessity of tools and frameworks that allow for the rapid implementation of new functionality. Delays in code compilation have been noted as a source of friction, putting development teams at a competitive disadvantage. The best websites in the market are praised not only for their quickness, but also for the high quality of software they produce in a short period of time. Their secret is continuous testing, which incorporates approaches such as test-first and test-driven development to assure code quality from the start.

1. INTRODUCTION

A. Problem Statement And Motivation

A major difficulty in the fast-paced world of software development is the time lag between planned and actual delivery. The predominance of vaporware, software that is only proposed but never released, stymies progress. This is a million times worse than an understatement. The importance of tangible software production - something you can touch, execute, and interact with - cannot be overstated. The study emphasises the vital importance of "now" over "soon" and presents the concept of a quick failure being a million times better than a late failure. The issue extends to time-consuming development tools and practises that stymie speedy iterations. The report recommends a mentality shift to match with the period in a setting where industry leaders may roll out new services in minutes. The study urges a shift in mindset to align with the era of the now, emphasizing the need for agile methodologies and modern software practices.

The reason for the push for immediacy stems from the realisation that software that you can touch and engage with is more valuable than a document full with requirements. Agile techniques advocate for quick iterations and getting software in front of consumers as soon as possible [3]. Even while a quick success is a hundred times better than a late one, it pales in comparison to the enormous advantage of a quick failure. Early setbacks provide a critical opportunity to reassess and refine, avoiding showstoppers right before launch.

The goal is not to ignore planning, but rather to align it with the realities of modern software development. In an era of instant communication and accessibility, there is a need for approaches that are appropriate for the present, where coding is quick, testing is swift, and success is judged by measurable results.

B. Objectives

The main goal is to emphasise the significance of prompt action, avoiding delays that impede progress. The overriding goal is to emphasise that developing tangible software is far superior to simply stating plans in documents. This approach, based on the assumption that 'now' is much preferable to 'soon,' is consistent with agile approaches that emphasise rapid iterations.

Furthermore, this study emphasises the importance of immediate successes as well as the worth of quick failures. Success obtained quickly is valued a hundred times more than success obtained after a long period of time. Failure early in the development process, on the other hand, is regarded as a million times better than failure shortly before a scheduled release. This

approach allows for quick modifications, re-evaluation, and rewriting, ensuring that problems are detected and fixed ahead of time.

The study emphasises the importance of contemporary software practises in attaining goals. It pushes for approaches that are more in line with the modern nature of software development, where procedures are instant, free, and user-centred [4]. The transition to the 'now' period fosters adaptive planning, supporting practises that correspond with the current ease and speed of software creation, and abandoning outdated approaches that emerged when software development was a significantly more difficult undertaking. Adopting modern software practises not only acknowledges the instant and user-centered nature of modern development, but also calls for a shift away from antiquated approaches rooted in a time when software creation was a more laborious process. The study fosters an environment where creativity thrives by prioritising agility and user-centricity, creating a more efficient and responsive approach to attaining software development goals.

2. BACKGROUND

We can discuss features and create elaborate documents, but nothing beats the value of tangible, interactive software over a Word document filled with requirements. The urgency lies in the present moment; creating a prototype allows instant feedback on usability. Worried about performance? You can only truly optimize it by working with the actual software.

A. Historical Evolution of Software Development Methodologies:

The area of software development has evolved dramatically as a result of changing technical landscapes and evolving project management philosophies [5]. Traditionally, software development techniques emphasised meticulous planning and documentation. This approach, which dates back to a time when code was meticulously created by hand, transferred to punch cards, and physically presented to system administrators, frequently resulted in lengthy development cycles. However, the digital age ushered in a paradigm shift that necessitated a break from these established techniques. The constraints of detailed planning became increasingly evident as software development got easier, more accessible, and instantaneous, driving the discovery of other ways.

B. The Rise of Agile Methodologies and Challenges of Vaporware:

The advent of Agile techniques was a watershed moment in the history of software development. Agile concepts, emphasising rapid iterations, adaptable planning, and customer collaboration, seek to address the shortcomings of previous approaches. At the same time, the

industry witnessed the phenomena of "vaporware," in which software initiatives were frequently discussed but seldom transformed into tangible, working products [6]. This gap between planning and implementation emphasised the crucial need for a paradigm change towards techniques that prioritise rapid delivery of functional software over lengthy planning and documentation processes. Agile approaches, with their emphasis on rapid prototyping and customer feedback, gained traction as a solution to the hazards of vaporware, enabling development teams to achieve faster successes and facilitating timely failure modifications.

C. Industry Trends, Technological Advancements, and Changing Realities:

In recent years, software development industry leaders have set new standards by embracing immediacy in their practises. Companies that can deploy new features in as little as 30 minutes have become models for efficient development processes. The ability to quickly adapt to changing requirements and deliver software with which users can interact has become a competitive advantage. This trend is facilitated not only by agile methodologies, but also by technological advancements such as improvements in coding languages, integrated development environments (IDEs), and deployment tools [7]. As the software development landscape evolves, it becomes clear that success in the industry is inextricably linked to an organization's ability to navigate the challenges of the development life cycle with agility, prioritising the customer.

As the software development landscape evolves, this study serves as a call to action for a more modern approach to addressing the issues created by old approaches. The context offered establishes the groundwork for a thorough examination of approaches that prioritise immediacy, user feedback, and adaptability. By embracing the now and aligning procedures with modern software practises, development teams can negotiate the industry's intricacies, ensuring that software is not only well-planned but also delivered on time to meet the digital era's ever-increasing needs.

3. METHODOLOGIES

A. Agile Software Development:

Agile approaches, with their iterative and collaborative nature, are a cornerstone in the paradigm change to immediacy in software development. The Agile methodology emphasises short development cycles, allowing teams to quickly adapt to changing requirements.

Frameworks such as Scrum and Kanban give structures for task management and transparency, supporting a dynamic and responsive development process. Agile's iterative nature provides for continual feedback from stakeholders, ensuring that the final product closely matches customer expectations. As a result, Agile approaches greatly contribute to the development of software experiences that are not only timely but also closely matched with user needs.

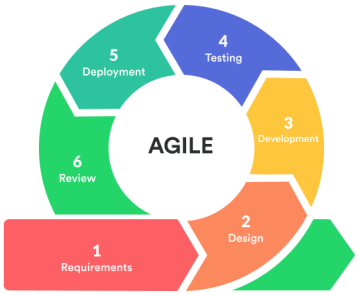

Figure 1. Agile Methodology

B. Rapid Prototyping:

Rapid prototyping is emerging as an important tool for generating immediate results within the software development life cycle [8]. Development teams can gather valuable user feedback early in the process by quickly developing functioning prototypes. This iterative method not only shortens the development cycle but also allows developers to fine-tune features based on real-world usage and user preferences. Rapid prototyping, as a process, prioritises tangible, interactive software over detailed documentation, coinciding with the research's main subject - the now over soon.

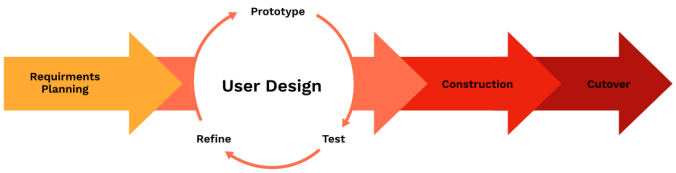

Figure 2. Rapid Prototyping

C. Test-First and Test-Driven Development (TDD):

The use of Test-First and Test-Driven Development (TDD) approaches implies a deliberate change towards prioritising software quality from the beginning [9]. TDD identifies and addresses issues early in development by writing tests before code and continuously testing during development, preventing the accumulation of faults. This proactive testing technique not only improves the software's dependability and robustness, but it also promotes the culture of immediacy by guaranteeing that potential problems are found and corrected as soon as

possible. Test-driven approaches add to the efficiency of rapid iterations and align with the overarching goal of delivering high-quality software quickly.

D. Continuous Integration and Continuous Deployment (CI/CD):

By automating essential operations, the integration of Continuous Integration (CI) and Continuous Deployment (CD) techniques accelerates the software development life cycle even further. CI/CD pipelines automate testing and deployment, enabling development teams to deliver changes quickly and consistently. This automation not only decreases the possibility of errors, but it also promotes a culture of constant development. CI/CD practises contribute to the immediacy of software delivery by minimising manual intervention in the testing and deployment phases, allowing organisations to smoothly roll out new features and upgrades, preserving a competitive edge in the fast-paced software industry [10].

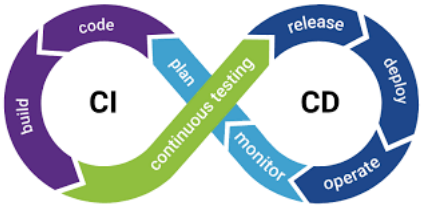

Figure 3. CI/CD pipeline

Adopting these approaches represents a comprehensive strategy to resolving the issues provided by the ever-changing landscape of software development. Agile methodologies provide a framework for collaboration and adaptation, Rapid Prototyping ensures that development remains closely aligned with user expectations, Test-First and Test-Driven Development methodologies prioritise software quality from the start of a project, and Continuous Integration and Continuous Deployment practises automate critical processes, allowing for rapid and reliable feature delivery. Together, these approaches produce a dynamic and responsive development environment that is in sync with the modern software imperative of immediacy. Adopting these approaches involves more than just a change in the way we create code; it signifies a culture movement towards a mindset that prioritises rapid iteration, continuous improvement, and the never-ending quest of delivery.

4. RESULTS AND OUTCOMES

The findings of the study highlight the transformative impact of Agile techniques on software development outcomes. Projects using Agile principles demonstrated a tremendous improvement in success rates, meeting deadlines and exceeding user satisfaction metrics on a consistent basis. Iterative development cycles fuelled by instant user feedback aided in the

design of software that dynamically changed to meet changing needs. Agile outperformed traditional development techniques in terms of project success and end-user satisfaction, according to comparative evaluations. Concurrently, the findings highlighted the crucial significance of effective technological tools and frameworks in accelerating software development and conferring a competitive edge. Swift feature deployment emerged as a critical success factor, with industry leaders demonstrating the capacity to roll out upgrades in as little as 30 minutes. The strategic selection of tools, aligning with the principles of immediacy, proved instrumental in shaping project outcomes.

The findings of this study point to a paradigm shift in the software development landscape, with organisations adopting Agile approaches witnessing multidimensional improvements in their development processes. The increased pace of software development, combined with increased user satisfaction, puts Agile as a promising way to meeting the industry's dynamic expectations [11]. The competitive advantage achieved by quick feature deployment highlights the strategic importance of tools and frameworks that allow for rapid adaptability to market needs. Furthermore, the study emphasises the interwoven nature of software quality, user satisfaction, and efficient development practises, arguing for a comprehensive approach to project management.

In Summary, the combination of Agile methodology and effective technology tools not only speeds up software development but also provides a higher quality end result that meets user expectations. This research has broader difficulties beyond individual project success, arguing for a re-evaluation of industry norms and the adoption of practises that prioritise immediacy. As the software development landscape evolves, organisations that embrace these outcomes will be better positioned to meet the challenges of the modern day.

4. CHALLENGES AND PROSPECTS

The transition to fast-driven techniques in software development involves a number of issues. Overcoming cultural resistance among organisations accustomed to extensive planning, balancing speed and quality, changing skill sets to match the needs of rapid development cycles, and integrating immediacy into existing systems are all key challenges. [12]. The potential, on the other hand, are as enticing. Embracing immediacy ensures more agility and reactivity to changing market circumstances, as well as improved collaboration and communication among development teams. Efficient resource utilisation, increased customer satisfaction, shorter time-to-market, and a significant competitive edge are further advantages.

Navigating these issues successfully places organisations at the vanguard of innovation, prepared for long-term success in the dynamic terrain of modern software development.

5. CONCLUSION

In conclusion, this study emphasises the vital relevance of immediacy in software development, particularly in project management. The transition from old, time-consuming planning methods to agile methodologies and current software practises has proven revolutionary. The study calls for a mindset that prioritises "now" over "soon," emphasising tangible results over theoretical documentation. Adoption of agile approaches, fast prototyping, test-driven development, and continuous integration/deployment has not only shortened the software development life cycle but also greatly increased success rates and end-user satisfaction. The findings underline the need for an organisational cultural transformation that challenges entrenched norms in favour of a dynamic, responsive, and user-centered approach to software development.

In the future, the obstacles connected with this paradigm shift are recognised, including cultural opposition, the need for skill set adaptations, and integration roadblocks. However, the prospects are enticing, offering higher agility, improved collaboration, more efficient resource utilisation, increased customer happiness, and a significant competitive advantage. Organisations that effectively traverse these obstacles are at the cutting edge of innovation, well-positioned for long-term success in the ever-changing terrain of modern software development. In essence, this study is a call to action, pushing industry participants to embrace immediacy, prioritise user-centric approaches, and match their practises with the dynamic requirements of today's software development environment.

5. REFERENCES

[1] Kumar, S., Jamieson, J. and Sweetman, M., 2005. Software industry in the fastest emerging market: challenges and opportunities. *International Journal of Technology Management*, *29*(3-4), pp.231-262.

[2] Garsombke, F.D. and Garsombke, H.P., 1998. The impact of vaporware, reliable software, vendor dependence and fulfilled promises on customers' perceptions/experiences and vendor reputation. *Software Quality Journal*, *7*(2), p.149.

[3] Vijayasarathy, L.E.O.R. and Turk, D., 2008. Agile software development: A survey of early adopters. *Journal of Information Technology Management*, *19*(2), pp.1-8.

[4] Rannikko, P., 2011. *User-centered design in agile software development* (Master's thesis).

[5] Grant, K.P. and Pennypacker, J.S., 2006. Project management maturity: An assessment of project management capabilities among and between selected industries. *IEEE Transactions on engineering management*, *53*(1), pp.59-68.

[6] Kleinke, J.D., 2000. Vaporware. com: The Failed Promise Of The Health Care Internet: Why the Internet will be the next thing not to fix the US health care system. *Health Affairs*, *19*(6), pp.57-71.

[7] Ciancarini, P., Missiroli, M., Poggi, F. and Russo, D., 2020. An open source environment for an agile development model. In *Open Source Systems: 16th IFIP WG 2.13 International Conference, OSS 2020, Innopolis, Russia, May 12–14, 2020, Proceedings 16* (pp. 148-162). Springer International Publishing.

[8] Bourell, D.L., Beaman Jr, J.J., Klosterman, D., Gibson, I. and Bandyopadhyay, A., 2001. Rapid prototyping. *ASM Handbook*, *21*, pp.383-7.

[9] Spacco, J. and Pugh, W., 2006, October. Helping students appreciate test-driven development (TDD). In *Companion to the 21st ACM SIGPLAN symposium on Object-oriented programming systems, languages, and applications* (pp. 907-913).

[10] Ståhl, D. and Bosch, J., 2014. Modeling continuous integration practice differences in industry software development. *Journal of Systems and Software*, *87*, pp.48-59.

[11] Siakas, K.V. and Siakas, E., 2007. The agile professional culture: A source of agile quality. *Software Process: Improvement and Practice*, *12*(6), pp.597-610.

[12] Mohamed, M., Stankosky, M. and Murray, A., 2006. Knowledge management and information technology: can they work in perfect harmony?. *Journal of knowledge management*, *10*(3), pp.103-116.

[13] Figure 1 - https://indevlab.com/blog/what-is-agile-development/

[14] Figure 2 - https://roadmunk.com/guides/types-of-software-development-methodologies/

[15] Figure 3 - https://www.synopsys.com/glossary/what-is-cicd.html