A NEW PROJECT MANAGEMENT FRAMEWORK FOR STARTUPS: AGILE WORKSYSTEMS

Arun Aryal
California State University, Los Angeles

SUMMARY: Agile development methods have become very popular, specifically with software startups. Extant literature does not emphasize the decision-making processes of agile developers, particularly in startups. The goal of this research is to propose a framework where startup developers make these decisions and how those processes align with customer needs.

Introduction

Despite the growth in startups, the Bureau of Labor Statistics claims that more than half fail in the first few years (Mansfield, 2016). Surprisingly, more startups fail for management issues such as bad teams, miscommunication with customers, and incompatible product design than technical issues (Cantamessa et al., 2018), highlighting the need for startups to balance technological issues with management issues. Startups have been adopting a newer project management methodology called Agile, which has become very popular over the last ten years, especially when it comes to rapidly changing business requirements (Giardino et al., 2014).

Models for comparing software development life cycle methods have been well established (Davis et al., 1988), and methods for assessing the personalities of agile developers have been developed (Balijepally et al., 2006). But there has been scant literature focusing on the decision-making processes of agile developers, particularly in analysts working in startups.

In traditional software development approaches, analysts developed detailed requirements to satisfy customer needs, and then programmers implemented those, with little decision-space of their own. Programmer decisions were usually limited to technical implementation details. However, in agile development, programmers are given very high-level, brief descriptions of user needs and are given significant leeway to implement them as they see fit (Anwer et al., 2017). Startups seek to balance the technological and managerial aspects of startup project management (Devadiga, 2017); and startups need a lens through which to examine complex decision-making processes engaged in by developers participating in agile software development teams. The following sections will describe the agile development and examine the Work System Theory as a way of understanding how agile developers make decisions.

Agile Development

Even with the tremendous strides made in software development, as evidenced by the rich array of methods, tools, and techniques, about 19 percent of all software projects are never completed, while another 46 percent are categorized as "challenged." These challenged projects are operational, but over budget, missed deadline, and completed with fewer features and functions than originally specified (Hughes et al., 2016; Hughes et al., 2017). With the growing frustration spurred by the low success rates of software projects, in addition to the imperative for

responsiveness and agility, developers are obliged to think about and invest in an array of development methods that differs appreciably from traditional software practices.

These new methods, labeled Agile Development Methodologies, aim to expeditiously deliver software of high-quality and value to customers by emphasizing the following: (1) collaborative and empowered teams unfettered by rigorous processes; (2) simplicity of design and minimal critical specifications, while documenting only what is absolutely necessary; (3) active involvement of preferably collocated customers; and (4) inevitability of change and an understanding that it may be leveraged through rapid iterations, feedback, and constant reflection on the consequences of actions (Cockburn et al., 2016; Cockburn and Highsmith, 2001). A singularly distinctive feature of these methods is the premium they place on collaborating and self-organizing teams (Cockburn, 2006). The popularity of Agile has been growing for more than ten years. In a survey of software developers conducted during March 2007, 69 percent of respondents indicated that their organizations were using agile methods and another 7.3 percent hinted that they should be going agile within the next year (Boehm, 2007). Since then, Agile methodologies have become widespread in the industry (Bonner et al., 2016).

The goal of agile methods is to help software development organizations to quickly develop and change their products and services, thereby providing the ability to adapt to dynamic market conditions (Boehm, 2002). Agile methods such as Extreme Programming (XP) advocate iterative development, frequent consultation with the customer, small and frequent releases, and rigorously tested code (Ambler, 2002).

Challenges to adopting agile methods

When startups seek to use agile methodologies, they face the following challenges.

1. Development Process-related Challenges

Agile methods value a working system over documents. No formal architecture design is included in a typical agile approach. However, the lack of architectural scalability can raise a serious concern for relatively large projects. Without a formal design phase, many design problems may be ignored (Erickson et al., 2005). For example, agile development teams may make irrecoverable architectural mistakes due to inadequate attention paid to architectural design (Boehm, 2002).

2. Customer-related Challenges

Agile methods rely on inputs from on-site customers instead of predefined requirements documents (Beck and Gamma, 2000). The focus is rather on how to better handle inevitable changes throughout the life cycle than to minimize changes in a project. Agile methods respond to this expectation by adopting strategies designed to reduce the cost of change throughout a project (Cockburn and Highsmith, 2001). The team can obtain immediate feedback and information by closely working with on-site customers. However, customers' insufficient knowledge of the requirements due to the complexity and size of the system poses significant challenges (Cao and Ramesh, 2008). These challenges are even more pronounced when customers are not available or not willing to commit to the project (Fitzgerald et al., 2006).

3. Developer-related Challenges

As agile methods rely heavily on tacit knowledge embodied in development teams, all team members co-locate in the same room and stand-up meetings among team members take place daily, but critical decisions may be left undocumented. There is a lack of formal history of the project for team members to trace and understand the evolution of the system. Communication strategies adopted by agile methods work well for small, highly cohesive teams. However, their use in large, complex projects may result in several challenges. Informal communication may not be effective when dealing with a large number of stakeholders and vast amounts of information (Fitzgerald et al., 2006).

4. Organization/Management-related Challenges

Agile methods recommend decentralized decision-making. Every team member is informed of the progress of the project and is empowered to make decisions on his/ her own. Agile methods work well in organizations that have a flat organizational structure. However, in organizations that are used to deep hierarchical and centralized decision-making structure, they may conflict with the organizational culture, causing resistance between top management and team members (Boehm and Turner, 2003).

5. Agile Work Systems

Agile methods are described as lightweight processes that employ short iterative cycles that actively involve users (Boehm and Turner, 2005). The agile developers' involvement fosters the environment for collaboration to establish, prioritize, and verify requirements. Development process relies on a team's knowledge and collaboration as opposed to documentation. The Agile method must be iterative, incremental, self-organizing, and emergent (Williams, 2010).

Work System Framework and Agile Development

Given these persistent challenges, this study proposes that Agile methodologies, especially in startups, utilize a fundamentally different lens of work system theory (Alter, 2010). Instead of agile being seen as a "development" methodology, it can be viewed as a "work system." A work system is a system in which human participants and/or machines perform work (processes and activities) using information, technology, and other resources to produce specific products/services for specific internal and/or external customers (Alter, 2018). Thus, human participants are viewed as part of a work system rather than as users of a work system. From this definition, it can be seen that in an agile delivery, a team is performing some kind of "work." Applying this work system theory to agile startup teams would bind all involved stakeholders more clearly.

This work system can easily be adapted to agile environments for startup teams. Note that WorkSystem does not require a detailed requirement or any other traditional methodology. In terms of WorkSystem Theory (WST), an agile can be seen as an "agile work system" in which agile developers in startups perform sprints using the information gathered from customers for those customers.

Figure 1:

WorkSystem Framework Components (Alter, 2013; Alter, 2018)

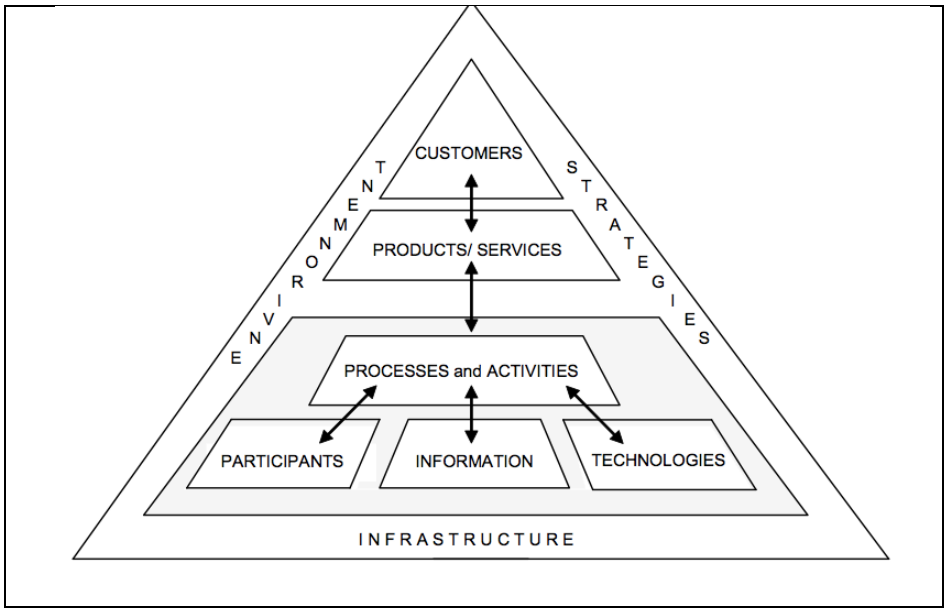


Figure 1 describes the nine main components of the WorkSystem. Table 1 summarizes the work system framework components and provides the description and matches the similar concepts from agile manifesto as discussed in Fowler and Highsmith (2001) as guidelines for startup teams.

Table 1

WorkSystem Components Mapping to Agile Startups

Framework Component	Description and example (adapted from Alter (2013)	Agile Example (Adapted from Agile Manifesto (Fowler and Highsmith, 2001)
Processes and activities	Processes and activities occur in a work system to produce products/services for its customers. Many important work systems perform organized activities that rely heavily on human judgment and improvisation (e.g., Hall & Johnson, 2009; Hill, Yates, Jones, & Kogan, 2006)	Agile processes harness change for the customer's competitive advantage. For startup projects, processes would be customer-centric rather than technology-centric.
Participants	Participants are people who perform work within the work system, including both users and non-users of IT.	Developers and Customers work together in an agile environment. Forces startup teams to view customers as participants.

Information	All work systems use or create information, which in the context of work system analysis is expressed as informational entities that are used, created, captured, transmitted, stored, retrieved, manipulated, updated, displayed, and/or deleted by processes and activities.	The most efficient and effective method of conveying information with and within a development team is face-to-face conversations.
Technology	Almost all significant work systems rely on technology to operate. Work systems are not all about the tools (Alter, 2013)	Deploy all the tools, technologies and processes, but even in agile processes, in the end, it is people who make the difference between success and failure.
Products/ services	Work systems exist to produce things for their customers. Ignoring what a work system produces is equivalent to ignoring its effectiveness. Products/services consist of information, physical things, or actions produced by a work system for the benefit and use of its customers.	The main priority for agile is to satisfy the customer through early and continuous delivery of valuable software. Customer-centric product development would lead to more success for startups.
Customers	Customers are recipients of a work system's products/ services for purposes other than performing work activities within the work system. Since work systems exist to produce products/services for their customers, an analysis of a work system should consider who the customers are, what they want, and how they use whatever the work system produces.	The volatility associated with today's projects demands that customer's value be re-evaluated frequently, and meeting original project plans may not have much bearing on a project's ultimate success.
Environment	The Environment includes the relevant organizational, cultural, competitive, technical, regulatory, and demographic environment within which the work system operates, and that affects the work system's effectiveness and efficiency.	Build projects around motivated individuals, give them the environment and support they need and trust them to get the job done.
Infrastructure	Includes human infrastructure, informational infrastructure, and technical infrastructure	Agile processes promote sustainable development; sponsors, developers, and users should be able to maintain a constant pace indefinitely.
Strategy	In general, strategies at the three levels (work, department, enterprise) should be in alignment, and work system strategies should support department and enterprise strategies.	Giving people a simple set of rules and encouraging their creativity will produce far better outcomes than imposing complex, rigid regulations.

As can be seen from Table 1, the main elements from agile manifesto can be represented in WorkSystems and used by startup teams. Both agile and WorkSystem framework are customer-centric systems. By combining these two approaches, startup teams can manage projects better via agile methodology and bridge the technology-business gap via WorkSystem Theory. Many managers or organizations may feel trepidation in using agile because of the perceived lack of control and processes. However, adapting WorkSystem components in agile would provide some structure while allowing the main elements of agile to function efficiently.

Conclusion and Future Directions

Agile project management methodology has become mainstream over the last decade. Several studies have outlined the benefits of Agile over traditional project management, such as greater customer involvement and better cohesion in teams. In startups, teams need to quickly become acclimated to become business environment and offer customer-centric products. Hence, in addition to a project management methodology, startups need a business/customer management primer that bridges the technology-business gap. This short study proposes WorkSystem framework as a tool for startup agile teams.

This study has a number of limitations. This study does not distinguish different variations of Agile methods such as Scrum, Feature Driven Development (FDD), and Crystal. Similarly, this study applies the generic term *startups* ignoring differences between tech startups and non-tech startups. Future studies could focus on implementing these frameworks in different agile methodologies as well as in various types of startups. The framework illustrates how startups can involve customers during the product development lifecycle. Follow up studies could validate and refine the framework further.

References

Alter, S. 2010. "Work Systems as the Core of the Design Space for Organisational Design and Engineering," *International Journal of Organisational Design and Engineering* (1:1-2), pp. 5-28.

Alter, S. 2013. "Using Work System Theory to Link Managerial and Technical Perspectives on Bpm," *2013 IEEE 15th Conference on Business Informatics*: IEEE, pp. 222-227.

Alter, S. 2018. "System Interaction Theory: Describing Interactions between Work Systems," *Communications of the Association for Information Systems* (42).

Ambler, S. 2002. *Agile Modeling: Effective Practices for Extreme Programming and the Unified Process*. John Wiley & Sons.

Anwer, F., Aftab, S., Waheed, U., and Muhammad, S. S. 2017. "Agile Software Development Models Tdd, Fdd, Dsdm, and Crystal Methods: A Survey," *International journal of multidisciplinary sciences and engineering* (8:2), pp. 1-10.

Balijepally, V., Mahapatra, R., and Nerur, S. P. 2006. "Assessing Personality Profiles of Software Developers in Agile Development Teams," *Communications of the Association for Information Systems* (18:1), p. 4.

Beck, K., and Gamma, E. 2000. *Extreme Programming Explained: Embrace Change*. addison-wesley professional.

Boehm, B. 2002. "Get Ready for Agile Methods, with Care," *Computer*:1), pp. 64-69.

Boehm, B. 2007. "A Survey of Agile Development Methodologies," *Laurie Williams* (45), p. 119.
Boehm, B., and Turner, R. 2003. "Using Risk to Balance Agile and Plan-Driven Methods," *Computer* (36:6), pp. 57-66.
Boehm, B., and Turner, R. 2005. "Management Challenges to Implementing Agile Processes in Traditional Development Organizations," *IEEE software* (22:5), pp. 30-39.
Bonner, N. A., Kulangara, N., Nerur, S., and Teng, J. T. 2016. "An Empirical Investigation of the Perceived Benefits of Agile Methodologies Using an Innovation-Theoretical Model," *Journal of Database Management (JDM)* (27:3), pp. 38-63.
Cantamessa, M., Gatteschi, V., Perboli, G., and Rosano, M. 2018. "Startups' Roads to Failure," *Sustainability* (10:7), p. 2346.
Cao, L., and Ramesh, B. 2008. "Agile Requirements Engineering Practices: An Empirical Study," *IEEE software* (25:1), pp. 60-67.
Cockburn, A. 2006. *Agile Software Development: The Cooperative Game*. Pearson Education.
Cockburn, A., Carlson, D., Gallagher, B., Nidiffer, K., and Sega, R. 2016. "Beyond the Agile Manifesto," *CrossTalk*).
Cockburn, A., and Highsmith, J. 2001. "Agile Software Development: The People Factor," *Computer*:11), pp. 131-133.
Davis, A. M., Bersoff, E. H., and Comer, E. R. 1988. "A Strategy for Comparing Alternative Software Development Life Cycle Models," *Software Engineering, IEEE Transactions on* (14:10), pp. 1453-1461.
Devadiga, N. M. 2017. "Software Engineering Education: Converging with the Startup Industry," *2017 IEEE 30th Conference on Software Engineering Education and Training (CSEE&T)*: IEEE, pp. 192-196.
Erickson, J., Lyytinen, K., and Siau, K. 2005. "Agile Modeling, Agile Software Development, and Extreme Programming: The State of Research," *Journal of Database Management (JDM)* (16:4), pp. 88-100.
Fitzgerald, B., Hartnett, G., and Conboy, K. 2006. "Customising Agile Methods to Software Practices at Intel Shannon," *European Journal of Information Systems* (15:2), pp. 200-213.
Fowler, M., and Highsmith, J. 2001. "The Agile Manifesto," *Software Development* (9:8), pp. 28-35.
Giardino, C., Unterkalmsteiner, M., Paternoster, N., Gorschek, T., and Abrahamsson, P. 2014. "What Do We Know About Software Development in Startups?," *IEEE software* (31:5), pp. 28-32.
Hughes, D. L., Dwivedi, Y. K., Rana, N. P., and Simintiras, A. C. 2016. "Information Systems Project Failure–Analysis of Causal Links Using Interpretive Structural Modelling," *Production Planning & Control* (27:16), pp. 1313-1333.
Hughes, D. L., Rana, N. P., and Simintiras, A. C. 2017. "The Changing Landscape of Is Project Failure: An Examination of the Key Factors," *Journal of Enterprise Information Management* (30:1), pp. 142-165.
Mansfield, M. 2016. "Startup Statistics-the Numbers You Need to Know," *Small Business Trends*).
Williams, L. 2010. "Agile Software Development Methodologies and Practices," in *Advances in Computers*. Elsevier, pp. 1-44.