4-1-2022

What more can software development learn from Agile manufacturing? A roadmap on the 20th anniversary of the Agile manifesto

Akshay Kakar
West Virginia University, akshay.uh@gmail.com

Ashish Kakar
Texas Tech University, ashish.kakar@ttu.edu

Bismita Choudhury
Assam Downtown University, bismitachoudhury@ttu.edu

Adarsh Kumar Kakar
Alabama State University, akakar@alasu.edu

What more can software development learn from Agile manufacturing? A roadmap on the 20th anniversary of the Agile manifesto

Ashish Kakar
Texas Tech University
Ashish.kakar@ttu.edu

Akshay Kakar
West Virginia University
akshay.uh@gmail.com

Bismita Chaudhary
Assam Downtown University
bismitachoudhury@gmail.com

Adarsh Kumar Kakar
Alabama Stare University
akakar@alasu.edu

ABSTRACT

The concept of agility originated in manufacturing and was later adopted by the software development discipline. In this article we argue that in the process some important aspects of the agility theory have been either ignored or misinterpreted. A historical review of the evolving paradigms and practices in software development and manufacturing on the 20th anniversary of the Agile Manifesto (2001) suggests that if the ideas and principles underlying agility are faithfully implemented it would lead to significant improvement in the software development process.

KEYWORDS

Software Development, Manufacturing, Agility

INTRODUCTION

For many decades, software engineering was focused on heavy-weight approaches aimed at success in developing increasingly complex business applications speedily, at lesser costs and of higher quality. Formal methods based on scientific management principles using a variety of tools and techniques for measurement and standardization of the software process were adopted in the belief that it would result in success in software development activities (Kakar and Kakar, 2020). These tools and techniques included Structured Systems Analysis and Design Methodology, Information Engineering, Unified Software Development Process) and OPEN. However, in the late 1990s, as disenchantment with the heavy-weight engineering methods grew, suggestions for improvement came from practitioners culminating in the Agile manifesto (2001)

The Agile Manifesto caught on quickly with the software development community. By 2007 84% of the respondent organizations were using agile methods within their organizations which rose to an impressive 97% by 2018 (Hoda, Salleh and Grundy, 2018). Scholars and practitioners are now working to transfer the success of agile software development methods in other functions and domains, However, in their article Rigby, Sutherland and Takeuchi (2016) noted that "Agile has indisputably transformed software development, and many experts believe it is now poised to expand far beyond IT. Ironically, that's where it began — outside of IT. ". In this article we examine whether this observation is accurate and ask the following questions:

Is Agility as defined in manufacturing (AM) and Agile Software Development (ASD) aligned? What if any are their differences? Is our current understanding of Agility in ASD helpful or is there an opportunity to learn from manufacturing once again? On the occasion of the 20th anniversary of the Agile Manifesto (2001) we find by hindsight that with a right understanding of agility in the context of ASD would have saved us years in its evolution to the present state and would help the Agile movement realize its full potential in the future quickly without much experimentation.

LITERATURE REVIEW

AM is considered the next logical step in the evolutionary chain from craft production to mass production and lean manufacturing. Similarly, ASD is the next logical step to the evolutionary chain from cod-and-fix to plan-driven to lean methods of software development. Further, the change in software development methods have lagged the change in manufacturing paradigms indicating the source of inspiration for software development methods is manufacturing and not the other way around (Table 1). The concept of Agility in software development has also trailed the concept of Agility first introduced in manufacturing (Conboy, 2009):

Manufacturing Paradigms	Software Development Approaches
Craft Production (pre-1910s)	Code and Fix (1950s)
Taylorism and Mass Production (1910s)	Plan-driven approaches such as Waterfall or V Model (1970s)
Lean Manufacturing (1970s)	Lean Software Development (1990s)
Agile Manufacturing (1990s)	Agile Software Development (2000s)

Table 1. Evolution in software development and manufacturing (Kakar, 2000)

However, a deeper review shows that there are some important differences agility in manufacturing and software development. The term agile manufacturing can be traced back to the publication of the report 21st Century Manufacturing Enterprise Strategy (Iococca Institute, 1992). The origins of the "agility movement" stems from US government concerns that domestic defense manufacturing capability would be diminished following the end of cold war in 1989. The following phenomena underscore the reasons for putting agility at the core of manufacturing strategy for the twenty-first century (Goldman et al., 1995):
1. Increasing market fragmentation
2. Growth in the need to produce to order
3. Shrinking product life cycles
4. Globalization of production
5. Distribution infrastructures which support greater customization

AM is an organization level strategy designed to respond quickly to changing customer requirements through mass customization. "It demands a manufacturing system that is able to produce effectively a large variety of products and to be reconfigurable to accommodate changes in the product mix and product designs." (Gunasekaran and Yusuf, 2002). Manufacturing system re-configurability and product variety are critical aspects of agile manufacturing.

Further in AM organizational business processes are integrated with the production process to avoid local optimizations at the expense of agility at global level. "In its fully developed form, agility in manufacturing exemplifies the collaborative capability of an organization to proactively establish virtual manufacturing where a group of independent geographically distributed firms form suitable and temporary alliances based on complementary competencies to address customer/ market needs" (Gunasekaran and Yusuf, 2002).

The word agile was first used in ASD when the 17 participants who huddled together for three days at on February 11-13, 2001, at The Lodge at Snowbird ski resort in the Wasatch mountains of Utah were searching for the right word to use in their manifesto. It was then that the term Agile was suggested by one of the participants who was reading the book "Agile Competitors and Virtual Organizations: Strategies for Enriching the Customer" at that time (Rigby, Sutherland and Takeuchi, 2016).

However, even though the term Agility first used in the context of software development in the Agile manifesto (2001) was derived from manufacturing it was restricted to the project level. This core principle of agile manufacturing of implementing agility beyond the production process to the organization and the supply chain to derive maximum benefits is not so well developed in ASD. Conboy and Morgan (2010) note that ASD has not focused on the role of other stakeholders besides the customer. Further, they argued that a single customer/ user representative on the agile development team is too

narrow a focus to adopt. There was also no mention of sub-contractors, suppliers, service providers and value-added resellers. Therefore, for almost a decade since their introduction agile methods were considered largely restricted to small, co-located development teams, for noncritical system development when compared to the plan-driven heavy weight methods they sought to replace (Conboy, 2009).

AM on the other hand did not make a total break from the past during its evolution. For example, agile manufacturing although advocating organization level flexibility in response to uncertainty in customer/ / supply chain;/ market requirements never abandoned the useful lean manufacturing and tayloristic principles and practices such as the assembly line, common parts, modular design and well-defined production processes. The current trend of integrating the synergistic practices of plan driven and ASD methods as a way forward was already well understood in AM. However, ASD was presented as revolutionary with a total disregard of plan-driven practices that came before it (Boehm and Turner, 2003). The summary of significant differences in the concept of agility in AM and ASD is shown in Table 2.

Agile Manufacturing	Agile Software Development
Agility in Manufacturing is a philosophy not a set of practices (Gunasekaran et al. (2002)	The concept of Agility in software development evolved from a set of practices and was driven by practitioners (Conboy, 2009, Kakar, 2018abcde)
AM is a busines wide context (Goldman and Nagel, 1993)	ASD restricted to software development projects (Conboy and Morgan, 2010; Kakar, 2020)
AM is focused on design (new product development process), production, sourcing, distribution and temporary alliances to meet customer/ supply chain/ market requirements.	ASD focused primarily on production process and activities of software development (Kakar, 2018ab)
The ideal of AM is virtual manufacturing and mass customization	The ideal of ASD is the use of light weight processes for enriching customer and making them more competitive (Kakar, 2017abcde)
AM was an evolutionary concept; did not disown the useful Lean and Mass production methods but further built on them	ASD represented a dichotomic split between agile and every other method that went before and was projected as revolutionary (Boehm and Turner, 2003; Kakar, 2013; Kakar, 2015abc))
The concept of Agility in AM has matured (Conboy, 2009)	The concept of Agility in ASD is still evolving with the research efforts current trend focused to address scalability, global agile development, distributed agile development, Agile-DevOps, Agile automation, automated testing and continuous integration (see Ebert, Gallardo Hernantes, Serrano, 2016; Alqudah and Razali, 2016; Dingsøyr and Lassenius, 2016)

Table 2. Differences in the concept of Agility in Manufacturing and Software Development

FUTURE ROAD MAP: WHAT ASD CAN STILL LEARN FROM AM

Thus, the concept agility when introduced in software development through the Agile Manifesto (2001) was a truncated version and is still in the process of coping up to the full level and scope of Agility in AM from where it originated. Reference disciplines are usually more mature than the software engineering discipline because they have a longer history (Niederman, Gregor, Grover, Lyytinen and Saunders, 2009). There is therefore value in learning more from AM. A deeper understanding of agility in AM would have saved two decades of efforts in reinventing and can help quicken the pace of its future developments in ASD.

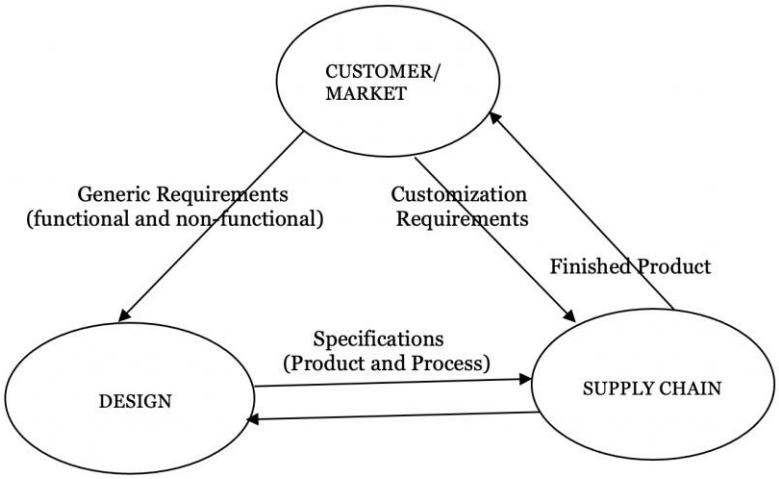

Figure 1. Schematic of agility in AM

Agility in manufacturing deploys structured and unstructured upstream and downstream processes for product design and production (see Figures 1). The structured processes include practices for concurrent engineering, mass customization, product portfolio management and supply chain management, The structured processes are backed by an organization level culture promoting internal and external collaboration; cross-functional communication, coordination, and knowledge sharing; customer/ market focus. Further, an environment is created at team level for enhancing cohesion, reflexivity, self-organization, and conflict resolution in work groups. Agile organizations recognize the value of both organic and mechanistic structures in managing uncertainty in customer demand and turbulence in the competitive landscape by providing variety in products and services aligned with the organization's strategic goals. The goal of AM is to design, manufacture, distribute, sell and service a variety of products at low cost and high quality so that customers find exactly what they want and reap the benefits of customization.

This broad approach to agility in AM is relevant for ASD and provides a future roadmap for ASD. Although, attempts have been made in some of these areas in recent years, yet a comprehensive approach is needed to reap benefits at the organization level whether the organization is a software development firm or an internal MIS department. For example, although most literature on ASD have tended to be written with the overriding assumption that the projects are managed as single projects, it does not reflect the real-life situation as project boundaries are pliable internally within the organization as well as external to the organization (see Figure 2). Focusing solely on individual project performance is suboptimal as organization manages a basket of projects each having different priorities within budgeted resources. The projects include development projects, deployment projects and maintenance projects. New projects are continually added to the basket and existing projects prematurely discontinued or retired in alignment with the strategic goals of the organization. Just as an individual project fulfills customer/ supply chain/ market requirements by building them into a software product to provide value to the customer, a software organization fulfills its business goals through its products and services using portfolio management to maximize the business value for the organization.

Further, as Conboy and Morgan (2010) noted a decade earlier, ASD has not focused on the role of other stakeholders besides the customer. They argued that a single customer/ user representative on the agile development team is too narrow a focus to adopt. There was also no mention of sub-contractors, suppliers, service providers and value-added resellers. This lacuna continues to persist today despite other developments in ASD. The root of this problem can be traced to the misinterpretation that has prevailed about agility the context of software development. Lean and agile manufacturing focus on creating processes at the level of supply chain for rapid mass customization of products through modularization and late differentiation (Huh, 2001). The ultimate goal is flexibility in meeting the needs and desires of individual customers at low cost and high quality. This ideal of Agility is stated evocatively by Toyota where Toyota visualizes its ideal plant as "one where a Toyota customer could drive up to a shipping dock, ask for a customized product or service, and get it at once at the lowest possible price and with no defects.

To the extent that a Toyota plant or a Toyota worker's activity falls short of this ideal, that shortcoming is a source of creative tension for further improvement efforts" (Spear and Bowen, 1999). Figure 2 provides guidance and roadmap for agility through supply chain focus and virtual manufacturing in the software development context.

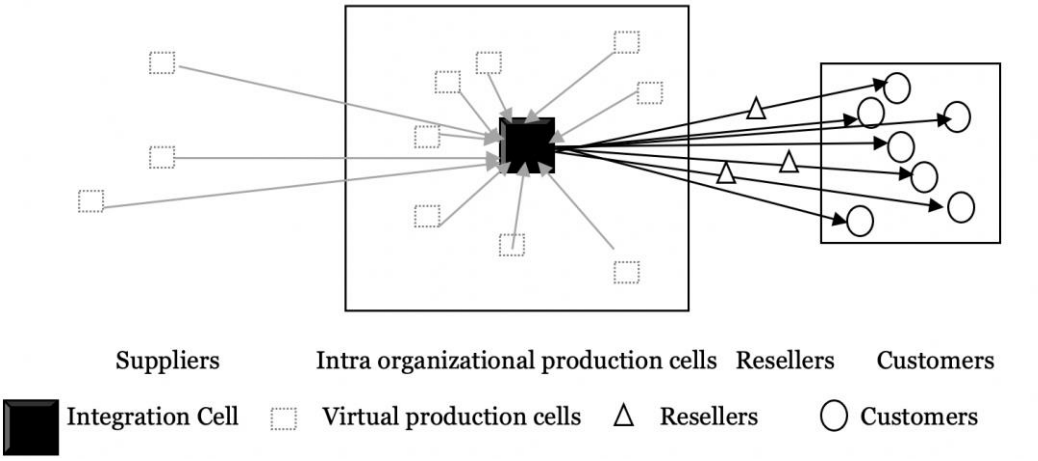

Figure 2. Schematic for agility in software development

CONCLUSION

On the occasion of the 20th anniversary of the Agile Manifesto (2001) we find by hindsight that with a deeper understanding of agility in AM would have saved us years in its evolution to the present state. A correct interpretation of Agility would help the ASD realize its full potential in the future quickly without reinventing the wheel and without much experimentation. The main obstacle is the narrow project focus in ASD. Lean and agile manufacturing focus on creating processes at the level of supply chain for rapid mass customization of products through modularization and late differentiation. The ultimate goal is flexibility in meeting the needs and desires of individual customers at low cost and high quality. Agile Software development does not even talk about mass customization as a goal. Until that is done and the agile processes to accomplish that is understood, agility may not find its full expression in software develoopment and efforts in the area of hybrid methods, global software development will either fail or produce sub-optimal results or achieve maturity through trial and error after another long struggle and time. Interpreting the concept agility correctly in the context of software development can accelerate improvements.

REFERENCES

Alqudah, M., & Razali, R. (2016). A review of scaling agile methods in large software development. *International Journal on Advanced Science, Engineering and Information Technology*, *6*(6), 828-837

Conboy, K. (2009). "Agility from first principles: reconstructing the concept of agility in information systems development," *Information Systems Research* (20:3), pp. 329–354.

Conboy, K. and Morgan, L. (2010). "Future research in agile systems development: applying open innovation principles within the agile organization," In *Agile Software Development* pp.223-235.

Dingsoyer, T., & Lessensius, C. (2016). Emerging themes in agile software development: Introduction to the special section on continuous value delivery. *Information and Software Technology*, *77*, 56-60.

Goldman, S.L., Nagel, R.N. and Preiss, K. (1995). Agile Competitors and Virtual Organizations: Strategies for Enriching the Customer. Van Nostrand Reinhold, New York, NY, USA.

Gunasekaran, A. and Yusuf, Y.Y. (2002}. "Agile manufacturing: a taxonomy of strategic and technological imperatives," *International Journal of Production Research* (40:6), pp. 1357–1385.

Hoda, R., Salleh, N. and Grundy, J. (2018). The rise and evolution of agile software development. *IEEE software*, *35*(5), 58-63.

Hofmeister, C., Nord, R. L. and Soni, D. (2000). *Applied Software Architecture*, Addison-Wesley.

Huh, W. T. (2001). "Software process improvement: operations perspectives," In *Management of Engineering and Technology, PICMET'01. Portland International Conference* (1), pp. 428-429.

Iococca Institute. (1992). *21 st Century manufacturing strategy*, Lehigh University, Bethlehem, PA.

Kakar, A. K. (2014). When form and function combine: Hedonizing business information systems for enhanced ease of use. In *2014 47th Hawaii International Conference on System Sciences* (pp. 432-441). IEEE.

Kakar, A. K. (2015a). Why do users speak more positively about Mac OS X but are more loyal to Windows 7?. *Computers in Human Behavior*, *44*, 166-173.

Kakar, A. K. (2015b). Investigating the penalty reward calculus of software users and its impact on requirements prioritization. *Information and Software Technology*, *65*, 56-68.

Kakar, A. K. (2015c). Software product features: should we focus on the attractive or the important?. *Journal of Decision Systems*, *24*(4), 449-469.

Kakar, A. K. (2017a). Do reflexive software development teams perform better?. *Business & information systems engineering*, *59*(5), 347-359.

Kakar, A. K. (2017b). Investigating the prevalence and performance correlates of vertical versus shared leadership in emergent software development teams. *Information Systems Management*, *34*(2), 172-184.

Kakar, A. K. (2017c). Assessing self-organization in agile software development teams. *Journal of computer information systems*, *57*(3), 208-217.

Kakar, A. K. S. (2017d). Why do users prefer the hedonic but choose the Utilitarian? Investigating user dilemma of hedonic-utilitarian choice. *INTERNATIONAL JOURNAL OF HUMAN-COMPUTER STUDIES*, *108*, 50-61.

Kakar, A. K. (2017e). Investigating the relationships between the use contexts, user perceived values, and loyalty to a software product. *ACM Transactions on Management Information Systems (TMIS)*, *8*(1), 1-23.

Kakar, A. K., & Kakar, A. (2018a). GTTA 2: AN EXTENSION OF THE GENERAL THEORY OF TECHNOLOGY ADOPTION (GTTA), *Southern Association of Information Systems*.

Kakar, A. K., & Kakar, A. (2018b). GTTA 3: AN EXTENSION OF THE GENERAL THEORY OF TECHNOLOGY ADOPTION (GTTA), *Southern Association of Information Systems*.

Kakar, A. K. (2018c). How does the value provided by a software product and users' psychological needs interact to impact user loyalty. *Information and Software Technology*, *97*, 135-145.

Kakar, A., & Kakar, A. K. (2018d). Assessing Shopper's Penalty Reward Calculus in Online versus Instore Shopping. *e-Service Journal*, *10*(3), 24-45.

Kakar, A. K., & Kakar, A. (2018e). IS THE TIME RIPE FOR BRANDING OF SOFTWARE PRODUCTS. *Southern Association of Information Systems*.

Kakar, A. K. S. (2018a). Engendering cohesive software development teams: Should we focus on interdependence or autonomy?. *International Journal of Human-Computer Studies*, *111*, 1-11.

Kakar, A., & kumar Kakar, A. (2018b). Is team cohesion a double edged sword for promoting innovation in software development projects?. *Pacific Asia Journal of the Association for Information Systems*, *10*(4).

Kakar, A. and Kakar, A. (2020). A Brief History of Software Development and Manufacturing. In *Proceedings of the Southern Association for Information Systems Conference*).

Kakar, A. K. (2020). A Theory of Effectiveness of Agile Software Development. AMCIS (2020) Virtual.

Niederman, F., Gregor, S., Grover, V., Lyytinen, K. and Saunders, C. (2009). ICIS 2008 Panel report: IS has outgrown the need for reference discipline theories, or has it? *Communications of the Association for Information Systems*, (*24:*1), pp. 37.

Rigby, D. K., Sutherland, J., & Takeuchi, H. (2016). The secret history of agile innovation. *Harvard Business Review*, *4*.