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*

Follow this and additional works at: [https://aisel.aisnet.org/sais2022](https://aisel.aisnet.org/sais2022)
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
bimitachoudhury@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 code-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):

<table>
<thead>
<tr>
<th>Manufacturing Paradigms</th>
<th>Software Development Approaches</th>
</tr>
</thead>
<tbody>
<tr>
<td>Craft Production (pre-1910s)</td>
<td>Code and Fix (1950s)</td>
</tr>
<tr>
<td>Taylorism and Mass Production (1910s)</td>
<td>Plan-driven approaches such as Waterfall or V Model (1970s)</td>
</tr>
<tr>
<td>Lean Manufacturing (1970s)</td>
<td>Lean Software Development (1990s)</td>
</tr>
<tr>
<td>Agile Manufacturing (1990s)</td>
<td>Agile Software Development (2000s)</td>
</tr>
</tbody>
</table>

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

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.

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.

<table>
<thead>
<tr>
<th>Agile Manufacturing</th>
<th>Agile Software Development</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agility in Manufacturing is a philosophy not a set of practices (Gunasekaran et al. (2002))</td>
<td>The concept of Agility in software development evolved from a set of practices and was driven by practitioners (Conboy, 2009, Kakar, 2018ab)</td>
</tr>
<tr>
<td>AM is a business wide context (Goldman and Nagel, 1993)</td>
<td>ASD restricted to software development projects (Conboy and Morgan, 2010; Kakar, 2020)</td>
</tr>
<tr>
<td>AM is focused on design (new product development process), production, sourcing, distribution and temporary alliances to meet customer/supply chain/market requirements.</td>
<td>ASD focused primarily on production process and activities of software development (Kakar, 2018ab)</td>
</tr>
<tr>
<td>The ideal of AM is virtual manufacturing and mass customization</td>
<td>The ideal of ASD is the use of light weight processes for enriching customer and making them more competitive (Kakar, 2017ab)</td>
</tr>
<tr>
<td>AM was an evolutionary concept; did not disown the useful Lean and Mass production methods but further built on them</td>
<td>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, 2015ab)</td>
</tr>
<tr>
<td>The concept of Agility in AM has matured (Conboy, 2009)</td>
<td>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)</td>
</tr>
</tbody>
</table>

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, Lytinen 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.
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 development 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**


