Why Agile Fails: Great Methodology Can't Circumvent Complex Design

In the past decade, agile practitioners have maintained that improving software development processes is the path towards efficient software production. Agile methodology has created great progress in developing and monitoring a productive software culture. Still, many organizations are running into a wall: applying lean principles can only increase progress so much. This is because velocity measurement, planning poker, attacking defect backlogs, Kanban cards, pair programming, or sprint-based planning does little to attack the root cause of inherently structural problems. If humans can't easily understand or modify their code, teams might be using the best agile practices, but their ability to respond to market demands will be far from agile. Technical debt will weigh down the value stream's performance.

Silverthread's bread-and-butter is the visualization and quantification of software modularity and its erosion. Insight into this typically invisible element of software production has shown strong correlations between architecturally complex software and organizations in which agile methodologies have been attempted and failed. Opaque internal complexity prevents timely additions by increasing the effort needed to change, adding non-value-add effort to each new element. Additionally, architecture degradation can lead to technical bankruptcy. Typical change in US gov code base takes 80 days, which is fundamentally incompatible with the 15-day Agile sprint.

If you use agile processes in non-agile product architecture, you'll get faster at delivering non-value. The architecture will become the biggest bottleneck to your DevOps transformation. A balanced focus on agile processes and agile architecture is crucial. With this approach, your organization can sustain excellence and succeed at the pace of delivery enabled by DevOps.

Happy Holidays,
The Silverthread Team