Highpoints From the Agile Software Development Forum

Pamela Bowers
CrossTalk

There is much confusion in the software industry about what agile software development is and is not, and what it implies. This article reports on the keynote talks at the "Creating Competitive Advantage Through Agile Development Practices" technology forum held at Westminster College in Salt Lake City in March. Nearly 110 attendees at this initial annual forum participated in work sessions and networking breaks, and heard speakers and panel discussions about increasing return on investment, decreasing time to market, increasing innovation, and more.

Agile software development is not new. It has been around since the beginning of software development, but did not show a competitive advantage in the 1970s and 1980s, said Alistair Cockburn, a consulting fellow at Cockburn and Associates. However, it did win the development races in the turbulent 1990s, he said, and methodologies that began appearing in 1993-95 were Rapid Application Development, eXtreme Programming (XP), Scrum, Dynamic Systems Development Method, Crystal, and adaptive.

Cockburn was one of the speakers at the "Creating Competitive Advantage Through Agile Development Practices" technology forum held recently at Westminster College in Salt Lake City. More than 150 attendees learned about agile software development and networked with peers during breakout sessions. The Westminster College Gore School of Business and Wasatch Digital IQ co-sponsored the forum.

In his talk, Cockburn explained that agile software development is about putting value on "maneuverability." He said, "Its [agile development] being able to respond quickly became its advantage, however, agility is a value statement." Cockburn stressed that agile is not appropriate for every project. Some projects want predictability and repeatability, and cost and agility go against each other, he said. "With agile, it costs more to deliver products faster. There are trade-offs, and each project will make its own appropriate value decisions."

Jim Highsmith, director of Cutter Consortium's Agile Project Management Advisory Service, cited two main reasons for the popularity of agile software development today. "Agile addresses a problem domain where speed and flexibility are paramount," he said. "And it addresses a culture and workplace we would like – that Dilbert would like to work in – a community."

Cockburn noted that agile methods make greater use of the following: individuals and interactions, working software, customer collaboration, and responding to change. Plus, agile means different techniques in different situations, he said. "Within agility, different tactics fit different situations. Agile is not just a re-titling of eXtreme Programming."

Highsmith noted that agile works best in "exploration" environments. He compared it to drilling for oil: Drilling to known oil reserves requires very different techniques to be cost effective than does exploration drilling to find oil. In each case, projects are managed differently and success is measured differently, he said. "Agile software development finds a way around problems to get successful projects."

> *"Agile software developers are not hackers. Agilists plan regularly, test according to project priorities, and re-check results with users often. They talk to each other and customers as a matter of practice."*

Highsmith defined an exploratory project in the software world as "one to complete large projects that are both frontier (research-like) and mission critical in a turbulent business and technology environment. The characteristics include early product release, high customer involvement, and frequent testing."

What led Symantec to move to XP was not delivering the product that its customer needed, said Russell Stay, vice president of Product Delivery. The company was using a modified Waterfall technique consisting of up-front design then execution. Tight project management resulted in delivering the product on time and in budget, said Stay, but it was the wrong product. "We needed to adopt a new process."

Agile software development is a resource-limited cooperative game of invention and communication, said Cockburn. The key is adapting to reality with players – people – who are non-linear, unpredictable, spontaneous, and bring weaknesses and strengths to the game, which never repeats itself, he said. "Software succeeds when people notice errors and have enough pride in their work to step out of their job description to see that it gets fixed."

An important consideration, said Cockburn, is that people communicate most effectively interactively, i.e., face to face. "The richest form of communication is two people at a white board. The least effective form is on paper." Much is said in the communication that occurs with body language, voice inflection, facial expression, etc., he said.

Added to this is the fact that the project methodology gets restructured around the ecosystem details, which are always changing. The key is to pair workers who complement each other to achieve the desired results. For example, said Cockburn, if "Bill" only has the patience to take a project through the requirements stage, then he should be teamed with "Mary" who excels in implementing the process through to project completion, he explained. "This way, information gets from the marketplace to the programmers."

"Gone are the days of the saviors and cowboys," said Stay. Pair programming was stressed up front when Symantec began its agile implementation. Stay said that he was willing to accept a 15 percent attrition rate due to this change. However, after giving it a try, he said that fewer than 8 percent of

Successful Methodology Ensures Reuse

In the real world, organizations can mandate that Personal Software Process^{SM} (PSP^{SM}) be used on a project and install the trainers to make sure it is used. However, software developers can either refuse to use PSP, or find many ways to subvert it, warned Alistair Cockburn, a recognized expert on software project management in an interview with CrossTalk.

"Since 1991, my views on methodology have been that programmers can at anytime opt not to use this [process] either overtly or covertly," said Cockburn, a consulting fellow at Cockburn and Associates. "Therefore, the definition of the successful methodology for me includes that the people agree to use it the next time."

Cockburn said that he is looking for the way [methodology] that "puts the least requirements for consistency and discipline on software teams, yet beats the odds." He pointed to Crystal as a solution. There are three core principles in Crystal that make it successful. First, it works in increments, which allow you to recover from almost any catastrophe. Second, Crystal calls for reflection after every increment to discuss what to keep and what to change, which develops a process that adapts to change. Third, the team must tailor itself to create its own process. He added that Crystal also has a strong emphasis on personal communications, tacit knowledge, close worker proximity, and frequent delivery of running deliverables. It is all these elements, Cockburn said, that allow you to "beat the odds."

When asked, "What are the three things that most ensures agile success?" Cockburn said that experienced management is the No. 1 factor. "You need to have a project manager who is alert, sensing whether something is right or wrong, and with enough experience to steer the group. Second is having access to real users; developers need reliable information for requirements, to have someone handy to show results, and to get feedback on a reliable basis. Third most important is physical proximity. It is important to have people close enough together that they can talk to each other, he said.

Cockburn also advises management to pay attention to their fears. "They could be well founded." The key is to find out which fears are unfounded, he said. For example, he pointed to the fear of "hacking." In XP, the process check is that there are always two people working together, so it's a lot harder for one of them to hack, he said. Programmers also write their acceptance check before they write actual code, and this takes a lot of thinking, he added. A final rule is that any two people sitting together can change anybody else's work, as long as they agree, he said. "Call it common ownership."

With agile development, said Cockburn, "Programmers cannot just say, 'Go away and leave us alone'. Agile takes collaboration among project managers, users, programmers and testers. There is no privacy in the code."

employees left the company.

Stay stressed that it is important for the team to buy into the agile method and that coaching should be brought in house. Co-residence is also critical so that leaders and developers can work side by side, he said. "Individual office cubes get used less and less. Our average use is about one hour per day. The rest of the time, the development area is the hub of activity."

Unfortunately, the agile message is often misconstrued, said Cockburn. "Agile software developers are not hackers." Unlike hackers, agilists do plan regularly, he said. Agilists test according to project priorities, and re-check results with users often. They talk to each other and customers as a matter of practice. They expect management to provide priorities and to participate jointly in making project adjustments.

Stay concurred. The process at Symantec is "highly managed but flexible," he said. They use the "queing" theory of breaking down the process into small parts; iterations are done biweekly. Also, exit criteria and test procedures are defined first, before writing code, he said, and test automation is a priority.

It is also not true that agile only works with the best developers, Cockburn said. The critical success factor is to have at least one experienced and competent lead person, who can then carry four or five "average or learning" people. With that skill mix, agile techniques have been shown to work many times when the deck is "stacked" as follows:

- Hire good people.
- Seat them close together to help each other out, close to customers and users.
- Arrange for rapid feedback on decisions.
- Let them find fast ways to document their work.
- Cut out the bureaucracy.

"Agile software development has a lot to do with how much trust and communication is set up," said Cockburn.◆