Agile 2012 Conference

Key Take Aways

August 2012

SYNERZIP

www.synerzip.com

Conference Overview

- August 12-17 in Dallas
- Approx 1600 participants, 16 stages, over 200 sessions, plus inspiring keynotes
- We (Vinayak and Hemant) attended 20 sessions each plus Exhibit booths of about 40 vendors of tools and training services
- 2nd year of 1-day Executive Forum, with invited senior executives
- Like last year, big sponsors like Google or Microsoft were absent or kept out giving more relevant companies a bigger share of the lime light

Top “10” Take Aways

Disclaimer

1. Limited sample – 40 sessions out of 200

2. Entirely our viewpoint, what we took away

3. Some inconsistencies, representative of conference presentations

4. Where possible, we’ve mentioned the presenter name

1. Transformation vs. Adoption

- Concern about creating lasting Agile transformation, than just adoption. How to make it stick?
- Moving beyond superficial practices/ceremonies to deeper understanding of essence of Agile – “Being Agile” rather than “Doing Agile”
- Top challenges with Agile adoption
  1. Corp/org culture
  2. Enterprise wide adoption, beyond dev teams
  3. Poor technical practices
- How to create the needed org culture change?
  - Being a “Learning Organization” is a pre-condition to effective Agile Transformation
  - Do “Org Refactoring”, piece by piece, respecting interfaces
  - Put people in roles to leverage their strength (Gallup Group assessment)
- It begins with the leaders. Leaders need to understand software and its inherent uncertainty.
  - Don’t try to overmanage or overcontrol it.
  - “Focus on steering the ship, but need to understand the engine-room”

2. Agile for Lean Startup

- UDD = Agile + Lean Startup + Enterprise
- User driven development. MVP decides priority for this sprint. No need to labour over building a product backlog.
- Build- Measure – Learn using analytics
- MVP = 20% features that deliver 80% value
- Developers pair with the users for feedback hence **product owner not reqd.**
- Sprint planning and estimation meetings not needed. Kanban works very well.
- 2-3 day loops from MVT+dev to feedback.

3. Scaling Agile/Scrum

- Scaling up Excellence - Bob Sutton's keynote
  - Scaling is about spreading and sustaining a mindset (shared beliefs), not just footprint
  - Catholicism vs. Buddhism
- Dean Leffingwell's Scaled Agile Framework
  - http://scaledagileframework.com/
  - Lean/Agile transformation at John Deere http://scalingsoftwareagilityblog.com/john-deeres-isg-gets-results/
- Scott Ambler's Agile Scaling Model
- Need better answer on scaling Agile, beyond 30 person teams, organization wide (incl. mktng, executives, etc)
  - Scrum of Scrums is a "non-Answer"?
  - Think of scaling scrum along the lines of Object Oriented techniques (Jeff Sutherland)

4. Living with Black Swans

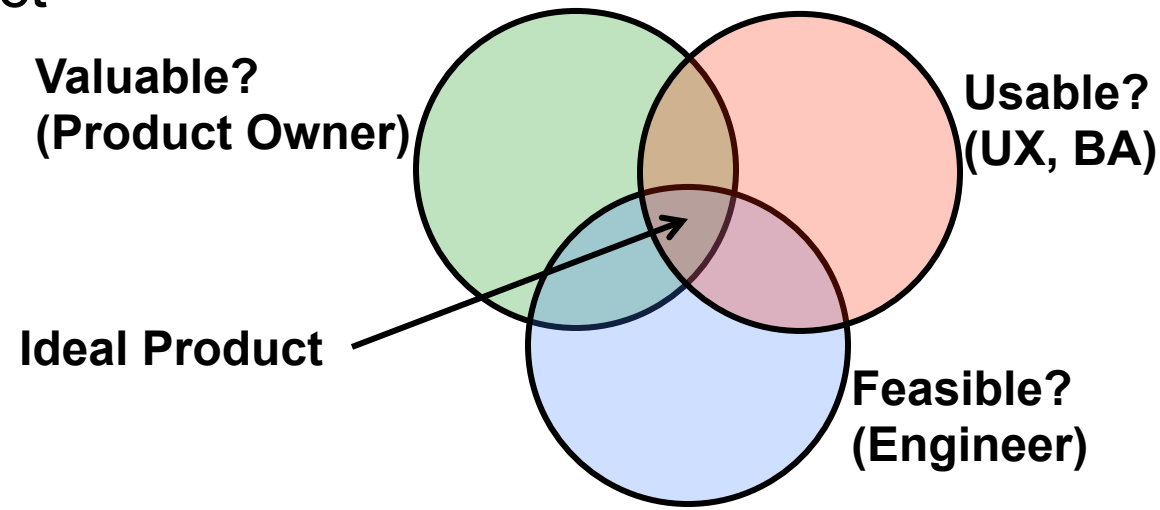
- Point of inflection 10X improvement e.g. skype (10x reduction in cost), wikipedia
- Continous release (Flikr, Gmail) learn from fast feedback. Don’t ask what is needed but study the outcome of experiments.
- If one link in adoption chain breaks the product fails (MP3 player 1998 without music/ 3G cell phone without provider)
- Experts can’t predict consumer behavior. Test early , test often, learn from failures.

4. Living with Black Swans (Cont'd)

- Learning to adopt to change by innovating
  - Step 1: Compelling offer by observing. (Press release)
  - Step 2: Immediate Connection by questioning (immediate vs delayed feedback , directly vs indirectly from customers)
  - Step 3: Adoption chain by networking (Innovator, Distributor, retailer , consumer)
  - Step 4: Validate assumptions by experimenting (connections= assumptions when? who? how?)

5. Product Owner

- Jeff Patton's session, comakewith.us, http://www.agileproductdesign.com
- Product Owner shouldn't be considered the single "wringable neck" for product success
- The Agile team should collectively take that responsibility
- The entire team should strive to discover & learn the customer needs and address them – they should "Co-make" the product



6. Continuous Delivery

- How long does it take for one line change in code by dev to get deployed in production?
- Dev Complete is meaningless unless what is developed is delivered & users give f/b
- For most deployments MTRS is more important than MTBF. Resilience needed.
- Everyone commits to mainline every day.
- Deployment pipeline, single click deployments, blue green deployments, feature toggles, dark launching.

7. Don’t Listen to Customers!

- To prioritize features/stories, don’t just listen to customers, look at their actions also
- Be hard-nosed and question the requirements
  - Analyze usage data of current features in the product
  - Observe and analyze what customers do rather than what they say
- Routinely and proactively retire under-used/ unused features
  - Like code refactoring
  - Analyze why a feature is underused – poor design or no need for it

8. Cross-functional Pairing

- Collaborative innovation by pairing UX designers and developers. Strategic work requires more c/f pairing. PM/ UX/ Dev/ QA
- Ideas are fragile when they first emerge. Culture that nurtures ideas needed.
- The myth of solitary genius has eclipsed creative innovation-need to collaborate.
- Keynote by Joe Justice. Agile applied to car manufacturing. Other industries will adopt agile practice.

9. Don’t Focus on Velocity

- Sessions by Jeff Patton, Kenny Rubin, Analyst Forum
- The focus on measuring and monitoring velocity is misguided
- Velocity is a measure of “capacity”, should not be used to measure “productivity”
  - Capacity should be measured at team level
- Don’t try to maximally loading the team members
  - Focus on “idle work”, not “idle people”
  - To win 4x100m race, focus on the baton, not on maximizing the utilization of runners!
- “In Agile teams should focus on getting work done, rather than getting work started”
- Strive for stable velocity, sustainable pace

10. Driving Innovation

- Session by @dneighbors
- Chaos sparks creativity
- Collaboration – surround yourself with good people – conducive to innovation.
- Fun and playful environment
- Excellence – set the bar higher.
- Heirarchy/designations are preventing big companies from bringing in innovations.
- Don’t invest on building swanky offices for today- instead make configurable offices that teams can modify as needed.

11. Use Value Points

- Session by Jim Highsmith, Pat Reed
- To move management focus away from “Velocity” use “Value Points”
  - Use $ at Portfolio level, value points at story level
  - “It is better to have a fuzzy measure about something important, rather than a precise measure of something unimportant”
- Example:
  - $5M NPV project is approved
  - Assign 5000 value points to it overall
  - Allocate down these points to various epics, stories etc.
  - If scope increases later, either add more points (if NPV has increased) or reallocate all value points
  - On each story card, show value points along with story points (cost estimate)
- Regularly show Value Burn-up charts

12. Maintainable Acceptance Tests

- Typically ratio of code to test code is 1:2.5
- Tests are result of dev-QA dialogue. Having them in silos is not productive
- Having 2 tiered tests (e.g page object) make them more maintainable.
- Code tests unit tests-addl tests not reqd.
- Story based tests are brittle and represent how the application was behaving at the time of writing the tests- Journey tests that depict user's journey thru the app are better.

13. Cost/Budget

- Sessions by Johanna Rothman, Ron Jeffries, Chet Hendrickson
- Agile is not about managing costs, it is about delivering higher business value for a given cost (a constraint)
- There is no special way to do estimates in Agile
- For a big project, if you need to do upfront cost estimation, then Agile can't help you there
  - Use conventional methods – prior experience w/ similar projects, etc
  - Then use Agile to deliver higher business value per sprint
- At sprint level
  - The team who does the work is the team who estimates
  - Don't spend too much time estimating – 2hrs per sprint is adequate
  - "Better yet, make your stories small (1 SP/story), and just count the stories, with 50% confidence!"

14. Testing Big Data

- Hadoop is optimized for big stuff but poorly optimized for small stuff
- Riak doesn’t provide special harness for unit testing like mrunit.
- Unit tests are significantly faster than integration tests
- Riak and Cassandra tests run faster than Hadoop
- Netapps has special plugins for Hadoop to clone clustered environment in seconds

15. All About the (Distributed) Team

- Agile is all about the team
  - Defiintion of “one team” includes the customer on it, not just dev, QA, product owner, scrum master, etc.
  - Stable, collaborative teams, with mutual trust, inspect & adapt mindset
- Behind every high-performance scrum team is high-performance scrum-master
- Get rid of individual performance appraisals – they are demotivating and don’t’ accomplish anything!
- Workable team structure for distributed teams
  - Co-located Dev+QA, but PO can be remote
  - 1 PO per team
  - 0.5 Scrum Master per team
  - All other variations, on exception and short-term basis only
- Most dropped practice: Potentially Shippable Increment!

16. Simple Design

- Four principles
  - All tests must pass – given that TDD is being practiced
  - Minimize duplication. Do commonality-variability analysis. (DRY principle)
  - Maximize clarity- code should be self explanatory. (Apply telephone test)
  - No superfluous code exists. Work done by fewer elements.
- Remove duplicates= structured code
- Remove bad names = clear responsibility.

17. Technical Debt

- Techical Debt concept and vocabulary is now well understood and widely practiced
- Managing Technical Debt book by Chris Sterling
- Keep it visible (on product backlog) and concrete
- What matters more is the ongoing cost to service this debt, not its absolute amount!
- Debt is more relevant in context of meeting your customers' needs. If you are meeting your customers' needs, then don't bother about Technical Debt.

18. Beautiful HTML/CSS Code

- Consider all form factors (Most HTML is written for desktop)- Why not mobile first?
- Content is more important than look & feel
- Users should control UX instead of designers- think of HTML as an API
- Consider who you are excluding- e.g. commuter drinking coffee, low b/width
- Think infinite number of browsers – progressive enhancement
- Composable pages,reusable module, oocss

19. Personal Kanban

- Session by Sandi Mamoli
- Multi tasking /Context switching reduces productivity
- Personal Kanban board – WIP limit imposed by the space available.
- Bigger tasks are broken down into smaller tasks.
- Time boxing – Pomodero technique
- Can be used for improving collaboration.

Contact Information

- Hemant Elhence *(Dallas based)*
  - hemant@synerzip.com
  - Cell Phone: 214.762.4873
- www.synerzip.com
- HQ and US office in Dallas, TX
  - 14228 Midway Rd, #130, Dallas, TX 75244
  - Office Tel: 469.322.0349
  - Office Fax: 469.322.0490
- Development center in Pune, India.