



The Agile Philosophy

- ▶ Reduce risk and increase customer satisfaction by delivering working versions of the software at frequent, short intervals
- ▶ Reduce communication overhead and misunderstandings by encouraging face-to-face communication among project team members
- ▶ Gracefully incorporate changed requirements into the product at any point during the development cycle
- ▶ Give the team room to self-organize and evolve its own project plan

PLATEAU

The Project Goals

- ▶ Improve communication between product management and the technical teams
- ▶ Improve development sizing estimates
- ▶ Reduce the length of the development and testing cycles
- ▶ Empower nascent technical leaders
- ▶ Build trusting working relationships across departmental boundaries

PLATEAU

The Project Team

- ▶ Product Management (The Customer)
 - Product manager
 - Business analyst
 - Interaction designer
- ▶ Engineering
 - Developers
 - Architect
 - Technical lead (in-house agile coach)
- ▶ Quality Assurance
 - Testers
- ▶ Technical Communications
 - Technical writer
- ▶ External Agile Coach

PLAT=AU

The Process

- ▶ Customer Presents Product Vision
- ▶ Develop User Role Model
- ▶ Develop User Stories
- ▶ Size User Stories
- ▶ Prioritize User Stories
- ▶ Iterate
 - Do the planning game
 - Document stories in the iteration
 - Develop and test
 - Daily stand-up meetings
 - Adjust requirements and/or UI specification if necessary
 - Evaluate
 - Customer acceptance testing
 - Sign off

PLAT=AU

Early Problems

- ▶ Everybody does everything together
 - Communal role modeling effort collapsed in confusion
 - Communal User Story development much proved more expensive and time consuming than originally anticipated
- ▶ Lack of clarity about User Stories – *what are user stories anyway?*
 - What they are
 - Descriptions of developer work assignments that can be used as building blocks to size and plan iterations and developer workloads
 - What they aren't
 - Usage scenarios
 - User tasks
 - Sometimes not even an action a user would take (e.g., "display summary data", "maintain dirty state")
 - Use cases
- ▶ Persistent tension between story-based iteration planning and interaction design activities
 - Granularity of stories is very small, design is holistic
- ▶ Documentation was too heavy
- ▶ End of iteration evaluation required the Customer to make judgments about quality that were out of context

PLAT=AU

Adjustments

- ▶ Project meetings changed to include only those whose work touched the meeting topic
- ▶ More work was done outside of meetings with the results presented during meetings
- ▶ Daily stand-up meetings became the only “all team” gathering
- ▶ Design activities pushed ahead of the rest of the project to ensure coherence of the user experience
- ▶ Documentation was lightened
- ▶ Out-of-context issues were deferred

PLAT=AU

Lessons Learned

- ▶ Develop a project charter that clearly defines roles and responsibilities
- ▶ Stay in touch as a team
- ▶ Start with a stable set of requirements
 - It’s easier to break use cases down into stories than to build up a coherent set of requirements from individual stories
- ▶ Start with a stable interaction design
 - Stories exercise very small, isolated pieces of interaction that may not have connections made to each other for several iterations
- ▶ Pause at the end of an iteration (of whatever length) to re-think, reprioritize, and celebrate success

PLAT=AU
