

## Discussion Slides

9/16/2007

Agile Process and eXtreme Programming

25

## Deeper? ●

- Why should a waterfall project have a predictable schedule?  
*We try to get the requirements right before we start the design.*  
*We try to get the design right before we start the construction.*  
*This should minimize surprises and the associated wasted effort.*

9/16/2007

Agile Process and eXtreme Programming

26

## Deeper? ●

- What happens, in a waterfall project, if the requirements are wrong?  
*We will build and deliver the project on schedule, after which we will either*
  - a. Go back, reexamine the requirements, and try to redo the project correctly (but very late, and very over budget).*
  - b. accept that the project has failed (and that we have wasted a great deal of time and money).*

9/16/2007

Agile Process and eXtreme Programming

27

## Deeper? ●

- Why might it be impossible to develop correct (necessary and sufficient) requirements?
  1. *The people you interview either don't know, or can't articulate what they need (e.g. we are in terra incognita).*
  2. *Technology, process, and applications evolve so that requirements are constantly changing.*
  3. *People are careless about what they call "requirements".*

9/16/2007

Agile Process and eXtreme Programming

28

## Deeper? ●

- How does an incremental development model help (or not help) with this problem?  
*It does allow us to postpone requirements for subsequent releases ... which gives us (and the market) longer to understand those needs.*  
*It still, however, assumes good requirements for the release we are currently working on.*

9/16/2007

Agile Process and eXtreme Programming

29

## Deeper? ●

- How can formal process over-burden small projects?  
*By requiring inappropriate (and ineffective) levels of approval and review.*  
*By requiring work products that are not necessary or applicable to a particular project.*  
*By making people spend more time on reporting than they do on actual work.*  
*Remember what McConnell said about quality.*

9/16/2007

Agile Process and eXtreme Programming

30

## Deeper? 🟡

- How can process help a new (or weak) team?

*Standardized process ensures adequate communication.*

*If they lack the experience to know what the right steps are, a formal process can guide them.*

*If they lack the skills to do the job correctly, check-lists and reviews can find mistakes as quickly as possible.*

9/16/2007

Agile Process and eXtreme Programming

31

## Deeper? 🟡

- How can process hurt a strong team?

*Some of the steps mandated by the process may not actually be necessary for a particular process.*

*It may take much more time to document step-by-step compliance than it takes to do the right things.*

*Some people can get to the right answer without following all the intermediate steps.*

9/16/2007

Agile Process and eXtreme Programming

32

## Deeper? 🟡

- Why might incremental prototypes and regular meetings with users be more effective than exhaustive requirements development?

*Feedback on a prototype is probably more reliable than speculations on requirements for software that doesn't yet exist.*

*Successive iterations are based on real experience, rather than on further hypothetical speculation.*

9/16/2007

Agile Process and eXtreme Programming

33

## Deeper? 🟡

- Why might smaller prototype iterations be more effective than more complete incremental releases?

*Less waste - we get feedback sooner and so waste less time going down the wrong paths.*

*Much faster convergence on right approach.*

*Higher return on investment - if the primary goal is concept validation, doing the effort to create a formal release would be mis-spent. Don't productize it until it is right.*

9/16/2007

Agile Process and eXtreme Programming

34

## Deeper? 🟡

- Is the Agile Manifesto techno-elitist?

*It is almost entirely dependent on experienced and motivated individual contributors. It cannot work without them.*

- Is this OK?

*Ignoring the issue of the team's experience would clearly be foolish ... it is clearly the "elephant in the room".*

*Saying "if you don't have experts, you shouldn't build software" is also unrealistic.*

9/16/2007

Agile Process and eXtreme Programming

35

## Deeper? 🟡

- How do agile processes address people and teamwork issues?

*By focusing on face-to-face communication rather than written specifications.*

*By allowing developers broad latitude in how they should organize their efforts.*

*By specifying team processes (as will be seen in XP).*

9/16/2007

Agile Process and eXtreme Programming

36

## Deeper? ☺

- How do agile processes focus on the real goals?

*They define the real goal to be the delivery of working software that satisfies user needs.*

*They focus on this goal by mandating regular interaction with customers, and organizing software development activity around specific requested features.*

*They measure progress in terms of delivered features.*

9/16/2007

Agile Process and eXtreme Programming

37

## Deeper? ☺

- What is the difference between principles, methodology, and process?

*Principles are broadly applicable axioms of goodness and good behavior.*

*Methodology is a collection of tools and techniques for solving specified problems.*

*Process is a sequence of steps to be followed.*

*Their point is that the steps to be followed should be dictated by the applicable principles and methodology.*

9/16/2007

Agile Process and eXtreme Programming

38

## Deeper? ☺

- How does the planning game compare with requirements analysis?

*It is very similar in the questions it asks, and the factors it considers to prioritize tasks.*

*It combines work scheduling with prioritization, enabling the estimates to feed back into that process.*

*It happens in the context of much smaller (incremental) releases, and only extends a few weeks into the future.*

9/16/2007

Agile Process and eXtreme Programming

39

## Deeper? ☺

- What is good about Test Driven Development?

*We have tests for everything, and new code is tested as soon as it is written.*

*The testing will surely result in our delivering code that works better.*

*The focus on defining correctness will probably lead us to write code that is more correct to begin with.*

9/16/2007

Agile Process and eXtreme Programming

40

## Deeper? ☺

- One XP principle is that we should “write code for today, not for tomorrow”.
- When would this be a good approach?

*If we are still trying to understand what is needed, it doesn't make sense to build features we might never need.*

- When would this be a bad approach?

*If we know what will be needed in the future, ignoring that knowledge today will result in a lot of code being thrown away tomorrow.*

9/16/2007

Agile Process and eXtreme Programming

41

## Deeper? ☺

- An XP principle is periodic refactoring.
- Why would this be a good approach?

*It is much easier to solve real problems than hypothetical ones.*

*We can waste a great deal of time designing and coding around problems that will never happen.*

- When would this be a bad approach?

*If we can reasonably anticipate a problem that can easily be solved by the right architecture.*

9/16/2007

Agile Process and eXtreme Programming

42

## Deeper ☺

- How is an agile prototyping cycle different from an incremental release cycle?  
*In an incremental release cycle, we actually deploy and support a product. In an agile prototyping cycle, we are just looking for feedback.*
- How is an agile prototyping cycle different from a spiral iteration cycle (ala Boehm)?  
*Agile approaches suggest a specific way of planning spiral iteration cycles (based on feature prioritization and feedback from the previous cycle).*

9/16/2007

Agile Process and eXtreme Programming

43

## Deeper ☺

- Is one of these approaches monotonically better than the other?  
*If the requirements are truly unclear, a strict waterfall model is a plan for failure.*  
*If we need specific functionality by a specific date, we can have little confidence that a set of successive build and evaluate cycles will converge in time.*

9/16/2007

Agile Process and eXtreme Programming

44

## Deeper? ☺

- A project where agile prototyping cycles would be the best approach?  
*User Interface intensive projects like a music catalog or a new social web service.*
- A project where thorough up-front analysis would be the best approach?  
*Infrastructural or "correct-computation" projects like a new file system, cellular call routing, fly-by-wire aircraft control.*

9/16/2007

Agile Process and eXtreme Programming

45

## Deeper? ☺

- A project where formal process (lots of planning, reviewing, reporting) would be essential?  
*A project that will combine the work of hundreds of people from many locations and organizations.*  
*A project where requirements are clear, and correctness and on-time delivery are critical.*  
*A project whose customer demands visibility into the development process.*

9/16/2007

Agile Process and eXtreme Programming

46

## Deeper? ☺

- A project where formal process (lots of planning, reviewing, reporting) would be counter productive?  
*Developing and validating a consumer product concept ... where success will be the result of creativity and good feedback.*  
*Prototyping a new mechanism ... where we are exploring new mechanisms, and success will result from inspiration and fast (vs. good) code production.*

9/16/2007

Agile Process and eXtreme Programming

47