


**MICHIGAN STATE UNIVERSITY**

## 8. Software Tools

CSE 498, Collaborative Design

Wayne Dyksen  
Brian Loomis  
Department of Computer Science and Engineering  
Michigan State University  
Spring 2006



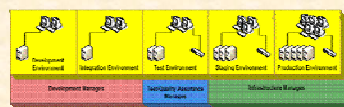
### S Overview

- Overview of a lifecycle or development process
- Tools used in each phase
  - Design
  - Development
  - Test/Stabilization
  - Deployment

3.2

### S Why have a process?

- PRO: Methodology tells you how to do something
  - Seems like a good idea...
  - Carries lessons learned forward so we don't repeat mistakes
  - Makes sure we're thoroughly understanding the problem
- CON: Most people do not have the experience to know which parts of the method help get them to the goal
  - Methodology can lead to "analysis paralysis"
  - Methodology cannot solve fundamental people issues like communication
- What's your methodology? How do you do team projects?



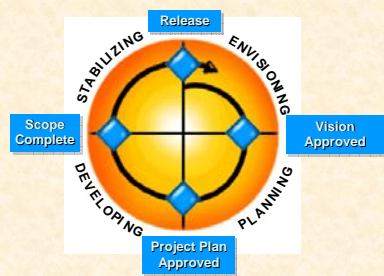
3.3

### S Lots of processes exist...

- Mil-Std 2167A, 10000 series...
- Capability Maturity Model (SEI)
- PMBOK (IEEE)
- Extreme programming (XP), agile methods
- "Corporate custom"
  - I'll use Microsoft Solutions Framework terminology
  - Most methods have similar steps but often call them by different names

3.4

### S Process Model for Application Development



3.5

### S Milestone-Driven Process

- Milestones are review and synchronization points, not freeze points
- Milestones enable the team to assess progress and make mid-course corrections
- The process model uses two sorts of milestones
  - Major milestones
  - Interim milestones
- Achieving a major milestone represents team and customer agreement to proceed
- Deliverables are physical evidence that the team has reached a milestone

3.6

## S Milestones

- 1/23: Teams: Progress Reports
- 1/30: Teams: Technical Specifications
  - PowerPoint presentation due by noon
- 2/01: Teams: Technical Specifications document due by 3 pm
- 2/02: Finish setup of PHP/MYSQL
- 2/07: Project-----
  - A webpage that will control a QuickTime instance, play, stop, alert current position
  - Example PHP scripts that store and recall data from the MySQL database, movie title, clip URLs, clip start time, clip stop time
  - Have a PHP script that generates an arbitrary SMIL file given dummy data
- 2/09: Have a plan for final UI
- 2/13: Project-----
  - Webpage JavaScript interface for POSTING to PHP program
  - PHP script for generating SMIL files now reads from database
  - PHP script to swap positions of video clips
- 2/15: Teams: Progress Reports
- 2/20 - 3/01: Teams: Prototypes
  - Combine previous projects into a prototype

## S Project Plan Approved Milestone

Signals agreement on

- Project trade-off strategy
- Project risks
- What will be built (tech spec)
- When it will be built
- How it will be built
- Who will build it

## S Drive the architecture

## S Scope Complete Milestone

Signals agreement on

- The planned feature set
- Whether the planned feature set has been developed
- Baselined materials to support user performance
- The stabilization process, including betas and testing

## S Team Focus During Developing

Role	Focus
Product management	Customer expectations management; communication plan execution; beta planning
Program management	Project tracking; team communication and coordination; beta planning
Development	Feature development; testing
User education	User performance support development and testing; beta planning; product usability testing
Testing	Test specifications, cases, and scripts development; testing
Logistics management	Operational support documentation; beta planning; internal team support

## S Internal Releases

Getting the product to a known state and incrementally building upon it

## S Daily Build

**Building the product in an executable form on a daily basis**



A public daily build is

- A strong indicator that a team is functional
- A way to make the product and its progress visible
- The heartbeat of the development process

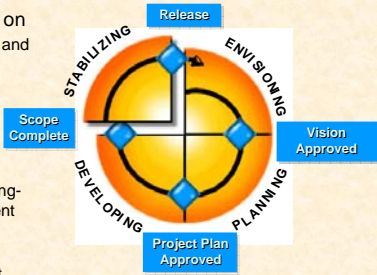
**Video**

3-13

## S Release Milestone


Signals agreement on

- Product stability and resolution of all known bugs
- Customer acceptance of the product
- Transfer of ownership for long-term management and support
- Change in team focus to the next release



3-14

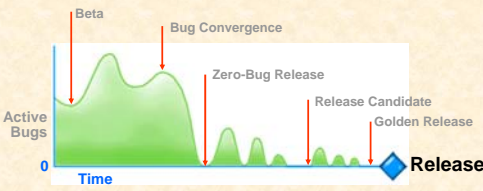
## S Suggested Interim Milestones



Entering the stabilizing phase marks the transition from a schedule-driven focus to a ship-driven focus

3-15


## S Focus on Shipping



3-16

## S Beta Testing

**Testing of a stabilized product by external end users**




- Provides actual end-user usage testing in the expected environment
- Requires greater team effort than alpha testing
- Occurs with different frequency and size depending on a number of factors

3-17

## S Bug Triaging

**Evaluating and prioritizing bugs to determine their appropriate resolution**



- Uses a review committee to prioritize and assign bugs
- Determines what new bug fixing, if any, will be done
- Balances stability against customer needs
- Can result in loss of features for the sake of stability

3-18



Tools we use....



## S Design tools

- Word, Excel (noun analysis, contracts)
- Visio (UML and ORM), Erwin, Rational XDE
- MS Project (schedule)
- Team portal and bug tracker set up
  - SharePoint, raw HTML, FrontPage
  - TaskVision
- Whiteboard...

## S Development tools

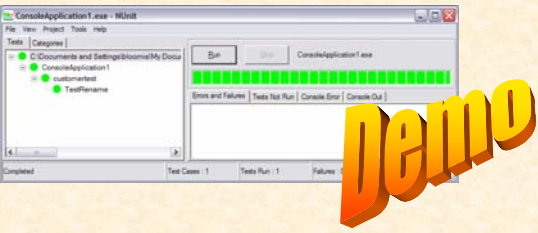

- (your compiler ☺)
- NAnt, Ant, Kinook (builds)
- FxCop (secure code reviews)
- Visual SourceSafe, PVCS, Clearcase (code mgmt)
- NDoc, HTMLHelp, Robohelp (documentation)

## S Test and stabilization tools


- VSS, Clearcase, PVCS
- NUnit, NUnitASP, JUnit
- App Center Test
- WinRunner, XRunner
- NDoc, HTMLHelp, Robohelp
- FxCop
- MSI or InstallShield
- CLR Profiler

## S Show me the code!

- Testing a simple customer app

BACKUP SLIDES



## S Testing in development

- Coverage testing
  - Attempts to thoroughly test every feature of the product
  - Attempts to thoroughly test the code base of the product
  - Is used primarily during the developing phase
- Usage testing
  - Attempts to successfully complete usage scenarios
  - Attempts to test the product in its expected environment as users might stress it
  - Is used primarily during the stabilizing phase

3-25

## S Types of Testing

- Unit tests
- Functional tests
- Check-in tests
- Build verification tests
- Regression tests
- Configuration tests
- Compatibility tests
- Stress tests
- Performance tests
- Documentation and help file tests
- Alpha and beta tests

3-26

## S Zero-Defect Mindset

**Committing to the highest possible level of quality within project constraints**

- Team members must understand the required quality level for their work
  - Articulate the quality bar for all work performed – who does this?
- Work is not complete until it reaches that level of quality
- The zero-defect mindset is embodied in
  - Task deliverables
  - Milestones



3-27

## S Benefits of a Zero-Defect Mindset

- Increases accountability for the quality of the product
- Increases stability of the product
- Improves schedule predictability
- Decreases the cost of addressing issues
- Allows testing to shift focus to quality assurance
- Rewards quality developers



3-28

## S Techniques for Zero-Defect Development

- Write unit test cases before debugging
- Assume the code is broken, then prove that it isn't
- Fix bugs before moving on
- Use competing designs and implementations
- Assign bugs to other developers
- Reassess the code in light of bugs
- Document code
- Conduct code reviews
- Perform daily builds

3-29

## S Code Reviews

**Assessing code to improve its quality and to improve the capabilities of the development team**

- Some ways to conduct code reviews
- A comprehensive, formal review
  - A more casual, peer-based review
  - An independent, third-party review
  - Tools plus people...
  - Daily build status and source control are evidence of a working process



3-30

## **S** Guidelines for Internal Releases

- Treat internal releases within a single project like versioned releases of a product
- Address high-priority and high-risk features in the earliest possible release
- Define a quality bar to determine when the product has met the standards for internal release
- Make each release as cohesive and yet independent as possible
- Conduct postmortem reviews of each internal release

3-31