The Microsoft Software Development Process

Scott Guthrie
Program Manager
Microsoft Corporation

“Natural” Phases of a Software Project

- Enthusiasm
- Disillusionment
- Panic
- Search for the Guilty
- Punishment of the Innocent
- Praise and Honors for Non-Participants

Successful Projects

- Not all software projects have to progress this way!
- Those that are successful typically share three outstanding characteristics:
  - People
  - Poise
  - Process

Today's Agenda:
The Microsoft Development Process

- Origin of a MS Product
- The Product Team
- Designing the Product
- Scheduling the Product
- Implementing the Product
- Testing the Product
- Shipping the Product

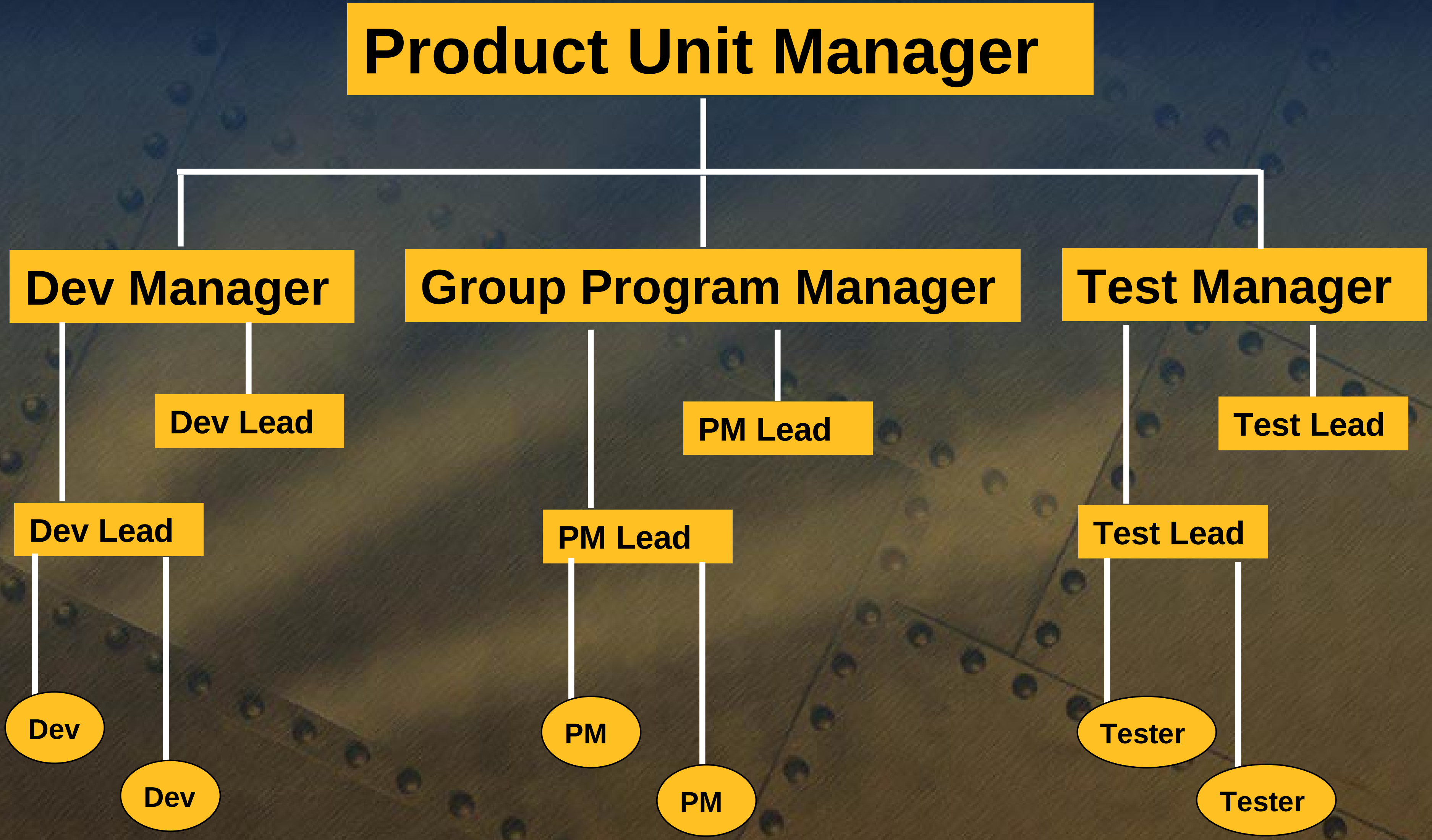
Origin of a MS Product

How to Start a MS Product

- Step 1: Identify market opportunity
  - Customers, Competitors, Market Dynamics
- Step 2: Determine viability of market entry
  - Volume, price/cost margins, fixed costs, etc.
- Step 3: Define vision statement
  - Crisp enunciation of goals + issue ownership
  - Explain strategic importance to company
- Step 4: Make a lot of noise!

The Product Team

The Product Team



Designing the Product

Product Design

- Thoroughly understand your customers
  - How do they work? What do they really do?
  - Visit, observe, listen & meticulously document
- Thoroughly understand your competitors
  - Evaluate their product strengths/weaknesses?
- Identify the *strategic* and *tactical* themes and requirements that your features should be thinking about
  - Ensure that they are inline w/ vision statement

Feature Design

- Drill down on feature specifics
  - Focus on “what it does” vs. “how we build it”
- Questions to consider:
  - How do we make a feature usable/simple?
  - How do we make a feature visible?
  - How do we integrate other parts of a product?
- Document scenarios, assumptions and design proposal in a detailed spec
  - Maintain tight feedback/evaluation loop

Implementation Issues

- Developers own thinking through the implementation issues of a feature
- Questions to consider:
  - How factorable is the feature?
  - Can the feature be delivered in stages?
  - What dependencies does it have?
  - What other features are dependent on it?
  - How many developer weeks are required?

Scheduling the Product

Scheduling/Planning

- Schedules are done after the initial design document is ready for review
- There is an inherit tension between the schedule and the design document
  - Each needs to be constantly re-evaluated and re-calibrated against the other
- Software scheduling in general is something of an imprecise science
  - Concatenation of educated guesses

Scheduling Questions

- Is the ship date driven by features or a hard schedule?
- Can/should the product vision be staged over multiple product releases?
- How long has the product team worked together? What size will it be?
  - Big != Good. Keep in mind the N-1 rule...
- Will the team be working at a normal pace or in "Death-March" mode?

Milestones

- Milestones are used to logically segment development into 9-12 week periods
  - Early Milestones: Critical features & core code
  - Later Milestones: Functionality that can be cut
- Milestones help maintain “ship-mode” focus/atmosphere over long projects
- Milestones encourage staging of products
  - Enable review of progress (“Postmortems”)
  - Facilitate corrections to master schedule

Rules for Picking Dates

- Whatever date you publish will be the **earliest** you possibly ship
  - Date should be aggressive *and* realistic
- Budget vacations and sick-leave
- Plan for unexpected absences
  - maternity/paternity leave
- Pad schedule for stabilization and non-deterministic progress delays

Implementing the Product

Establish Best Practices

- Source code management
  - Whatever happened to Microsoft Pascal?
- Coding Standards
  - What dialect of Hungarian do you use?
- Code Reviews
  - Every line of code should be peer reviewed
- Localization Guidelines
  - If you plan ahead it is money in the bank…

First Implementation Steps

- Define overall code-base structure:
  - Specify directory hierarchy (headers, libs, etc.)
  - Setup Makefile and build environment
  - Come up with common Macros and Ifdefs

- Define overall code-base architecture:
  - Design core APIs, interfaces and structures

Builds

- Products are compiled and released daily
  - Forcing factor for code interoperation
  - Provides steady progress measurement
  - Enables daily test coverage of entire product
- Builds can often take a long time…
  - “Clean Build” of Windows NT takes 36 hours
- It is critical that delays are minimized
  - Strict check-in procedures typically enforced

Check-in Procedures

- Step 1: Finish writing code
- Step 2: Code review with a team member
- Step 3: “Buddy build” on 2 clean systems
- Step 4: Send “check-in request” mail to the Build Technician and daily “Build Meister”
- Step 5: If check-in is approved, the build technician will check-out appropriate files into the build tree

Build Problems

- Build Breaks (compile/linking error)
  - Basically means some bozo screwed up
  - Punishment should fit the crime… :-)
- Build Verification Test (BVT) Failures
  - Automated test indicates functionality failure
- Each build classified at release:
  - “Self Host”
  - “Self Test”
  - “Self Toast”

Ensuring Product Quality

Software Testing

- Testing is critical to software development
  - Must be analytical, methodical and thorough
- Test plan documents must be developed before code is even written
- Automation is key to stabilizing a product
  - Comprehensive code coverage
  - Enables quick verification of product health
  - Enables easy reproducibility of errors

Bug Triage

- Discovered bugs are logged to a database
- Senior team members meet at least once a day to review/rank active bugs
- Bugs assigned severity, priority, owner
  - Must-fix bugs marked as “showstoppers”
- “Scrubbing” the bug list
  - Process of upgrading bugs to future releases
  - Done when a bug is just too dangerous to fix

Getting It Out The Door

The End Game

- Alpha Release
- Beta1 Release
- Code Complete
- Beta2 Release
- Zero Bug-Bounce
- Release Candidate (RC)
- Release to Manufacturing (RTM)

End Game Responsibilities

- Program Management: Ensure that all scenarios documented in design spec are fully operational.
- Test: Ensure that all features implemented are at 0 showstoppers.
- Development: Resolve critical bugs as they appear. Ensure that the build remains stable

The End

- Once the build hits zero showstoppers, it will be “escrowed” while the team spends several days verifying that no new nasty bugs are lurking.
- If no new showstopper bugs are identified, a “master” or “golden” CD will be burned with the product bits.
- The CD will then be released to a manufacturing factory where shrink-wrapped products will be produced.

where do you want to go today?

Microsoft®