

Gamefest

MICROSOFT GAME TECHNOLOGY CONFERENCE

August 14-15, 2006



Agile Game Development: Tales from the Trenches

Noel Llopis

llopis@convexhull.com

Senior Architect
High Moon Studios

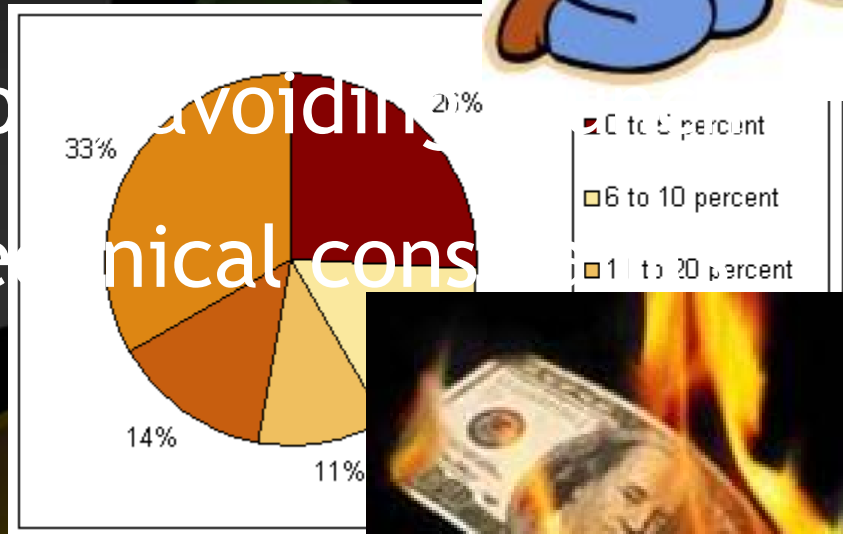
Coming up

- Agile Development
- Organization-Wide Agile Techniques
- Programming Agile Techniques
- Lessons Learned

PART 1: Agile Development

Motivation

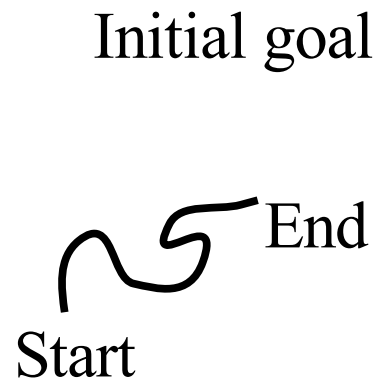
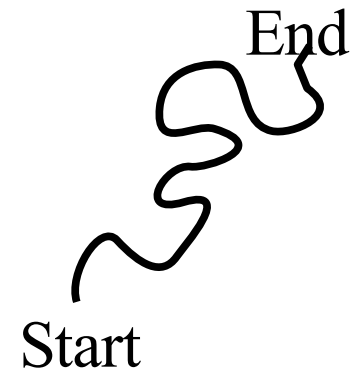
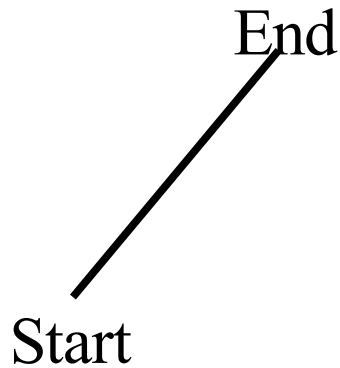
- Changing requirements
- Finding the fun factor
- Wasted effort avoiding
- Unknown technical constraints



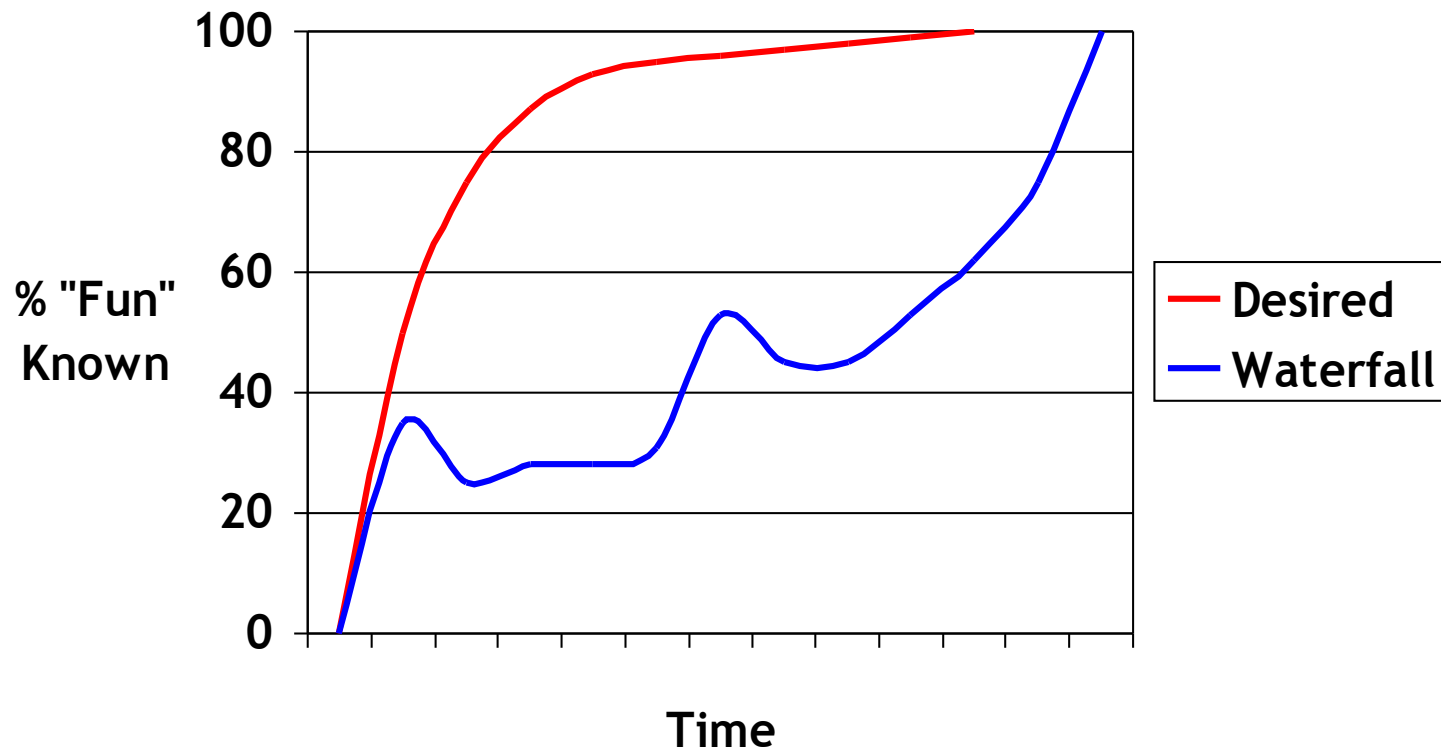
What Is Agile Development?

- Just-in-time decisions
- Adapt to change
- Maximize work not done

What Is Agile Development?



What Is Agile Development?



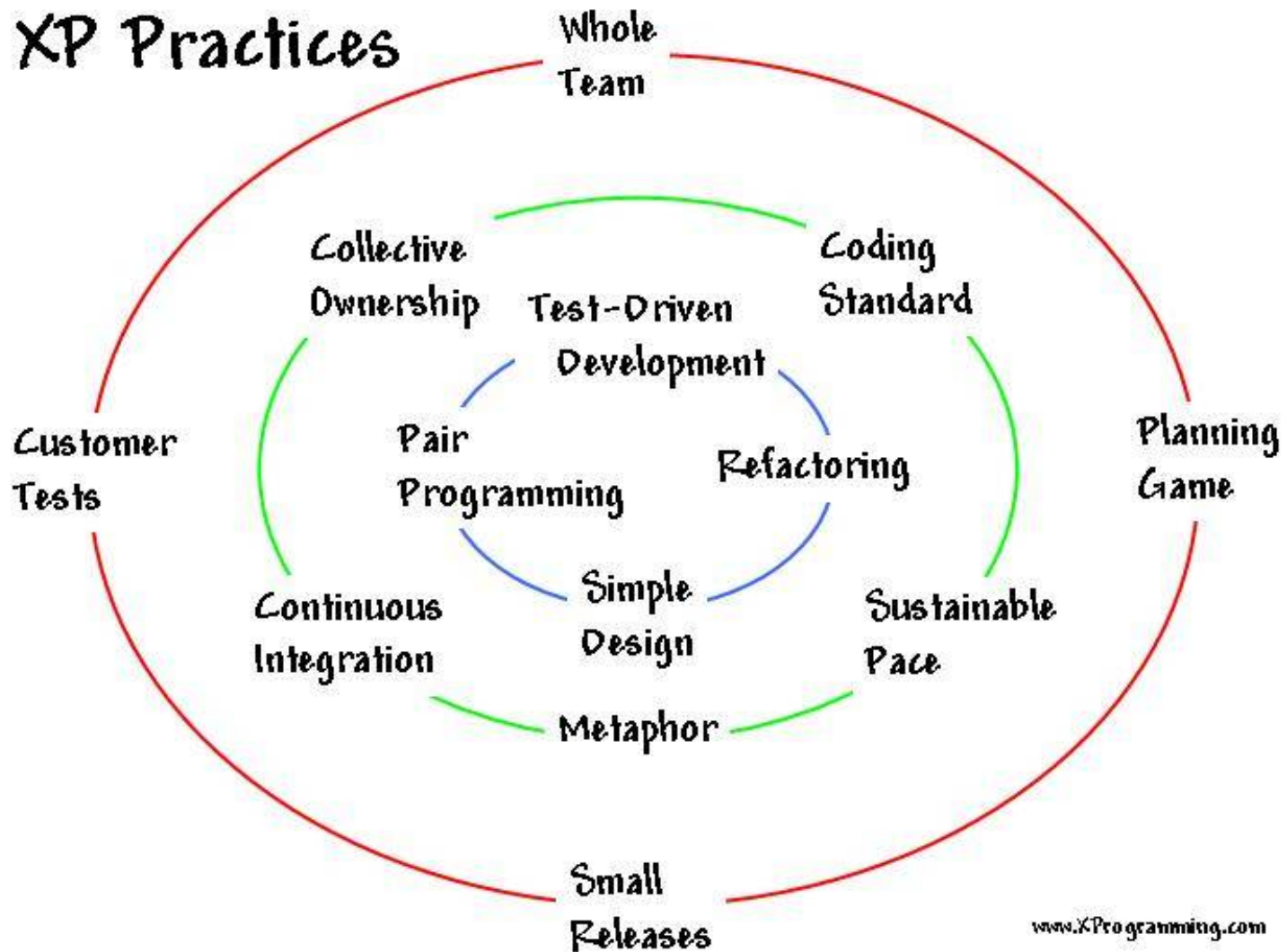
Scrum



FOR INFORMATION GOAT SL

Extreme Programming (XP)

XP Practices



PART 2: Organization-Wide Agile Techniques

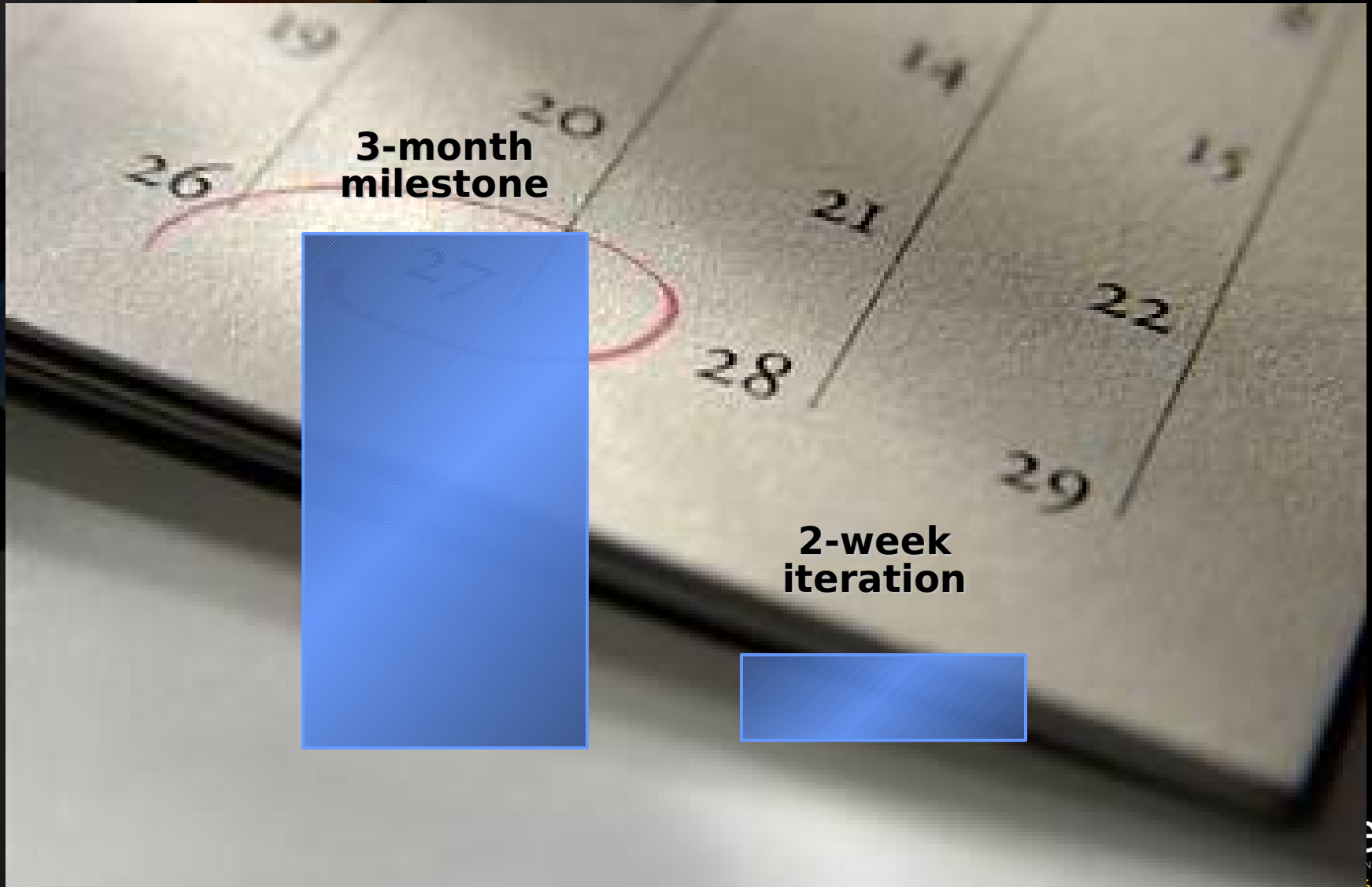
Deciding on Priorities

- “Customer” is the person responsible for the direction of the team.
- Customer decides what are the most important things to do next (“stories”)
- Team estimates duration of those stories.
- Communication between customer and team is essential.

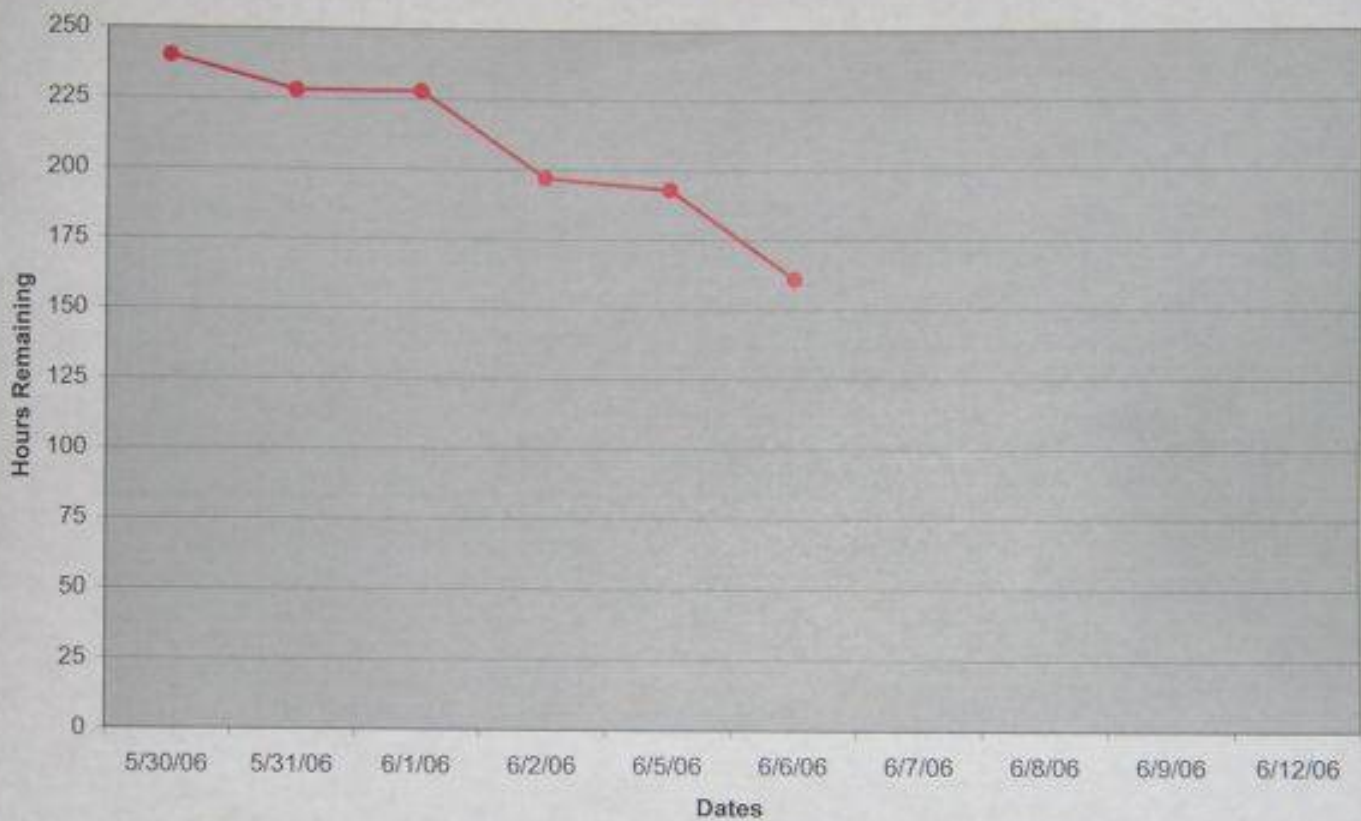
Short Iterations

- Scrum suggests 30 days. XP 2 weeks.
- We tried both and we're doing 2-week iterations.
- Each iteration attempts to create a true "vertical slice"
- Use results to feed into next iteration

Short Iterations



Apartment Burndown for Sprint 5/30/06 - 6/12/06



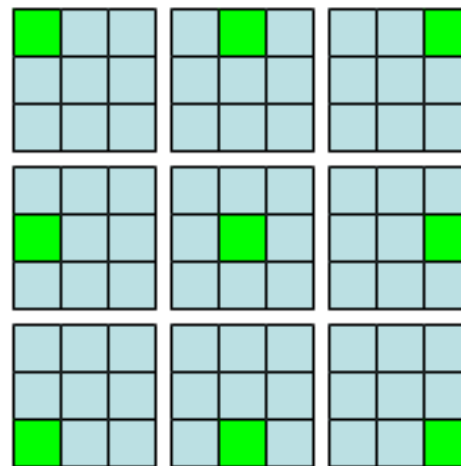
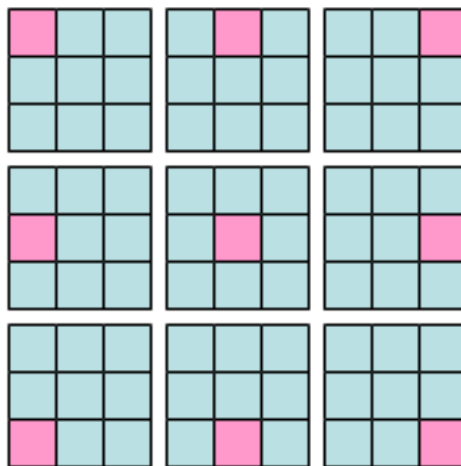
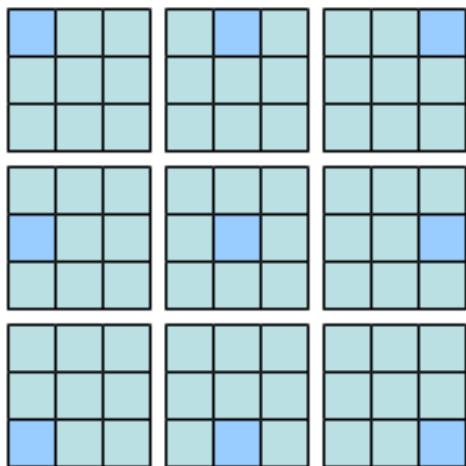
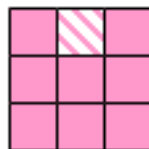
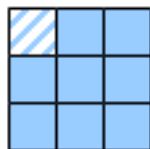
Who Does What

- People pick whatever task they want to work on
- Great for morale and spreading knowledge.
- Integrate domain experts.

Team Composition

- Small teams (8-12 people)
- Collocated in open work environment. This is huge for quick discussions and communication.
- Originally only programmers. Now totally cross-functional.
- Scrum of scrums to scale to larger team sizes

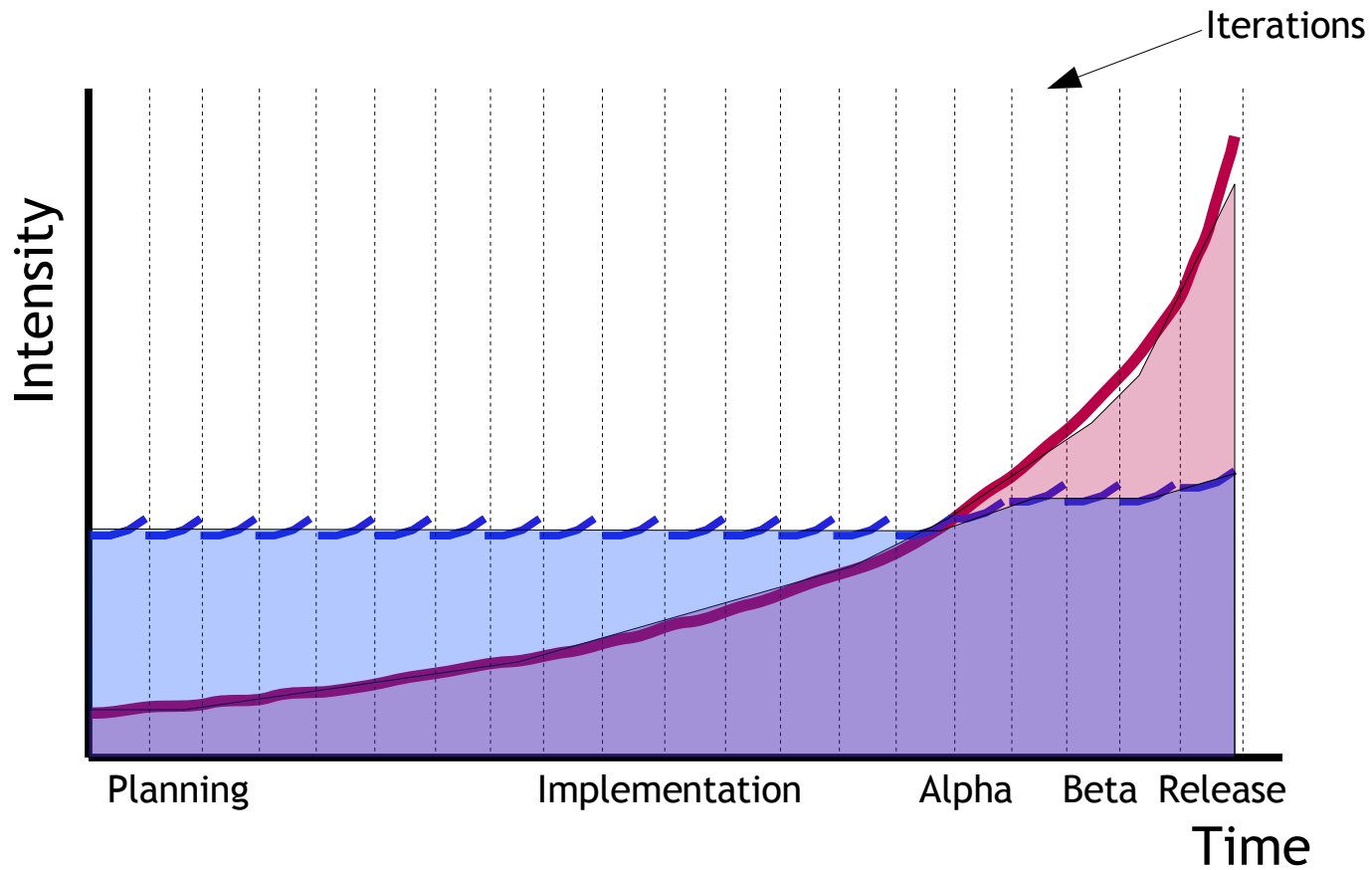
Team Composition



Sustainable Pace



Sustainable Pace



PART 3: Programming Agile Techniques

Pair Programming





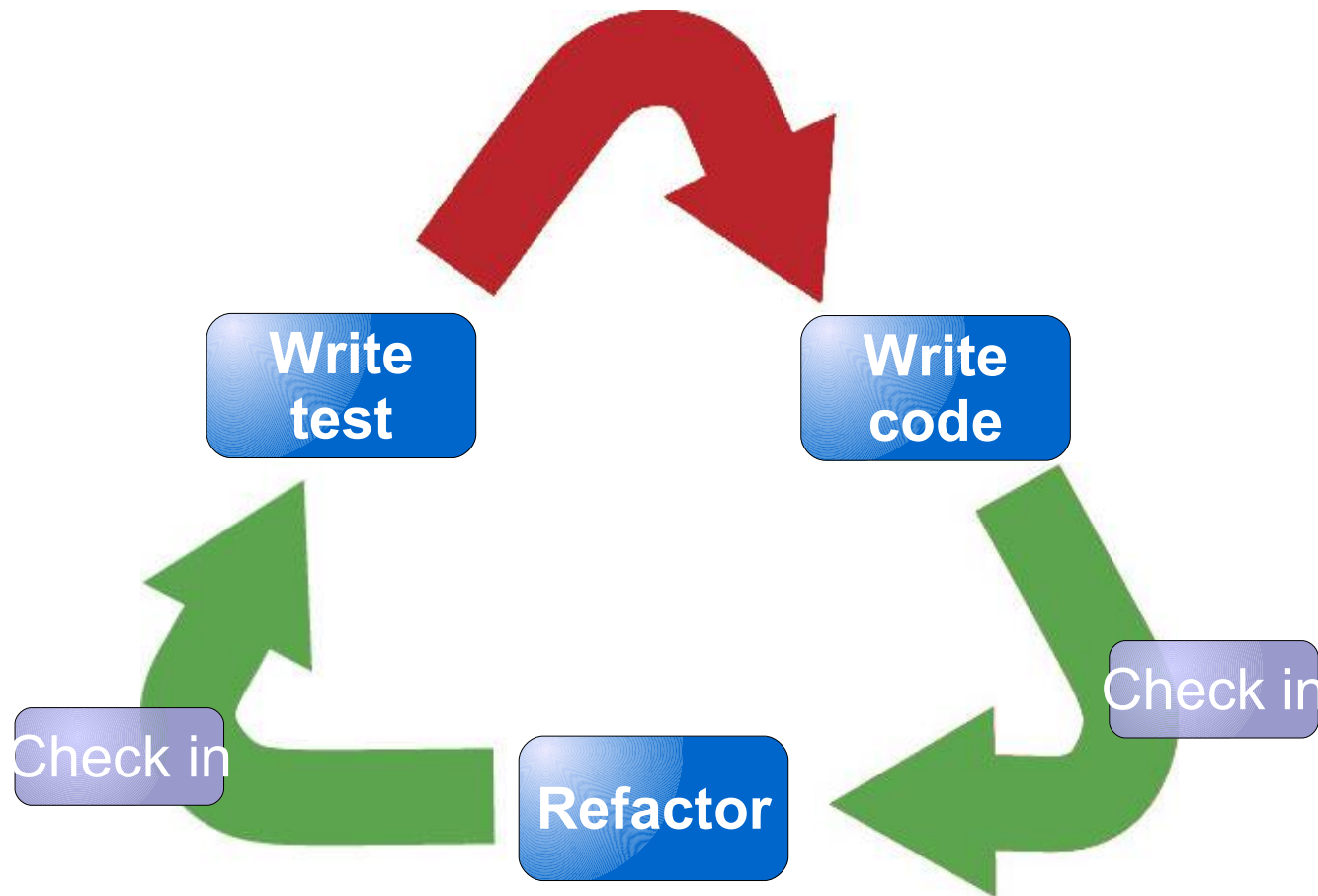
Pair Programming

- Is output 2x of what one programmer can do? Not quite.
- It's a bit less (around 1.7x), but also has other huge benefits:
 - Much higher quality
 - Spread knowledge/philosophy
 - Team spirit
- In the long term is a huge win

Pair Programming

- Not everybody likes it, but a very large number of programmers do.
- Some teams use it almost all the time, some other teams not as much.
- Another benefit: Intensity and concentration. You never have low moments, little breaks, or anything. You work 8 hours and you end up exhausted!

Test-Driven Development



Test-Driven Development

- Quick idea: test-code-refactor cycle.
- All code unit-tested. Written by programmers.
- Design methodology.
- Much easier to refactor/optimize

Test-Driven Development

- Get a unit-test framework (UnitTest++)
- Running on the 360 is a bit more challenging, but doable.
- Need to be able to run a program, capture its output and return code.
- Xbreboot should really do it.
- We used the XboxManagerClass API

Continuous Integration

- Many small, very frequent check-ins (maybe every 3-5 minutes).
- As soon as any code is checked-in, build is triggered.
- People are notified right away of any failed builds
- Unit tests very useful keeping things stable.
- We use CruiseControl.Net. It's great!

CruiseControl.NET - Mozilla Firefox

file:///C:/Documents%20and%20Settings/shoughton/Desktop/failure.aspx.htm

Slack

ilovebacon.com - dumb but fun | CruiseControl.NET | CruiseControl.NET

[Latest](#)
[Next](#)
[Previous](#)

[Build Report](#)
[View Build Log](#)
[NAnt Output](#)
[NAnt Timings](#)

Recent Builds

- 2006-03-20 16:52:48 (CCnet-Ue3-PC-Dbg-1771)
- 2006-03-20 16:34:35 (CCnet-Ue3-PC-Dbg-1770)
- 2006-03-20 16:20:33 (CCnet-Ue3-PC-Dbg-1769)
- 2006-03-20 15:36:11 (Failed)
- 2006-03-20 15:15:06 (Failed)
- 2006-03-20 15:07:07 (Failed)
- 2006-03-20 14:58:35 (Failed)
- 2006-03-20 14:34:58 (CCnet-Ue3-PC-Dbg-1768)
- 2006-03-20 14:06:10

BUILD FAILED

Project: UE3 Incr. Build (PC-Debug)
Date of build: 3/17/2006 4:08:14 PM
Running time: 00:19:26
Build condition: Modifications Detected
Last changed: 2006-03-17 16:01:58
Last log entry: - Added discriminator for idle-to-walk left and right
 - Added turn quadrant enum and selector

[rory] [gary]

Errors: (9)

Error: Warnings occurred that are considered unacceptable. Please look at the out

Unit Test Failure in TestSynAiPawn::TestControllerFocalPointIsUpdated :
 < 1.00,0.00,0.00 > and < 0.99,0.12,0.00 > are not within 0.0001

External Program Failed: C:\Program Files\Microsoft Visual Studio .NET 2003\Comm

Warnings: (1)

LINK : warning LNK4098: defaultlib 'MSVCRTD' conflicts with use of other libs; us

Unit Test Summary (C++ and Script)

Tests Run:
 2055 test(s) run
 ScriptLog: UnitTest: 2348 test(s) run

Continuous Integration

- House rule: Nobody leaves without making sure his last check-in resulted in a successful build.
- The faster your builds, the better, so pay attention to logical and physical dependencies (feedback time for normal checkins is around 10-20 seconds).
- Much slower for Unreal projects.

Collective Code Ownership

- True collective code ownership, not "no code ownership".
- Really relies on unit tests and shared knowledge.



Automated Tests

- Unit tests from TDD are only part of all the automated tests.
- Functional tests: Test whole program or parts of it
- Fully automated. Run them on the build server at least once a day (we have them running every couple of hours).
- Run them on your target platforms as well.

Automated Tests



PART 4: Lessons Learned

How Is It Working?

- We're discovering the important stuff first
- Much improved code quality
- More robust builds
- People are really happy

What Can Be Improved?

- Team composition
 - How many resources are shared
 - How to fully integrate people with different ways of working and thinking (programmers, artists, designers)

What Can Be Improved?

- Team self-organization
 - Big goal of scrum. Made lot of progress but can be taken a step beyond.
 - Team needs to realize they're empowered to do whatever it takes to get their tasks done.
 - Sometimes some leadership will still be needed

What Can Be Improved?

- Non-productive time
 - Scrum recommends 1 day for reviews and 1 day for planning.
 - That's 2 “wasted” days for every 10 working days. Even worse with multiple teams.
 - We recently changed things to minimize down time and we do reviews and planning the same day.

Adopting Agile Development

- We have customized a lot of the standard scrum rules to fit game development.
- Start with standard process, then adapt it.
- Try it out in small team working on a sub-project.
- XP was easy adopt without any major changes.

Resources

Games from Within

<http://www.gamesfromwithin.com>

Agile Game Development

<http://www.agilegamedevelopment.com>

llopi@convexhull.com

Questions?