About Me

• Training & coaching in Test-driven Development, Refactoring, OO Design Principles & Advanced Techniques for developers
• Programming since 1982
• Founder of the original Software Craftsmanship conference
• Apprentice Will Price studying Comp Sci at Uni of Bristol
• Patron of Bletchley Park Trust
• Available for Weddings and Bar Mitzvahs
The 5 C’s of Continuous Delivery

- Continuous Integration
- Continuous Testing
- Continuous Learning
- Continuous Inspection
- Clean Code
Sustainable Pace

Copyright 2002 Scott W. Amrhein
What is Continuous Inspection?

Very frequent inspection of software to provide early warning of maintainability and other code quality issues.
3 Approaches to Code Inspections

- Code Reviews
- Pair Programming
- Automated Inspections
Code Reviews
Pair Programming
Automated Inspections

...because an extensive code review every 10-15 minutes is robot’s work
Example Automated Continuous Inspection Process

- Change Code
  - Run automated inspection
- [before check-in]
- Check-in
- Confirm changes pass inspection on build server
- Success

Fix code quality issues raised

run inspections with updates from VCS

If critical issues raised, build fails

run inspections with updates from VCS
The Continuous Inspection Automation Process

- Requirement
- Acceptance Tests (examples)
- Implement quality gate
- Integration into Continuous Delivery cycle
- Feedback
Quality Gates

Soft Gate
- Quality issues raised do not break the build, but may prompt further review by peers

Hard Gate
- Breaks the build
- Can be overridden after team review
Feature Envy

As a developer, I need to know about any methods that have more than one dependency on features of another class so I can move them to where they belong and reduce coupling.
Example Examples

```csharp
public string Summary {
    get {
        return name + "", " + address.House + " " + address.Street + "", " + address.City + ", " + address.Postcode;
    }
}
```

This is feature envy

```csharp
public void RentFor(Customer customer) {
    if(isUnderAge(customer))
        throw new CustomerUnderageException();
    customer.AddRental(this);
}
```

This is NOT feature envy

```csharp
public string Sumary {
    get {
        return this.House + " " + this.Street + "", " + this.City + ", " + this.Postcode;
    }
}
```

This is NOT feature envy
Choose Your Weapon

- Cecil
- Gendarme
Implementing Your Quality Gate

exploratory testing → Pick an example → Implement just enough

Test your gate ← team discussion
Examples We Didn’t Think Of

```csharp
public string IsThisFeatureEnvy()
{
    Video video = new Video("X Men", Rating.TWELVE);
    return video.Title;
}
```
When Are We Done?
Integrating Your Quality Gate

1. Choose Example
2. Agree Policy
3. Implement Build Step
4. Test Policy
Example FxCop Build Pipeline

1. Get source files from VCS
2. Compile csc.exe
3. Test NUnit
4. Analyse assemblies FxCopCmd
5. Diff XML report with previous version mdcxml.exe
6. Test diff XML file for new problems NUnit

If any tests fail, build fails

If new errors, build fails

If new warnings, build held for review

custom rule set
Continuous Inspection Patterns
Our House, Our Rules

Beware of reusing someone else’s ruleset

Involve all developers
Code Analysis Code Is Code

Apply the same care to the analysis code and Continuous Inspection “plumbing” that you apply to production code.
Differential Inspections

Apply Continuous Inspection to new/changed code
Clean Code Check-in

Run code rules against changes before committing them
Call To Action

Report problems that developers know how to fix
Rising Tide

On legacy code, fail builds if quality gets worse. Raise the bar if quality improves.
Adopt By Stealth

Be ready to run before making it visible
www.codemanship.com