

PHP & AGILE DEVELOPMENT

By Eddo Rotman



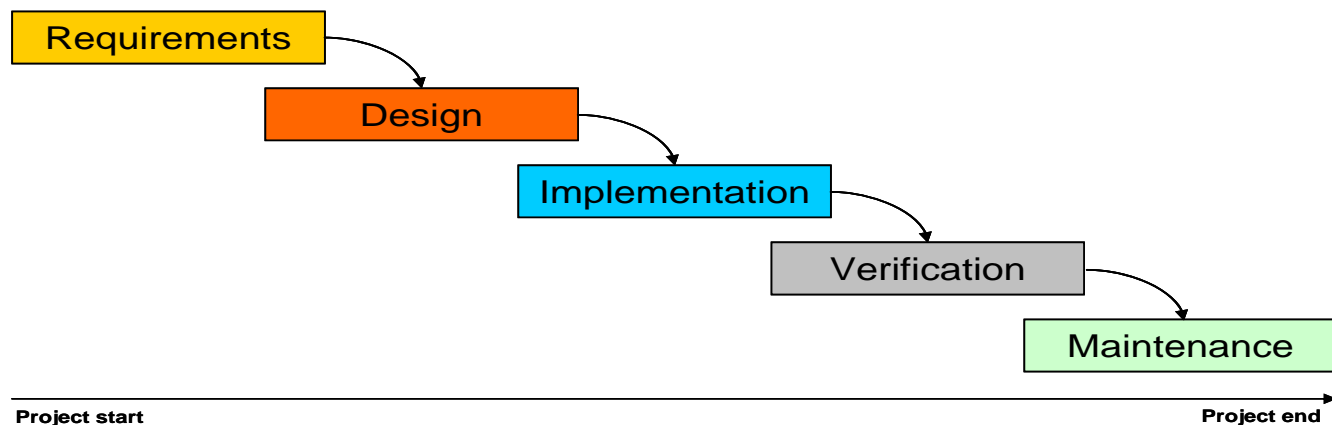
ZEND/PHP CONFERENCE & EXPO 8-11 NOVEMBER 2007

Introduction

- Developing applications without a process:
 - Is common in the PHP world though should be acceptable only for *very small projects*
 - Causes unexpected delays to planned delivery dates
 - Causes the bottom line to be over the budget
 - Could result in messy and unmaintainable code
 - In extreme cases, the result is an application that is never used – not matching requirements.

Traditional Methodologies

- Predictive – they assume that much is known before the application is created.
- They address the problems at hand with processes
- Typically large phases with lots of documentation
- They are inflexible to requirement changes



Agile Methodologies

- Adaptive processes (even the process may change on the way)
- Empirical processes (you both learn and improve as you go)
- Accept from the start that requirements will change
- Short iterations



Agile Manifesto

- Individuals & Interactions over Processes & Tools
- Working Software over Comprehensive Documentation
- Customer Collaboration over Contract Negotiation
- Responding to Change over Following a Plan

*While there is value in the items on the right,
we value the items on the left more.*

<http://agilemanifesto.org/>

Test Driven Development (TDD)

- Not an Agile Method, but a development best practice
- Tools
 - PHPfit – by Luiz Floreani
 - PHPUnit – by Sebastian Bergmann
- Logic and GUI separation
- Any domain / product owner can update the requirements with PHPfit
- Test writing is an art

Agile Methods Name Dropping

- SCRUM
- Crystal Clear
- Extreme Programming (XP)
- Lean Software Development (LSD)
- Feature Driven Development (FDD)
- Adaptive Software Development (ASD)
- Dynamic Systems Development Method (DSDM)

Method Tailoring

Agile Practices

- Coding Standards
- Collective Code Ownership
- Refactoring
- Continuous Integration
- Small Releases
- Sustainable Pace
- Pair Programming
- On-Site Customer
- Test Driven Development

Agile is *not* a magic solution

- There is a need to create a good simulation environment (for example, database mock-ups)
- There is a need to simulate the application's flow
- Requires a strong, flexible Object Oriented language such as PHP5
- Human interaction should not be taken for granted
- Relies on absolute commitment from all team members
- Needs tools like Zend Studio for Eclipse, PHPfit & PHPUnit to enforce best practices

General

- Agile development helps create code with less bugs, but it *doesn't replace testing*
- *The switch from no-methodology to agile is easier for PHP teams because:*
 - *PHP teams are usually small*
 - *Coders get their hands dirty quickly*
 - *Coders are part of the planning process*
- *Limit the risk of not delivering, since at the end of each iteration you have a subset of the working application*
- *Can-Do attitude at its best*

Conclusion

- Choose a method that fits your needs, and if there isn't one – create one.
 - Know the people you work with
 - Quality
 - Keep your customers satisfied
 - Adapt to changes
- Focus on the target – the application
 - Don't become Agile for the sake of becoming Agile
 - Use proven practices

References

- Agile Manifesto - <http://agilemanifesto.org>
- Agile Alliance - <http://www.agilealliance.org>
- Agile Journal - <http://www.agilejournal.com>
- PHPfit - <http://developer.berlios.de/projects/phpfit/>
- PHPUnit – <http://www.phpunit.de>

More questions? eddo@zend.com

