

The Generic Company
Software Development Methodology
Version 1.1

Contact: Rick Bateman
Manager, Development

Software Development Methodology

TABLE OF CONTENTS

EXTERNAL RELATIONSHIPS	3
INTERNAL RELATIONSHIPS	3
AGILE APPROACH GUIDING PRINCIPLES	3
GUIDING PRINCIPLES	3
VALUES	3
CONVENTIONS & STANDARDS	4
SOFTWARE DEVELOPMENT METHODOLOGY	4
Phase One – System Design	4
Phase Two – Iterative Feature Analysis, Design, Build & Test	5
Phase Three – Implementation & Deployment	6
REFERENCES	7

Software Development Methodology

External Relationships

1. This is an Agile Methodology and is most similar to the “Feature Driven Design” approach
2. It responds to majority of SEI/SWEBOK Knowledge Areas
3. We are currently at Level One of the CMM and plan to be at Level Two within two years
4. Our Software Documentation Specification follows the IEEE requirements outline
5. Compliance with our internal implementation of PMI/PMBOK is prerequisite to any development work

Internal Relationships

1. Project Management Methodology
2. Technical Documentation Specification
3. Design and Code Conventions
4. Technology Standards
5. Business Plans

Agile Approach Guiding Principles

1. Individuals and interactions over processes and tools
2. Working software over comprehensive documentation
3. Customer collaboration over contract negotiation
4. Responding to change over following a plan

Guiding Principles

1. Keep communications open
2. Simplicity (maintainability and managing complexity)
3. Speed – we put business goals first
4. Iterative approach
5. Frequent feature releases

Values

All our work, including methodologies and documentation, must be supported by one or more of our core values:

- it is our responsibility as professionals
- it will result in something genuinely useful
- it will result in something valuable

Software Development Methodology

Conventions & Standards

1. All our code is Object Oriented
2. We practice pair programming when useful
3. We proactively support each other by volunteering our skill sets, backgrounds and experience
4. Our roles & responsibilities are documented
5. We practice frequent peer and stakeholder reviews at strategic points
6. We have and follow conventions and standards (see appendices)
7. All our products are modular, extensible, reusable
8. We use a CVS for all projects
9. We document our technologies (see appendices)
10. We use Open Source when possible

Software Development Methodology

Preamble

Following a detailed review of industry standards and practices, the following methodology elements have been included because they comply with our values - we believe they are our professional responsibility as software developers or they are genuinely useful or valuable. Other previously common methodology components considered to be *not* in keeping with either Agile or companies guiding principals or values have not been included.

Not all steps below will be appropriate or required for every project. The following assumes a professional approach which seeks to comply with the “spirit of the law” rather than the “letter of the law”. Therefore judgment plays a significant role in the application of this methodology to each project.

What follows is largely document driven i.e. most points below refer to a document to be created and if you do the work required to create the documents identified you will be following the methodology.

Phase One – System Design

All the steps in Phase One are done at the system level. This series of steps should be reviewed for inclusion again in the Phase Two or feature level. *Judgment* must be used to determine which elements you chose to include in either level. Keep in mind that software development is not linear but that the entire process will be iterative, overlapping and necessarily involve a real world level of ambiguity.

1. Requirements Definition

- 1.1. Requirements Specifications document
 - 1.1.1. Non Functional Requirements
 - 1.1.2. User Interface Requirements
 - 1.1.3. Feature Set
 - 1.1.4. Enhancements (by version)
 - 1.1.5. Deployment Requirements
- 1.2. Base Assumptions
- 1.3. Use Cases
- 1.4. Peer Review

Software Development Methodology

2. Design & Modeling

2.1. Design Intent Document

2.2. Domain Model

2.2.1. ERD of major entities

2.2.2. A static diagram of the system including any data stores.

2.3. Architecture

2.3.1. Structural Design Diagram

2.3.2. Behavioral Design Diagram

2.4. User Interface Model

2.4.1. Drawings, html, or screenshots to dictate any major interface components.

2.4.2. Interface flowchart(s)

2.4.3. Peer review

3. Customer Review & Agreement

3.1. **Sign off documents:** documents signed off by stakeholders agreeing to things such as sanitation suite, design intent, feature set, and interface model.

4. Features & Tasks list - aka Work Breakdown Structure (WBS)

5. Prioritizing, Time Boxes and Estimating

5.1. Prioritizing & Time boxes – rather than basing our project schedules on “a project with this many features will take this much time”, our approach is to first determine with the customer what their feature priorities are and then negotiate “time boxes” in which features that can fit in the boxes, usually reflecting business driven milestones, will be delivered. These become a projects phases (not to be confused with our methodology phases).

5.2. Project time estimating – our estimating approach is simple yet reflects the most commonly used method; that of developer experience being the deciding factor.

5.2.1. First, break the project down into high level units *such as* features, components or requirements

5.2.2. Break those down further into smaller units *such as* tasks, functions, or code-elements.

5.2.3. Assign your time estimates to each in terms of hours, days or weeks as appropriate.

5.2.4. Add an amount of time *to each* as a risk factor (Murphy time) *based on your judgment*.

5.2.5. Add time amounts to account for design time and testing time.

5.2.6. Total the numbers. This total is referred to as “Ideal Time”. Now double the total amount. This doubled figure is referred to as “Calendar Time” and accounts for the fact that you will really only get to work on most projects half the time. The other half will be meetings, other project bug fixes, and other distractions. Calendar times are the estimates you give to the customer.

5.3. Peer Review

Phase Two – Iterative Feature Analysis, Design, Build & Test

1. For each feature, review the steps in Phase One for inclusion in the Phase Two. *Judgment* must be used to determine which elements you chose to include from Phase One in this level.

2. Create Use Case

3. Define Test(s) (and plan if required)

4. Peer Review

5. Build

6. Test

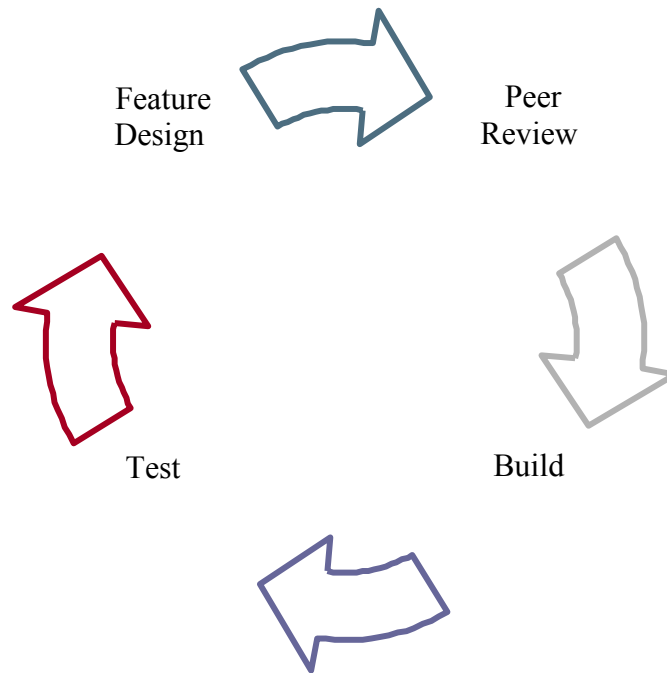
6.1. Test the build yourself, on all applicable platforms, against your Use Cases

6.2. Have another developer test, on all applicable platforms, against the Use Cases

6.3. Facilitate end user beta testing

Software Development Methodology

Phase Two Iterative Process



Phase Three – Implementation & Deployment

Source

- Key Directories and Files in Dev Working Copies (Paths, Version Controls, Description)
- Configuration Items and explanations

Deploy

- Location Information
 - Staging Location
 - Projection Location
- Client Requirements
 - Browser or Hardware Requirements
- Server Requirements
 - Software Requirements (PHP version, modules and config information, apache etc ...)
 - Deployment to server
 - Backup and Restoration plan

Data Store

- Data Stores and implementation

Software Development Methodology

References

The Agile Alliance

<http://agilealliance.org>

Black Box/White Box Testing

<http://www.faqs.org/faqs/software-eng/testing-faq/section-13.html>

Software Capability Maturity Model (CMM) Documents

<http://www.sei.cmu.edu/cmm/>

Software Engineering Institute (SEI) Software Engineering Body of Knowledge (SWEBOK)

<http://www.sei.cmu.edu/publications/documents/99.reports/99tr004/99tr004abstract.html>

Software Engineering Body of Knowledge at Wikipedia

http://en.wikipedia.org/wiki/Software_development

Project Management Body of Knowledge (PMI/PMBOK)

http://www.pmi.org/info/pp_pmbok2000welcome.asp

Software Project Survival Guide (Construx Software - Steve McConnell)

<http://www.construx.com/survivalguide/desspec.htm>

Steve is Editor in Chief Emeritus of IEEE Software magazine, serves on the Panel of Experts of the SWEBOK project, and is Chair of the IEEE Computer Society's Professional Practices Committee.