

# Towards a More Efficacious Strategy for Successful Software Development

Glenn Engstrand, Dynamical Software, Inc.

July 2008

## **Abstract**

Successful software development is inherently difficult. The primary challenge is in maintaining a sustained focus on the goals of the project. A twenty year participant in software development presents a solution that will help organizations successfully produce their own software.

## **1 The Impact of Project Failure**

Studying various reports on the I.T. industry shows a distressingly high failure rate for software development. One organization surveyed five different industry reports to conclude that 30% to 70% of software development projects fail catastrophically and only 25% are considered fully successful [6]. One of the most prestigious groups reported a 31% success rate in 2003 [5]. Not only do many projects fail but the cost of failure is high. A report on I.T. activity in 1997 - 2001 showed more than \$75 billion lost on failed projects [1].

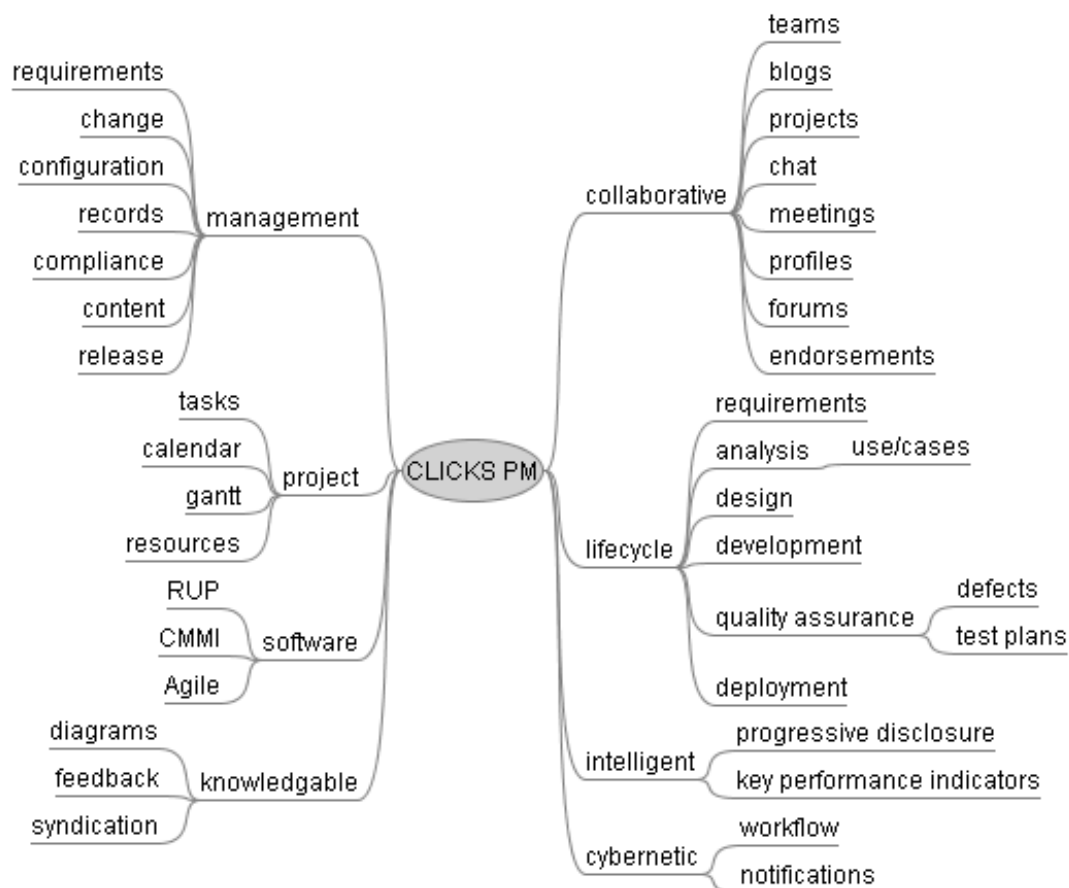
## **2 Dynamical Software, Inc.**

Dynamical Software is committed to producing software that is powerful and remains so by evolving over time to market conditions in order to stay relevant. Its founder has demonstrated over 20 years of participation in the field of business application software development in the engineer, architect, and director roles. Dynamical Software possesses a large body of knowledge and expertise over what works and what doesn't when it comes to writing successful software.

## **3 Code Roller™**

Dynamical Software introduces the Code Roller™ solution. The key to writing successful software is to get really clear and focused on the goals that the funding organization wants the software to achieve. The problem is that those goals can get really elaborate and complex and decision makers get tired having to think

about all that. So, they get overwhelmed, burn out, and just let those with less oversight take over the decision making process. That is when the project is in the most jeopardy. The Code Roller™ solution is designed to keep the team focused without getting overwhelmed by using the CLICKS PPM approach.



Mind Map of the Code Roller™ Solution

### 3.1 Collaborative

People build profiles that shows their relevance to candidate projects. Endorsements are made. Projects get formed. Teams are built by inviting people to join. People and teams also share their values, opinions, and knowledge through blogging, forum participation, and online chat. Meetings are scheduled in the calendar and can be conducted through the online chat facilities [13].

### **3.2 Lifecycle**

The lifecycle of a software development project is well understood and is reflected and represented in this solution. In the inception phase, requirements are captured. In the elaboration phase, analysis is done on those requirements in order to prioritize and predict the outcome of solving the business needs. In the construction phase, designs of the intended software are made and executed on. In the transition phase, testing is planned and executed until a reasonable and acceptable assurance of quality is achieved. After which, the product is deployed [11].

These phases are not mutually exclusive. Rather, they overlap, get refactored and revisited in an iterative and incremental way [12].

### **3.3 Intelligent**

This solution encourages collective intelligence through its problem solving, goal oriented, business focused, social networking features [9]. Through its use of the visual design paradigm known as progressive disclosure [10]. Through its Key Performance Indicator oriented reports such as burn down and earned value charts [8].

### **3.4 Cybernetic**

Cybernetics is the study of the structure of complex systems as it relates to communication, control, and feedback [2]. The Code Roller™ solution uses cybernetic principals in its workflow component where any and all milestones and deliverables can be configured to go through a feedback, review, and approval process that harnesses a complete email and dashboard notification component.

### **3.5 Knowledgeable**

Knowledge management as it relates to organizational learning and intellectual capital is exploited by the Code Roller™ solution through the content management and syndication components, the elaborate feedback system, and the collaborative diagramming component [7].

### **3.6 Software**

Without getting pedagogical or evangelical, the Code Roller™ solution encourages the Agile principals of embracing change, close collaboration, measuring progress, promoting sustainability, and achieving a tight feedback loop through continuous integration [3]. The Code Roller™ solution is designed with the understanding that each organization currently operates at its own level of maturity [4]. Code Roller™ is a web based solution that depends on Apache, MySQL, and PHP.

### 3.7 Product

Product management is a major focus of the Code Roller™ solution. Projects are organized into workflow empowered releases which are collections of requirements and subsequent feedback.

### 3.8 Project

Project management is another major focus of the Code Roller™ solution. Requirements sift through a matrix of analysis and design activities in an easy to track and verify way. These activities create milestones and deliverables which take time and attention. Both time and attention affect a cost on an organization and so they must be tracked and measured. Tasks are assigned to resources and scheduled in a calendar. Traditional Gantt and resource leveling tools are also included.

### 3.9 Management

Code Roller™ provides components for the intelligent and effective facilitation of the many different aspects of managing a software development project.

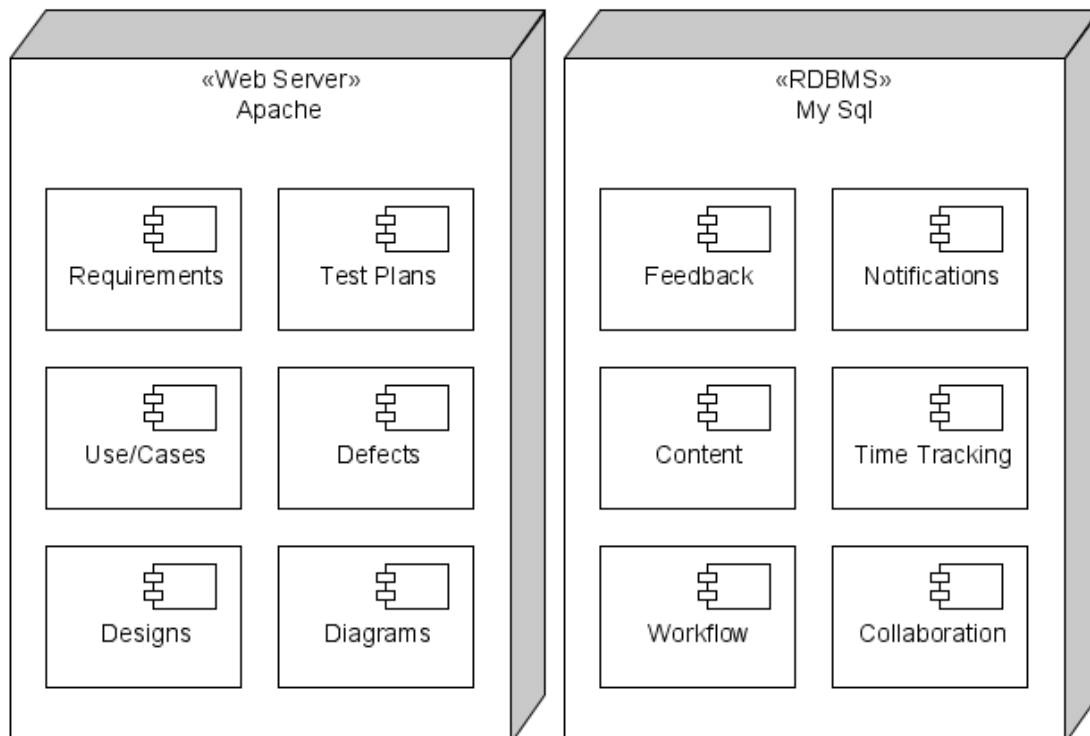
- **Requirements Management** tracks and measures the flow of requirements through the process and ensures that no important requirement be dropped or misinterpreted. Code Roller™ handles requirements management through its requirements component, its workflow component, and its feedback component. Requirements are tied to design personas through a hierarchy of need in order to facilitate requirements capture and prioritization.
- **Change Management** tracks and measures the flow of feature change requests and defects through the process such that management can gain corrective insight and the quality and quantity of output from its developers and analysts. Code Roller™ handles change management through its defects component and through the versioning features of its use-case component.
- **Configuration Management** assures quality of the candidate software project under the various configurations that it is designed to operate. Code Roller™ handles configuration management through its test plan component.
- **Records Management** tracks and measures the recording of the filing, storage, and archival of deliverables for the purposes of retention and expiration. Code Roller™ handles records management as a part of its content management component.
- **Compliance Management** tracks and measures various performance indicators that are important to governing organizations and oversight committees. Depending on the level of maturity of the operating organization,

Code Roller™ provides compliance management through its records management features, through its workflow component, and through its time tracking component.

- **Content Management** organizes documentation under polyhierarchical taxonomies that are searchable. Code Roller™ has an advanced content management component.
- **Release Management** gives management predictive insight as to when a release will be available. In Code Roller™, requirements are tied to releases and use-cases are tied to requirements. Designs and test plans are tied to use-cases and defects are tied to test plans. All tasks are tied to one of these deliverables so all quality assurance is related back to and therefore grouped by release.

## 4 Components For Success

Here are the components to Code Roller™ and how each component contributes to a successful software development project.



Architecture Diagram for Code Roller™ Components

- Requirements are captured and tracked to avoid losing any important features in the long road from inception to delivery.
- use-cases are the analyzed requirements that can be prioritized and estimated. use-cases are reviewed for quality.
- Designs on how to implement the use-cases are reviewed for quality.
- Test plans are built from the use-cases to ensure quality across all configurations. Test plans are reviewed for accuracy.
- Defects are the measured discrepancies between test plans and the actual implementations. Quality is measured from the injection and removal of defects.
- Collaborative diagrams help teams intelligently analyze and reason about the solutions that the candidate software project is attempting to make. Any deliverable can be tied to a diagram.
- An elaborate feedback system of all the deliverables is in place to facilitate the interactivity of the review process.
- Team members on a project can elect to receive dashboard and email notifications when a deliverable has been reviewed or has incurred a change in status or when an event of theirs is upcoming or a task of theirs is scheduled to complete.
- All content is managed with support for workflow, syndication, expiration, multiple categorization and searchability. Tag clouds are also available for documents.
- Tasks and their dependencies can be entered and team member time can be tracked, both estimated and actual. This facilitates predictions as to the completion of releases for general availability.
- Reviewed deliverables go through a workflow process in order to facilitate peer review and executive oversight.
- Teams are formed and disbursed to handle projects. Users collaborate through traditional social networking features to collectively overcome the challenges inherent to software development.

## 5 Feature Matrix

The first column list the various objects in the system and the subsequent columns are the high level actions that the system can perform. Each cell describes the level of applicability of the particular action to the specified object.

<b>Objects</b>	<b>Review</b>	<b>Rate</b>	<b>Endorse</b>	<b>Notify</b>	<b>Approve</b>	<b>Restrict</b>
Blog	Entries	No	No	Entries	No	Public
Chat	Messages	No	No	No	No	Yes
Contact	No	No	No	No	No	Yes
Content	No	No	No	No	No	Yes
Defect	Yes	No	No	Yes	No	Project
Design	Detailed	Yes	No	Detailed	Yes	Project
Diagram	No	No	No	No	No	Yes
Document	No	No	No	Yes	No	Content
Endorsement	Yes	No	No	No	No	No
Event	No	No	Yes	Attendees	No	Historical
Impact	Yes	No	No	Yes	No	Project
Need	Yes	Yes	No	Yes	No	Project
Profile	No	No	No	No	No	Public
Project	No	No	Per Team	No	No	Yes
Release	No	No	No	No	Yes	Project
Requirement	Yes	Yes	No	Yes	No	Project
Revision	Yes	No	No	Yes	Yes	Content
Task	Yes	No	No	No	Yes	Project
Team	No	No	Member	No	No	Public
Test Plan	step	No	No	step	Yes	Project
Topic	Yes	No	No	Yes	No	Yes
Use Case	part	Yes	No	part	Yes	Project

## References

- [1] TIWANA, AMRIT, KEIL, MARK, *The One Minute Risk Assessment Tool*,  
<http://www.acmqueue.com/modules.php?name=Content&pa=showpage&pid=239>
- [2] AMERICAN SOCIETY FOR CYBERNETICS,  
*Foundations: The Subject of Cybernetics*,  
<http://www.asc-cybernetics.org/foundations/definitions.htm>
- [3] AGILE ALLIANCE, *Principles Behind the Agile Manifesto*,  
<http://agilemanifesto.org/principles.html>
- [4] CARNEGIA MELLON SOFTWARE ENGINEERING INSTITUTE,  
*Process Maturity Profile*,  
<http://www.sei.cmu.edu/appraisal-program/profile/pdf/CMMI/2006sepCMMI.pdf>
- [5] JOE MARASCO, *Software development productivity and project success rates: Are we attacking the right problem?*,  
<http://www.ibm.com/developerworks/rational/library/feb06/marasco/index.html>
- [6] IT CORTEX, *Statistics Over IT Projects Failure Rate*,  
[http://www.it-cortex.com/Stat\\_Failure\\_Rate.htm](http://www.it-cortex.com/Stat_Failure_Rate.htm)
- [7] DORFMAN, PETER, *Executive Validation for KM*,  
<http://www.kmworld.com/Articles/News/News-Analysis/ITIL-3-executive-validation-for-KM-40807.aspx>
- [8] COCKBURN, ALISTAIR, *Earned-value and burn charts*,  
[http://alistair.cockburn.us/index.php/Earned-value\\_and\\_burn\\_charts](http://alistair.cockburn.us/index.php/Earned-value_and_burn_charts)
- [9] MALONE, THOMAS, *MIT Center for Collective Intelligence*,  
<http://cci.mit.edu/about/MaloneLaunchRemarks.html>
- [10] NIELSEN, JAKOB, *Progressive Disclosure*,  
<http://www.useit.com/alertbox/progressive-disclosure.html>
- [11] RATIONAL SOFTWARE, *Rational Unified Process*,  
[http://www.ibm.com/developerworks/rational/library/content/03July/1000/1251/1251\\_bestpractices\\_TP026B.pdf](http://www.ibm.com/developerworks/rational/library/content/03July/1000/1251/1251_bestpractices_TP026B.pdf)
- [12] CONTROL CHAOS, *What is Scrum?*, <http://www.controlchaos.com/about/?SID=8ef7eb5b2a069a2710abef27d02c851f&SID=7da824062baf60b8e78ec5f99836f092>
- [13] HINCHCLIFFE, DION, *A checkpoint on Web 2.0 in the enterprise*,  
<http://blogs.zdnet.com/Hinchcliffe/?p=130>