

## C3P – C3 Programming

Author: Mr. Shimon Rothschild

Email: [Shimon@Shimon.US](mailto:Shimon@Shimon.US)

web: <http://www.shimon.us>

Copyright 2007 All rights reserved.

### The Problem Defined

Most software projects fail. The definition of failure is: 1) fails to meet the stakeholder (end user) minimum requirement or (2) A project significantly over budget and/or significantly late. This is not because of negligence and incompetence. Some projects will fail because of the high degree of uncertainty associated with leading edge applications. Most will fail because of miscommunication. The largest area of miscommunication is between these roles. The business analyst transfers business requirement to programmers and programmers are expected to define for quality assurance when the software is ready for testing. This is the area where business requirements and the software implementation meet. This is also an area where the cost of repair is higher than fixing programming defects. If there would be a method to better communicate expectations and visualized implementation, this would greatly reduce miscommunications.

One area identified as a significant source for miscommunications leading to failure is the transfer of knowledge from business analysis to programming. A second source identified as a significant source for failure is the fog of software expectations. Both of these sources are identified with the role of programmer and specifically the relationship between the programmer and other roles in the software development cycle.

### The Solution

Programmers focus on coding and the role of programmer need to be expanded. The programmer needs to take responsibility for ensuring that the business requirements will be correctly interpreted into software. The programmer also needs to be fully cognizant of

C3 Programming (C3P) addresses these sources for failure by explicitly assigning responsibility to the programmer. Programmers will feel comfortable assuming these additional roles because it is intuitive to programming. On a macro level it is analogous to a function call, input, operations and output. The inputs are a contract of expectations by business and the output is expected results. Coding is the operation.

### What is C3 Programming (C3P)?

C3 is a concrete process for fostering better communications. C3 programming, (C3P), defines how to successfully write software. C3P defines programming as three phases, **C**ontract, **C**ode and **C**lose. Programming is a component of the software development lifecycle (SDL) and C3P breaks down the programming component into a more pronounced phases.

## Overview Software Development Lifecycle (SDL)

### Software Development Lifecycle (SDL) Overview

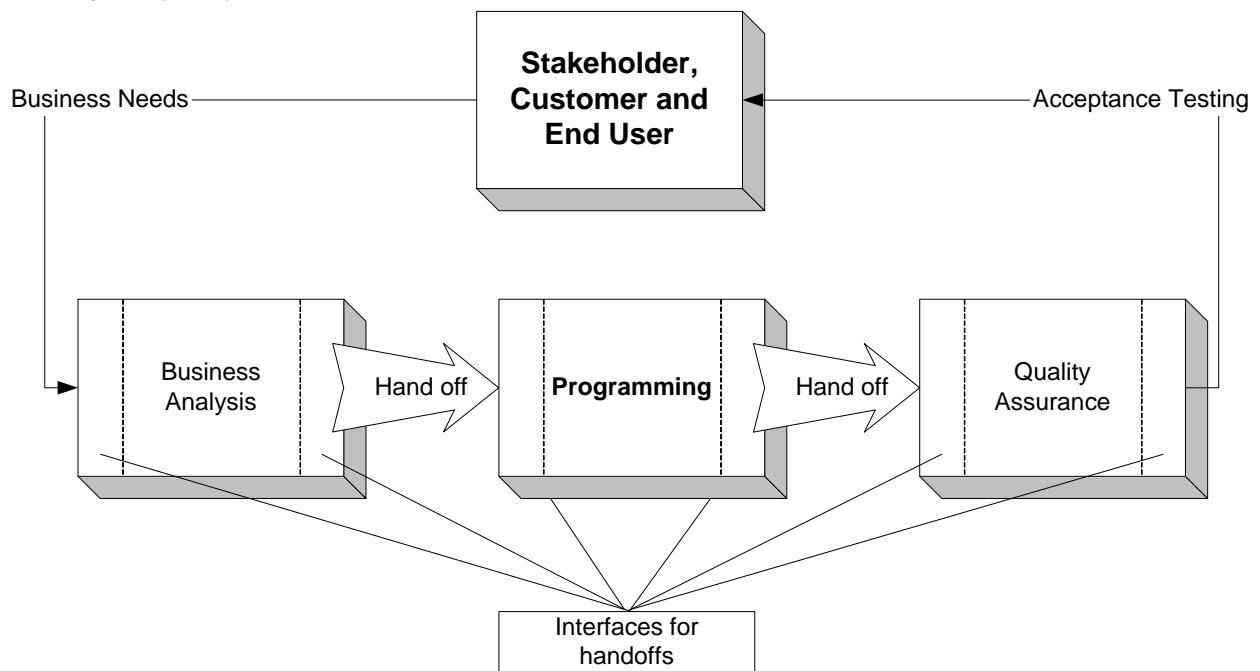


Figure 1: Software Development Lifecycle Roles

The SDL is broadly broken up into four roles. Every major SDL model (CMMI, Waterfall, Test driven “TDD”, Agile and Extreme “XP”) encompasses these roles in this order. Models differentiate themselves by assigning roles and by frequency of cycling through the loop. The stakeholder is the ‘boss’. The remaining roles are designated to implement the vision of the stakeholder. The business analysis role is to make practical the vision and capture contradictions and gaps in the vision as well as prioritizing the work. Programming is the design and writing of software code to accomplish the work. C3 programming defines the internal processes in programming. The quality assurance role validates that the software code meets specified minimum conditions.

### Overview of the Programming Role

Programming is more than writing code. Writing functional specifications are the blueprints for the code and this is still fundamentally part of writing code. C3 programming defines two additional components in the programming role. One is backward facing, working with business analysis and the second is forward facing, working with quality assurance. Both of these additional components expressly focus on interfacing with other roles. Better communication between the roles will result in fewer defects. The programmer reflects back to the business analyst the perceived business need as implemented with software. The programmer defines for quality assurance, before coding, under what condition the code is ready for testing. Of the two additional roles, interfacing with quality assurance is being implemented with test driven development (TDD). Ironically, the significantly more expensive errors are between business analysis and the programmer and this needs further development.

## C3 Programming (C3P) Contract, Code, Close

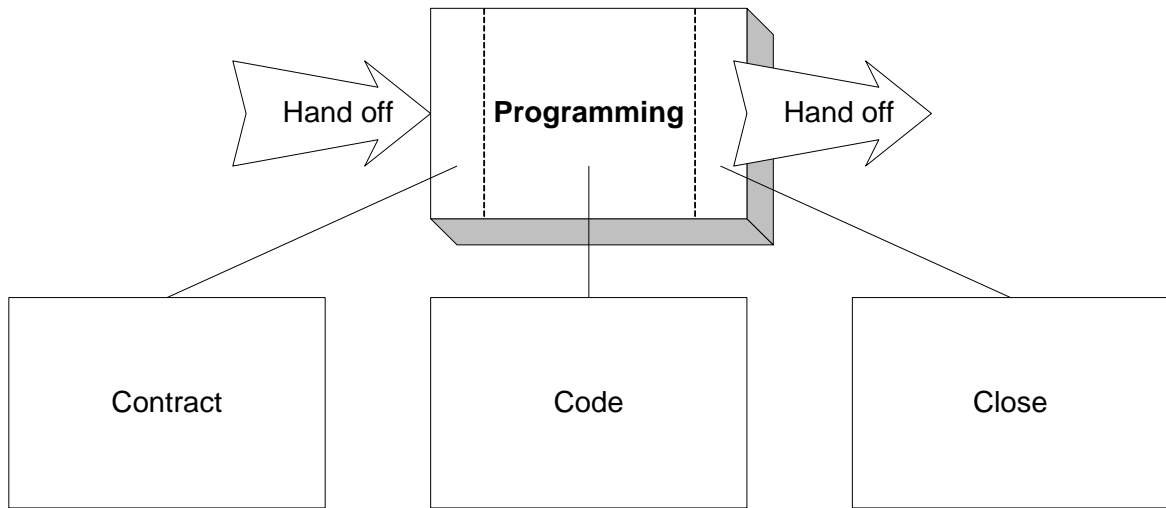


Figure 2: The Three “C”s Contract, Code and Close

C3 Programming focuses on the three components of the programming phase. The three components are the two interfaces between other teams and the internal development or ‘coding’. The middle component, coding, is the most practiced and most familiar to programmers and software development teams. The contract component represents the interface with business analysis and the close component represents the interface with quality assurance. The three components make up the role of “programmer”.

### C3P Components

Each of the C3P components could happen in any order and even simultaneously. Most commonly there is some contract component prior to coding or close. Test driven development (TDD) emphasizes the close component prior to coding. Programmers are proficient in coding. Programming tools and development environments have been designed to facilitate writing code. There exist add-on modules to facilitate closing and tools to create contracts, but the bulk of programming effort is in coding. This balance needs reevaluation.

#### Contract

This is a binding agreement between business and programming. Business provides requirements in some format, spreadsheet lists, business use case diagrams or process diagrams. The process of transferring the business requirements to a software “plan of action” is informally the contract phase. This phase needs to be formally framed. The contract reflects back to the business that which the programmer will provide. The contract defines from a programming perspective, in programming/business jargon, how the programmer understands the business needs. The programmer then works from this document in developing the code.

## Code

This component is the most familiar component to the programmer. This is the functional requirements, architecture, software interface design, algorithms and coding of instructions that make up software. Existing programming tools have focused almost exclusively on this component of programming. Code generators, debuggers and code analysis tools all contribute to reducing defects in code and have done admirably well in this domain. In C3P, this is the “C” that requires the least attention.

## Close

When is software complete? When there is no clear, unambiguous, finish line, the software is never complete. Incomplete software does not get delivered. For this reason, ‘the finish line’ must be clearly demarcated for the programmer. Most of the time the finish line is implied and as rules in business change, the finish line gets moved. The solution to control shifting finish lines is an explicit set of objectives that shall be reached to conclude that the software is complete. Test driven development (TDD) controls this by requiring that the programmer formally state the tests that the software shall pass to qualify the software as finished.

## C3 Integration with the Software Development Lifecycle

C3 integrates seamlessly and complements the SDL. C3 more finely defines the interface and expectations in the hand off between different roles in the cycle. The roles of contract and close can be assigned to others on behalf of the programmer. Business analysis could take responsibility for contract and quality assurance for close. However, the value of having the programmer fill these roles is valuable because it both crystallizes the solution in the mind of the programmer and reduces risk of misunderstandings. It’s analogous to asking a person to repeat, in their own words, what is understood.

## C3 and Uncertainty

Peer review is a common method to reduce error and identify to resolve uncertainty. C3 improves upon peer review by providing a view from a different perspective. Individuals in the same organization performing similar tasks, tend to follow similar routines and overlook similar events. By bringing the programmer to overlap with the sphere of business analysis and quality assurance, this provides an orthogonal view to a proposed solution. This alternative perception can strengthen both sides of the interface. The input from business analysis and quality assurance can serve to guide the programmer in focusing on core requirements and ignoring non critical and non testable features.

## Summary

C3P is a more fine grain implementation of the programming phase and can be implemented in any software development lifecycle (SDL) model. It adds necessary responsibility to programmers to both ensure that the programmer understands the scope of the business problem and limits the programmer to solving only the problem. An additional value is that programmer perspective to business requirements and quality assurance will provide observations not generally made by those dedicated in to those roles. This perspective can identify areas of ambiguity and opportunities where programming can provide additional added value.