

# Software Development Methodology

## Table of Contents

1. BACKGROUND .....	2
2. SOFTWARE AND CONSTRUCTION .....	3
3. METHODOLOGY PHASES .....	5
4. CONCLUSION & ACCREDITATIONS.....	6

## 1. Background

The global software development scoreboard is dismal. Consider these facts from Software Engineering Australia (SEA);

- The price business is paying for comparable applications can vary by a factor of 15:1.
- 32% of projects are cancelled before they deliver anything.
- Approximately two thirds of every software project undertaken goes over budget in terms of time and cost.
- The average project cost blowout is 87%.
- Each hour spent on quality-assurance activities such as design reviews saves from 3 to 10 hours in downstream costs.
- A requirements defect that is left undetected until construction or maintenance will cost 50 to 200 times as much to fix as it would have cost to fix at requirements time.
- More generally, a defect that isn't detected upstream (during requirements or design) will cost from 10 to 100 times as much to fix downstream (during testing) as it would have cost to fix at its origin. The further from its origin that a defect is detected, the more it will cost to fix.

These facts occur because software development is a creative process that happens between the user (ie the client), the analyst and the developer – and unless the delivery process is managed in a structured way it is all too easy for this creative development to spiral out of control and not deliver on expectations.

It is for these reasons that CSA adopts a structured software development delivery methodology based on the Capability Maturity Model (CMM) and elements of Rapid Development teachings. The CMM was developed by the Software Engineering Institute (SEI) to offer software developers a quality assured framework by which to work from. This framework minimises risk to the client and risk to the software developer.

The phases of CSA's methodology are summarised in section 3 below. The methodology begins with managing the client's expectations of delivery right from the initial phases of scope and estimation. As mentioned above, software development is a creative process. It is a process of gradual refinement. You begin with a picture of what you want to build and then spend the rest of the project trying to bring that picture into clearer focus. Because the picture of the software you are trying to build is fuzzy, the estimate of the time and effort needed to build it is fuzzy also. The estimate can only come into focus along with the software itself, which means that software-project estimation is also a process of gradual refinement.

## 2. Software and Construction

Suppose you go to your friend Stan, who's an architect, and say that you want to build a house. You start by asking Stan whether he can build a three-bedroom house for under \$100,000. He'll say yes, but he'll also say that the specific cost will vary depending on the detailed characteristics you want.

If you are willing to accept whatever Stan designs, it will be possible for him to deliver on his estimate. But if you have specific ideas about what kind of house you want - if you later insist on a three car garage, gourmet kitchen, sunroom, sauna, swimming pool, study, gold plated fixtures, floor to ceiling Italian marble, and a building site with the best view in the suburb - your home could cost several times \$100,000 even though the architect told you it was possible to build a three bedroom home for under \$100,000.

It is difficult to know whether you can build the product that the customer wants in the desired time frame and budget until you have a detailed understanding of what the customer wants.

How much does a new house cost? It depends on the house. How much does a new billing system cost? It depends on the billing system. Some organisations want cost estimates to within +/- 10% before they'll fund work on requirements analysis. Although that degree of precision would be nice to have that early in a project, it isn't even theoretically possible. That early, estimates do well to fall within a factor of 2.

Until each feature is understood in detail, you cannot estimate the cost of a program precisely. Software development is a process of making increasingly detailed decisions. You refine the product concept or scope into a statement of requirements, the requirements into a design, and the design into working code. At each of these stages you make decisions that affect the project's ultimate cost and schedule. Because you cannot know how each of these decisions will be made until you actually make them, uncertainty about the nature of the product contributes to uncertainty in the estimate.

Here are some examples of the kinds of questions that contribute to estimation uncertainty:

- Will the customer want Feature X?
- Will the customer want the cheap or expensive version of Feature X? There is typically at least a factor of 10 difference in the implementation difficulty of different versions of the same feature.
- If you implement the cheap version of Feature X, will the customer later want the expensive version after all?
- How will Feature X be designed? There is typically at least a factor of 10 difference in the design complexity of different designs for the same feature.
- What will be the quality level of Feature X? Depending on the care taken during implementation, there can be a factor of 10 difference in the number of defects contained in the original implementation.
- How long will it take to debug and correct mistakes made in the implementation of Feature X? Individual performance among different programmers with the same

level of experience has been found to vary by at least a factor of 10 in debugging and correcting the same problems.

- How long will it take to integrate Feature X with all the other features?

As you can see, the uncertainty about even a single feature can introduce a lot of uncertainty into an early-in-the-project estimate. Multiplied across an entire project, literally thousands of specification, design, construction and implementation decisions have to be made before the ultimate cost of the project can be determined. As you do make a greater percentage of those decisions, however, you can narrow the estimation range as shown in figure 1 below i.e. for a large project the scope phase has an estimate range of 0.5x to 2x whereas the next phase, requirements analysis, has a range of 0.67x to 1.5x. These ranges are based on the estimate convergence graph.

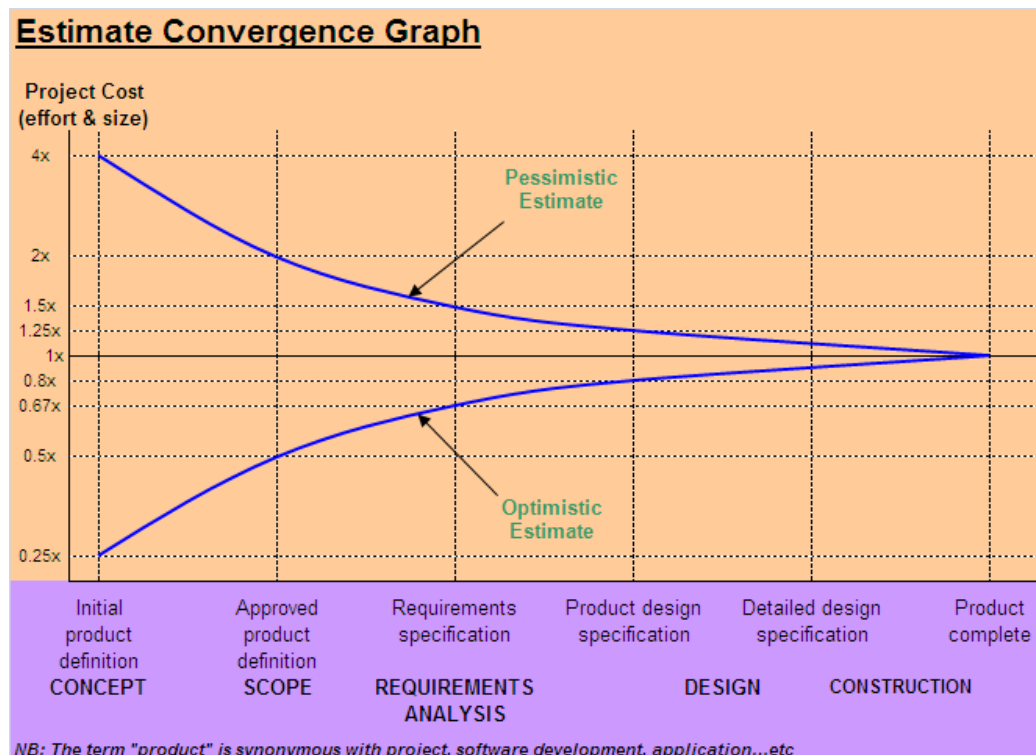


Figure 1. Estimate-convergence graph. Shows that for any given set, estimation precision can only improve as the software itself becomes more refined. Estimates can fall anywhere between the two curves. Source: Adapted from "Cost Models for Future Life Cycle Processes: COCOMO 2.0" (Boehm et al. 1995)

### 3. Methodology Phases

Phase	Deliverables
Scope	<ul style="list-style-type: none"><li>• A scope document that defines what is being looked at and what is not being looked at.</li><li>• A quote for the next phase (Requirements Analysis) and an estimate for the solution. This estimate could vary anywhere between 0.5x and 2x where 'x' = the actual cost of the solution.</li></ul>
Requirements Analysis	<ul style="list-style-type: none"><li>• The detailed definition of requirements for the particular area of the business as defined in the scope document.</li><li>• Dependent on the size and complexity of the project, either a quote (small project &lt; \$100k or low complexity) or an estimate (large project &gt; \$100k or high complexity) is given for the complete solution. The estimate could vary anywhere from 0.67x and 1.5x where 'x' = the actual cost of the solution.</li><li>• This document also includes projected return on investment and payback periods.</li></ul>
Design	<ul style="list-style-type: none"><li>• The specific application of technology to the requirements defined during the requirements analysis.</li><li>• A quote for the complete solution (if large project &gt; \$100k or high complexity).</li></ul>
Build	<ul style="list-style-type: none"><li>• Solution operating in a development environment.</li><li>• User &amp; administrator manuals.</li></ul>
System Test	<ul style="list-style-type: none"><li>• Solution operating in a test environment.</li></ul>
Implementation	<ul style="list-style-type: none"><li>• Solution operating in a live environment.</li><li>• On-site training.</li><li>• User acceptance testing.</li><li>• Initiate performance measures</li></ul>
Review	<ul style="list-style-type: none"><li>• Client satisfaction project review.</li><li>• Software engineering project review.</li><li>• Initiate continual improvement loop</li></ul>

The challenge developers' face is that the majority of customers are after a fixed price as early as possible in a project – which you can't blame them given the track record of the software development industry. So, for the majority of projects, CSA can provide a fixed price once the requirements analysis has been done. This is a detailed requirements analysis which documents process, data and roles – from which an analyst programmer can estimate and design from. The requirements document is used by the;

- Analyst programmers to determine development tasks and task effort. This allows us to provide a fixed price quote for the development of the solution.
- Project Managers to determine variations and manage the scope of the project. The estimates provided by the analyst programmers, together with the teams work schedule, create the project schedule.
- System Testers to create a system test plan and assist the customer in defining a user acceptance test plan.
- Contract management to qualify any warranty claims

## 4. Conclusion & Accreditations

CSA's goal is to seek a convergence between estimates and reality. We believe our methodology and skill sets allow us to offer a firm quote for the majority of projects once the requirements analysis has been completed. This allows us to produce the shortest possible software schedules with the most accurate estimates possible. Our methodology ensures the lowest possible risk to our clients and conversely the highest probability that the solution we offer will more than meet user's expectations and acceptance.

Computer Systems Australia has been certified to the ISO 9001 quality assurance standard, recognised as a Government Endorsed Supplier, PD 50 Endorsed Supplier and is one of the top 500 privately owned companies in NSW. While this is encouraging, we are not resting, but striving for greater success, which we believe to be directly related to our ability to deliver technology to provide a business advantage.

In our pursuit of excellence, we have developed much expertise and are proud to be recognised and acknowledged by our Business Partners for our capabilities with the assigning of industry certifications.



Advanced Infrastructure Solutions  
Networking Infrastructure Solutions  
Business Process and Integration  
Information Worker Solutions