Why Software Projects Fail and How to Make Them Succeed
~ By Duncan Haughey

Let’s start with a worrying statistic. According to the Standish Group in 1995, only 16% of software projects were successful, 53% challenged (that is cost overruns, budget overruns or content deficiencies) and 31% cancelled.

Furthermore, they say the average software project runs 222% late, 189% over budget and delivers only 61% of the specified functions. Evidence suggests little has changed since then.

Failure has become the IT industry norm. So what can we do about it?

A good starting point is by addressing some of the key reasons software projects fail.

Not Enough Time

Often, the deadline date is decided before the project starts and is non-negotiable. This deadline results in a headlong rush to get started on the assumption, the sooner you begin coding, the sooner you’ll finish.

A rush to start coding is almost always the wrong approach. It is important to spend the time to create a good design. Not having a good design leads to continuing changes throughout the development phase. When this happens, time and budget are consumed at a rapid rate.

Solution: Make time to create a good design. Don’t be tempted to jump straight in and begin coding. Assign time to this task and the rest of the project will run much better. It will improve your reputation when you deliver something that fulfils the customers’ expectations and works the first time correctly.

Insufficient Budget

Many projects have a lowest price most successful candidate policy, or an unrealistically low budget, not based on the actual requirements. When this happens, everything slows down. Resources are slow to arrive, or never arrive; corners get cut and quality suffers.

Solution: Be realistic about the budget and base it on the full requirements. Avoid basing selection of a supplier solely on lowest price. Go to a supplier or team that has a proven track record of delivery within budget.
Poor Communication

There’s an old saying, “never assume anything” and this is particularly the case for software projects. Good communication with the customer, users and especially the development team is critical to project success. Does everyone in the team understand you? Do they know what you expect of them or have you assumed they know? Do they communicate well with one another, with users and with other departments?

Solution: Find any communication breakdowns now. These can lead to confusion and complications later in the project. Never assume that everyone understands. Take time to create an environment that will bring the project in on time, on budget and to the customers’ expectations.

Never Reviewing Project Progress

As a project progresses, things change and these changes can have a significant impact. It is important to keep examining the project progress so you can overcome challenges early, and warn stakeholders of possible delays and changes to the product.

Solution: Set various milestones during the project when you can review progress with your team and adjust as necessary to stay on course. Stay close to your team so you understand what is going on, and the challenges they face.

Inadequate Testing

When the pressure to deliver is on, it is often testing that suffers. All the testing gets left until the end of the development cycle and only lip service paid to it. Often, the result is a product filled with bugs and an unhappy customer.

Solution: Carry out testing throughout the development lifecycle, testing each module or component as it’s developed. This approach leaves just the integration testing to be carried out at the end of the development lifecycle.

Testing in the Production Environment

It’s surprising how many organisations test products in their production environment. Using the production environment in this way is a high-risk strategy that can lead to security breaches and release without testing, disrupting the production systems.

Solution: Develop a process for quality assurance and release of new software products. Provide an environment separated from production for testing and bug fixing.

Lack of Quality Assurance

Often in the haste to deliver the software, quality assurance suffers. Documentation is incomplete for code changes, the design contains flaws, and implementations can be unfinished. These all lead to rework, lost time and eventually unhappy customers.

Solution: Take time to quality check and document software before it is released.
Not Conforming to Industry Standards

Conforming to industry standards in your software projects can prove beneficial by ensuring accessibility, portability, usability, robustness and reducing problems now and in the future. Bodies such as the International Organisation for Standardisation (ISO) and World Wide Web Consortium (W3C) have developed open standards that when used are hard to challenge.

**Solution:** Take time to introduce standards for your projects. Find what works well and keep doing it. Change anything that isn't working. Review and update your standards regularly.

Next time you run a software project, review this list first and remind yourself what is needed to ensure your success. You’ll be surprised at the difference it makes.