

Reducing Costs to Increase Quality

Hard times offer opportunities

Keith Braithwaite

The present difficult times for business offer great opportunities for those who are prepared to embrace change. These opportunities extend into the realm of IT. In-house development groups can use the current focus on business survival to overcome resistance to change. Purchasers can demand that external suppliers change to deliver more value for less. Suppliers can take competitive advantage of customer's desire to reduce costs and improve their own business.

In some industries the downturn has directly generated new IT work. Integrating the information systems of two banks after an acquisition or merger is a huge undertaking, for example. In other areas the desire to save costs through automation of existing business processes becomes all the more pressing. Throughout the economy line-of-business work and the stream of enhancements and new features required to support it does continue, albeit at a reduced pace. Everywhere expenditure is becoming harder to justify and clear benefits more in demand.

This is fertile ground for process improvement. Specifically, wasteful working practices are a luxury that IT departments can no longer afford. Consider projects with a large systems development component, although similar principles apply to other kinds of project. During the good times a development project can be late and over budget without causing serious concern. Most projects do end up that way, after all, and the sky didn't fall. The delivered system can be defect ridden and inadequate to its users' needs, likewise. With plenty of money in the bank and plenty of work in the pipeline there is a tendency to chalk a failure up to experience and move on. We can imagine that if high quality were delivered on time then the project would be more valuable, but even a late or buggy new trading system (for example) can enable enough extra revenue that the late or buggy parts of the delivery may be overlooked.

That sort of project is still unsatisfying for all concerned, albeit tolerable during the good times. In the bad times, however, it becomes increasingly intolerable. If the state of the world is perceived to be bad enough and the desire to add value strong enough then it can be that the status quo becomes completely unacceptable. Then, previously unthinkable changes become possible.

Most mainstream development organisations work in extremely wasteful ways. "Waste" here is to be understood in the sense borrowed from Lean manufacturing: activities that do not

directly add value. For good historical reasons and with the best of intentions most development teams spend a lot of time involved in activities that do not add value. This test has to be applied very strictly: for example, planning a project does not add value and so is "waste". Corporate governance may require non-the-less that a certain amount of planning activity goes on, or we may have some other reason to do some planning. In the interests of efficiency we should then look to do the least possible amount of planning, do whatever planning remains in the most effective way possible, for the least outlay, and do it at the most effective time.

Many current IT organisations, particularly in large corporate settings, have slowly accreted a range of procedures for developing software that have left them needing to spend weeks or months preparing a project. One bank development organisation spent six months performing analysis activities on the requirement for a single web page, a "configurator" style workflow with a handful of steps. Even after that exercise the development team did not feel ready to being construction.

When a business has moved into a survival mindset it is remarkable how much of the habitual project planning activity that seems so crucial during the easy times turns out to be dispensable. This is avoidable waste that the organisation has accumulated over time and by choice. With a publicly visible deadline looming the configurator project was restarted using Agile methods. By bringing the line-of-business people and developers together for a week to directly explore the requirement, rather than work through paper exercises, a prototype page was working within ten days and brought to deliverable quality in six weeks.

There is also avoidable waste that appears to be built in to software development. For example, the unplanned re-work of defect fixing is not a value-adding activity. However, the defect rate of a system does have to be below some threshold for the business sponsors to be happy.

There are several approaches to that. Some can challenge long-held beliefs. In 1970 W.W. Royce first described what came to be known as the "Waterfall" approach to software development, later institutionalised as the Software (or System) Development Life Cycle. Royce said back then that the waterfall approach "is risky and invites failure." The principle source of risk that he describes is in post-hoc testing, in which the developers' assumptions, as realised in the system, are compared against reality. Reality always wins. A failing test towards the end of a

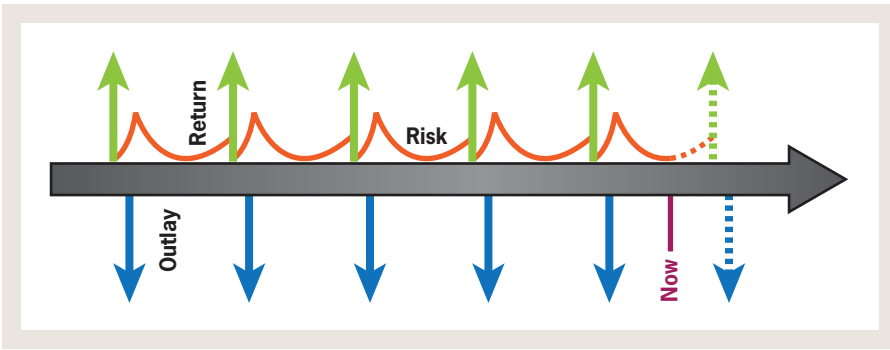


Figure 1:
**Risk Profile of an Iterative,
 Incremental Development
 Project**

Source: Keith Braithwaite

development episode injects an unknown amount of unplanned re-work into the project, invalidating the planning assumptions. Following the lead of manufacturing, it turns out to be much less wasteful to design quality in to systems than to inspect defects out. Contemporary practices such as Test Driven Development achieve this (TDD has been shown to reduce defect rates up to 90% for a moderate increase in development time) but require sometimes radical changes in thinking. The current pressure to increase efficiency can provide the spur to overcome that resistance.

One team had been dogged by user complaints regarding low quality, even with intensive and lengthy manual testing phases. By adopting the Agile practice of automated testing, both at unit (or developer) and acceptance (or user) level they were able to deliver an increment of functionality that exhibited no failures in use. And revealed several latent defects in the existing systems to which it interfaced. The time spent not diagnosing, fixing and re-testing the defect reports not raised was saved.

The so-called “Agile” development approaches are demonstrably aligned with the Lean principles of minimising waste, and so should be attractive in straightened times. However, being able to use them requires that the questions posed at the inception of a project allow them as an answer. The worst case is to demand a fixed price for a fixed scope.

Government IT projects are amongst the worst (and most visible) offenders in this. For example, by asking suppliers to bid to build “an” IT system for “the” NHS, a gigantic undertaking, and then attempting to manage risk through punitive contract terms the conditions for efficient delivery are ruled out from the beginning. Too many businesses ask similar questions on a smaller scale. This is doubly punishing for the business as risk and outlay are both increased. Large systems built by long projects delay (and therefore reduce) any return on investment. By building a small system more quickly, technical and commercial risk are constrained and the return delivered nearer the outlay.

One team working in a finance house were able to bring new features to their users only two or three times per year. By moving to incremental delivery using Agile they were able to bring new revenue-earning features into use within six weeks. The Agile practice of intensive automated

testing both assured their internal quality team that this system would not damage the business, even though it handled very high value transactions, and satisfied industry regulators that suitable validation had taken place.

Most successful projects do not deliver all of their initial scope (estimates vary, but a gap of approximately 20% is typical), most successful products have features that no user needs or wants. By running a series of short projects, or running a larger project in several iterative steps, the waste of building un-needed features can be reduced. Each iteration boundary is an opportunity to choose what not to develop in the next iteration. Internal development projects run this way are easier to manage, less risky and offer improved ROI. But organisations that are not used to this way of working will resist the change.

Projects developed by external suppliers often contain all the wastes seen in internal projects, plus another entire class of waste: contract negotiation and change control. Prolonged wrangling of contracts is an expensive symptom of lack of trust between vendor and purchaser. This can seem necessary if the large project route is chosen, whereas an iterative approach allows for trust to be developed through demonstrated delivery.

Change control processes are a symptom of lack of trust. Suppliers don’t trust customers to have told them what they need, customers don’t trust the supplier to be quoting a fair price. The effort that goes into change control is pure waste. This waste can be reduced to zero, eliminated completely, if only the customer gives their best effort to manage their needs (and the supplier believes that they do) and the supplier gives their best effort to deliver (and the customer believes that). Even on fixed-price projects, even with outsource development, an open, trusting relationship between customer and supplier can eliminate the waste of change control. Agile development, being iterative, incremental and evolutionary, opens the door lower cost styles of interaction. Lengthy projects can be managed as a sequence of short fix-price episodes (perhaps as short as two weeks). This reduces the exposure of both purchaser to under- or mis-delivery and the supplier to outlays on development that might not be paid for. At the end of each episode delivery and direction can be assessed and modified is necessary. This sort of arrangement can require a painful change to the administration of contracts so one would need a good reason to do that, such as an urgent need to reduce costs and risk. Such a need exists now.

All of these efficiencies and more are available to teams, departments and enterprises that choose to adopt them. Hard times are a test of the resilience of an organisation. One approach is to retreat into the comfort zone, to carry on doing the things that worked during the good times but do them somehow “better”. Another is to take advantage of the pressures of the new operating regime and make radical changes for the better.



Keith Braithwaite
 Principal Consultant,
 Zuhlke Engineering Ltd
 kbr@zuhlke.com