Guest blog on **http://ukcampaign4change.com/** on 13 June 2011

Agile can fix failed GovIT says lawyer

In a guest blog, commercial lawyer Susan Atkinson argues that agile development is not an evangelical fad ill-suited to government IT.

The blog by Alistair Maughan in *Computer Weekly* in which he argues that *'Agile will fail GovIT'* is quite extraordinary [1]. It is extraordinary in that it completely overlooks the poor track record of GovIT to date. It also makes a damning attack on the adoption by government of agile without explaining the potential benefits.

The state of GovIT

The Government spends about £16bn per year on IT. The spend has been growing steadily in recent years and, without radical intervention, shows no sign of abating. [2] A compelling number of studies has found that about one quarter of all IT projects (in both the public and private sector) are cancelled and about half are delivered late, over budget or both. [3] This would suggest that public funds in the order of several billion pounds per year are being invested by the Government in failed IT projects.

In 2005 Edward Leith, Chairman of the Public Accounts Committee, commented that:

"far too often major IT enabled projects in government departments are late, well over budget, or do not work at all – an enormous waste of taxpayers' money" [4]

The problem is so serious that shortly after coming into power the Coalition Government introduced the ICT Moratorium, under which any new ICT contracts and contract extensions/modifications above a value of £1m could not be entered into without specific agreement by the Treasury.

The waterfall model

Why is the track record of IT projects so dreadful? Until fairly recently virtually all IT projects have been managed using the waterfall model. The waterfall model enshrines a sequential development process, in which development is seen as flowing steadily downwards – like a waterfall – through the phases of conception, initiation, analysis, design, construction and testing. The output of each phase provides the input for the next stage.

There are two very significant consequences of the waterfall model. Firstly, all of the requirements of the customer are specified before the project starts. However, this fetters the ability of the customer to respond to change and to exploit emergent opportunities over the course of the project. Secondly, the customer does not receive anything of tangible value until all of the requirements have completed testing at the end of the project. This means that it can be many months and possibly years before the customer can realise its investment in the project.

The waterfall model has come under increasing criticism for a number of reasons over recent years. Major studies point to the use of the waterfall model as the cause of failure for many IT projects.

Agile is not *"an evangelical fad"*

Agile has developed from a grassroots movement in the US in the 1990s as a backlash to the waterfall model, and its influences originate in Japan. The theory of agile is based upon, and supported by, complexity science, systems dynamics, economic theory and behavioural studies, amongst others.

The adoption of agile has steadily increased since 2001 when the Agile Manifesto was created. Originally agile was largely the premise of the IT departments (even though agile is not necessarily IT-specific), but it is now widely used on an organisational basis, in virtually all industry sectors, and extensively in North America, Japan and the Scandinavian countries.

Some of the most remarkable examples of the use of agile are found in Google, Yahoo! and salesforce.com. Indeed, salesforce.com has delivered a 41% annual return to shareholders over a sustained period, and it credits this result in no small part to its adoption of Scrum in 2006. [5]

BSkyB v EDS and DeBeers v Atos

The cases of BSkyB v EDS and DeBeers v Atos do not show that *"when Agile projects go wrong, they can go spectacularly wrong"* .The decision in BSkyB v EDS doesn't make any reference to agile. The project was actually based on the waterfall model and used rapid application development (RAD) – which isn't agile – for rapid development of prototypes, the feedback from which was fed back in to the requirements.

The project outsourced by DeBeers to Atos began, ostensibly, with an agile approach, but then switched to a more traditional approach after the project began to run into difficulties. However, the parties do not appear to have contracted on an agile basis. It is very difficult to run a project on an agile basis within the constraints of a traditional contract, because the waterfall model and agile model are quite different. Despite the references by the principle technical architect to DSDM (Dynamic Systems Development Method), which is an agile methodology, it is not clear from the decision how the project was in fact run in an agile way. For example, it appears to have always been the intention that a sequential model of development would be used, which is wholly inconsistent with an agile approach.

A general lack of understanding of agile best practice

The problem stems from the fact that agile is merely an umbrella term for lightweight methodologies, of which Scrum and Extreme Programming (XP) are the most widely used. Each of the methodologies is quite different. For example, the Rational Unified Process (RUP) is far more prescriptive than Scrum. For there to be a sensible debate on agile we need to ensure that the participants share a common understanding of agile best practice.

Agile *is* compatible with fixed price

Contrary to what is suggested, it is possible to agree a fixed price for an agile contract. Under an agile contract the project is sub-divided into modules or releases, each of which is initiated by means of a statement of work. The releases can be charged for on a fixed price basis, or the units of work (often measured by reference to story points) can be charged for on a fixed price basis.

However, *"a watertight contract, clear deliverables … with a fixed price and appropriate remedies"* is a fallacy. Any project must be performed and delivered under certain constraints, which have traditionally been identified as: (i) Scope (features and functionality), (ii) Resources (cost and budget) and (iii) Schedule (time). These three constraints compete against each other and exist in an 'unbreakable' relationship, as illustrated by the *'Iron Triangle'*. [6]

For example, bringing forward the scheduled end date by adding more resources will increase cost, or, adding to the scope will increase time and cost. So, if all three constraints are fixed, there is no give in any of them if there is any uncertainty or unforeseen events arise during the project, and it has been proven that this will inevitably adversely impact on quality and the project objectives.

Most customers want to fix Resources and Schedule which means that Scope must be allowed to vary. The question, therefore, is how can the customer derive value if the Scope may change? Agile solves this problem by prioritising the requirements of the customer on an ongoing basis throughout the project, ensuring that the highest priority requirements are delivered on time and within budget.

In any event, many projects are actually over-specified. It has been shown that 64% of software features are rarely or never used. [7] So it may well be the case that the overall needs of the customer can be met without the need to deliver the lowest priority requirements, in which case it may be possible to achieve significant cost-savings by ending the project earlier than originally planned.

This can be contrasted with the traditional waterfall method under which the customer doesn't receive anything of tangible value until all of the requirements have been delivered.

Government is right to want to manage its budgets tightly. However, it has been proven that uncertainty is inherent in the process of software development.

For this reason any estimates regarding price (or, indeed, regarding the amount of effort involved or schedule) are subject to large amounts of uncertainty at the start of the

project. This amount of uncertainty is only reduced as the software definition is refined over the course of the project, as illustrated by the *'Cone of Uncertainty'*. [8]

It is unrealistic to rely on estimates made at the start of the project when the level of uncertainty is at its greatest. The Government has experienced so many problems with overruns on fixed price contracts that today many of its contracts for software development are no more than a variation of time and materials.

Compliance with public procurement rules

EU public procurement rules require public sector bodies (PSBs) to select the most economically advantageous tender using pre-defined and objective criteria. Detailed up-front specifications and a fixed price are not a requirement. Contracts awarded for the provision of consultancy services or, indeed, legal services, are a good example of this. In any event, in an agile contract the scope of the project is outlined in the form of the objectives of the project, the metrics for success and the constraints.

In Finland, which is also subject to the EU public procurement rules, there are a number of agile contracts that have been awarded by PSBs in full compliance with these rules.

Contractual rights and remedies in an agile contract

I disagree that *"Agile contracts lack clear contractual delivery obligations or remedies"*. An agile contract only differs from a traditional contract in terms of how the solution is delivered. There is no reason why, for example, provisions regarding the treatment of intellectual property rights, data protection, assignment and so on should be treated any differently.

In fact a customer has more remedies under an agile contract than under a traditional contract. Under a traditional contract it is very difficult for a customer to enforce any of its rights before the acceptance date – which can be many months or even years away – because up until then all of the requirements are merely work in progress. Often it is difficult to determine in the acceptance tests whether the software delivered meets the requirements because there have been so many change requests to the requirements in the intervening period that it is hard to establish what the requirements are.

Under an agile contract there are contractual rights and remedies at the end of each release. The supplier commits to deliver by each release completion date fully tested and working software that is ready to deploy and which represents an agreed number of completed units of work. As mentioned above, units of work are often measured by reference to story points.

Equally important is the ability of the customer to plan adaptively throughout the development project, re-focusing the work of the supplier at the start of each iteration based on its findings from the work delivered to date. Not only does this give the customer much greater flexibility, but it also means that many disputes can be avoided by correcting misunderstandings at an early stage.

The discrete roles of the customer and the supplier

Whilst agile advocates the collaboration of the customer and the supplier, their roles are quite different and – contrary to what is suggested - clearly defined.

The supplier has responsibility for the technical domain and the customer has responsibility for the business domain. In other words, the customer is responsible for articulating the business processes to be codified, and the supplier is responsible for designing and writing the code. To that extent the roles are clearly demarcated.

However, input from both parties is essential, as software development is nothing more than the codification of business processes. It is unrealistic for the customer to transfer all responsibility for delivery to the supplier.

Cross functional teams

The blog states that agile *"is not suited to public sector management structures"* for the reason that decision-making is centralised in government, whereas *"agile decision-making … flows down"*. Arguably, it is not agile that is not suited to public sector management structures, but public sector management structures that are not suited to agile. The Institute for Government acknowledges that organisational culture within government is a significant barrier to the adoption of agile:

"The existing governance and commercial processes, not to mention the fundamental mindset shift required, pose specific and difficult challenges." [9]

However, that is not a reason for rejecting the new IT strategy. There is currently a trend for organisations in many different industries and disciplines to move away from hierarchical and siloed departmental structures and towards decentralised cross functional teams. This approach is advocated by TQM (total quality management), lean, systems thinking, and in business management books such as *'The Leader's Guide to Radical Management: Reinventing the Workplace for the 21st Century'* by Stephen Denning.

Conclusion

The age of the Internet has made possible collaborative working and joined-up thinking on a scale never previously experienced. But it has also brought about innovation and a pace of change at a rate that is pushing traditional project methods and contracts to breaking point. Agile offers a solution for managing projects in this increasingly dynamic environment.

The UK Government should be applauded for taking the bold step to change its IT strategy to adopt agile. However, it is inevitable that, like any innovation, such a significant change in strategy will be met with resistance.

It will require changes to be made not only on the part of the government, as highlighted by the Institute for Government, but also on the part of suppliers and supporting partners, including the legal profession.

But there is already evidence that agile can fix failed GovIT. A number of public sector bodies, including the Ministry of Defence and the Metropolitan Police, are already using agile with great effect. We now need to move forward the debate to discuss how the challenges to the adoption by government of agile can be overcome.

---

[1] The blog *'Agile will fail GovIT, says corporate lawyer'* published by Computer Weekly on 26 April 2011.

[2] Latest total spend based on the estimate in the *'Operational Efficiency Programme'* final report published by HM Treasury in April 2009.

[3] 'Software Estimation: Demystifying the Black Art' by Steve McConnell.

[4] As reported in The Telegraph in 2005.

[5] The blog *'Six common mistakes that salesforce.com didn't make'* by Steve Denning and published on the Forbes website on 18 April 2011.

[6] The *'Iron Triangle'* was invented by Dr Martin Barnes in 1969 and popularised in the Project Management Body of Knowledge (PMBOK Guide) issued by the Project Management Institute (PMI).

[7] Standish Group study reported at XP 2002 by Jim Jonson, Chairman.

[8] The original conceptual basis of the *'Cone of Uncertainty'* was developed by Barry Boehm in 1981. The model has since been validated, based on data from a set of software projects at the US Air Force, NASA's Software Engineering Lab and other sources.

[9] *'System Error: Fixing the flaws in government IT'* published by the Institute for Government in March 2011.

**

Susan Atkinson is a Legal Director at gallenalliance Solicitors, based in London. She is a commercial lawyer specialising in IT and with a particular interest in Agile and Lean. She has been advising the Institute for Government on an ad hoc basis on the contractual implications for the government in outsourcing agile projects. She contributed to the Institute's report 'System Error: Fixing the Flaws in Government IT'.