

**ICA 40<sup>th</sup> Conference**  
**Agile Government.**  
**Line Departments Can Achieve a Lot Too:**  
**Philip O'Reilly, Ireland**

*Phillip O'Reilly has been working in the government for some years on a range of operational transaction agencies and, more lately has been involved in implementation of systems and transformation of older legacy systems into new, more flexible agile systems.*

I'm delighted and privileged to be here talking to you today. I want to talk to you about implementing an approach to agility in an operational department. I'm very conscious that I'm standing before an audience that believe that most things are best driven from the centre so I'm in dangerous territory here since I'll be suggesting that when it comes to agility, most of what needs to be done has be driven from within an agency. What is agility and why would we want it anyway? I suppose a good starting point is to start with the definition and I've chosen two.

McKenzie defined agility of an organization as an ability to change tactics or direction quickly. That is to anticipate, adapt to and react decisively to events in the business environment. They define speed as a measure of how rapidly an organization executes an operational or strategic objective. Enrique Barque, a vice president of HP suggests that a measure of agility for a government agency can be defined according to terms of time, range and ease. That is, the time taken by an agency to affect a needed change, the breadth and complexity of changes an agency can handle and the effort and manageability of change. McKenzie, in a most recent survey there published in the last number of weeks, noted that executives around the world overwhelmingly agree that agility and speed of their companies are urgent business issues and are taking steps to capture the benefits for the company performance.

I'm being simplistic here just to illustrate a point, but government administration is about getting policies implemented effectively and any government administration must do two things to be considered a high performer. It must: a. excel at delivering on it's business as usual, that's the existing service it delivers every day to it's customers, and b. it must have the ability or agility to adapt to it's environment by implementing new policies, that is the ability to operationalize it's new initiatives quickly and efficiently so that they too can become business as usual. Knowing what to do is just half the battle, the second half is getting it done and this is where government administrations most often fall down and where they are most often criticized. A minister will announce to the public a wonderful new policy, devised no doubt by a civil servant to address some issue or other but it is a long time before the citizen sees the policy in operation, in other words action on the ground. So where is agile government needed most.

Well, agility in government is not usually about redefining the boundaries of the prime responsibility of ministries or departments that happens, it's true, but it doesn't happen so frequently. Agility is more often about individual departments ability to anticipate and react to changes in their environment in a fast and effective manner. I would say that if departments are agile in their own processes then their integration with the processes of others, that's other departments their customers, agents, or service delivery partners, will most likely be simpler and more effective. So what makes a government agile or what stops it from being agile? There are factors at 2 levels. There are areas where changes can and should be made at the centre, particularly in HR, in structures and scrutiny, but time doesn't permit me to look at those today. While the centre can do much to facilitate and remove the obstacles I would say that line operational departments can do even more themselves to improve their agility.

Here today I will focus primarily on an approach to agility for operational departments using what I call agile process development. I'm basing this on first hand experience with an approach to agility in my own

department, the Department of Agriculture and Food, which has many operational functions. This is an experimental project, which is well advanced. So what I would like to address is the following.

First the questions: Why do we need something new like agile process development? I will talk about two problems and an opportunity. Also, what is agile process development and how does it differ from traditional business process of reengineering? Then I will look at the benefits, the risks and the experiences so far. So why do we need something new? Well, with the conventional approach to deployment of ICT to support business processes, ICT people receive a request from a business area, quite typically, they develop a business case with that area to start an ICT project, be it a waterfall method or prototyping, they'll establish requirements, functional design, technical design, they'll specify, they'll test, they'll deploy...so what's wrong with all of that? In a conventional operational department, or a typical operational department, it's not unusual for ICT application development teams to be aligned in some way with a particular business unit. Through the conventional approach, ICT receives a request, or may receive requests, from multiple business units, for separate systems that at a level of abstraction do the same thing.

The conventional focus of analysis, understandably, is on the individual business unit that is requesting the system and on the processes within that business unit. Conventional methods suggest that the analyst should bottom out the requirements for that unit. This solves the business problem, with a bespoke solution for the specific processes. But there may be similar processes in some far-flung corner of the organization with potential synergies, even if they make look and sound quite different. The conventional approach does not take these into account. The conventional methods don't cater for the requirement of other potential users of the process or if they do, they do so on the basis of bottoming out all of the requirements in a very elaborate functional requirement with potential scope, scale and complexity issues for any software implementation that might subsequently arise to support the process. This is in fact the essential difference between in house systems and commercial off the shelf software. In the case of commercial off the shelf software the most abstract, the requirements of as yet unknown potential users, and our aim as IT people I would argue must think more like the Microsoft or the SAP design engineers.

Now to the problem areas: if you take the problem area number one, cost and workload, each new system means a future main man's load which absorbs resources and escalates costs on an ongoing basis. The faster we deliver new systems the faster the main man's burden grows. Furthermore, some processes never get automated because they're too hard to justify on an individual basis. Problem number two: agility, which we're focusing on today. A new business requirement means new development, and given the main man's load that already exists this leads to reduced responsiveness or agility over time. The lays in operationalizing new initiatives are often down to the need to engineer complex new systems or to reengineer old ones.

So much for the problems, there is a new opportunity here for ICT function to add new value. The ICT function must move up or continue to move up the value chain from code to systems to processes. It can add major business value in that way. ICT has, in the past, added business value through corporate data. In the case of my own organization it has added spectacular value through integration with the processes of the businesses we work with corporate customers, agents and delivery partners. Now there is an opportunity through corporate processes to add to the organizations own agility. So what is agile process development? I suppose agile process development could be described as an engineering approach to abstract the generic architecture of multiple processes that at a high level are essentially the same to create a generic high-level process. And most importantly to incorporate the richness of the current variations in the process to create a rich agile process, and the design and verify process architecture and prove the value of the process in the business unit before building the ICT solution.

So now for the sequence involved in agile process development. We start with an operational strategy. This fits between an organizations overall strategy and it's ICT strategy. This will identify the major operational activities that an organization wishes to carry out to achieve its corporate vision or strategy and how the organization would like to do each of these. In our case, it identifies whether we wish to have an activity centralized or decentralized, whether we want it to be outsourced or in-sourced, and the reason why, and a number of other issues like that. In finalizing the operational strategy it is opportune to

identify processes, which are candidates for support or redesign and to prioritise these. Then the priority candidate process is selected and reviewed with the chosen business unit to carry out the abstraction and design a high level process for review. The chosen business unit should ideally be the one with the greatest number of variations on the process in mind but there are other factors that I will mention later. During this stage it ICT process analyst is exploring for other activities or processes within the business unit that may at a level of abstraction be the same. Furthermore, while incorporating the variations to each step from within the business unit, the analyst based on a general knowledge of the enterprise, is constantly considering the existence of similar activities elsewhere in the organization and the potential use of the new process by other business units. And this is injected into discussions, which are typically white board sessions at the early stages.

This element is quite critical and is as much of an art as it is a science as there is no formula here for this. Remember, the individual business unit will only be conscious of their own use of the process. But as they become more accustomed to this approach, they will appreciate and become more conscious of the organization-wide perspective on their processes. The sketch process is walked through with a number of users of the various individual processes within the business unit to see if the abstraction captures in general terms their processes.

The variations are picked up on at this stage and explored in terms of their essential differences and some voluntary optimisation will typically occur at this stage. The generic process may be walked through with somebody from another business unit to get a reaction. This should spark some recognition of the process, and perhaps elicit some suggestions for further key variations.

Next we must prove the viability and value of the process and this is quite important. To do this an area is chosen in which to carry out a proof of value. This will be in a live environment on a small scale with the process supported by anything that is available, it doesn't matter, it can be an excel spread sheet, it can be a quick prototype done up in MS access, it can be formatted documents and e-mails or perhaps even a few junior staff to key in some documents to simulate the software that will support the process. It just needs to be an approximation so that those involved in the proof of value can assess the viability and value of the process. It doesn't matter at this point that the IT underneath can't scale and is not architecturally pretty, this is not the end solution.

Following this stage which can be quite short, perhaps a few weeks a decision is taken on the viability of the process for the various activities and any shortcomings are noted and decisions taken on the architecture and functional design of the software solution. The IT project to deliver the software only starts now.

When the final software is built it is rolled out to the full business unit in whatever phase manner is suited to task, be it by geography or group of activities or whatever. Subsequent rollouts to other business units only occur when the process and the ICT product are stable in the initial business unit. So how does application process development, I will refer to it as APD, differ from conventional 1980s or 1990s business processes re-engineering?

Well business processes re-engineering, or BPR, tries to re-engineer the processes before supporting or automating them, and ignores some of the human elements, in my view. The BPR focus is on the current process needs. It aims to remove unnecessary slack and optimise the current process. There is less focus on potential of future needs. BPR tends to focus on achieving a single standardised process, a "one size fits all". For this reason, BPR may well automate a new process that will not be accepted and may not work when implemented. BPR was well suited to the downsizing of the 1980's and the early 90's. While agile process development have some similarities it is a very different philosophy, and its key aim is on future agility. The philosophy behind agile process development is to develop processes that are sufficiently rich to accommodate the varying needs of different business units and for disparate needs within units.

It is this richness that is likely to facilitate future unknown business demands. This richness also facilitates ongoing improvement, post implementation. Process improvements during the process

development are voluntary and arise naturally. The business units processes are not under threat and there will be much less natural resistance. This is important in agile process development's success. Remember what William Heath said to us yesterday, if people don't like what they think you are trying to do they would have nothing to do with it.

The benefits of agile process development are the development of rich agile processes including the rich variations in the first instances rather than simply ICT automation. It gives best value for IT development spend, it gives containment of ongoing software maintenance costs, more integrated management information, because many previously separate processes will use the same underlying support, an ability to provide process automation support in future that would not be feasible for individual processes, a generic ICT product rather than a business unit specific ICT system is likely to give greater flexibility and agility in the future.

Further benefits of agile process development are: its less disruptive to the business area than business process re-engineering, it generates less resistance than BPR since any pre-implementation re-engineering is voluntary, and it captures the varied way one might want to carry out a step. There are often very sound reasons for these variations, and the local experts will know this. There is reduced risks since the process functionality is clear and proven before software construction begins. The total initial implementation cost is probably less since there are no loop backs to pick up changes. It generates more future value as it may accommodate new business activities, it allows ongoing re-engineering of existing ones, and it gives greater organisational choice and agility.

The main risks are resistance due to tradition, each function is accustomed to owning its own systems and processes, failure to communicate the benefits, failure to demonstrate the benefits, cynism towards any new approach, pressure to implement on a full scale before building the permanent solution: don't be flattered or tempted by such demands. And finally, unrealistic expectations is not the answer to everything. There are implications for ICT people, firstly on the positive side: skilled people get to build on some of their old analysis skills to great advantage, they get early involvement to get focus on high value added activity, they capture business knowledge before any outsourcing occurs, reducing external dependence and loss of key business knowledge. They get more rewarding work if they can rise to the challenge. However, it requires a different approach to a typical waterfall method, and it means working without all the requirements. You've only got those requirements for the initial business area. The ICT analyst must develop a flexible architecture to cater for the unknown. As I said earlier, he must think more like the authors of commercial off-the-shelf software. It requires IT people to have a more proactive role and affords them that opportunity and where an organisation is structured along the line of business the IT may be required to take ownership of processes and the resulting IT products. There are development implications for both ICT people and for business people.

The ICT people are forced to move beyond the comfort zone of business defines and ICT delivers to a more fluid development model where they must propose process models. Agile process development forces business people to move beyond their current requirements to think about what happens in other business units and what might happen in the future. In both cases, it gets them thinking in a more agile mindset, which is of benefit in itself. How to promote it by identifying areas where the approach can be of real value by demonstrating that it can work well and by delivering early wins.

Now to the experiences so far the Department of Agriculture and Food: well, the pilot area was our veterinary inspector. These vets carry out business to farms, factories, export points, pharmacies, private veterinary practices and whatever. They carry out inspections, samples, tests in both live animals and animals slaughtered at meat plants. They are involved in disease prevention, control and eradication, control of animal remedies and residues, enforcement of animal welfare standards. 63 different types of activities that these specialised vets carry out. It's not possible to justify ICT support for all of these individually. They also have inadequate management information to manage these very expensive resources.

These activities are all very different but, to some degree, all 63 activities involve some form of test or sample or inspection. The progress the days when we set up a joint veterinary ICT team, a small team, to

work on the process directed by a steering group co-chaired by the Deputy Chief Vet and myself, the process has been extracted at the highest level to five key steps, albeit with significant variations within them, within each the process has been designed and verified in the veterinary area taken to the stage of proof of value that I mentioned earlier and the process has been verified at a high level with two other major business areas. That's plant health and agricultural inspectorate, which are quite different activities. The resulting process and ICT product will be known as AFOD.

Just to give you an idea of what this means in reality, here is the proof of value of AFOD, the architecture involved. We have over 60 different operations reduced to a simple architecture, five steps albeit with many variations, maybe 4 or 5 in each, which gives you a lot of variations when you multiply that out. As you can see, the proof of value has integrated with existing facilities like MS Active Directory and MS Exchange and an underlying database to capture all of that activity. So it's still a proof of value and the objective is to prove the process, software is irrelevant at this stage it must just be adequate to support the proof of value. It doesn't have to be able to scale but the conceptual architecture must be capable of scaling the final software will only be built when the results of the proof of value are there. Much could change in the meanwhile. But so far the feedback has been positive from proof of value, in particular the business units believes that it will give them greater ability to implement new measures and response to any animal health issue or threat.

The two other major business areas are going to use the process and the ICT product that emerges for very different functions and this will mean a single, rich, ICT product to be supported, giving much greater agility to each of the business units because of the rich processes that they have taken on. So some APT dooms: pick a business unit that is receptive to the new ideas with open minded senior management, pick a business area that has a lot to gain from this success, pick team members, ITM business very carefully for the initial pilot, it means a significant change on business and IT side, ensure that the head of the organisation fully understands and supports the approach, spend time communicating the approach, even to those not immediately impacted, and expect resistance from some business units as it means sharing the processes rather than owning them, and expect resistance from IT units because it means working in a very different way: more is expected of them and its outside their comfort zone, and its counter to their traditional methods and, of course, there are some APD don'ts: don't pick a very resistant business area for the first trial, it could put you back years. Don't pick a project with an absolute deadline for your first pilot project. For this approach, it's a learning experience for all, and it's more important that you get the approach right, short circuit it and you may ruin it. Don't forget to spend time communicating the approach. And concluding comments: there are measures that can be taken at the centre to remove obstacles to agility in areas like HR scrutiny and structures, but there is more that can be done in line departments. Agile processes development can offer a key to future agility, it has also as benefits in efficiency and ICT costs, but our human elements that are as important as the technology, it is philosophically very different and it should therefore be introduced to an organisation with care and patience to achieve its potential. Thank you very much for your time and attention.