

An Introduction to Agile Development



Version 1.0
September 2005

An introduction to agile development

This Brief provides a high-level view of the 'agile' approach to software development. The Brief covers what agile development is about, how it compares to other approaches to software development, why organisations are adopting an agile approach, and who is particularly investing in agile development.

What is agile development?

'Agile development' is an approach to software development that emphasises frequent and sustained delivery of business-valuable software. There are a number of software development methodologies that fall into the category of 'agile development' such as eXtreme Programming, SCRUM, Crystal Family and DSDM. Each of these has a different emphasis on, and combination of, practices and techniques, but all exhibit common features:

- **Iterative Delivery.** The project is divided up into a number of short (usually two-four week long) iterations. What will be delivered at the end of each iteration is decided at the start of that iteration, and the customer can expect to take delivery of working, production-quality code when the iteration finishes.
- **'Continuousness'**. Activities such as requirements definition, design, testing, integration, system build and quality assurance are carried out all the time rather than confined to 'phases' at the beginning or end of the project. 'All the time' tends to mean every day or more frequently.
- **Emphasis on business-valuable deliverables.** The primary value of a software project comes from the delivery of a software system that meets the business needs of the customer. Agile projects prioritise software delivery over the delivery of 'incidental' artefacts such as detailed status reports, detailed designs model, and detailed long-term project plans.
- **Frequent lightweight communication.** Rather than spending a lot of time writing lengthy memos covering project status, highly-detailed requirements specifications, or intricate design models, agile development projects devote time to face-to-face communication involving the 'whole team'. 'Whole team' usually means customers, users, developers, project managers, QA engineers, managed operations staff and anyone else directly involved with or affected by the system delivery.

Agile development methodologies evolved in reaction to other styles of software development prevalent throughout the eighties and nineties: waterfall development, RAD and 'just do it'.

‘Waterfall development’ (and its derivatives)

Waterfall development demands that all the requirements are identified and detailed at the start of the project. The system architecture is then defined, and detailed design of the system carried out before programming starts. The system is usually delivered in modules that, once finished are then integrated and tested.

The big problem with waterfall development is that it assumes all the requirements, the architecture and the design can be identified, fixed and signed off up front. If this is done correctly, development and delivery of the system should be little more than the correct translation of the design into code. Experience has shown that it is incredibly difficult to produce requirements of such detail and completeness that the system delivered is exactly what the customer wanted. We have also seen that the transition from design to programming, from programming to integration, and from integration to testing can be fraught with problems; usually caused by misunderstandings, incorrect assumptions and discrepancies not identified during the requirements, architecture and design phases.

Waterfall development can, and does work, but doing it right is typically very expensive. However, even when waterfall is practiced well, it is not very good at dealing with change. Every change in the requirements requires a ‘change request’, often submitted to a ‘change control board’. Waterfall works best when change is restricted.

RAD (Rapid Application Development)

RAD is superficially similar to agile development. In a RAD project, the customer is involved early on in working with technologists to quickly produce prototypes that embody the requirements. This gives the customer and users a chance to ‘play’ with the technology early on to firm up their requirements and identify any gaps or incorrect assumptions.

RAD projects then typically head in one of two directions. Some will become waterfall development projects (and suffer the problems described above), with the prototypes being formalised as requirements. Others will seek to develop the prototypes into ‘proper’ implementations. The latter approach often leads to a number of poorly engineered and poorly integrated solutions as ‘fleshing out’ a prototype is not the same as developing a ‘proper’ solution from the ground up.

As with waterfall development, RAD can and does work. However, successful RAD projects tend to be those where the final solution is actually relatively close to the prototype in terms of implementation. Front-office solutions for exotic financial markets built around spreadsheets, and one-off, limited scope, database-and-forms applications are good examples.

‘Just do it’ (AKA hacking)

A number of software projects have been successfully delivered by getting a very small number of high-skilled developers together, getting them to work closely with the customer (or hiring developers with excellent domain knowledge), and foregoing any formal development process.

The big problem with this approach is that it doesn't scale. If one or two developers leave, the project slows dramatically or grinds to a halt. If the system grows much larger than expected the need for more architectural or design consideration suddenly becomes apparent but it is very hard to retrofit. If the project team needs to grow to incorporate more developers and more delivery staff (such as QA engineers or build and release staff) – plus a more formal process to ensure they are successfully involved in delivery – the tight-knit core of developers finds it hard or impossible to adapt.

Why are organisations adopting agile development?

Modern business has woken up to the idea of being agile: reacting to changing customer needs and marketing conditions on a daily, weekly and monthly basis. Software projects that last two years (and don't even get close to delivering any working software until month twenty) have no hope of supporting any degree of genuine business agility. Long-term waterfall projects cannot deliver the benefits these businesses need.

The problem is hacking is not an acceptable alternative. Whilst these businesses need to be agile, they also need to be sustainable. Being reactive today at the expense of long-term viability is not an option.

Agile development evolved to provide the short-term benefits of a 'just do it' solution in a sustainable and scalable model. Sceptics who look only at the lightweight communication aspects of agile development (or the "lack of documentation" as its often referred to) tend to dismiss it as hacking but the reality is that agile development projects are extremely rigorous in the application of practices and techniques that all the project team have to follow all the time. Many experienced developers and project managers who work on an agile development project for the first time comment on the degree of discipline and rigor inherent in the process.

Having said that, agile development is not a silver bullet; it is not for everyone. Waterfall development still suits projects with a massive scale (the NHS infrastructure project for example) or with extremely well-defined and unchanging requirements (such as medical systems and control systems). RAD will always be a good option where the prototype and the final system are very close in terms of their implementation. And some organisations will always choose to 'just do it'.



But businesses that rely heavily on software systems to deliver direct business benefit, and that have to be agile in reacting to changes in customer needs and the market place, are starting to become interested in – and invest in – agile development.

Who is adopting agile development?

Some of the methodologies that are now grouped under the agile development banner (particularly DSDM) have been around for a number of years. However, the first project to be particularly identified as using an agile development process was run for Chrysler in the mid 1990's. On the back of the reported success of this project, a number of small projects were run using the Chrysler methodology (or 'eXtreme Programming' as it became known; much to the chagrin of developers and project managers trying to sell the benefits to their bosses).

Adoption of agile development processes really took off in the Internet boom of the late 1990's. Dotcoms and traditional businesses alike tried to adopt agile business practices based on the hype and reality of this incredible new business channel. Many businesses (and many systems) failed, but, after the bubble burst, there was a general acceptance that business agility depended directly on IT systems, and that agile development processes could (and did) deliver IT systems in a manner that supports business agility.

Today, agile development is gaining acceptance in businesses that recognise the need for this agility. Many financial institutions are investing in agile development initiatives as they recognise the ability to roll out new products, to react to regulatory changes, and to break into new markets, is underpinned by their ability to evolve and adapt their IT systems in parallel with their business. Large corporations in domains as diverse as consumer electronics, petroleum, pharmaceuticals and telecoms – all of whom need to become more agile in the way they do business – are also adopting agile development processes.

About e2x

e2x is a consultancy specialising in helping organisations to adopt agile development processes. e2x consultants were among the very earliest adopters of agile development (one e2x consultant led the first eXtreme Programming project to run in the UK) and have wide-ranging, in-depth experience of the various agile processes. e2x has enjoyed particular success in helping blue-chip clients adopt and adapt agile development processes and deliver software systems using these processes. e2x clients include Philips, Royal Mail Group, Lockheed Martin and UBS Investment Bank.