Bird & Bird & Contracting for agile software development projects

Contents

Introduction

Agile software development methods are now being widely used in the IT sector and are increasingly being advocated as preferable to the traditional waterfall development model.

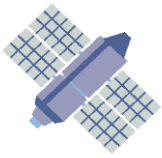
However, contracting for Agile software development projects remains a challenge. Most standard software development contracts were designed for use with the waterfall model and can be difficult to reconcile with the principles that underpin Agile working practices.

At Bird & Bird, we have been exploring the legal questions raised by Agile projects. We believe that a fresh approach is needed to ensure that the contracts being used for Agile projects are fit for purpose. Our aim is to develop a new template contract that will be a more effective starting point for Agile projects than the traditional waterfall-based contracts.

Purpose of this paper

In this paper, we identify a number of areas relating to contracting for Agile projects where we believe a fresh approach may be helpful. In each area, we have:

- identified the challenges raised by traditional IT contracting principles; and
- proposed an alternative approach that may be more suitable for Agile projects.



A brief overview of Agile

Before considering the contractual issues raised by Agile, it is worthwhile being clear about what we mean by Agile and how it differs from the traditional waterfall development model.

What is the waterfall development model?

Under the waterfall development model, the project is divided into a sequence of distinct phases (see Figure 1 below), starting with a detailed planning phase in which the project requirements are analysed and documented in detail. Once the requirements have been fully specified, the project then continues through the design, coding, testing and integration phases, finally leading to deployment of the finished product.

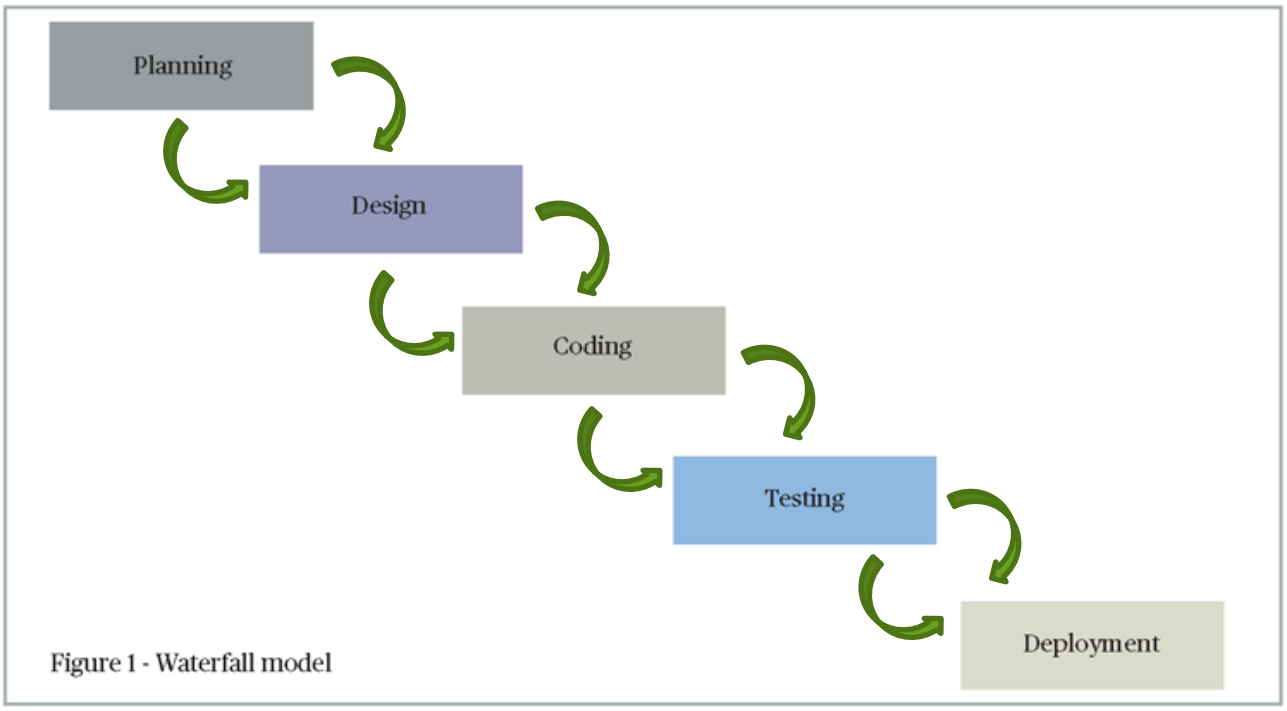


Figure 1 - Waterfall model

Some of the main arguments in favour of the waterfall model are that:

- it provides a very clear and structured way to approach development projects;
- getting the requirements and design correct at the outset of the project will save time and money later on; and
- it lends itself to identifying clear milestones to track the progress of the project.

Perceived problems with the waterfall model

Critics of the waterfall model argue that, while it may work well for stable projects where the requirements are reasonably certain, it all too often fails to deliver for projects where the requirements are more uncertain or fluid.

Particular problems with the waterfall model which are often mentioned include that:

- given the intangible nature of software, it is difficult for the customer to define its requirements in a clear and unambiguous way at the start of the project (particularly without the benefit of reviewing a prototype); and
- it does not adapt well to change – the model assumes that the customer's requirements can be "frozen" at the end of the planning phase before moving on to the design and build phases whereas, in practice, the customer's requirements are likely to change over time, perhaps significantly.

The Agile model

"Agile" is an umbrella term for a number of different software development models including Scrum, Extreme Programming (XP), Crystal Clear and Dynamic Systems Development Method (DSDM). However, the key feature that these development models have in common is that they are iterative. In other words, they advocate short, frequent development cycles with working software delivered at the end of each cycle.

A good starting point for understanding how these Agile models work is the Agile manifesto.

The Agile manifesto

In 2001, a group of software developers published the Manifesto for Agile Software Development to define the values of the Agile movement (see Figure 2).

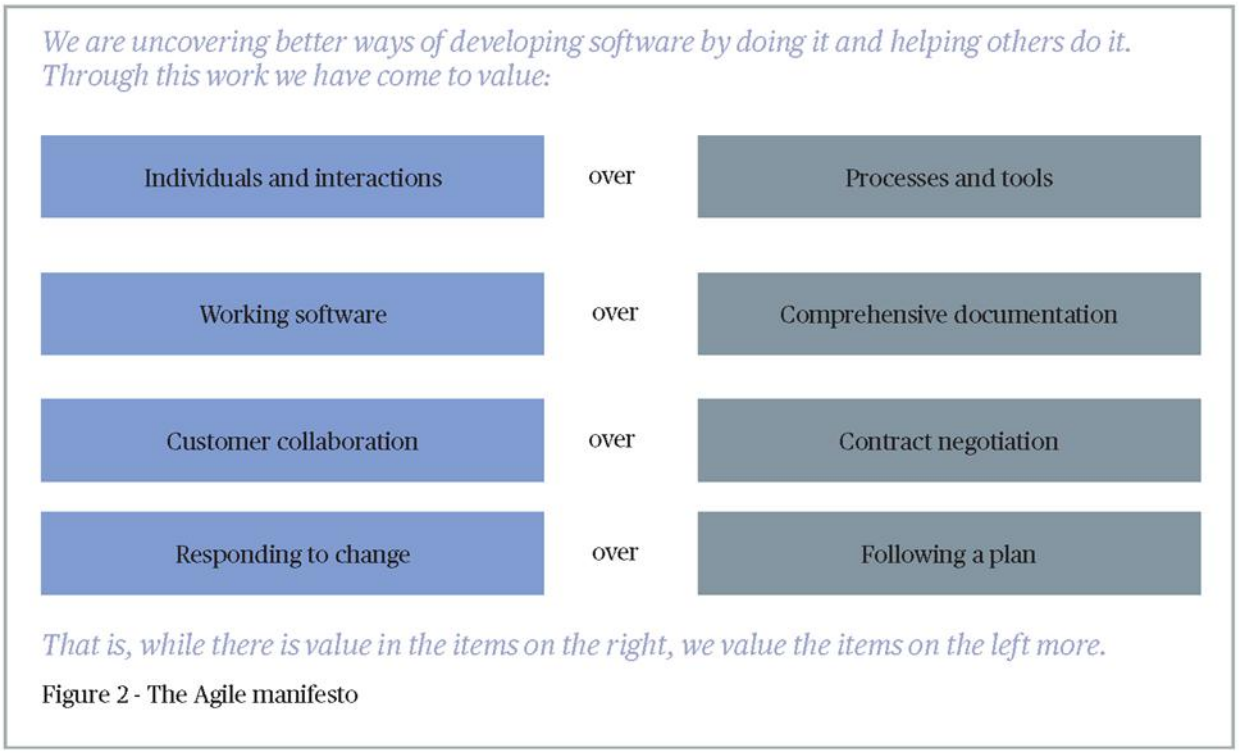


Figure 2 - The Agile manifesto

There are 12 principles that underpin the Agile manifesto. Some of the key principles are:

- Customer satisfaction is best achieved through rapid delivery of useful software.
- Working software should be the primary measure of progress and should be delivered frequently (i.e. weeks not months).
- Changing requirements should be welcomed, even if late in the development process.
- There should be close interaction between developers and business teams.

How does Agile work in practice?

As mentioned above, the key to the Agile model is that the overall project is broken down into a series of short development cycles (commonly known as "iterations" or, in Scrum terminology, "Sprints") of approx. 2 to 4 weeks each. As can be seen from Figures 3 and 4 below, each iteration involves the development team carrying out essentially the same activities as under the waterfall model (i.e. planning, design, coding, testing and deployment), but under Agile working practices:

- these activities are focused on the current iteration rather than the overall project (although there may be some general planning activities, as we shall discuss later in this paper); and
- although there will be some upfront planning in each iteration, the design, coding and testing activities are effectively performed simultaneously. (A key element of this is a process known as test driven development (TDD)[1].)

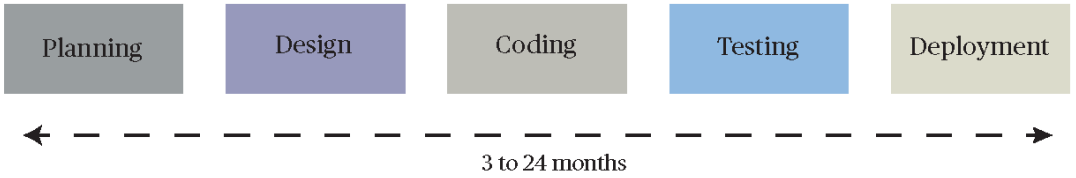


Figure 3 - The waterfall model

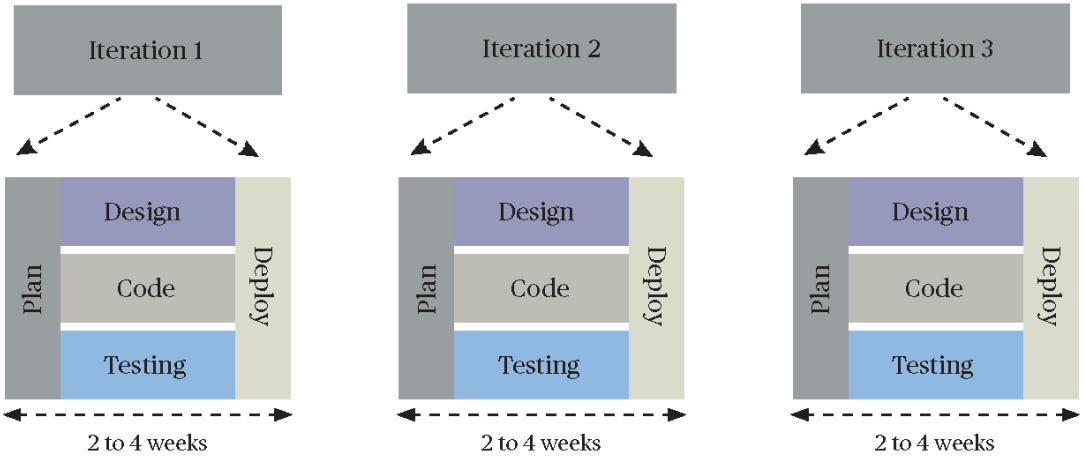


Figure 4 - The Agile model

Source: *The Art of Agile Development*, James Shore and Shane Warden

[1] A detailed description of TDD is outside the scope of this paper, although there are a number of explanations readily available (e.g. Chapter 9, *Succeeding with Agile* by Mike Cohn (2010, Addison Wesley)).

The Scrum methodology

As mentioned above, there are many different software development models which fall under the banner of Agile. For the purposes of this paper, we have adopted the concepts and language used in the Scrum methodology (which is the most popular form of Agile). However, many of the principles discussed in this paper will also apply to other Agile development models.

Before turning to the issues surrounding contracting for Agile projects, it is useful to explain some of the key concepts used in Scrum. Figure 5 is a diagram prepared by Mountain Goat Software that shows how the Scrum model works in practice.

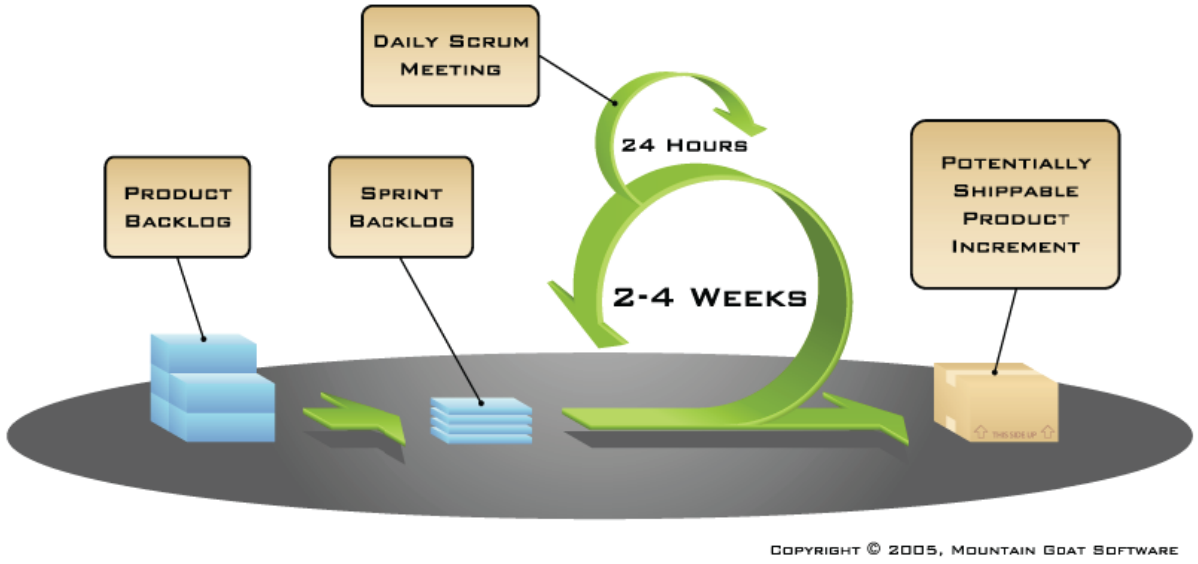


Figure 5 - The Scrum model

Starting from the left hand side of Figure 5:

- the Product Backlog" is a prioritised list of all of the individual items for development in the overall project (we will discuss this document in more detail further on in this paper);
- for each "Sprint" (i.e. a development cycle of 2-4 weeks), the development team will prepare a "Sprint Backlog". This document sets out which of the development items from the Product Backlog will be developed during the current Sprint (starting with the highest priority items). In other words, it is effectively a "to do" list for the current Sprint;
- the development items listed in the Sprint Backlog will then be designed, coded and tested during the 2-4 week Sprint (which will typically involve a daily meeting of the development team to ensure that the Sprint activities are progressing as scheduled); and
- the output of each Sprint should be a "potentially shippable product increment" (i.e. a working piece of software that can be deployed within the Customer's business).

Contracting for Agile projects – a change of approach?

Relationship between the waterfall model and standard development contracts

Most standard software development contracts were designed for use in traditional waterfall projects and, as a result, are based on the principles and philosophy of the waterfall model. In particular, such contracts are typically structured to reflect the waterfall model's sequential approach, based around long delivery cycles (or often just one delivery) and distinct development phases.

Another important feature of many waterfall-based contracts is their approach to problems and failures during the development project. In many cases, the contract reflects a key assumption of the waterfall model that each phase of the project can (and should) be completed perfectly before starting the next phase.

This basic principle often leads to a presumption in the contract that:

- provided the supplier complies with the sequential approach of the waterfall model (and the associated procedures of the contract), then development failures ought to be capable of being avoided altogether; and
- consequently, if a failure does occur, it must be due to the supplier failing to follow the relevant contractual procedures properly (and, as such, the customer should be entitled to treat the failure as a breach of the contract, with all of the remedies that this entails).

Agile proponents would argue that this approach to development failures is unrealistic (and ultimately unproductive for both parties). Instead, they argue that:

- software development is an inherently unpredictable process and that some degree of failure is inevitable in any project; and
- consequently, the contract needs to provide a more flexible regime under which failures can be accommodated and resolved efficiently and cost-effectively.

Agile contracting – a fresh start

In our view, the fundamental differences between the Agile and waterfall models (including the different approach to failures) demands a fresh approach to contracting for Agile projects – existing templates and contracting approaches cannot be easily adapted to properly reflect and support the requirements and philosophy of the Agile model.

Back to basics

If we are to make a fresh start to contracting for Agile projects, then it is useful to first take a step backwards and ask ourselves a fundamental question, namely what are software development contracts for? What is their purpose?

We think there are 3 key goals for any software development contract:

- to define the purpose of the project (i.e. what are the parties trying to do?);
- to define how the project is to be established and run; and
- to define what happens if the project goes wrong.

Many standard waterfall contracts focus heavily on the 3rd bullet (for the reasons mentioned above) while overlooking the importance of the 1st and 2nd bullets. For example:

- in many cases, it is the functional specification which bears the burden of defining the scope of the project and determining whether or not the software will be accepted at the end of the project. When (as often happens) the functional specification fails to provide the level of clarity and detail expected of it, the parties' rights and obligations under the contract can be left unclear, leading to disputes that may be difficult to resolve; and
- many standard waterfall contracts devote little attention to establishing how the project is actually going to be run by the parties on a day-to-day basis.

By contrast, most Agile models have a detailed methodology that clearly defines the "rules of engagement" of the project. For example, the Scrum model sets out clear requirements for each of the following aspects of the project:

- the key project roles (i.e. the Product Owner, Development Team and ScrumMaster);
- the key planning and management meetings (i.e. the Sprint planning meeting, the daily meeting and the Sprint review meeting); and
- the key project documentation (i.e. the Product Backlog, the Sprint Backlog and the Sprint Burndown chart).

The advantage of this approach – which we believe Agile development contracts should seek to replicate – lies in promoting "real time" visibility and control of the project by the parties as opposed to simply "after the fact" legal remedies (i.e. when the project has already failed).

We will come back to these issues as we discuss some of the key aspects relating to contracting for Agile development projects.

Contracting for Agile projects – some key aspects

Introduction

In this section, we consider some of the main elements of an Agile software development contract and discuss some of the key contractual issues to be addressed.

As mentioned above, for the purposes of this paper, we have adopted the concepts and language used in the Scrum methodology. However, many of the principles discussed below will also be relevant to other Agile development models.

Key roles

There are 3 fundamental roles within the Scrum model: the Product Owner, the Development Team and the ScrumMaster.

1. The Product Owner

The Product Owner is the key representative of the Customer. He or she is responsible for communicating the Customer's vision of, and requirements for, the project to the Development Team.

The Product Owner will also take primary responsibility for the Product Backlog, including its initial development and its ongoing revision during the project – this process is discussed further below.

He or she will participate in meetings with the Development Team during each Sprint, including to assess developed items (although other Customer stakeholders may also be involved in these activities).

Key issues to be addressed in the contract

- The Product Owner is a representative of the Customer and as such, the contract should provide for the Product Owner to be nominated and appointed by the Customer.
- The Supplier may seek some assurance from the Customer that the nominated Product Owner is suitably experienced in Scrum development projects.
- The contract should set out the key responsibilities of the Product Owner, including:
  - the initial development and prioritisation of the Product Backlog;
  - the ongoing revision and re-prioritisation of the Product Backlog as the project develops; and
  - participation, on behalf of the Customer, in the relevant Scrum planning and review meetings (discussed further below).
- Given the importance of the Product Owner role in the overall project, the contract should require the Customer to use reasonable endeavours to ensure that the Product Owner:
  - dedicates a reasonable amount of their time and efforts to the above activities;
  - responds to questions from the Development Team as soon as possible; and
  - is not removed from the project without good cause.

2. The Development Team

The Development Team will be responsible for the actual development activities within each Sprint. As such, the team will need to be cross-functional and include members who are skilled in areas such as coding, testing etc. It will also be important that the Development Team are experienced in Scrum development projects.

A key question is whether the Development Team should comprise only Supplier personnel or whether it should also include Customer personnel. A Development Team comprising both Supplier and Customer personnel may, strictly speaking, reflect Agile principles and goals more fully. However, we suspect that this approach may in many cases be problematic in practice. For example:

- the Customer may not have sufficient resources to dedicate to the project on a day-to-day basis;
- equally, the Customer may not have personnel with the necessary technical skills to participate effectively in the Development Team; and
- from a legal perspective, a combined Development Team raises difficulties in having a clear allocation of risk and liability between the Customer and the Supplier (this issue is discussed further below).

Key issues to be addressed in the contract

- The contract should identify the proposed Development Team (or, alternatively, the Supplier should be obliged to notify the Customer of the proposed Development Team shortly after the contract is signed).
- Each member of the Development Team should be appropriately skilled and experienced to carry out the development project[2].
- The Customer should be entitled to confirm whether or not the proposed Development Team is acceptable. If the proposed team is not agreed by the Customer:
  - the parties should use reasonable endeavours to agree the composition of the Development Team; and
  - if the parties are unable to agree the Development Team within a specified period, then either party should be able to terminate the contract without liability.
- The Supplier should ensure that each member of the Development Team:
  - is dedicated to the project during the development period (unless otherwise agreed); and
  - is not re-assigned from the project without the prior written consent of the Customer.

[2] Training and certifications are available from organisations such as the Scrum Alliance (www.scrumalliance.org) and Scrum.org (www.Scrum.org)

3. The ScrumMaster

The role of the ScrumMaster is perhaps the most challenging to capture in the contract. In broad terms, the ScrumMaster's role is similar to that of a coach or a mentor – ensuring that the Development Team and the Product Owner are working co-operatively and following the Scrum processes. Importantly, the ScrumMaster is not a project manager – his or her role is not to assign tasks and measure progress against goals, but rather to support the Product Owner and the Development Team in doing this.

Given this, potential contractual approaches could include:

- having no ScrumMaster role at all;
- the ScrumMaster being a member of the Supplier personnel (or, perhaps less likely, a member of the Customer personnel); or
- the ScrumMaster being a third party unconnected to either the Supplier or the Customer.

Doing away with the ScrumMaster role entirely may lead to problems in practice, particularly where the parties are relatively new to Scrum projects. The identity of the ScrumMaster will depend on a range of factors including:

- whether the Customer has personnel available, and with the right skill set, to act as the ScrumMaster; and
- whether an external consultant will be able to build the necessary relationships with the Product Owner and the Development Team to act as an effective ScrumMaster (and whether the project budget can justify an additional external resource).

Key issues to be addressed in the contract

- The contract should identify the proposed ScrumMaster (or include a process for the ScrumMaster to be agreed by the parties). The contract should also set out the level of skill, experience and qualifications required of any ScrumMaster (such as x years of experience of software development projects using the Scrum methodology).
- The key responsibilities of the ScrumMaster should be set out in the contract.
- The contract should require that the ScrumMaster:
  - is dedicated to the project during the development period (unless otherwise agreed); and
  - is not re-assigned from the project without the prior written consent of the Customer.

Product Vision

The starting point for a Scrum project is the Product Vision. This is a statement setting out the overarching goals of the project and the high-level benefits that are sought. Ideally, the Product Vision will have been developed before the contract negotiations start in order to help the negotiation team understand the intended end result of the project.

Key issues to be addressed in the contract

- The Product Vision should be included in an appendix to the contract as a reference point for the development of the Product Backlog and the project generally.

Product Backlog

The Product Backlog is the Agile equivalent of a "statement of requirements". In essence, it is a prioritised list of all of the items that are to be developed during the course of the project. One way to imagine the Product Backlog is to picture it as a "stack" of development items, ordered by importance, with the highest priority items at the top of the stack and the lowest at the bottom.

The Product Backlog will be based on the Product Vision and should include the following elements:

- **items:** the list of features to be developed. As the project develops, these may also include defects to be rectified and/or areas for further improvement;
- **estimate of business value:** an estimate by the Product Owner of the value to the Customer's business of each item (presented in relative terms by comparison to other items);
- **estimate of effort:** an estimate of the effort required by the Development Team to develop each item (presented in relative terms by comparison to other items); and
- **priority:** the priority for each of the items, taking account of the estimates of business value and effort.

In Agile projects, the development items are often articulated as "user stories" (i.e. capturing succinctly what the end user wants to achieve). A typical format is:

"As a <*describe end user role*>, I want <*describe feature or goal*> so that <*describe reason or benefit*>".

The intended purpose of this format is to keep the description of the development items short and clear. (There are a number of textbooks and blogs devoted to the art of writing user stories.)

Some of the other key features of the Product Backlog are:

- the high priority items should be clearly defined (by contrast, the low priority items may be more general or vague);
- items can be re-prioritised by the Product Owner at any time;
- new items can be added by the Product Owner from time to time (and prioritised as necessary); and
- items can also be removed from the Product Backlog by the Product Owner at any time.

It is common for the Product Owner and the Development Team to devote approx. 5-10% of each Sprint to jointly reviewing and refining the Product Backlog, including by preparing more detailed explanations of individual items or dividing more general items into smaller, specific items.

Key issues to be addressed in the contract

- The contract will need to specify how the initial version of the Product Backlog will be developed. There are a number of possible approaches, including:
  - the initial Product Backlog could be developed in parallel with the negotiation of the contract, in which case, it can be attached as a schedule to the contract; or
  - the Product Backlog could be developed following contract signature. In this case, the contract could provide for an initial workshop to be held between the Product Owner and the Development Team to discuss the development of the Product Backlog and, where necessary, carry out some detailed requirements analysis.
- Once the initial version of the Product Backlog has been developed, the Development Team should be obliged to provide the Product Owner with an estimate of the effort required to develop each item in the Product Backlog.
- The contract should require that these estimates are prepared with appropriate care and skill and on the basis of fair and reasonable assumptions. (It may also be appropriate for this process to be subject to the dispute resolution procedure in the contract where there is a dispute between the parties as to whether the estimates from the Development Team are appropriate.)
- Once the estimates of effort have been finalised, the contract should require the Product Owner to assign a priority to each item based on the estimates of effort and business value.
- If the parties will hold a workshop during each Sprint in order to refine the Product Backlog (see above), this should also be addressed in the contract.
- The contract should also make it clear that the Product Owner is free to amend the Product Backlog at any time. The only exceptions to this are:
  - the Product Owner cannot unilaterally change the estimates of effort received from the Development Team (although they could be discussed during the above workshops); and
  - as discussed further below, the scope or priority of items cannot be changed once they are under development as part of a Sprint.

The Sprint process

Duration

The parties are free to choose the duration of Sprints that will suit them, although these should be kept relatively short (e.g. 2-4 weeks). However, it is a key principle of Scrum that the duration of individual Sprints should not be changed, even if the progress is running behind schedule – unfinished items should instead be re-inserted into the Product Backlog and prioritised accordingly.

Key issues to be addressed in the contract

- The contract should specify the agreed duration of each Sprint.
- The contract should also include an acknowledgement from both parties that the duration of an individual Sprint cannot be extended.

Sprint meetings

Each Sprint will typically feature 3 types of meeting:

1. Sprint planning meeting

At the start of each Sprint, the Product Owner, the Development Team and the ScrumMaster will typically hold a planning meeting to address the following points:

- the Product Owner will explain to the Development Team which of the items from the Product Backlog are of a high priority for the current Sprint and the goals and business context for each of those items; and
- following this, the Development Team will determine how many of the high-priority items identified by the Product Owner can be developed during the current Sprint.

Following the planning meeting, the Development Team will prepare a "Sprint Backlog". This will specify (for each item to be developed during that Sprint):

- a breakdown of that item into individual tasks;
- an estimate of the time required to complete each task; and
- an allocation of the tasks within the Development Team.

2. Daily meetings

During the course of each Sprint, the Development Team will generally hold a short, daily meeting during which each member of the Development Team reports on the following matters:

- what work they have completed since the last meeting;
- what work they are planning to complete before the next meeting; and
- any obstacles to completing their work.

As part of this meeting, the Development Team should also update the estimates of effort in the Sprint Backlog to reflect progress to date.

A useful tool at these meetings is a "Sprint Burndown" chart which measures (i) the current estimate of outstanding work required to complete each task in the Sprint Backlog against (ii) the total available time remaining in the current Sprint.

If progress is running to schedule, the 2 elements above should match with all tasks being completed by the end of the Sprint (and with no "left over" time). However, where things are not going to plan, the Development Team, with the support of the ScrumMaster, will need to decide what action to take to improve their progress.

3. Sprint review meeting

At the end of each Sprint, the Product Owner, the Development Team and the ScrumMaster will typically attend a review meeting at which the items which have been developed during the Sprint will be assessed. A key element for this meeting will be the "Definition of Done" (see below). Another element of this meeting should be discussing the success of the Sprint generally and what improvement should be implemented in future Sprints.

Starting the next Sprint

Typically, each Sprint will follow on immediately from the previous one (although, in some projects, the parties may want to build in a short period from time to time to "pause" and reflect on overall progress). As such, following each Sprint:

- the Product Owner should promptly update the Product Backlog; and
- the parties should begin the next Sprint.

Key issues to be addressed in the contract

As mentioned at the start of this paper, it is likely that the parties will want to capture the Sprint process in the contract in some form. However, a key question is whether compliance with the Sprint process should be contractually binding (see discussion below on this issue).

Subject to this, the contract should address the following aspects of the Sprint process.

- In relation to the Sprint planning meeting, the contract will need to include a regime for the Development Team to determine how many of the high priority items identified by the Product Owner can be developed during the current Sprint.
- The contract should also include an acknowledgement from the Customer that, once the items to be developed in each Sprint have been identified, they are fixed for that Sprint.
- The contract should also set out a regime for the development of the Sprint Backlog by the Development Team following each planning meeting. It may be useful to include an agreed format for the Sprint Backlog in an appendix to the contract.
- The contract should also require that any improvements agreed at a Sprint review meeting are implemented by the Development Team in the next Sprint (and that the effectiveness of these improvements is reviewed at the next Sprint review meeting).
- The contract should provide for the parties to move directly into the next Sprint, following the same procedure as set out above. The Sprint process should then continue until either:
  - the project is completed; or
  - the contract is terminated.

Should compliance with the Sprint process be contractually binding?

As mentioned above, one question that arises is whether compliance with the more detailed aspects of the Sprint process (e.g. daily Sprint meetings and/or Sprint Burndown charts) should be contractually binding. In some projects, this may be desirable to ensure that the Sprint process is properly followed by the parties. However, we anticipate that this may be overkill in many projects.

One possible solution is to provide that the key responsibilities of the parties (such as those mentioned in the "Key issues" boxes above) are contractually binding, while leaving the more detailed day-to-day aspects of the Sprint process as non-binding.

The "Definition of Done"

As discussed above, an important Agile principle is that each Sprint should result in a "potentially shippable product increment". But how will the parties determine whether or not this has been achieved?

In Scrum projects, the key to this is establishing an agreed "Definition of Done". One Agile practitioner described this concept as the "soul" of the entire process. Some of the elements of a "Definition of Done" may be:

- the scope of tests to be conducted and passed (e.g. user acceptance tests and non-functional tests);
- that all code has been reviewed (or pair programmed);
- that all coding standards have been met and code has been re-factored where necessary; and
- any necessary documentation has been completed.

Key issues to be addressed in the contract

- The "Definition of Done" should ideally be developed in parallel with the negotiation of the contract and attached as an appendix.
- The contract should provide that:
  - during each Sprint planning meeting, the Product Owner and the Development Team should review how the "Definition of Done" will be applied to the items to be developed during that Sprint; and
  - the ScrumMaster should be responsible for ensuring that all items presented by the Development Team at the Sprint review meeting have been completed in accordance with the "Definition of Done".
- The contract should also include an appropriate dispute resolution procedure if there is a dispute between the parties as to whether any item has been completed in accordance with the "Definition of Done".

Project completion

The development project will be completed if all of the items listed in the Product Backlog have been developed and completed in accordance with the "Definition of Done" (see above). Importantly, the list of items in the Product Backlog at the end of the project may not be the same as at the start – during the course of the project, the Product Owner may have decided that some of the features identified at the start of the project are no longer needed.

Key issues to be addressed in the contract

- The contract should identify when the project will be deemed to have been completed (as set out above).

Pricing

Customers and suppliers are likely to have very different perspectives on how an Agile development project should be priced. In many cases, the Customer will be seeking to agree a fixed price while the Supplier will want to work on a time and materials basis.

From the Customer's perspective, one of the major concerns for any software development project is cost. Many Customers worry that by agreeing to use Agile, which may involve an unknown number of iterations, they are effectively writing a "blank cheque" for the project costs with few constraints on costs escalation.

Some of the counter-arguments from Agile proponents are that:

- it is unrealistic to expect any development project, whether based on a waterfall or Agile model, to be delivered on a fixed price basis – there will always be changes in scope which will affect the original price;
- a fixed price model erodes the intended benefits of Agile by encouraging the parties to retreat to the traditional approach of building to rigid specifications and adversarial change management; and
- if the Customer has a fixed budget, this can be managed within an Agile project (unlike many waterfall projects) by focusing on development of the high priority items first, allowing the Customer to remove lower priority (or "nice to have") items from scope at a later date.

It is equally unlikely that a pricing model based solely on time and materials will drive the right behaviour by the parties. For example, a T&M pricing model is likely to result in disincentives for the Supplier to create realistic estimates and to stick to them.

All of these issues need to be taken into account when determining the pricing model to be used for an Agile development project. In particular, the parties need to be conscious that pricing needs to be addressed in relation to both individual iterations and the project as a whole.

Some potential pricing models include:

- fixed price per iteration (perhaps calculated by reference to the amount of work required for that iteration or the business value of the relevant development items);
- fixed price per user story (although this may be difficult if the user stories have a different scope or value); and
- fixed price for an agreed number of user stories.

As part of your comments on this paper, we would be very keen to hear about the types of pricing model you have seen in Agile projects (and how they have worked in practice).

One of the key issues for the Supplier in agreeing a pricing model will likely be revenue recognition. Under a traditional waterfall contract, the Supplier may be able to recognise the full contract price upfront whereas, under an Agile contract, the Supplier may only be entitled to recognise the charges that the Customer is contractually required to pay. This issue may be particularly relevant if the Customer is entitled to terminate the contract at the end of each iteration (or at other agreed points) without an obligation to pay the full contract price.

Key issues to be addressed in the contract

- The agreed pricing model will need to be clearly set out in the contract.
- This will need to include:
  - a description of the pricing methodology (e.g. fixed price per iteration);
  - when fees can be invoiced;
  - who bears the costs for items which have not been completed during an iteration (or which have not met the "Definition of Done" or equivalent); and
  - the impact of scope reductions or early termination (see separate section below).

Warranties and indemnities

Given the more iterative and collaborative nature of Agile development projects, what warranties and indemnities can the Supplier be expected to give? There are two key issues that it is useful to consider in this context.

"Compliance with specification" warranties

One of the key warranties in a traditional software development contract is that the finished product will comply with the functional specification. But one of the features of Agile is that a comprehensive functional specification is not developed at the outset of the project.

To bridge this gap, it may be useful if, on completion of the project, the Supplier prepares a "Product Description" which (amongst other things):

- contains a detailed description of the design and functions of the completed product; and
- demonstrates how the completed product is consistent with the Product Vision.

This document will not only provide a useful overview of the new product and the development project, but can also be used as the basis of Supplier warranties.

As with a functional specification, the Product Description should be subject to review and comment by the Customer, with any disputes between the parties being subject to the agreed dispute resolution procedure.

Composition of the Development Team

Earlier in this paper, we discussed whether the Development Team should include Customer personnel. One of the potential problems mentioned in relation to a combined Development Team was the difficulty in establishing a clear allocation of risk and liability between the Customer and Supplier. This problem is particularly acute when it comes to negotiating warranties and indemnities.

If Customer personnel are to be involved in the development activities on a day-to-day basis, then we suspect that the Supplier is going to be very reluctant to:

- offer any substantive warranties that the developed product (or individual product increments) will be free from defects, fit for purpose or of satisfactory quality; or
- offer a substantive indemnity against third party IP infringement claims.

For these reasons, combined with the possibility that the Customer may not have resources with the required skill set available to participate in the Development Team, it may be more appropriate for the Development Team to comprise only Supplier personnel. This would then put the Supplier in the position to offer the above warranties and indemnities.

Key issues to be addressed in the contract

- The contract should include appropriate warranties from the Supplier. These could include that:
  - the product is free from defects and of satisfactory quality; and
  - the product will comply with the agreed Product Description.
- If agreed, these warranties could be limited to a defined "warranty period" as per standard contracting approaches.
- If appropriate, these warranties could be given by the Supplier in relation to individual product increments at the end of each iteration.
- If a "Product Description" will be prepared, the contract should include a regime for the development and agreement of this document (including a dispute resolution mechanism where necessary).
- The contract may also include warranties from the Supplier regarding:
  - use of open source software by the Development Team; and
  - virus protection.
- The contract should also include a standard IP infringement indemnity from the Supplier. This can be supported by standard provisions dealing with conduct of proceedings and rights for the Supplier to modify the product so that it becomes non-infringing.

Liability

Just as with a traditional development project, the Supplier in an Agile contract will want to include appropriate limitations of its liability. The issues and discussions will, in many areas, be largely the same for Agile projects as for waterfall projects and will focus on issues such as:

- the overall cap on the Supplier's liability;
- liability of the Supplier for loss of profits, revenues etc. incurred by the Customer; and
- areas where the Customer may want the Supplier to have unlimited liability (e.g. under the IP infringement indemnity).

One area that may require special consideration is project delays. As mentioned above, each Sprint will be of fixed duration, but what if the Customer has deadlines for the overall project (or distinct phases within the project)? In this case, it may be possible for the parties to agree a timetable (which would sit across multiple iterations) for the development of a list of specific items within the Product Backlog.

Under this approach, the parties could then measure any delays against this overall timetable and allocate responsibility for such delays, with the possibility of the Supplier paying liquidated damages if the overall timetable (or individual milestone dates) are missed as a result of acts or omissions of the Development Team. By contrast, if the timetable is delayed as a result of the acts or omissions of the Product Owner (or other Customer representatives), the Supplier would be entitled to relief from liability.

Key issues to be addressed in the contract

- The contract should address limitations of liability using similar tools and concepts as standard development contracts.
- If appropriate, the contract should also address the impact of, and responsibility for, delays in the overall project (see above).

Termination

One of the key advantages of Agile development projects from the Customer's perspective is that it is not tied into long delivery cycles. Instead, the Customer should receive workable product increments at the end of each Sprint.

With this in mind, when should each party be entitled to bring an end to the project?

In particular, is it realistic in a project of any size for the Customer (and potentially the Supplier) to have the right to walk away from the project after each iteration?

Arguably, this is an inherent right for the Customer in an Agile project – at any time, the Product Owner could amend the Product Backlog to de-scope any outstanding items and declare the project is complete.

On the other hand, the Supplier may have invested significant time and resources in dedicating a Development Team to the project and feel that it is entitled to some form of compensation if the project is cancelled earlier than expected.

Key issues to be addressed in the contract

- The contract should clearly deal with each party's right to terminate the project.
- It is likely that the Customer will insist on having a right to terminate after each iteration (or, possibly, after defined "groups" of iterations).
- By contrast, the Customer may be reluctant to allow the Supplier to have the same flexibility as this creates the risk of the Customer being "left in the lurch" by the Supplier mid-project or, potentially, the Supplier using the threat of termination to renegotiate the pricing arrangements.
- The contract should also include standard rights to terminate immediately, such as a material breach by, or the insolvency of, either party.
- The contract should also address the consequences of termination, including the delivery to the Customer of work-in-progress including software code and copies of other working materials in their current state of development.
- The contract should also make it clear if and when any compensation will be payable to the Supplier if the contract is terminated early (e.g. some or all of the profit that the Supplier may have made from future iterations).

Intellectual Property Rights

The ownership of the IP rights developed by the Supplier is a key area in any development project. In particular, the contract will need to identify whether the Customer will own the IP rights in the developed product or whether the IP rights will be retained by the Supplier and licensed to the Customer.

The issues are largely the same for Agile projects as for waterfall projects, except that the IP ownership/licensing regime in an Agile development project will need to reflect the fact that the Supplier will be delivering product increments on a regular basis.

Key issues to be addressed in the contract

- The contract will need to clearly set out whether the Customer will own the IP rights in the developed product (and the individual product increments):
  - If so, the contract should include assignments of the relevant IP rights from the Supplier (and the Development Team). The Supplier may also be required to provide copies of the relevant source code to the Customer.
  - If not, the contract will need to include an appropriate licence in favour of the Customer. The Customer will also need to consider how the product will be supported going forward and whether software escrow arrangements are required.
- In any event, the contract should provide for the Customer to own all IP rights in the Product Vision and the Product Backlog (and, again, the contract may need to include assignments of IP rights from the Supplier if the Development Team has been involved in the development of the Product Backlog).

Dispute resolution

Given the more collaborative approach to Agile projects, it will be important for the contract to include a procedure that promotes the quick and efficient resolution of disputes without destroying the relationship between the parties.

It may be that this can be best achieved through a combination of:

- informal discussions by the parties, escalating up to senior management; and
- where this process does not resolve the dispute, determination of technical or financial issues by an independent expert,

with only the most serious or intractable disputes being referred to arbitration or court proceedings. Mediation should also be considered by the parties as a means of resolving disputes in an efficient and "non-destructive" manner.

Key issues to be addressed in the contract

- The contract should set out in a clear manner how disputes should be resolved. Initially, this could be done as follows:
  - informal discussions between the Product Owner and the ScrumMaster (assuming the ScrumMaster is a member of the Supplier team); and
  - if these discussions fail to resolve the dispute, escalation to the senior management of each party.
- If these informal discussions do not resolve the dispute, it may be appropriate for certain issues to be referred to an independent expert for resolution, such as:
  - disputes as to whether a particular item has been completed in accordance with the "Definition of Done";
  - disputes as to whether the Product Description is a fair and accurate description of the design and functions of the completed product; or
  - invoicing disputes (this may require a financial, rather than technical, expert).

Key Contacts

Ian Edwards
London

Tel: +44 (0)20 7905 6377
ian.edwards@twobirds.com

Roger Bickerstaff
London

Tel: +44 (0)20 7415 6160
roger.bickerstaff@twobirds.com

Christian Bartsch
London

Tel: +44 (0)20 7982 6458
christian.bartsch@twobirds.com

twobirds.com

Abu Dhabi & Beijing & Bratislava & Brussels & Budapest & Copenhagen & Dubai & Dusseldorf & Frankfurt & The Hague & Hamburg & Helsinki & Hong Kong & London & Lyon & Madrid & Milan & Munich & Paris & Prague & Rome & Shanghai & Singapore & Skanderborg & Stockholm & Sydney & Warsaw

Bird & Bird is an international legal practice comprising Bird & Bird LLP and its affiliated and associated businesses.
Bird & Bird LLP is a limited liability partnership, registered in England and Wales with registered number OC340318 and is authorised and regulated by the Solicitors Regulation Authority. Its registered office and principal place of business is at 15 Fetter Lane, London EC4A 1JP. A list of members of Bird & Bird LLP and of any non-members who are designated as partners, and of their respective professional qualifications, is open to inspection at that address.