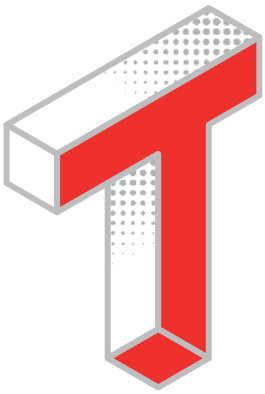
Contracting for Agile Services—a Paradoxical Journey

By Luke Fleming and Adam Herndon

There are various approaches to contracting for Agile services. Some shift risk between the parties, while others may undermine the benefits of Agile. As clients and vendors gain experience working with one another and mutual trust grows, their approach to contracting should evolve to support healthy Agile delivery.

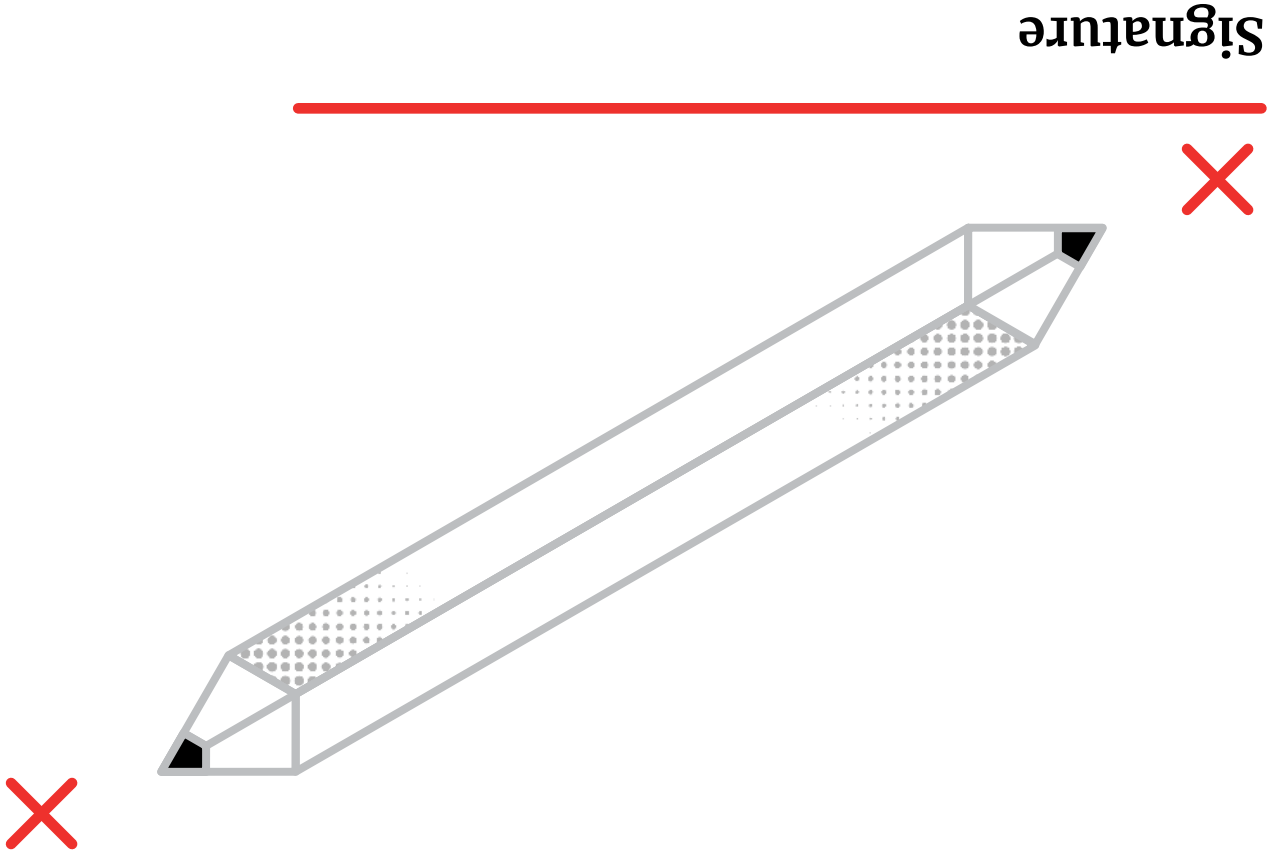


There's a disturbance in the Force. Have you felt it? Your organization has decided to bring in a third party to develop software or deliver a new system—and it isn't going well. At the outset, you and your vendor opted to conduct the project using Agile because you wanted to have more visibility into delivery, gain greater control over the finished product, or achieve another similar laudable goal.

Instead, you've found yourself constantly negotiating and renegotiating with your vendor, red flags are everywhere, and you're at risk of either not delivering the scope you committed to or significantly blowing your schedule and budget.

Too often, smart executives fall into a trap when it comes to Agile. They incorrectly assume that "being Agile" allows them to make changes to scope whenever they want and as often as they like—without making tradeoffs.

While Agile teams are able to pivot quickly to meet changing business needs, it isn't true that those changes can always be made without consequences to schedule and budget. Agile allows for changes, but with tradeoffs. The way companies contract for Agile services can exacerbate this problem.



Signature

Agile Agreements Are Different

Under Waterfall, the software development lifecycle is sequentially ordered: requirements are gathered, the solution is designed, build occurs, testing follows, and there is a "big bang" release to production at the end. Companies are able to contract for vendor support at each phase of this process. Because requirements are gathered at the outset, the parties typically have more information upon which to estimate their work. This creates a sense of "certainty," whether real or imagined, that allows for easier contracting.

The basic criticisms of Waterfall are that it is slow and that what was intended at the outset isn't what's delivered in the end, either because of translation noise in the process or because business needs, customer preferences, or other situations have changed. Agile can be an effective antidote for these challenges as it breaks down the development process into smaller increments of work with more frequent releases and scope can be adjusted to meet fluid situations. In theory, this translates into quicker and more frequent value delivery to clients and more responsiveness to business needs and customer preferences.

However, these benefits pose challenges for contracting parties. Because requirements are gathered in parallel with design, build, and test, and are not gathered in their entirety at the outset, there is less information upon which the parties can estimate their work. Requirements and scope are also subject to change, and thus, there is more ambiguity and potential complexity when contracting for Agile services.

Contracting Options

Organizations generally contract for Agile services using one of five approaches: *fixed price*; *time and material*; *shared benefit*; *fixed scope with a capacity buffer*; or *fixed capacity*. Each of these approaches, and numerous permutations thereof, has advantages and disadvantages that should be viewed in context and considered at the outset of any engagement.

By far, the two most common approaches companies use in contracting for Agile services are the fixed price and the time and material agreements. A fixed price contract, as the name implies, is an agreement by the vendor to provide a predefined scope, for a fixed sum, irrespective of the effort ultimately expended.

Clients often favor this type of arrangement because they can control costs while shifting risk to the vendor. Because the client is paying for an outcome and not effort, inaccurate estimates or unanticipated circumstances may be borne by the vendor. Because they are widely used, fixed price contracts also have the advantage of being familiar to both parties.

Unfortunately, fixed price contracts can also be antithetical to Agile. One of the primary benefits of Agile is the ability to develop and release incremental functionality to production, learn from those releases, and course correct as needed.

In this way, Agile software development may involve regular changes to and reprioritization of scope. Changes to scope—whether due to evolving requirements, revised priorities, or user-provided feedback—usually require negotiating and drafting change requests to fixed price agreements, which can adversely impact continuity of value delivery. Instead of focusing on delivering value to customers, clients are forced to spend time renegotiating contracts with vendors.

Redistributing Risk Between Client and Vendor

The other most common form of contracting for Agile services is the time and material contract. In a time and material contract, the client agrees to pay the vendor for performance at a predetermined rate per unit of time. Scope may be loosely defined, but because the client is paying for the vendor's time, shifts in scope usually won't necessitate contract renegotiation.

Staff augmentation contracts are usually structured as time and material agreements. Under those agreements, the client generally sets the direction for the vendor's work and is typically free to change the scope as needed. Generally, the client is also free to terminate the contract at any time.

Under time and material arrangements, risk is redistributed between client and vendor. The vendor faces less risk because the client has committed to paying for the time the vendor works. The client, on the other hand, is absorbing more risk because, unlike in the fixed contract relationship, the vendor doesn't commit to delivering specific scope.

Because vendors are paid for the time spent supporting the client, they may have little incentive to complete their work quickly.

In Agile, the increased risk of not having a guaranteed outcome is offset by the client's ability to prioritize vendor work.

In theory, this should translate into the highest priority items being completed first. In practice, it isn't that clean. Various factors, including external dependencies, may inhibit successful delivery of those higher priority items. Because vendors are paid for the time spent supporting the client, they may have little incentive to complete their work quickly.

Limited experience working with one another and asymmetric vendor knowledge may further exacerbate client suspicion that the vendor, to gain financial advantage, is taking longer to complete the work than is necessary. Service-level agreements may help, but they are a crude and imprecise means of addressing the problem, and they're only as good as how they're defined and the means by which they're enforced.

Tying Effort to Results

Less common forms of contracting for Agile services include what we'll call shared benefit, fixed scope with a capacity buffer, and fixed capacity agreements. Shared benefit contracts are outcome-based contracts in which the vendor and the client agree to share the anticipated benefits accrued.

Under this type of arrangement, the vendor and the client might agree to a base time and material rate, with some sharing of the anticipated benefits arising as a result of the project. Both sides must accept a certain level of risk under these agreements.

The vendor effectively agrees to tie more of its margins to the outcome of its work. The vendor must have a high degree of confidence in the outcome of its work, especially because the vendor usually has limited control over the rollout and monetization of the finished product.

The client, on the other hand, must be careful to accurately assess the anticipated benefits of the work performed. If the client underestimates those benefits and has shared some portion of the outcome with its vendor, the client may considerably overpay for the vendor's services.

If, however, the anticipated benefits do not materialize, the client may ultimately pay less for the vendor's services than it would have under a time and material or fixed price agreement.

In this way, this type of arrangement may serve as a hedge against uncertain and/or unrealized benefits from the work performed. This type of contract is rare for a reason: It requires considerable trust between the vendor and the client—which may not exist unless the two parties have already successfully worked together.

Addressing the Limits of Fixed Scope

An agreement that's fixed scope with a capacity buffer is a tacit acknowledgement of the complexities of Agile contracting. Under such an arrangement, the parties essentially agree to fix the scope of the project. This guarantees that the vendor will deliver a specified scope at an agreed-upon price.

For some, this type of contract is antithetical to Agile because it constrains the product owner's ability to adjust the scope to better meet user needs. Contract needs become a barrier to healthy Agile process.

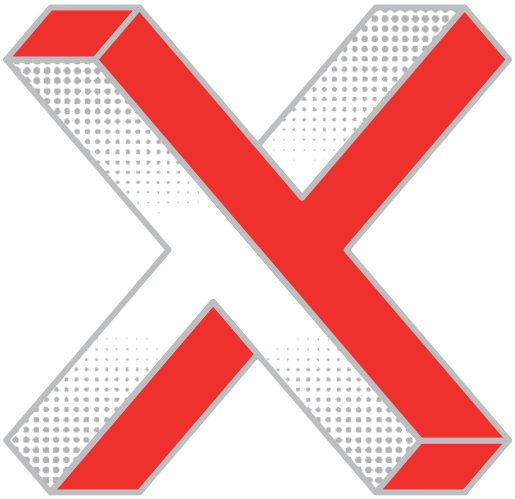
Because requirements are not gathered all at once, teams often have limited information upon which they can estimate the full "fixed scope." Teams typically won't have more than a few sprints worth of fully defined user stories. As such, estimates beyond those defined user stories are by necessity high level and less accurate.

Including a capacity buffer acknowledges the limitations of estimating the full scope with incomplete information. By retaining 10 to 20 percent of the team's bandwidth as a buffer, vendors are partially protected against inaccurate estimates.

Clients, on the other hand, get the certainty of a fixed scope agreement and, assuming the full buffer hasn't been used, the opportunity to add new scope or adjust existing scope to provide value to users. If, however, the capacity buffer is exceeded, the parties must resort to contract renegotiation.

Trust Is the Key

Fixed capacity agreements are, arguably, the most true to the spirit of Agile, because they allow clients to adjust scope to meet customer needs without resorting to tedious contract negotiation that can disrupt value delivery. Unfortunately, these types of agreements also require a deeper understanding of Agile and trust between the parties. Under this type of arrangement, the client and vendor agree on a fixed number of story points representing a level of effort by the team. Because the team's capacity—its ability to complete work within a given time frame—is finite, the parties are essentially contracting for a fixed amount of effort by the team.



In Agile, development work or effort is typically estimated using story points. Story points can be calculated in different ways, but one common way involves the use of Fibonacci numbers (e.g., 1, 2, 3, 5, 8, 13, etc.)—a sequence of numbers in which each number is the sum of the previous two. An Agile team might estimate an initial medium-sized user story to get a baseline. Subsequent user stories are then estimated relative to that initial user story.

Thus, estimation is partly subjective in that it may be relative to an initial estimate of work and, because it is done at the team level, may be unique to that team. Additionally, where team composition is fluid, estimates become less predictable while the new team is formed.

Because of this subjectivity and variability, clients may find it extremely difficult to understand what value they'll derive from a given number of story points unless they've worked with the team previously and that team's composition has remained steady. As this is frequently not the case with vendor teams, this type of contract is extremely difficult to put into practice.

How to Decide?

Deciding which contract approach to use is often a function of the circumstances between the parties and their respective tolerance for risk. When the vendor and client have a history and have developed mutual trust, they may be able to use a contract approach that supports healthy Agile processes and allows product owners to deliver continuous value to users—free of intrusive contract negotiations.

In situations where there is limited history and/or trust between the vendor and client, or where the client has limited experience with Agile, the parties may need to build toward effective contracting for Agile services. By beginning with a time and material arrangement, vendor risk is limited, and the client retains the flexibility to direct the vendor's work and alter the scope as needed.

While this type of contract may not provide the level of predictability clients crave, it can allow the parties to work with and learn from one another while still maintaining control over the work.

Over time, as the parties learn to work with one another and velocity or throughput becomes more predictable, an opportunity to move toward a fixed capacity model may arise. Under such circumstances, the parties would contract for a fixed amount of effort in the form of story points. Having worked together previously, both parties better understand the level of effort captured by a given number of story points and the client is forced to assess the relative value of work and make the types of trade-offs that Agile requires.

Contracting for Agile services is not easy, but by focusing on transparency, partnership, and continuous improvement, clients and vendors can evolve how they work with one another and move toward a contracting model that supports Agile processes and enables value delivery to users. ✕

Luke Fleming
luke.fleming@jabian.com

Adam Herndon
adam.herndon@jabian.com