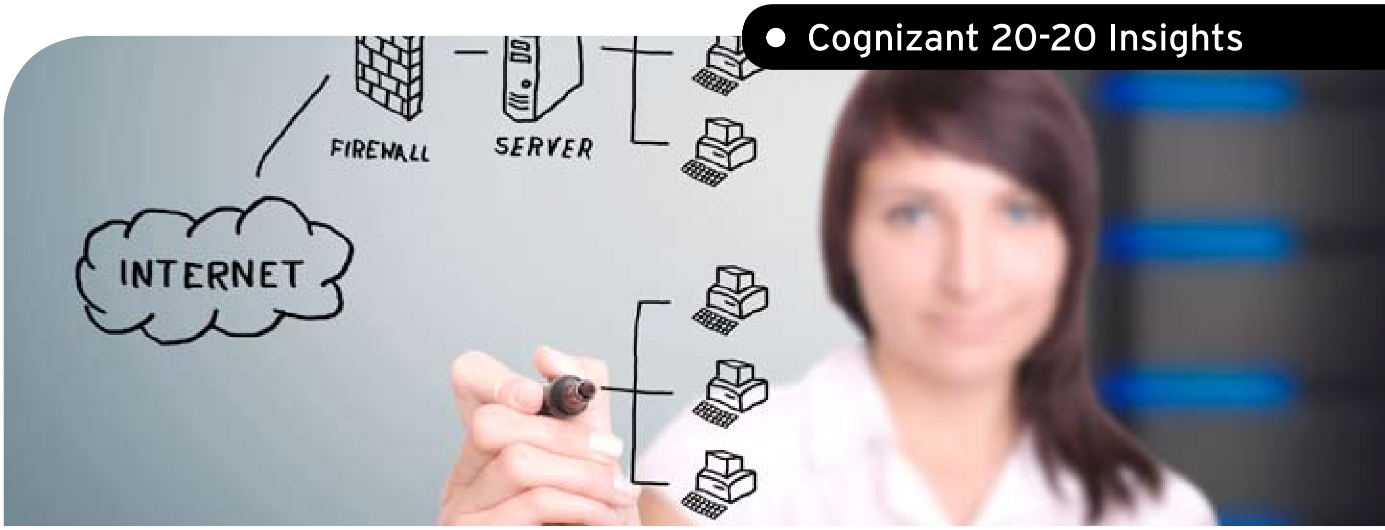
Cognizant 20-20 Insights



An Agile Twist: Fixed-Bid Pricing

Executive Summary

Have you recently been asked to create a proposal using one of the accepted Agile practices, such as Scrum or Kanban, to deliver a new project or program? Is your account manager or client asking you to fit this request into a traditional fixed-bid contract? Yet everything you've read says Agile practices do not support a fixed-scope, let alone a fixed-price, contract – and to make matters worse, the more research you do on Agile practices, the more you realize you are beginning to agree with the experts.

As business managers and CIOs in Fortune 1000 companies begin to hear about Agile software development and the benefits of being Agile, clients have adopted the view that Agile is a "silver bullet" that is going to make all things better. More and more large companies are seeking ways to use Agile principles to develop their next big program.

As you explore the ability to gain agility (which is the organization's take on the primary benefit of using Agile) while satisfying the need for fixed scope and price, you will run into many client-vendor relationship issues; quite frankly, some of these may be impossible to resolve.

This white paper will explore and discuss some of the base issues and investigate an alternative to the orthodox Agile pricing model that can work in a fixed-bid model while following as many Agility principles as possible to meet the organization's expectations. While you will still need to consider the unique organization issues, the foundational steps presented below should assist you in getting started.

The Difficulty of Pricing and Agile

In the past few years, we have all increasingly been receiving requests for proposals (RFPs) that include the use of the Agile methodology, with the expectation that we will use the normal fixed-bid style of contract. Yet, Agile states it is a time-boxed, flexible-scope methodology. A fixed-scope and fixed-priced contract is just as it sounds – it contains a contractual fixed statement of work and a set, predetermined price for the scope of work.

If your team is more efficient than expected at performing the work, your profitability numbers will look great. If your team is inefficient, or the leadership overlooked risks on the project (or scope-creep occurs), profitability drops, the project becomes extended, and you may even lose money on the deal.

The prospect of not accounting for all potential variances from the get-go typically forces many assumptions and risks to be written into traditional fixed-bid contracts. In the conventional world of IT development and services, we have learned – through much trial and error – that the goal is to develop a rock-solid statement of work (SoW) that is clear to both parties and from which neither party will deviate, accompanied by a series of immovable change controls.

This is where the problem begins. When following an Agile practice, the mindset is completely different: Change is meant to be embraced, not avoided. The experts will say (and most importantly, the client organization will expect) that "easy change" without traditional change control is a key benefit of Agile. To make matters more difficult, they will inform you that managers (particularly those on the business side) can change the scope or reprioritize the work as often as they wish. This is why many experts will tell you that Agile, by its very nature, is inherently an open-ended time-and-material (T&M) type of contract — which vendors love but clients are hesitant to sign up for.

This creates a situation in which the client organization is expecting to realize a key purported Agile benefit that change is easy. Management may even say, "There really is no change control. The business owners can change their mind at any time."

Can you already hear your legal office crying foul? And remember, both the software developer and client organization want to know how much the project will cost, not to mention how long it will take.

It is, in fact, possible to develop a fixed-price contractual agreement that also accommodates quick and easier change that results in higher agility. This solution does not match the model of pricing in the purists' Agile world, but it does navigate the troubled waters described above and calms the fears of corporate accountants, lawyers and top executives.

Agile Fixed-Bid Solution

Let's explore a few steps of how we can modify both the fixed-bid theory and the Agile practice to meet somewhere in the middle. First, it's important to determine whether you are trying to complete the entire large program as a fixed contract or a first (single) release-type project as a fixed contract. We recommend starting out with the latter, since it will allow both the vendor and the client to learn about the project goals and important Agile principles such as sustainable velocity. As we explore this issue, we will point out some key areas where the approach may differ slightly depending on which contract type you are working toward.

Here are the main steps for your consideration:

- **Step 1:** The first thing all contracts focus on is the scope or statement of work. There are different ways to approach this requirement, but consistent across all cases is that this step is key to a successful contract — as a foundation, we must have a solid list of stories from which to work. The story list or story backlog is the fundamental bottom-line artifact of Agile and can/will become the foundation of the fixed-bid contract, as well.

  To support a fixed-bid proposal, the client and vendor will need to spend one to three months working on the story backlog. These stories can be epic in nature, as long as the vendor and client can easily understand the basic scope or product being defined by the stories.

  Stories tend to be better written than the old traditional requirements, so they lend themselves to a clear English-language style of elaboration and definition.

  The fundamentals of a story are a clear statement of whom, what and why, along with a clear definition of acceptance criteria, relative sizing and business value rating or prioritization based on business value.

  Here is an example of an effective story: "As a customer service agent, I want to be able to see all the products a customer has purchased from our company, so that we can serve our customer as a whole, not just my single product line."

  Example of acceptance criteria: "All six of our separate and independent product line systems will have the customer data displayed in the single UI."

  Special note: Whether you are doing the entire project or the first-release approach within your contract, you need to ensure your team has decomposed the first release into true, workable stories that can be sized for a single iteration or sprint. The stories for the first release need to be relatively sized in preparation for Step 2.

- **Step 2:** Once we have completed the product/program story backlog, it becomes important to estimate velocity, if you do not already know your stable estimated velocity. Doing this requires a random set of stories and asking the Scrum team to break them into tasks and complete a detailed estimate for each task. Alternatively, if you have time, you can ask them to do the anticipated first iteration or two worth of stories.

  Special note: If you are taking the whole-contract approach, you might want to consider doing several iterations worth, up to the first release here. This is especially true if you are conducting a multi-year program. This allows the team to refine as it goes, and you can get a little more data for your velocity estimate. But we caution companies to time-box it to no more than a week's effort, because no matter how long this early effort takes, you will most likely still need to re-estimate based on the actual or sustainable velocity that comes into play after five to ten iterations.

Once you have done this, the team needs to determine what their "ideal" sprint looks like in hours. For example: Four developers will work 6.5 to 8 hours a day for five days a week, except for Sue and Amid. Sue will only be able to work four to six hours a day for five days a week, and Amid is available for six to eight hours, three days a week.

So if there is a two-week sprint, and we assume one day for sprint planning and two days for retro and demo, then we know we have seven work days available.

Taking the aforementioned hours, we see that the core four members of the team can work 182 to 224 hours. Sue can work 28 to 42 hours, and Amid can work 36 to 48 hours per sprint. So, that gives us a range of 246 to 314 hours in our "ideal" sprint.

Now that we know our ideal sprint size in hours, we can now estimate velocity. If we take the stories that have been broken into tasks and total the estimated hours for all tasks (per skill set) within each of the stories, we then can select how many story points we can put into a sprint.

Effectively, this is easy; you want the team to select the number of stories until the total hours for the team is somewhere within the low and high range. We would suggest for the first sprint to aim for closer to the low range than the high range.

- **Step 3:** This step is focused on release planning, driven by estimated velocity. Based on the average story points that you determine can be completed in the velocity estimation step, you can now look at your entire backlog and count how many sprints you will need to complete all of the backlog.

  By placing them into sprints based on the velocity and business value, you can then understand where your natural releases are. While it is optimal to have the same periods for releases, it is not necessary. So, for example, the first release may be a bit longer than subsequent releases.

  Now you should be able to develop your roadmap to your contract. But be wary that everything is based on rough estimates at this point, so this fixed-price contract really should be for the first release, only. I sometimes like to call it the learning-pilot project. Once you get through release one, you, the client and your team will be much better at estimating for the next release. Actually, at this point, we would suggest you always base your contracts on single releases and no more than two or three releases, based on not exceeding the magical six-month timeframe by a large amount.

The best approach is what we call the 60/40/20 rule.

- **Step 4:** At this point, the contract agility buffer comes into play. We believe there are two practical ways to introduce and plan for agility in a fixed-bid contract. The most common way is to include some level of capacity buffer, typically 20% to 40%. The other is some form of a scope buffer, which is basically identifying some percentage as the focus of the project scope, while the remaining is flexible.

  We would venture to guess anyone interested in this white paper would know how to apply a capacity buffer and understands the many issues and challenges that come with that type of buffer. So we are not going to spend any time here discussing the capacity buffer approach but instead will focus on the scope buffer approach.

  The best approach is what we call the 60/40/20 rule. To be sure, we are not the first to use this style of buffering, but this is our suggested way to approach inserting and planning for Agility in a fixed-bid contract. This approach focuses the decision process on product owners and their prioritization and negotiation of the backlog.

  Effectively, the 60/40/20 rule breaks the release backlog and project scope into a 60%, 40% and 20% prioritized list. The categories are: 60% is "must-have," 40% is "would like to have," and, finally, 20% is considered a "stretch goal."

  The big chunk of 60% is the main absolute target of the project, or what we call the "in-scope" backlog. So you would identify 60% as "must-have" and 40% as being the "would like to have." The 40% is also the target of the project, but this category also contains the negotiable items that can be removed and replaced during change management.

The final "stretch goal," or 20% of the total project, is a list of items we would work on if we finish the "in-scope" backlog early.

Note here: Some teams do not add this 20%, since the idea is that this is a fixed project, so there should be a way to benefit from early delivery.

Another note: The 60% "must-have" and 40% "would like to have" equals the "in-scope" backlog, or 100% of the projected delivered scope. Additionally, this effectively introduces what we call the negative and positive buffers.

The negative buffer is the "would like to have" 40%, as this is where we negotiate stories. The "stretch goal" 20% target is a positive for the customer, in that additional value is sometimes layered into the contract scope, beyond what would normally be provided, if time and budget permits. Basically, this prioritized list is where work is selected from if the "in-scope" backlog is finished ahead of schedule. We know this is not normal for fixed-bid contract projects, but we suggest you actually deliver from the positive to build extreme trust with the client.

> The foundation of Agile is that it is simply a capacity vs. need negotiation at each sprint planning session.

Always remember, the primary reason for the 60/40/20 rule is to allow for agility within a fixed-price change control process, while maintaining the security of a fixed-price contract. Careful and thoughtful consideration must be made with the client and product owner. Once you have completed this list, you simply include it, in keeping with the scope section of the project.

In summary: The 60% is what we absolutely need, so if you must remove or change something already completed, you select and remove something from the 40% "would like to have" list and replace it with the changed item of work.

With that said, if something in the 60% area is no longer needed, then even that area can be swapped out, by swapping with a story from the "would like to have" 40%.

- **Step 5:** Once you get through Step 4, you now have a sense of time and size. You should be able to create a decent fixed-price contract, with one outstanding issue: The change control management plan. In essence, there are three main components to consider:

1. There must be a high level of trust developed between the client and vendor regarding change control, because the next two required components will not be very successful without it.
2. The basis of the scope is the selected story backlog for the release. If the client decides to change a story that has been completed (ready for test) or add a new story, then some level of change control is required. Once the new story has been fully defined and accepted, the client should then select an equally sized item from the original backlog to be removed and placed back into the program backlog instead of the release backlog. Please see the 60/40/20 rule for more clarification. At that point, a zero-cost change control is issued to change the fixed-scope portion of the contract.
3. The alternative is that the client wants to add an item to or change a completed item on the backlog without removing anything from the release backlog. In this case, we would again fully develop the new story, and then once accepted, we would create a cost-type change control based on the sizing. At a minimum, the cost would change scope, but it would potentially change price and time, as well.

Moving Forward

While this approach will get you through the initial phases of Agile implementation, it is important to look to the future. Agile's finest benefits are found in the agility to change, so we need to work toward the development and negotiation of contracts that better suit Agile development practices.

As previously mentioned, Agile best suits a time-boxed, time-and-materials-style contract. The foundation of Agile is that it is simply a capacity vs. need negotiation at each sprint planning session. The client organization should be in direct control of what will be delivered with each sprint or iteration, and the movement of stories up and down the backlog is a simple business value decision and has no impact on the contract itself.

Note: Make sure the teams complete all of the 60% portion before beginning work on the 40% negotiable portion. However, there is a caveat — most client companies shy away from time-and-material contracts, feeling there is little incentive to produce and finish within a schedule, with cost overruns being very common. Agile actually does control these issues with the close interaction that occurs on a regular, if not daily basis with the client organization. Our 60/40/20 rule is a mid-way suggestion between T&M (i.e., time and materials) and a full fixed-bid type contract.

The next step toward the time-boxed T&M contract is a time-boxed, fixed-capacity style contract. This contract is for a certain length of time (commonly six months at a time), for a set number of resources during that period.

Additionally, if your team has experience with Agile, and you have confidence in the story sizing and level of your velocity estimation, you can include the estimate number for story points that can be delivered during the contract time period.

If you include an estimated number of story points to be delivered, be careful that you have completed the analysis phase to determine your product or release backlog for the contract in question and that your team has worked together long enough to have a stabilized velocity, so that you have reasonable confidence in the sizing of those stories. Ensuring that good story elaboration has occurred is the most important consideration, here.

Implementation Considerations

Here are a couple of items to include in your contracts:

- Include a special statement in the change control section. We suggest the wording used in the Step 4 section of this white paper as the foundation.
- Take special care in the assumptions section – we believe this is crucial to your success with this approach. Suggested wording for some of the assumptions are:
  - We assume that any enhancement/change request communicated after the product/release backlog sign-off that is the focal point of this agreement will be treated as a project change request (PCR). Management of a PCR will be outlined in the change management plan.
  - A change request should require removal of scope (i.e., a like-for-like or equal-size story) from the "would like to have" 40% scope section of this contract of equally valued items, or it becomes a re-pricing consideration. If new scope is added, it will increase the stated scope in the scope section of this agreement. Either way, the changes to scope will be handled as a project change request.

These two assumptions – and the careful wording of the change management process – will take you a long way toward managing organizational expectations around change management. Remember, this is the most contentious area for an organization when doing Agile. They will expect easy change control, so it is important to work closely with them and make the process as easy as possible.

Additionally, we would strongly suggest careful conversation early in the negotiations and RFP process to ensure that the client organization understands and agrees to the process well in advance of the contract delivery. Make sure they "own" this process – this is essential to the success of this aproach.

One major component to successfully transitioning to the Agile practice is to have a strong and trusting relationship.

Putting Theory into Practice

Agile development practices are becoming the rage throughout the information technology development community and, recently, within the business community. This is placing pressure on many established teams within client companies and pushing vendors to make dramatic changes to the way IT projects are scoped and managed.

Unfortunately, this change does not always manifest itself in QA and procurement practices to support the new way of thinking. But with the few simple techniques in this white paper, we believe you can successfully negotiate these murky waters.

There are many different Agile practices to consider, and each may have some nuances that would impact the suggestions discussed above, so it is important to note here that the Scrum Agile practice approach was the basis of this study. It is also important to note that your team should use our suggested approach only as a foundation and a means for starting the discussion. For example, the 60/40/20 rule is really more of a recommendation, not a hard-and-fast rule. We suggest that you set your percentages to what works for you.

The organization culture will always contain special nuances that require careful consideration, as well. The client organization's procurement and legal teams may take a lot of convincing that this suggested contract style for supporting Agile projects can and will be successful. Moreover, they may be very skeptical and untrusting, at the start. So having a senior Agile coach/consultant who understands the process and the multiple approaches to Agile will be very much worth the cost.

As a final take-away, remember that — as with all interactions with your client — the building of clear expectations and honest communication will go a long way toward developing a trusting partnership. One major component to successfully transitioning to the Agile practice is to have a strong and trusting relationship.

About the Author

Carl G. Shea is a Senior Manager and North American Agile Practice Lead with 25 years of IT experience, serving in Cognizant's Advanced Solutions Group (ASG). Carl is an accomplished IT leader with a solid history of successfully delivering top-quality products, resources, support and guidance to multimillion-dollar programs across both government and corporate industries. He is an active and contributing member of the Project Management Institute (PMI), Scrum and Agile alliances and a Fellow of the American Academy of Project Management. He combines training and hands-on system development expertise with proficiency in a myriad of technical tools to facilitate project success, and he is a creative leader and strong believer that "management is doing things right and leadership is doing the right things." Carl is a PMP, MPM and an Agile Coach with Certified Scrum Master and Scrum Professional designations. He can be reached at Carl.Shea@cognizant.com.

About Cognizant

Cognizant (NASDAQ: CTSH) is a leading provider of information technology, consulting, and business process outsourcing services, dedicated to helping the world's leading companies build stronger businesses. Headquartered in Teaneck, New Jersey (U.S.), Cognizant combines a passion for client satisfaction, technology innovation, deep industry and business process expertise, and a global, collaborative workforce that embodies the future of work. With over 50 delivery centers worldwide and approximately 111,000 employees as of March 31, 2011, Cognizant is a member of the NASDAQ-100, the S&P 500, the Forbes Global 2000, and the Fortune 500 and is ranked among the top performing and fastest growing companies in the world. Visit us online at www.cognizant.com or follow us on Twitter: Cognizant.

Cognizant

World Headquarters
500 Frank W. Burr Blvd.
Teaneck, NJ 07666 USA
Phone: +1 201 801 0233
Fax: +1 201 801 0243
Toll Free: +1 888 937 3277
Email: inquiry@cognizant.com

European Headquarters
Haymarket House
28-29 Haymarket
London SW1Y 4SP UK
Phone: +44 (0) 20 7321 4888
Fax: +44 (0) 20 7321 4890
Email: infouk@cognizant.com

India Operations Headquarters
#5/535, Old Mahabalipuram Road
Okkiyam Pettai, Thoraipakkam
Chennai, 600 096 India
Phone: +91 (0) 44 4209 6000
Fax: +91 (0) 44 4209 6060
Email: inquiryindia@cognizant.com