Dear Customer: t*he truth about IT projects*

By Allan Kelly

Dear customer,

I think it's time we in the IT industry come clean about how we charge you, why our bills are sometimes a bit higher than you might expect, and why so many IT projects result in disappointment. The truth is that when we start an IT project, we don't know how much time and effort it will take to complete. Consequently, we don't know how much it will cost. This may not be a message you like to hear, particularly since you are *absolutely certain* you know what you want.

Herein lies another truth, which I'll try to put as politely as I can. You are, after all, a customer, and, really, I shouldn't offend you. You know the saying "The customer is always right"? The thing is, you don't know what you want. You may know in general terms, but the devil is in the detail—and the more detail you try to give us beforehand, the more likely your desires are to change. Each time you give us more detail, you are offering more hostages to fortune.

Software engineering expert Capers Jones believes the things you want ("requirements," as we like to call them) change 2 percent per month. Personally, I'm surprised that number is so low!

Just to complicate matters, the world is uncertain. Things change, and companies go out of business. Remember *Enron*? Remember *Lehman Brothers*? Customer tastes change. Remember *Cabbage Patch Kids*? Fashion changes, governments change, and competitors do their best to make life hard. So, really, even if you do know absolutely what you want when you first speak to us, it is unlikely that it will stay the same for very long.

I'm afraid to say that there are people in the IT industry who will take advantage of this situation. They will smile and agree with you when you tell them what you want, right up to the point when you sign. From then on, it's a different story; they know that changes are inevitable,

and they plan to make a healthy profit from change requests and late additions at your expense.

While I'm being honest, it is true we sometimes gold plate things. You might not need a data warehouse for your online retailer on day one. Yes, some of our engineers like to do more than what is needed, and yes, we have vested interest in getting things added so we can charge you more.

It is also true that you quite legitimately think of features and functionality you would like after we've begun. You naturally assume something is "in" when we assume it is "out." And, in the spirit of openness, can you honestly say that you've never tried to put one over on us? (Let's not even talk about bugs right now; it just complicates everything.)

Frankly, given all of this, it is touching that you have so much faith in technology to deliver. But, when IT does deliver, boy, it delivers big. Look what it did for Bill Gates and Larry Page, or Amazon and FedEx. Isn't it interesting that when the IT industry develops things for itself, we end up with multi-millionaires. When we develop for other people, they end up loosing money.

How did we ever talk you into any of this? Well, we package up this unsightly mess and try to sell it to you. To do this, we have to hide all this unpleasantness. We start with a ritual called *estimation*—how much time we think the work will take. These "estimates" are little better than guesses. Humans can't estimate. We've known this since at least the late '70s, when Kahneman and Tversky described the "planning fallacy" in 1979 and went on to win a Nobel Prize. Basically, humans consistently underestimate how long work will take and are overconfident in their estimates.

To make things worse, we have a bad habit we really should kick. Between estimating the work and doing the work, we usually change the team. The estimate may be made by the IT equivalent of Manchester United or the New York Mets, but the team that actually does the work is more than likely a rag-tag bunch of coders, analysts, and managers who've never met before.

Historical data – data about estimates, actuals, costs, etc - can help inform planning but most companies that don't have their own data. For those that do have data, most of it is worse than useless. In fact, Capers Jones suggests that inaccurate historical data is a major course of project failure. For example, software engineers rarely get paid overtime so tracking systems often miss these extra hours. Indeed, some companies prohibit employees from logging more than their official hours in their systems.

So, we make this guess (sorry, *estimate*) and double it—we might even triple it. If the new number looks too much, we might reduce it. Once our engineers have finished massaging the number, we give it to the sales folk, who massage it some more. After all, we want you to

say yes to the biggest sticker price we can get. That might sound awful, but remember: We could have guessed higher in the first place.

Please don't shoot me, I'm only the messenger.

We don’t know which number is “right,” but to make it acceptable to you, we pretend it is certain and we take on the risk. We can only do this if the number is sufficiently padded (and, even then, we go wrong). If the risk pays off, then we get a fat profit. If it doesn’t, we don’t get any profit and may take a loss. If it’s really bad, you don’t get anything and we end up in court or bust.

The alternative is that you take on the risk—and the mess—and do it yourself. Unfortunately, another sad truth is that in-house IT is generally even worse than specialist providers. For a software company development is a core competency, such companies live or die by their ability to deliver software, if they are too bad they cease to trade. Evolution weeds out the poor performers.

Corporate IT on the other hand rarely destroys a business – although it may damage profits. Indeed Capers Jones research also suggests specialist providers are generally better than corporate IT departments.

Sales folk might be absent, but the whole estimation process is open to gaming from many other sources and for many other reasons. Bottom line: if you decide to take on the risk, you may actually increase risk.

I know this sounds like a no-win scenario. You could just sit on the fence and wait for Microsoft or Google to solve your problems with a packaged solution, but will your competitors stand still while you do? Will you still be running a business when Google produces a free version?

By the way, beware snake oil salesmen selling off-the-shelf applications. Once people start talking about "customization" or "configuration," you head down a slippery slope. Configuring a large SAP installation is not a matter of selecting Tools, Options then ticking a box. Configuring large packages is a major software development activity, no matter what you have been told. The people who undertake the configuration might be called “Consultants” but they are really specialist software developers.

There really isn’t a nice, simple solution to any of this. We can't solve this problem for you. We need you, but you have to work with us. As the customer, you have to be prepared to work with us, the supplier, again and again in order to reduce the risk. Addressing risks in a timely and cost effective manor involves business level decisions and trade-offs. If you aren’t there to help we can either make the decision for you (adding the risk that you disagree) or we will spend your time and money to address it.

You need to be prepared to accept and share the risk with us. If you aren't prepared to take on any risk, we will charge you a lot for all the risk we take on.
Sharing the risk has the effect of reducing the risk because once the risk is shared you, the customer, are motivated to reduce risk. One of the major risks on IT projects is a lack of customer involvement. You can help with that just by staying involved.

Ultimately all risk is your risk, you are the customer, you are paying for this project one way or the other, if it fails to deliver value it is your business that will suffer. When you share risks, when you are involved closely, risks can be addressed immediately rather than being allowed to fester and grow.

Finally, you may have grand ambitious, but we need to work in small chunks. I know this may not sound very sexy, but software creation works best when small. Economies of scale don't exist. In fact, we have diseconomies of scale, so we need to work in tiny pieces again, again, and again. If you are prepared to accept these suggestions, then let's press reset on our relationship and talk some more.

Yours sincerely,

The IT Industry

About the Author

Allan Kelly, BSc, MBA is London based independent consultant. Allan is the author of seven books, a conference keynote speaker, co-founder of the Agile on the Beach conference and pioneer of techniques such as *Value Poker*, *Time-Value Profiles* and *Retrospective Dialogue Sheets*.

Allan started his career as a software engineer and after working in the City of London and Silicon Valley moved into team leading and product management. He maintains an interest in technology and programming and recently open sourced his first product.

When not writing – or coding! - Allan advises, coaches and trains software teams. He helps companies embrace the agile and digital world and teams to improve their performance. His blog is at www.allankelly.net/blog, on Twitter he is @allankellynet and he can be contacted at allan@allankelly.net.

His books include:

- The Art of Agile Product Ownership (Apress, 2019)
- Continuous Digital (LeanPub, 2018)
- Project Myopia (LeanPub 2018)
- Xanpan: team centric agile software development (LeanPub 2015)

- Business Patterns for Software Developers (Wiley, 2012)

And his best seller, Little Book of Requirements and User Stories (LeanPub, 2017).

History

Dear customer was originally published in The Agile Journal (now Agile Connection) in March 2012. The essay forms the prologue to Xanpan and an audio version is available at on Allan's website.