FLIPPING THE SCRIPT ON CORE TRANSFORMATION

CAPCO

THE FUTURE. NOW.

INTRODUCTION

In my experience, core system transformation is a bit of misnomer because teams end up building projects backward, and as a result, they take far too long and cost far too much to implement. Most importantly, insurers lose some of the benefits of moving to the new system(s) because the team has spent too much time in the past instead of looking toward the future.

For the small and medium (SMBs) insurance market, the core system transformation model cannot work. SMBs cannot afford the luxury of investing two to four years and $20-50 million. Fortunately, there's a significantly better way to implement these projects. First, you must flip the script on how you execute them.

CURRENT 'AGILE' APPROACH

Core system replacement is often undertaken with a systems integration (SI) partner with experience in the selected software package. As such, you must manage the commercial constructs. One way to do that is to define the scope of work during contract negotiations (based on an RFP, for example) or after an inception period. In both cases, the frame of reference is the carrier's perceptions of need paired with the SIs perception about how the software meets those needs (or not). Sometimes, the software vendor is part of the discussion, especially in those cases when they are the SI.

From there, you create Agile, high-level sprint plans that primarily fill the gap between what the carrier believes they need, and the SI believes the selected software package can deliver. This process is sometimes referred to as 'as-is' compared to 'to be.' Also included in this plan are data migration and integration plans; these workstreams are critically important, and you must sequence them correctly, but functional requirements can often overshadow them. More on that later.

In the Agile approach, you execute sprints – usually, two to four weeks in length – that involve 'configuration' of the system, unit testing, integration testing (if possible), incremental user acceptance test (UAT) builds and migration work (usually managed semi-independently from the main development workstreams). These sprints tend to finish late from sprint to sprint (because it's tough to gauge how long something is going to take, especially early on). Also, an individual sprint is hard to put into the context of the end-to-end (E2E) process. Regardless, the team marches on until some end point when E2E UAT can begin. You can deploy the system once those sprints are complete.

Throughout the sprints, critical change management and DevOps deployment work are also taking place. The former to help the organization prepare for the upcoming change and the latter to ensure a defined, tested, and repeatable deployment occurs not only at 'go live,' but also thereafter.

THIS APPROACH IS PROBLEMATIC

There are very rational reasons these projects are done this way – primarily because of commercial constructs. As an SI, there can be pressured to agree to fixed-fee contracts; it's hard to agree to 'whatever' without an explicit scope. For the carrier, there's a significant (rational) fear of a pure Agile time-and-materials approach because 'whatever' can be equally problematic. Unfortunately, they tend to meet in the middle and perform the as-is/to-be gap analysis from which you create a relatively defined scope.

This Approach is Problematic for Four Critical reasons:

First, by nature, as humans we prefer to do things the way we've always done them. It's not personal – it's our nature to resist change. Therefore, the more often we (as a team) discuss 'as-is,' the more we entrenched it becomes for the carrier; it causes the carrier to stick to what they have and how they've always done things – even in situations where there are definite advantages to the new system. In the more difficult cases, humans fight against each 'to-be' function because of the bias of what was. But even when support for the new system is strong, there is still cognitive dissonance against the new system, which doesn't manifest until E2E UAT when the team assembles all the parts, and the reality of new system becomes evident.

The second problem is that the SI should be hired to bring in new perspectives and a natural bias towards the new system, which is often true. However, the as-is/to-be process very often subjects the SI to the carrier's bias and as such, pulls the SI into the past. Eventually, in the worst case, everyone is drawn back into what used to be versus what will be. This shift is incredibly subtle, showing up as change controls or significant functional requirements that are unintentionally aimed at making the new system behave more like the old one.

The third and probably the most prominent problem is that the actual (entire) system is virtually unknown until far too late in the development cycle – E2E UAT. It's one of the reasons why people often refer to the Agile method used in these projects as 'Agile Waterfall.' Again, as previously noted, there are commercial reasons why people do this. By the time the team gets to UAT and discovers all the big misses, significant pressures are working against them, namely time and money. It's uncommon (putting it mildly) for teams to be on track in terms of time and budget as they get to E2E UAT. Meaning, there's little tolerance for significant changes, and even if there's tolerance, big changes are difficult to incorporate into E2E UAT scripts that have been building for (many) months.

Remember the previous remark about how integration and migration are not attended to well enough? Well, that is the fourth problem. Migration and integration workstreams are heavily dependent on the incremental development of the system. In the case of migration, initially, the data is not entirely understood; it takes time and effort. In the case of integration – an integration point implies that whatever part of the process being fulfilled by the system exists (which may not be the case). It's far too often the case that these workstreams are flying blind because they make significant assumptions they have virtually no way of validating… until E2E UAT. To be blunt, migration and integration should be the primary development focus of these projects; it's where most enormous problems occur and as such, are the most significant risks to success.

FLIP THE SCRIPT

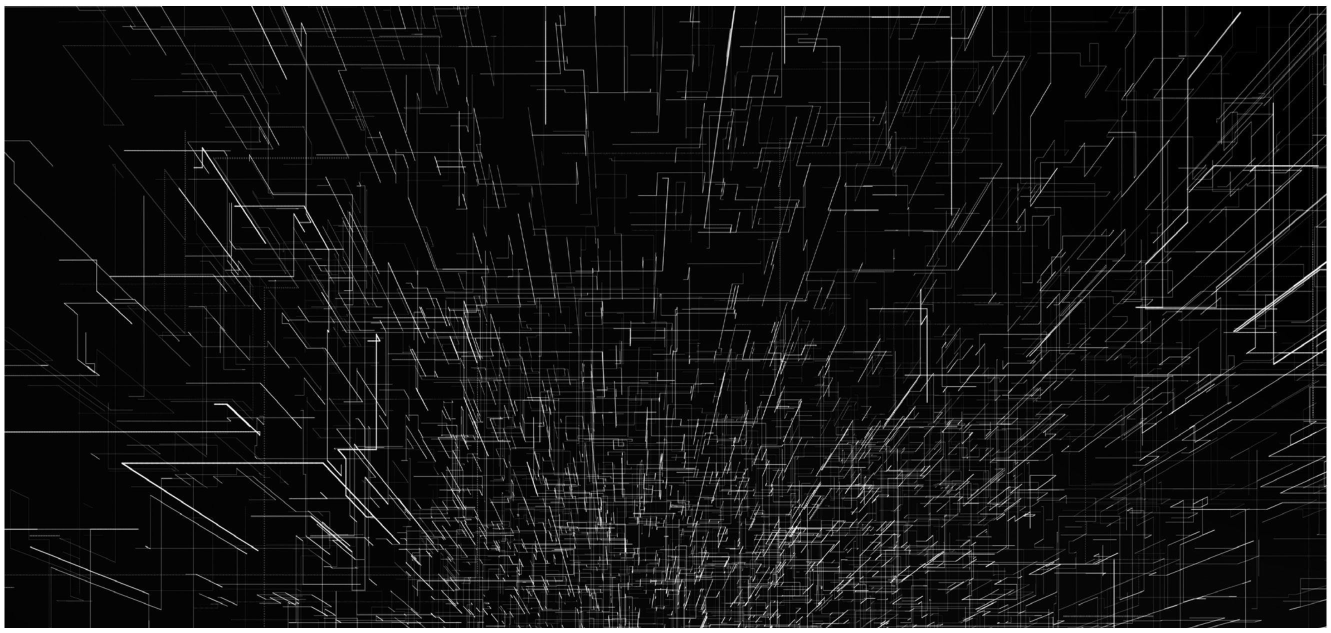
Looking at this problem in more 'scientific' terms, the central question currently asked is, "what are the gaps between your as-is and to-be systems?" This central question is almost the equivalent of the blind leading the blind because the SI doesn't know the 'as-is' and the carrier doesn't know to 'to-be.' By the time each understands the other, it's time for E2E UAT (OK, that's an overstatement). This question leads to more questions, such as "what is your as-is?" (And so forth, as noted previously.)

What if the central question was, "what's broken with the new system?" This question also gives rise to more questions, but they're a different set, including "how does the new system work?"

To answer this new central question, and to define 'flip the script' in this context, start the entire journey with E2E UAT. That's right – start with UAT. The assumption is that the new system meets the carrier's needs – truly 'out of the box.' The past is not relevant, and this approach makes that point very clear. It's the core system implementation equivalent of pulling off the Band-Aid quickly.

This approach saves significant time and money and lowers risk by putting everyone's focus on what will be (i.e., the new system), bypassing substantial time and energy on what used to be (i.e., the legacy system). One caveat: your line of business must exist in the vendor's system for this to work (which may be part of your selection criteria) because otherwise, there will be nothing out of the box to work with.

Before you think I'm (more) insane, Agile still matters with this approach. You must place significantly more focus on migration and integrations (because you have the time to do so). This work is built incrementally using Agile, but all the dependent pieces are in place (the true 'out of the box' system) on day one. There will most certainly be changes, but those should be resisted by default versus accepted by default. Again, the working assumption is that the new system is 'right' – it's up to the team to disprove this hypothesis. Changes are fed into Agile sprints, just as they are during 'traditional' UAT – but far sooner.



WHY THIS WORKS

STARTING WITH THE END IS SUPERIOR FOR MANY REASONS:

- It focuses everyone on the new system; for the carrier, they are learning it and focus on it instead of what they already have
- You spend no time on 'as-is' – this is a big time and money saver
- You dramatically reduce the need to configure the system – a great deal of time and money is spent configuring systems in ways that only serve to turn the new into the old
- The SI remains focused on the future without being brought into the carrier's past
- The entire E2E process is considered at the onset – this eliminates the "flying blind" scenario with incremental builds
- More (if not most) of the big problems are spotted during the sprints because UAT and the sprints are essentially the same thing

WHAT IT TAKES

Taking this approach is no trivial matter organizationally. Carriers are accustomed to being asked, "What do you want?" versus "What's wrong with this?" It will feel heavy-handed at first because the SI is prescribing the approach. But fundamentally, if there are so many things wrong with the new system that you must rebuild it for your needs, why use it? The likely truth is that the system you've chosen is a good one, and it is cognitive dissonance that gets in everyone's way. Again, this is nothing personal; it's not a function of experience or ability. It's a function of how the human brain works. You cannot overstate the value of organizational change management in this scenario; that group is a primary champion of the effort and as such, must fully accept the approach.

Support must start from the top. In particular, chief operating officers must support how the new system works operationally and how the SI is starting with E2E UAT. Without that deep support – because there will be many moments when staff want to revert to old practices without realizing that it's not even necessary with the new system – the team will be back to as-is and to-be (et al). Also, you need an open mindset. There will still be surprises; there will always be 'uh oh' moments. They will just come sooner. An open mindset creates an environment of trust where it's okay for things to go wrong because the team can fix them. It's okay to think differently because the team supports different – and it's okay to break all the rules sometimes if that's what the team needs.

AUTHOR

Frank Neugebauer, Partner, Head of US Insurance

ABOUT CAPCO

WORLDWIDE OFFICES

APAC	EUROPE	NORTH AMERICA	SOUTH AMERICA
Bangalore	Bratislava	Charlotte	São Paulo
Bangkok	Brussels	Chicago	
Hong Kong	Dusseldorf	Dallas	
Kuala Lumpur	Edinburgh	Houston	
Pune	Frankfurt	New York	
Singapore	Geneva	Orlando	
	London	Toronto	
	Paris	Tysons Corner	
	Vienna	Washington, DC	
	Warsaw		
	Zurich		

WWW.CAPCO.COM

CAPCO
THE FUTURE. NOW.