ITIL VS DEVOPS: WHAT'S THE DIFFERENCE?



With the rise of Agile and DevOps, what becomes of ITIL®? Gene Kim, the author of The Phoenix Project (along with Keven Behr & George Spafford), states:

"ITIL and ITSM still are best codifications of the business processes that underpin IT Operations, and actually describe many of the capabilities needed in order for IT Operations to support a DevOps-style work stream."

Download Now: ITIL 4 Best Practice e-Books

These all-new for 2020 ITIL e-books highlight important elements of ITIL 4 best practices. Quickly understand key changes and actionable concepts, written by ITIL 4 contributors.

Free Download ›



Free Download ›

But, with all due respect to Gene (who I consider a friend), I'm going to take the other side of the argument.

Giving Credit Where Due

Let's be clear: ITIL is important. Around two million people have been trained in it, and as the closest thing to an industry standard for IT management that currently exists, it has global reach. Lots of people *read* the ITIL volumes as guidance to their IT organizations. Throughout all its versions, ITIL has been framed as a complete approach to managing the IT function, with the specific exceptions of project methodology and systems architecture. Plus, it's worth noting that ITIL also informs the product directions of vendors selling IT management tools; in fact, they often market their IT service management tools as "supporting" the ITIL processes.

Also (despite some common misperceptions), ITIL is not *explicitly* opposed to Agile and DevOps. The Service Design volume supports iterative and incremental design, and mentions Agile and XP. And Service Strategy, with its strong grounding in current management theory, mentions the need for continual feedback between the ITIL service lifecycle stages (Strategy, Design, Transition, Operations, Improvement). And finally, I agree with my friend Jeff Sussna, who says that ITIL has made an indelible contribution in promoting service-centric, outside-in, customer-focused thinking as an alternative to overly technical and piecemeal approaches to supporting the customer.

Now, the Complaints

So, what are some of the problems with ITIL from an Agile and DevOps perspective? Here are some of my questions and observations:

What is ITIL really saying? Re-reading as sympathetically as I can, I find that the overall ITIL narrative is still sequential, plan-centric, and deterministic.

- We should first analytically determine a service strategy and charter the service.
- Then create a detailed Service Design Package, and organize resources into development projects that are designed, executed and transitioned in a high-ceremony fashion to operations (as a discrete function).
- And when that's done, continual service improvement can start.
- We ensure that everything is documented, and we assume that organizational resources are available and capable of generating and consuming this documentation.

Even with sporadic mentions of Agile and feedback, the deep, repeated narrative is one of planning, control, and documentation.

How is ITIL saying it? As a data modeler and liberal arts student, word choices are important to me. They indicate much about overall worldview. ITIL may supportively mention a concept, such as iterative design, but that doesn't mean it is primary in the overall ITIL narrative. As a simple test , if you compare various key Agile terms (feedback, iterative, incremental, collaboration, experiment) to terms more often associated with non-Agile approaches (plan, process, procedure, document), you'll see skewed proportions. For every mention of "iterative" or "feedback," there are ten mentions of "plan" or "planning." Notably, the word "experiment" appears only a few times in Service Strategy and not at all across the remaining volumes. This surprising omission raises legitimate questions about ITIL's relevance for Lean Startup philosophy (which one is likely to find influencing large enterprises as well as true start-ups). Somewhere between the service charter and the service design package is a need to build a minimum viable product and test the market hypothesis.

What are the organizational assumptions? The assumption is that functional silos will continue to exist, with specialized ticketing processes ensuring delivery and alignment. The idea of loosely coupled, cross functional, two-pizza product teams using lightweight, unified workflow (e.g. Kanban) is not well supported in ITIL, as far as I can see. For example, see the videos on Spotify Engineering Culture, part 1 and part 2. "Process" is barely mentioned.

What about this curious problem of configuration management? Configuration is important. ITIL's insistence on its centrality is a good thing, but the actual industry practices do not align well with how ITIL views the problem. Source control and package management, coupled with massive infrastructure automation, are increasingly how work gets done today. Perhaps you could call all that a "CMS," but that's not the terminology most practitioners are using. And then what of the structured CMDB that traditionally underpins the service management processes? What is presented as "best practice" and an industry consensus in ITIL is actually a complex, fast-evolving, and very unsettled problem. Even my friend Gene seemingly agrees that CMDB and ITIL are problematic with respect to Cloud.

Some Anecdotes from Our Field

ITIL is under scrutiny in IT organizations around the world. Representatives from ING Bank notoriously stated, "we kicked out most of the process managers…" (However, later in the same presentation they say that they still recognize the need for processes and, in private correspondence with me, have reinforced that the core ITIL processes are still key for them.) They also discourage the use of an ITIL Problem process if the Problem can be understood as a user story. Author Thomas Limoncelli, in *The Practice of Cloud Systems Administration*, advises using common ticketing across development and operations, while Scaled Agile Framework creator Dean Leffingwell advocates in his *Agile Software Requirements* "prevent the team from getting instructions from a variety of sources."

Also, I recently talked to a senior IT executive at a Fortune 50 corporation. He complained that the end result of significant investments in ITIL and PMBOK is 30+ "processes" that, in the aggregate, amount to a highly ineffective operating model.

The Central Failure

These anecdotes point out the most fundamental problems with ITIL: its failure to address the critical importance of managing work in process, batch sizes, fast feedback, multitasking, and queues (e.g. processes), resulting in gridlock. Lean product management specialist Don Reinertsen (whom I call the "man behind the curtain" of Agile, due to his influence on multiple Agile thought leaders) has made a compelling case for the **centrality** of these concerns in product management. And creating and delivering IT services is merely a specialized form of product management.

While ITIL briefly discusses queuing theory, I'd point out that there is a difference between a mention and the overall message. The overall ITIL takeaway is that:

- The IT pipeline consists of large batches of accurately planned work transitioned between strategy, development, and operations;
- Process solves your problems. ITIL enumerates 26, plus the queues implicit in formalized "services" in the catalog;
- Multi-tasking and context switching are not concerns with respect to enterprise IT processes;
- Risk is mitigated through planning and documentation.

ITIL is clear that one does not implement the full set of 26 ITIL processes just because they are in ITIL. But ITIL's guidance on exactly HOW one might decide which processes to implement is, again, very sequential; as discussed in the later parts of Service Strategy, one carefully analyzes, plans, and implements. There is little if any support, as far as I can see, for the kind of iterative experimentation on the operating model that (for example) the Lean Kanban method calls for.

What Comes Next?

We are all on a journey of discovery. I have great respect for the ITIL authors, who are human beings like the rest of us, and I know that a number of them are reading and keeping current with Agile theory. There is no question that the well-known core operational ITIL processes (Incident, Change, etc.) have added value, not diminished it, in many IT organizations.

As I noted in my last blog, IT organizations are complex, socio-technical systems. Radical interventions, such as "get rid of all processes," would be as unwise as going back to the bad old days of last-minute, large-scale release integrations. I also wish the just-announced ITIL Practitioner Architects team (including Phoenix Project author Kevin Behr) all the best.

But the bulk of ITIL was written 10 years ago; I know that the ideas I have this year are far removed from what I thought back then. Simply put, ***ITIL as a published artifact is becoming outdated***. New theories and understandings are emerging, led by people like Reinertsen and Mark Kennaley (*SDLC 3.0* author, who just released his new book *Value Stream*). In Agile terms, ITIL has been released as a series of very large batches across the decades, with no provision for feedback or ongoing refactoring. Compare this to the ongoing evolution of the Agile Alliance's Guide to Agile Practices.

So where does this leave us? I don't know. Perhaps Agile overlays and interpretations of ITIL can be developed, but I think this will be difficult. Retrofitting these ideas onto the existing ITIL body of knowledge is, to my mind, an impossible task. There are questions of basic mental models, culture, and the financial interests of a large, entrenched training ecosystem.

To fulfill its huge scope and ambition, ITIL **requires an improved foundational model** of IT delivery as a socio-technical system that is centrally concerned with **execution, feedback, and flow**. All process recommendations need to be founded on, and tempered by, such a core model. And yet a complete ITIL rewrite also seems unlikely, based on what I am hearing through industry channels.

I think the important thing we can do is to continue an open dialogue in the industry on our frameworks of understanding, and how they need to evolve to better support the challenges of our complex IT universe. I look forward to further discussions and correspondence with all those who are concerned and passionate about this topic.

Disclaimer: Charles Betz participates in the Open Group IT4IT Forum, which is developing an open, architectural standard for the business of IT, and where he currently leads the Agile Workstream. The opinions expressed here are his own.