DITY™ Weekly Newsletter
itSM Solutions®
IT Experience. Practical Solutions
The workable, practical guide to Do IT Yourself Vol. 4.23 • June 3, 2008

SDLC or ITIL? Wrong question.

By David Nichols

Ever-increasing business and technological complexity are driving successful IT organizations to search for methodologies to ensure that new or changed IT services meet the requirements of the business customers, and create value for the business by being designed, delivered and operated in an efficient and effective manner. Application groups point to the use of a Systems Development Lifecycle (SDLC) as their solution. Some of the technical functional groups have jumped on the ITIL® bandwagon. So which should it be; SDLC or ITIL?

Once upon a time, a programmer and a user would sit down and talk about what the user needed in a new program or wanted changed in an existing one. The programmer would go away and reappear sometime later with a new or changed program. As life, business and technology became more complicated, structured methodologies evolved to support the lifecycle of a software application or system. Commonly referred to as the Systems (or Software) Development Lifecycle, these methodologies have dominated how new software systems were built and maintained for over thirty years.

In parallel, the IT Infrastructure Library has struggled over this same time to rationalize the Systems Development and IT Service Lifecycles. Recently introduced to ITIL, many IT organizations who currently use a systems development lifecycle methodology have asked, "Should we use our SDLC or ITIL?" Although that may be an interesting question, it is the wrong question to ask.

SLDC - Some of the early adopters of systems development lifecycle methodologies often commented (with tongues firmly planted in their cheeks) that it seemed that they knew a phase of the software lifecycle was complete when the paperwork out weighed the project team. Most of the early SDLCs followed the waterfall method of sequential phases, where each phase had to be completed before the next could be started. This was okay when IT was automating clerks and financial folks.

As new systems started supporting information workers, the waterfall method gave way to subsequent generations of SDLC methods, most of which embedded some sort of iterative method such as the fountain, or spiral models. The spiral model (a series of waterfall cycles) gave way to rapid prototyping (a.k.a. Rapid Application Development or RAD) which prototypes the application, and builds the real application as soon as the prototype is approved.

An incremental model came along later, and built and tested separate sections or “chunks” of a system with close involvement of the user. Now we have the synchronize and stabilize method, which is fundamentally the spiral method enabled by oversight and code management tools. This method develops software components separately, assembled them into the “whole” system, and tests and tweaks it until it is stable before doing another round of development. This is also known as the F.W. I. T. W. method (Fiddle With It ‘Til it Works).

Although all of these methods are interesting and represent adaptation by the development folks to changing business conditions, they all share a similar approach; they ask and answer the following;

- Project Planning & Feasibility – how big is big and can it be done?
- Systems Analysis & Requirements Definition – what do we have and what do they want?
- Systems Design – how do we give them what they want (& need)?
- Implementation – how do we build it?
- Acceptance, Installation & Deployment – does it meet their needs, can we install and operate it?
- Maintenance – what is still needed or what has changed and is needed now?

IT Service Lifecycle – From the very beginning, the IT Information Library it has supported the idea of the delivery of the IT infrastructure as a set of IT Services as opposed to the individual hardware and software components that make up those services. This was refined in 2001 with the version 2.0 refresh of the library and expanded in the latest version 3.0. Version 3.0 (v3) extends and refines the concept of IT Services into a five-phase IT Service Lifecycle, expanding the core processes from ten to over twenty.

The IT Service Lifecycle phases ask and answer;

- Service Strategy – what new or changed services do we need and why?
- Service Design – will this design meet the requirements of the customer?
- Service Transition – how can we build this so that it achieves the requirements of the customer and can be operated and improved?
- Service Operation – how do we operate and support the new or changed service?
- Continual Service Improvement – what can we do to improve the service or the supporting processes?

With the v2 refresh, the library introduced Application Management to IT Service Management. It provided very little in the way of useful guidance and actually required interpolation of its guidance with the Information Communication Technology volume to make any sense on how to go about integrating software into IT Services and, in particular, to wrapper IT Services in the context of a larger lifecycle model. Version 3.0 addressed the issue by including the technical functional issues of application management in the Service Operation phase and the remainder of the "nuts & bolts" issues within the context of the service in which the software application participates.

Okay, so what is the problem?

Development & Infrastructure Dynamics – Most IT application development groups who use an SDLC often find themselves in conflict with the "infrastructure groups" within their own organization. In organizations with relatively immature design and support processes, the infrastructure folks do not respond or participate in "filling out the forms" used by the SDLC methodology to document and ensure consistency of phase deliverables.

This leads to the software folks thinking that the infrastructure folks are a bunch of "hip-shooting cowboys," and the infrastructure folks to think the application folks are "anally retentive, socially challenged geeks."

If the IT organization is more mature in its processes and has either adopted or attempted to adopt some of the support and transition-related IT Service Lifecycle processes, the conflict is more substantive and centers on the perceived relevance of the SDLC activities vis-à-vis those of the operation and transition processes and the technical functional areas outside of the application groups. This environment supports discourse, a prerequisite for healthy and productive disagreements among the technical functional groups (technical and application groups), process owners and the management team.

The Question Is? – The right question is not which method or process framework to use, but rather how to integrate SDLC-enabled activities in support of systems development in the larger context of the IT Service Lifecycle. This is similar in nature to integrating a governance framework such as CobiT, program/project methodology such as Prince2 or PMI, quality methods such as Six Sigma, Lean, Lean Six Sigma, Baldridge, and Deming into the IT Service Lifecycle.

Each method or framework addresses part, but only part, of what it takes to manage the lifecycle of an IT Service. The question of which one to use is specific to the individual circumstances of the implementing organization. The approach should be to understand the strengths and weaknesses of each, understand the full requirements of the IT Service Lifecycle and organizational maturity, and look for integration points, overlap and gaps.

Summary – Any article about ITIL that does not include the word "synergistic" cannot be considered buzzword compliant. However, in this case, the word fits, so we will use it. The ITIL framework, when integrated with other frameworks, methods and standards, is truly capable of enabling a level of organizational performance greater than the sum of its individual components.

So the right question is not one of "either-or," but one of integration.