

COLLECTIVE WISDOM

CHAPTER 2 EXCERPT

Four Models for Knowledge Management

It is useful to think about four different models of knowledge management: content specialist, product specialist, batch, and KCS. Many organizations choose to use a combination of models for different kinds of knowledge base solutions. Let's start by describing the models before we make recommendations on when and how they should be used.

- **Content specialist model.** In this model, there is a team dedicated to managing the knowledge base, including creating all content. The content team is separate from the support delivery team and is composed of individuals who are specifically skilled with creating solutions. Usually it's made up of technical writers.

Because the content team focuses exclusively on the knowledge base, it avoids the priority conflicts that often occur in the other models. It should also produce nicely-written solutions. On the other hand, the strict separation can become a big problem when the content specialists become detached from the concerns and hands-on knowledge of the support delivery team. It can also create a situation in which the support delivery staffers stop feeling any ownership in the knowledge base to the extent that they won't even make suggestions for improvement. It's also very expensive. We strongly believe that all support staffers should be able to suggest topics for solutions, even in a content specialist model.

- **Product specialist model.** In this model, unlike in the content specialist model, the support delivery team is involved in creating new solutions, or at least the product specialists are (tier 2 and above support staffers).

Having the senior support staffers write solutions pretty much guarantees that they will be technically accurate, and that they will match real customer questions, at least the most complex ones. The problem is that the solutions may never get written as the product specialists rush to the next customer emergency. And many don't like writing, or write poorly.

- **Batch model.** In the batch model, everyone in the support delivery team writes solutions, so it's very similar to the product specialist model with the difference that it's not just the senior staffers who are expected to contribute.

The difference between the batch model and the KCS model described below is that, in the batch model, support staffers may not complete writing solutions while resolving customer issues, but rather will make a note of an interesting topic, or perhaps create a draft solution or a stub solution to be completed later, during their project time. This is why we call it a batch model.

The pros and cons of the batch model are similar to the product specialist model. Solutions should be steeped in real customer situations, but they may

not get written at all, or poorly. Because there are more hands to do the work, the output could be better with the batch model, although since the product specialists usually do the reviews there can be serious bottlenecks anyway.

The batch model may not work at all for organizations that use outsourcers or channel partners to deliver support.

- **Knowledge-Centered Support.** Alone in the four models, KCS, as Knowledge-Centered Support is known, does not view knowledge management as a process separate from case resolution. The basic KCS belief is that knowledge management and case resolution are completely intermingled, and that the intermingling itself fosters quality and efficiency in the support center. KCS therefore dictates that each support delivery staffer create solutions (and use or improve solutions) during the case resolution process.

The major advantage of KCS is that knowledge is very literally created and maintained “just in time,” without the delays often associated with the other models. The main obstacle to KCS is that it requires a major culture change in the support organization.

Because KCS is an important development in support with interesting theoretical underpinning, because it’s the knowledge management model adopted by HDI, and because we love it, we devote a separate callout to it.

Knowledge-Centered Support (KCS)¹

Knowledge-Centered Support or KCS is a support philosophy and a set of practices that stem from the premise that support should not focus on cases but should focus on knowledge instead. Customer questions can either reuse an existing piece of knowledge or give rise to a new knowledge object that may be useful to other customers in the future.

The other critical idea in KCS is that knowledge creation happens during the resolution of a case. There is no batch work. The benefit of integrating knowledge creation with case resolution is that the information is unfiltered, better reflecting the real customer experience and there is no delay in making interesting information available.

¹ See www.serviceinnovation.org for more information.

If a cluster of similar cases occurs, as is often the case in support, whoever resolves the first one provides the solution (or part of the solution) to other staffers. The idea is that the support staffers will use an existing solution, fix (revise) an existing solution, or create a new one as they resolve a case.

Many organizations hesitate to use KCS because they believe that it will slow down case resolution, which they feel is unacceptable for customer satisfaction. This objection does not hold up very well if you consider that solutions are not lengthy, constructed documents that have little connection with an individual case. On the contrary, knowledge base solutions are very similar to what a support staffer would provide to a customer as part of the normal resolution of a case. There should be very little overhead in creating a solution, making it possible to create one as a step in the resolution of a case, with all the details freshly available.

We have seen our share of poor KCS implementations which can give KCS a bad name. It's too bad since KCS is so obviously a good idea.

A main misconception in KCS is that each case must be linked to a solution, either an existing one or a new one. This is silly! Support organizations that blindly adhere to the "link every case" belief quickly create "junk" solutions that are used as default links for hundreds of cases, fulfilling the (crazed and misguided) dictat that each and every case must be linked to a knowledge base solution. The junk solutions may be empty or mere placeholders. Or, they may be bona fide solutions. For instance, many cases are linked to the product release schedule, whether or not the main issue of the case was about the release schedule. The solution to junk solutions and junk links is to remove unhealthy incentives to link each case, as we will see later in this chapter, and to actively audit links (directly or aided by reports), requiring the support staffer to create a better link if a link is inappropriate.

Another common problem with KCS is the creation of multitudes of stub solutions, solutions that are created quickly upon closing cases, again to respond to the imperative of linking each case to a solution. The stub solutions are extremely sketchy and may never move beyond the draft status, so are useless to the support center as a whole. The solution to stub solutions is again a vigilant review of new solutions, combined with requirements for fully fleshed out solutions. In other words: there must be discipline around the program.

If you're interested in KCS, by all means use a reasonable implementation strategy. It makes no sense to mandate that each case be linked to a solution. Consider the common situation of a customer contacting support because he didn't understand how to use some feature that's fairly well documented in the help system. What kind of knowledge base solution should be linked to these kinds of cases? It would seem perfectly adequate to tag such a case as "user instruction in documented feature," or something similar, and avoid the need to link any solution to the case. Of course, if the caller's question reveals a shortcoming in the documentation, the best path may be to report a documentation bug and request a fix.

Beyond moving away from the mindless “one case-one link” idea, the critical success factor for KCS is driving the transformation through the organization, from job descriptions to metrics to performance assessment. Are support staffers sufficiently trained to recognize good solution candidates while resolving cases? Do managers have the time and the energy to audit solutions and links to ensure that there is no junk? Is the knowledge management tool set up to minimize the overhead of creating solutions? There is a real cost associated with creating solutions, whether in the KCS model or not, and that cost should be balanced by the ability to abbreviate lengthy troubleshooting sessions and to provide answers to customers without their having to contact support.

KCS is enjoying a small but growing ardent following, especially in high-complexity support centers, and is worth exploring.

Choosing a Knowledge Management Model

What model is best? We believe there is no best model for all support organizations. Further, we believe that many, if not most, organizations should combine models to match the various types of solutions required for their needs. Don't be afraid to mix and match.

Two factors seem to be important when deciding on a model: audience size and issue variety. A wide audience and many repeated issues suggest a content specialist approach, while a small audience and a wide variety of issues suggest a KCS approach. Here are a few examples.

If you support **low-complexity products** and the vast majority of questions are repeats from earlier questions, you may have to manage only a handful of new issues in any given month (past the initial build of the knowledge base, that is). In this situation, simply mining the few cases not linked to knowledge base solutions would be sufficient. Furthermore, the technical expertise required to create solutions is low. In this situation, using the content specialist model is best.

On the other hand, if you support **high-complexity products** and most questions are new, or seem new at first blush, your task is more complex. Many such centers use a KCS approach, but you should adopt KCS only if you are willing to make the commitment and manage the discipline to do it right. Support staffers must have enough time to capture knowledge along with resolving customer cases, and there must be proper control to guard against junk links and junk solutions.

In the same situation you could also adopt the batch model, asking support staffers to create solutions when they see fit as they stumble on new issues. The batch model also requires discipline so solutions are created when they are needed and
