



Tending your CMS Garden

Jim Howard, CEO of CrownPeak explains the importance of life after launch with your content management system.

Non-technical contributors use a content management system to add and modify Web content. However, an enterprise also needs to add to and modify the system itself to enable changes to the Website that weren't anticipated when the CMS was launched. For example, modifying the homepage layout or adding a new type of landing page often requires adjustments to CMS templates. Adding or subtracting a workflow step can require a specialist to adjust configurations or possibly modify code.

In short, you've built a nice flowering garden, but right away it needs to be tended. Whatever services (i.e., peoples' time) you needed to plant that garden, you will continue to need them going forward.

Gardening is not always easy. Supporting a Web content management project following a CMS launch takes much more than just patches and tech support. You should anticipate potentially significant post-implementation tweaks, improvements, and enhancements. These services need the same careful attention to management and budgeting that you exerted to implement the system in the first place. And it follows that the ease and cost of ongoing system changes will have a major impact on your long term success. In fact, most of the cost of IT projects is associated with post-launch efforts. Planting the garden is just the beginning.

Managing the Content Management System

You may launch your CMS to precisely meet written requirements, but almost by definition the system won't work as it should. When logging in for the first time and beginning to make content updates, contributors often find that they don't like elements of the interface, the workflow process they signed off on earlier now seems clunky, or the system requires three clicks to do the most common task (when one is possible, if the system were configured differently). This is the first place where a post-launch program is essential: making tweaks once editors have started using the system. [You don't have much of a window here](#) – if the system is crummy, people won't use it.

Next comes enhancements to the system: new page types, new site sections, new languages, new departments and contributors, new landing pages, and so on. If you are working with a flexible CMS – e.g., one that enables modification of the input screens and output templates – contributors can become familiar with the very healthy exercise of asking for new ways to improve the system.

Those changes should be quick and easy. Of course, "easy" does not always mean "good." Therefore, overseeing system modifications still requires the same sorts of management acumen that you applied to launch the CMS in the first place.

Many modern CMS systems have configuration screens to enable quick changes to various services, such as workflow, permissions, and templates. These changes can frequently be simple (like adjusting a chunk of HTML) or complex (like changing business logic for link placement or adjusting navigation).

► Tending your CMS Garden

A power user can often perform basic changes, while it may take a developer for more complex adjustments. The more platform-like the CMS package, the fewer of these configuration screens will exist, and the more readily you need recourse to a developer. But the truth is, business users often do not have the patience or time to learn how to master relatively infrequent configuration steps. So while not completely eliminating the IT bottleneck, you may be able to at least speed up the modification process. With an email to a system expert a workflow change can be accomplished in a few minutes rather than a few weeks.

Similarly, integration with other applications, data streams, or Web “widgets” has become an increasingly common requirement. Adjusting a template to accommodate a new chunk of functionality that comes from a separate data source is something that should be easy. Depending on how remote data is combined with CMS-generated content, a developer can often make this happen quickly, but unless the power user is a strong HTML developer, this task could prove difficult.

And then there is the classic “user error.” This is perhaps the most important support issue of them all. You want content contributors to like the system, to use it, and to give suggestions for extensions and improvements. When a system user pastes in some funky HTML-JavaScript-ColdFusion code combination and the published page gets wrecked or the preview mode chokes, is that a CMS problem, a contributor problem, or an implementation problem? It could be a mixture of all three. The key is responding to the issue and either fixing it or teaching the business manager how to avoid the issue in the future.

Otherwise, the system truly is broken. Any way you look at it, you need resources – a services regimen – to solve the problem.

Monitoring the Content Management System

In addition to managing the application and the environment, you must deal with the often underestimated task of application monitoring. Is the application working? Is it working quickly? Is it generating errors? Is it publishing? Is it publishing correctly? These can be quite difficult questions to answer.

The simplest questions often revolve around basic performance. Is the CMS working and is it working quickly enough? That can be fairly easy to monitor. But somebody needs to receive notification that the application isn’t working and do something about it. Diagnosing and fixing a problem is much more difficult, of course, than determining that there may be a problem.

The next question is more subjective: is the CMS working well? This question becomes very interesting when talking about a complex system like a publishing tool. Errors in publishing systems can occur within the contributor interface or at the publishing phase. Publishing can be especially tricky. For an automated system to tell if a page layout is mangled is a problem many developers have spent a great deal of time thinking about.

Error reporting seems like the natural answer – when error reporting exists within the application or can be provided by a third-party monitoring tool. Few commercial and almost no open source tools tackle this problem well. Again, the question quickly becomes, “who is going to care and do something about it if there is a problem”?

▶ Tending your CMS Garden

In any case, you'll need to decide:

- ▶ What's going to be monitored at first
- ▶ How to add monitoring systems to watch things that seem to be cropping up over time, and
- ▶ How you'll maintain solid response plans

For simple projects, you may be able to wait for complaints about the Web site and go stamp out the issues when they occur. For more high-profile efforts, though, you'd like to know when there are problems before your CEO's mother tells her about the broken page on your site.

Retaining your CMS

Internal IT departments are experts on keeping hardware and networks running. They usually excel at developing teams and processes to manage critical applications. When the commitment is there, the internal IT group can be the ideal partner for the business users of a CMS. The key here is to establish a *program* to manage the application and support system contributors, ideally led by the same IT group that directed the initial implementation, with at least part of that team carrying over into the post-launch support program. It's always hard to step in and try to manage somebody else's code.

When a third-party developer implements the CMS, it can be great because they are likely to be experts in the application, with dozens of projects under their belts. On the other hand, it's critical to have a transition plan to the post-launch support program. When experts swoop in and then hurry away, even a well-implemented application can fail for lack of support. It's possible to find third-party organizations that are willing to monitor and manage applications on retainer, or to transition to an internal team. Either way, it's critical to success.

A software as a service (SaaS) approach is also an option. SaaS vendors typically build post-launch services into their process, so the experts don't necessarily go away but more-or-less sit around waiting for the phone to ring with questions or to make changes. Such a service can be up to the level of a well-designed internal program when the application requirements mesh with the capabilities of the SaaS. On the other hand, if the CMS itself needs real-time integration with internal systems or when the CMS has to reside behind the corporate firewall for any reason, a SaaS approach loses its appeal.

The Case for Gardening

From a cost perspective, having no management program seems to be the cheapest. In IT, as in gardens, there are vendors who will promote "no-maintenance" installations. Trust me, untended types of gardens will become prohibitively overgrown and weedy. In reality, doing nothing is very expensive because the CMS application could quickly become shelfware.

▶ Tending your CMS Garden

Having an internal IT group establish a management program and commit resources to that program won't be cheap, but it's essential when the application is mission critical. Of course, when not considered mission critical, CMS projects are in danger of being poorly supported in these days of increasingly stretched IT resources.

A hosting partner can work to keep the application running, but supporting, managing, monitoring, upgrading, and modifying the application over time needs to be added on top of the other services. That can be expensive, especially if it's a one-off project for the hosting partner. Be sure to examine their process and their staff for expertise level and guaranteed availability.

In all cases, a smart enterprise will plan to tend – and not just plant – its CMS garden.