



TECHNOLOGY NETWORK

# Designing A Successful Portal Deployment

by Quinton Wall  
01/05/2006

## Abstract

Portals often fall into two major categories: those that consolidate services or applications into a single point of access, and those that are focused on building a community by providing communication tools (calendars, message boards, and so on). Very often these two categories are present in each portal implementation.

Going live on a portal project does not mean the completion of a project. This is especially true in the case of a community-driven site. This article intends to identify a number of best practices before and after go-live that may assist in the success of your project.

## Define the Portal's Purpose

Decide early on what you want to get out of the portal. Do you want to build a community or a one-stop shop for your employees? Ask yourself why you want someone to register. Is it for advertising, mailing, contacts, or community feedback? The answer to this question should drive how your application is implemented and promoted. Many organizations have a good idea of why they want to develop a portal solution. Such *wants* usually focus on sharing information and centralizing their resources. Unfortunately, it seems that organizations often don't recognize why they *need* a portal.

Portals, especially internal intranets, give people—your greatest resource—a way to communicate and be exposed to the artifacts of an organization. Portals provide a great way to centralize all this information. Modern portal technology promotes communication and sharing. Within an organization, networking, a term that is often overused, is fundamental to individual success.

In today's business environment we need to take advantage of virtual networks. Portals, whether they are an internal intranet, a partner site, or a public, community-based site, are a key factor in this success. It's all about communication and networking. Most important, it's about how to effectively manage these requirements in real time by the people who know the business the best—the business users themselves.

A successful portal implementation defines what network it wants to build and what message it wants to communicate.

## Community Building

In a portal project targeted toward building a community where individuals congregate and share stories and ideas, the ideal is to have the members influence the direction of the site and give it a life of its own. This cannot be achieved prior to launch, and even then the portal team must make careful decisions about how to grow this budding community. This section explores a number of ways to help you achieve this

ideal.

## **Avoid stale content**

Whether it is an external community-driven site or an internal intranet system, a portal is only as good as its content, and this content must be kept up-to-date. This requirement may sound obvious, but often grand plans of setting aside time to write new content each day or week fall by the wayside when a pressing project demands your attention. Sometimes goals are set that, when looked at analytically, were not realistic to begin with.

As a real-world example, let's look at a project I recently finished for a major academic institution. The purpose of the portal was to develop relationships with prospective students by engaging them in everything the campus had to offer and providing the option of applying online.

Shortly after launch it became apparent that the portal content was getting stale and something needed to be done to refresh the information. After quickly doing the math for the required number of new articles (the goal was eight) per section (six in total) per content editor (two at that time) per time period (week), it was obvious that I needed to take a different approach.

The example above shows that the best intentions may not be feasible as the system grows. At the time of writing, the business owners at the institution had decided to employ students as content editors. This step directly addresses the issue of human scalability encountered previously. Further, with this approach, the portal could be filled with content from many different perspectives (staff and students).

If we wanted to take this scenario even further, the ideal solution for community growth would be to leverage the prospective applicants themselves as a means to create a true community with multiple perspectives on the enrollment process.

Any of these solutions requires a number of key technical components including delegated administration and workflow functionality. Both components provide community involvement while still maintaining a central point of control for the business (see Portal Must Be Able to Be Administered by Business, Not IT).

## **Provide value for guest users**

A community portal should provide value to users who do not wish to register. I try to avoid the term "unregistered user" or "anonymous user" because these words imply that individuals are not important unless they register and, even worse, that you know nothing about them. Eventually, guests will register if they see there is value in doing so.

Businesses should recognize that there is much value in a person who has yet to register, or who is visiting. Just because they haven't registered doesn't mean these users can't provide valuable statistics. How do you determine the right level of what you give away "for free" versus what you offer to registered users? This is a delicate balancing act with no silver bullet, but a best practice may be to focus on the central purpose of the portal (see Define the Portal's Purpose). If, for example, the driving purpose of your portal is to sell your merchandise, then you may not want to limit this to registered users (doing so would greatly limit your prospective buyers), but rather offer them special savings or express shipping.

Guests who visit your site are often interested in what you have to offer but haven't yet decided whether it is worth subscribing to or joining the community.

Through investigation of HTTP requests, your organization may be able to determine information such as locale and browser agent. Cookies may also be used to determine whether an anonymous user is a regular guest. If so, you can offer *implicit personalization*, which means your business can start directing more targeted campaigns to these people. Through this type of personalization, guests may decide that they do want to register and become customers. It is important to recognize that many individuals consider the use of cookies an invasion of privacy. A socially conscious site should inform guests of the use of cookies and provide instructions for disabling this feature through their browsers if they so desire. Many organizations provide information about cookies and other topical information in a privacy policy publicly accessible on the Web site.

In addition to providing guest users with a more customized experience, BEA WebLogic Portal provides the ability to track behavior for all users, including guests or anonymous users. Such information is invaluable for businesses that are trying to determine traffic trends on the site. For example, if your business discovers through analysis that only one in ten anonymous or guest users eventually registers due to a perceived trouble with navigating to the registration page, you may decide to modify the site navigation to direct traffic more effectively. For more information, check out the online documentation for Anonymous Users.

### **Provide extra value for registered users**

Besides discussing anonymous and guest users, you need to consider your registered users and ask why someone should register. Registered users should be treated well. They have decided that your portal (therefore your business) is important to them. Make sure you show them that they are important to you by providing premium content, free downloads, discounts, frequent visitor incentives, the ability to comment online and interact with other registered users, and profile and portal personalization. Offer anything that makes them feel part of a community that you want them to be in and remain a part of. Businesses often put all their effort into using the portal to attract new members, forgetting that once these people are registered they become a valuable asset.

### **Registration must be easy!**

As identified in Provide extra value for registered users, registered users should be treated well. It sounds obvious, but many sites fail at this requirement. There is nothing worse than deciding that you finally want to register for a site, only to spend five minutes trying to find where you go to register and then another ten minutes filling out questionnaires that cover everything from the size of your company to what you had for breakfast. Your portal should provide a fast track to user registration and at the end of registering give users the opportunity to decide whether they want to participate in a survey or add additional information. *Don't* fall into the "check-by-default" scenario where you have users "opt-in" by default to subscriptions, mailings, and so on. Finally, always provide the ability for registered users to come back at a later date and update their registration. I often see portal applications that allow profile updates but never allow users to complete at a later date the surveys or questionnaires that were pressed upon them at initial registration. Always give users the chance to come back later and complete these forms if they feel it is important. Forcing reams of questions on users as they register may simply discourage them from continuing the registration process. Do it only if absolutely necessary.

## Learn to sow the seeds

When it comes to building communities, the best thing you can do is get out of the way. Plant the initial seeds—say a few news articles—and then allow the community to comment on these. You will be surprised by how quickly people will respond with their opinions, and these discussions will lead to further comments. A former colleague of mine used to call this sort of information dissemination "viral marketing." It is viral in the sense that feedback often encourages more feedback, infecting other readers until the initial conversation has mutated into something else of greater value (and obviously of greater interest to the community).

## Human scalability

The concerns of human scalability were briefly identified in [Avoid stale content](#), where the portal should not be restricted in its growth and adoption by technology or business processes. Such human scalability issues may manifest themselves in a variety of scenarios such as publishing too much content from too few resources or having a single point of approval (an issue if the only content approver is on vacation or sick).

Another important aspect to human scalability is the physical process of making a change to the site. These changes may be portal-related (see [Portal Must Be Able to Be Administered by Business, Not IT](#)) or more subtle. One such example is how the content editors produce and upload content. Portals often rely on a content management system (CMS). The problem with almost all existing CMSs is that the user interface for adding new content is minimal and not conducive to enhanced productivity. Business users often prefer to use specialized tools such as Dreamweaver to create their content. Any portal solution should not restrict the use of tools, and time should be taken to develop templates with appropriate metatags that facilitate their use. These tools often also include FTP or source-control integration that enables users to "submit" content to the system as needed. Many CMS vendors provide command-line utilities to load content en masse. These utilities may be utilized in conjunction with the content creation tools mentioned above to load content into the specific repository on a regular schedule (through cron or similar timing services). BEA, for example, provides the BulkLoader utility that can load content from the file system and insert it into the correct node based on supporting property files and meta information.

With the rise in popularity of weblogs and the power of RSS feeds, human scalability issues may also be approached in a more pervasive manner. WebLogic Portal provides an OOTB RSS portlet sample that, with a little bit of work, can be modified to be able to feed from multiple sources. This approach opens up endless possibilities for who can author content, with the portal acting as the centralized conduit to present information in a consistent and personalized format. For instance, the portal may determine that the current registered user is interested in tennis and not football, and, as a result of implicit personalization, you display the RSS portlet that feeds from various tennis-related sites and hide the one that feeds from football-related ones. Blogs on their own do not have this ability, but teaming them up with a portal adds increased depth and relevance.

As with any idea, using RSS feeds to populate your content brings up a number of issues. The first and probably most difficult to manage is the fact that, as the portal editor, you may not have control of the content in these blogs. When choosing which blogs to feed from a portal, the editor must determine whether the source is reputable and the comments are moderated or not. Remember, the contents of these blogs are going to appear on your site, branded with your logo, so take the time to consider whether you

can trust the source (and the people who comment on them). WebLogic 8.1 Service Pack 4 introduced the notion of library services or workflow that enables the portal editor to approve or reject submitted content. Using direct feeds from blogs bypasses this approval process.

In addition, using blogs as the content for your portal reduces the effectiveness of searching and campaign creation because the blog data is transient with respect to your system; it is getting sucked in directly to the portal and not into your content management system. As a result, you cannot tag this content with meta information or even have it available for site searches. As the portal editor, you must determine the correct balance between human scalability and rich portal functionality.

Moving forward, I can envision portal functionality that would allow a content editor, within the CMS, to define a content folder node as being an RSS feed rather than a particular piece of content. Each time a new article is added to the blog that the RSS node points to, the article is downloaded into the CMS and creates a new content node. This content may become automatically flagged as "Awaiting Approval" (or whatever state the portal editor configures in the system). The portal editor can then approve or reject the article as if it were content created directly in the CMS. This approach would address the two concerns defined above with regard to using transient content.

## **Portal Must Be Able to Be Administered by Business, Not IT**

The holy grail of many IT projects is to provide a business with the ability to manage its own needs. Portals offer a great leap forward in this respect. At its core a portal is about presenting certain information to certain people based on certain criteria. This sounds simple enough, but the how and when are often determined by IT (we can't deploy this yet because of...), and the why is determined by the business (I need this functionality because the business lifecycle requires it). Such scenarios may result in competing goals, both operating in their best respective interests. A successful portal implementation attempts to change this. By providing rich, interactive, business-focused administration, business owners can publish content, enable rules, and target audiences on a real-time basis. This functionality can be further enhanced through the use of delegated administration, putting the ability to do one's job in the hands of the right people.

To take full advantage of a portal's real-time configuration, IT personnel should design flexible systems that further enhance the business's ability to deploy change. One such example may be designing portlets that take advantage of preferences (administering portlet preferences and JSP tags for portlet preferences) that may be utilized to dynamically configure their operation. This loose coupling of portlet functionality to specific nodes (for example, specific content types in a content management repository) enables runtime reuse.

## **Put *Everything* Into the CMS**

Although the decision to leverage a content management system likely has been made prior to launching the portal, many organizations continue to use the CMS in a less than effective manner. Advertisements and features are prime candidates for inclusion in the CMS as these usually are the drivers for campaigns and require more frequent updates, but this is where it stops. Portals often have large amounts of static content or assets that inevitably change (probably more often than you think) and are never placed into the repository. This may include JSP pages, documents, and images—anything you can (and cannot) think of.

Although just about every organization keeps its information in some form of version control system, this

is not always the most business-friendly approach and requires custom scripting to push to the production system. Placing all assets into the CMS gives your business control of its systems in a more unified manner. The small investment of making portlets out of each static page, for instance, will be repaid quickly the first time your business wants to change the wording on a static page that otherwise would require the deployment team to push the change out when it suits the team's schedule.

Of course, there are times when it may not be feasible to place everything into a content management system. When this is the case, best practices may be to deploy the application in an exploded format or serve your static content from a separate, often network-mounted drive. This may provide the ability to update content via the file system versus potentially having to redeploy the application due to content changes (which will happen more than you expect).

## **Turn the Business Inside Out**

Historically, businesses have been the keeper of information that their clients provide. Things such as release dates, current policies, brochures, and personal information are distributed and maintained as an internal process. For example, if clients wanted to update their information, they would call a customer service representative to accomplish this. With the implementation of a portal solution, internal processes may be exposed to the general public.

The first candidate for exposing to the public is self-service utilities such as those used to address the example above. These services may seem simple at first glance (let's create a portlet that allows a client to update the personal information that will, in turn, update our CRM system), but often they involve the orchestration of a more complex and more subtle business process. If disparate internal systems are not coordinated automatically, a triggering event (for example, an information update) may require notifying internal users to take action to complete what was once an internal process.

To provide the best opportunities for success, your business should work closely with analysts to ensure that the existing business processes can accommodate the changes that a self-service model bestows upon them. Remember, externally exposing business processes highlights the good and bad points in your processes. It's best to be prepared!

## **Conclusion**

This article focused primarily on the business aspects of a successful portal implementation—in particular, those targeted at building communities by leveraging the functionality provided by BEA WebLogic Portal. Portal applications provide an organic system that should be designed to continue to grow long after the portal's initial release.

Through careful planning by your business units in conjunction with IT, a portal solution will become the focal point for a new level of service inside and outside your organization.

*Quinton Wallis a Sr. Product Marketing Manager for Integration at BEA where he is responsible for articulating the strategic vision and direction of the products such as WebLogic Integration and AquaLogic Integration.*