MANAGING WEB PROJECTS: RECIPES FOR SUCCESS

Author: *Geoff Hewson*
Software Productivity Center Inc.

Developing web applications shares many similarities with conventional software development. However, the business realities of web development present the following significant challenges that conventional methods don't readily solve:

- Delivery timelines are slashed due to "time-to-market" pressures.
- Requirements are often vague initially, and only become clearer over time.
- Internet technology is evolving rapidly.
- The user interface is critical - your competitor is only one click away.
- Web development teams meld together a wide range of skills and cultures: marketing, artists, software developers, content authors, and managers. These people all have different mindsets and rely on different styles of communication.
- Web development teams are often geographically dispersed and include subcontractors.

Conventional software development methods (like the Software Engineering Institute's Capability Maturity Model for Software) are primarily focused on the needs of large- scale development projects that typically involving large teams and long development cycles. Controlling these types of projects is usually very document and process intensive, which simply doesn't work in the rapidly moving, dynamic world of web development. Instead, the web project manager needs to organize and manage a project so that it emphasizes delivery and is flexible to change, yet at the same time keeps the team focused on the immediate work at hand. This is a challenge. Here are four recipes for success:

Identify Critical Functionality

While being quick to market in a particular e-business space is usually the critical success factor for an organization, understand that you don't have to implement 100% of the desired functionality in the first release. Entering the market quickly allows you to remain competitive, build brand awareness and obtain a market presence. Market research shows clearly that your web application doesn't have to have all its bells and whistles when it goes live, as long as it provides the important business functionality users are looking for.

What you need to do to implement this strategy is to inventory the business functions you want to provide through your web application, sort them in order of business importance, and determine the minimum set of functions you can afford to go live with. This allows you to plan the evolution of your web application over time, delivering sub-sets of this functionality in a series of development projects and enhancements, starting with the highest level of priority and moving downward.

This is a surprisingly powerful strategy, as it makes it much easier to stomach the de-scoping of your development project because you already have a road map that shows you where and when new functionality is to be introduced.

Building a Web Site Is an Evolutionary Process

Building a new web site/application, or making a major update to an existing site/application, is best approached as an evolution of ideas, rather than the more straightforward "analyze -> build -> test -> implement" approach of conventional software development. There are two keys for succeeding at this evolutionary strategy.

The first is to get your project into "integration mode" as quickly as possible. This is the point where most of the difficult technical issues and strategic uncertainties have been resolved, and the project team is focused on delivering the "right stuff" in increments. Each increment builds and stabilizes new business functionality on top of an operational prototype of your web application.

The second key is the approach to getting your project into integration mode. You need a clear enough definition of the web application you're going to develop in order to move forward to construction. In many respects this is a miniature version of planning the overall evolution of your web application. In this case, you aim to get as complete an architectural picture of the web application/site as possible. This tells you the software components that need to be developed, the content pages that need to be written, and the User Interface (UI) look and feel. Typically you would develop an exploratory prototype to demonstrate the appearance of the UI and the connectivity between key systems. Tying these elements back to the prioritized list of business functions should allow you to get customer and management approval.

At this point, integration mode kicks in. Functionality and content are assigned to each increment of development in order of decreasing business priority and technical risk. If the business climate changes, this prioritized, iterative approach allows you to change tack mid-stream by introducing new or changed requirements in additional iterations, at the appropriate level of priority. Of course, adding new work to your project is usually done at the expense of the lower priority work you had planned to do in later stages of your project, which leads us to...

Aggressive Scope Control

Aggressive Scope Control is the "slash and burn" approach to project management. Given the tight time constraints of most web projects, there always comes a time when you have to trim scope or deliver late. Delivering late is rarely an option, and the short development timelines limit the effectiveness of adding resources to your project, as it takes too long to bring new team members up to speed. Your only realistic option is to be aggressive about cutting back on the lower priority functionality and content of your web application to bring the project back on target.

Human nature makes us try to avoid cutting scope for fear that the functionality will be lost forever. This is where the previous two strategies help. Having a long-term plan for the evolution of your web application allows you to find a place where de-scoped functionality can be scheduled for later delivery.

Be sure, however, to have a clear understanding of the absolute minimum set of business functions the web application must provide for it to be viable. You don't want to cut so much functionality that your web application becomes inoperable. If you're facing this, all you may be able to do is admit that your delivery target is unrealistic and "grin and bear it."

Ruthless Execution

The last strategy I want to discuss helps you maintain project team focus. Although web applications are often developed in a highly changeable environment, it's important to avoid spinning your wheels unnecessarily by dealing with issues that are of lower priority than the immediate work at hand.

The strategy of Ruthless Execution allows your project to remain open to change, while at the same time keeps the team's work efforts focused on the highest priority work at all times. As changes to the project are proposed (new or changing requirements, new deadlines etc), you capture them immediately in a change request database (using a software tool or a simple list in a spreadsheet).

All potential changes are prioritized relative to one another, and to the current development plan. Changes are introduced into the project only if they are of higher priority than currently planned work. If this occurs, re-evaluate the revised schedule and immediately apply aggressive scope control to re-align the project with your delivery timeline.

While change requests are being captured and reviewed, the project team continues to focus on the work of the current iteration of development. Wherever possible, only apply changes to later project iterations, leaving your current iteration intact. If you've done a good job of business prioritization, this will be feasible more often than not.

Conclusion

There you have it. Four straightforward strategies to help you cope with some of the uncertainties and fluidity of work hinges on being able to keep an objective view of the true business priorities for your web application, and aligning your initial development plan, scope control and change decisions to support those priorities. You won't be able to eliminate the dynamic, high-speed, high-change nature of a web project, but you should be able to bring a little order to the chaos.