

3055

A Safety Net for Web Development

Douglas Lowe

**Hewlett-Packard
3404 E. Harmony Rd., Ft. Collins CO 80525
Phone: 970-229-6510**

Introduction

Have you ever worried about accidentally overwriting your application files? Have you been a part of a small team development project where coordinating the workflow seemed like a big hassle and there were a lot of mistakes being made? Have you had to try to re-build a prior release of the software to incorporate a new fix, and found that the source files weren't in a known condition?

Whether you are developing web applets for Intranet, extranet or electronic commerce, you will likely be facing many of the same problems in managing the code and documents needed to create these end products. Regardless of the type of web application, the following categories of problems will generally exist:

- Versioning of applets and page hierarchies during development;
- Production release of a "tested set" of web pages for end-user operations;
- Enhancement, maintenance and re-release of the web applets and HTML pages;
- Maintaining the mapping of large web hierarchies to directory trees.

This paper will describe one organization's view of what the minimum requirements for Configuration Management (CM) would be to reduce the risks in web development, and create a "safety net" for the organization. In addition we propose some basic questions that should be reviewed to assess the risk exposure for your particular development environment. Finally, we explore how one organization in HP has integrated this tool into the processes they use for web development and what the benefits have been as a result of doing this.

Processes for Software and the Web

Most organizations have approached web development in the same way that software development was approached 10 years ago - using a very ad-hoc process, with little thought for formally managing the assets created by the process. In software engineering we know that ignoring the issues raised by configuration management procedures is, in the long-term, a good way to introduce significant risks to the future of the business

The steps involved in web application development are similar to those involved in the typical software development cycle. In either of these domains, the requirements for the application are understood and agreed upon and then the application is designed and reviewed. The design and review can take place using a prototype UI or other functionality, using design documentation or by a combination of these. Design is followed by implementation, which then leads into a testing stage. Depending on the type of application and the project manager's preferences, iteration across these stages may be an important factor in the development process. Also, it is generally true that the

activities in each stage of the process tend to overlap, so for example there is some design work still going on in the implementation stage, and some implementation, or changes, going on during the testing stage. All of this indicates that the development process can be complex.

Typically, web applet development processes are on a shorter time-scale (e.g., days or weeks) than large software applications (e.g., months or years). A shorter time-scale for development does not imply less risk to the project or the organization, however. In our experience, shorter projects tended to be more chaotic, with control of artifacts managed by one individual, very little review of work by management and very few controls on the source code or artifacts after the project was delivered. Also, additional risk is incurred for the shorter projects because a delivered baseline (product or applet) is quickly established, but then it is often subject to more frequent updates than the longer development projects. If the source code, object libraries and other documents are not carefully controlled, this can potentially cause the organization to "lose the recipe" for making the product.

Synchronous / Asynchronous Development

To reduce risk of losing the capability to update or support a software product, it is useful to understand precisely how this process occurs in the organization.

Earlier discussion of the development process for applets or larger scale applications was focused on what is here referred to as the "synchronous" development model. This model assumes that each stage's work products are synchronized (completed, reviewed and corrected) prior to a checkpoint for completing that stage. This is also known as "serialized" change to a product. [Figure 1, 2] For code implementation this could be viewed as successive implementations with intermediate milestone builds of the product, followed by unit testing and other types of testing appropriate for that stage. Synchronous updating of products occurs when the changes (fixes, enhancements, and patches) are accumulated and applied to a specific release cycle, where the product is then re-tested and re-certified. Both synchronous development and synchronous updating are typical in large-scale development projects in order to control change and reduce the risks of introducing new defects, although large-scale projects are often also required to update files asynchronously.

Asynchronous development or updating is when changes to a released baseline of software are made in parallel with the development of a second release of the product. [Figure 3] Asynchronous development and updating frequently occur on smaller projects, especially in web development. In this model, patches or updates occur at various intervals after release 1 of the software, but prior to release 2 of the software. The changes are applied to release 1 stream, while a second branch for release 2 is under development. Enhancements or patches to release 2 may also be started prior to the completion of release 2. All of this can be occurring independently (or asynchronous to the "release points") by applying the changes and testing to a specific branch of the source files (HTML, scripts, Java, etc).

Examples of Risks in Web Development

Synchronous development and updating are approaches designed to minimize the risks to the development process, by keeping careful control of baseline sources under a configuration management system. The scenarios for asynchronous approaches discussed in the previous section indicates an even greater need for change control of branches of the source files.

Risks are defined to be "situations or potential situations that might result in negative consequences". Refer to Table 1 for examples of this. There may be many reasons why a risk exists, rather than just a single contributing condition. For purposes of discussion here, the general reasons that the risks exist will be assumed to be:

- The development process is too complex - involving asynchronous changes, and more than

- one person
- There is confusion about where the "master" set of source files exists
- There is a failure in the control of source code for a released product

Example Risk Condition	Table 1 Potential Consequences
♦ A change in a released system causes product failure	♦ Customer can't place an order ♦ Customer downloads wrong version of product ♦ Hyperlink page isn't accessible
♦ A previous version of a working Java script is over-written or destroyed.	♦ New product development is halted while "DB library" is reconstructed and re-tested.
♦ Corruption of source code on a released system	♦ Complete system failure ♦ Can't rebuild / fix parts of system
♦ Two developers write changes to the same file.	♦ First person to save the file will lose the work when the next person saves the file

Process and Tools Reduce Risk

In dealing with risk conditions such as those indicated above, the Software Engineering Institute at CMU has created some recommended goals for software development processes in a model called the Capability Maturity Model (CMM). A subset of CMM goals and the practices that are recommended constitute excellent guidance for establishing standard processes for configuration management.

Initiating improvements in the configuration management area is sometimes viewed as "a waste of time" or "no immediate payback" - especially for smaller sized projects. Starting the improvement process by addressing a few simple but critical goals and practices may be the right approach for your organization. Two of the CMM goals and a few of the important practices for achieving risk reduction in the configuration management area are listed below, in under Critical Goals and Practices. This is our organization's interpretation of this key area of the CMM for application in project teams of fewer than four people.

Critical Goals and Practices

-
- ♦ Goal: Selected Work Products are Identified and Controlled
 - ❑ A CM Library is established as a Repository
 - ❑ All Work products for CM control are Identified
 - ❑ A Procedure is used to authorize release of products
 - ♦ Goal: Changes to Identified Work Products are Controlled
 - ❑ A Procedure is followed to manage change requests
 - ❑ Product Changes are coordinated across individuals
-

We have also learned that other goals in the configuration management key process area are very

useful as the project size scales up. Specifically these goals and the practices are listed below. They are useful to ensure that appropriate CM planning, communication and work activities occur during the project.

Useful Goals and Practices

-
- ◆ Goal: CM Activities are Planned
 - ❑ A CM Plan is prepared using a procedure
 - ❑ A CM Plan covers work to be performed
 - ◆ Goal: People affected by Changes are kept Informed
 - ❑ A Procedure is used to track changes
 - ❑ Audits of CM activities and work are conducted
 - ❑ Results of audits are reported to management
-

In a web application development project, the procedures and tools need to be easy to understand and simple to apply. In designing the procedures and work instructions for projects teams doing web development, it is important to keep the process simple and the documents short (i.e., less than one page).

HP Group's Case Study Example

A real example project within HP occurred when we staffed a project to web-publish advertising and ordering information about our products. Initially this project consisted of a HP outbound web manager, a contracted web page designer, and third party Web Server / Publisher. Refer to Figure 4. The basic workflow for the project consisted of the following steps:

1. The Web Manager analyzed and specified the requirements for the web pages.
2. The Web Designer created the web page content and information architecture and then moved these pages to the system integration and test machine.
3. The Web Publisher applied production standards and changes necessary for the production web server environment.
4. At steps 2 and 3, the Web Designer and Web Manager would log into the system integration and test machine, or the production web server to test the "release" of web content.
5. If problems were identified or revisions in content required, the Web Publisher would create a "tar file" of files for the Web Designer to ftp back to his machine to revise. Once the changes were made, the Web Designer would create another "tar file" to ftp back to the Web Publisher.

This scenario caused many types of problems for the people on the project. Refer to Table 2 for examples. Basically the issue was the existence of multiple "file systems" where the web page files could reside in different "states". It was difficult to determine the version or "state" of any file. This situation caused many types of problems. The table below lists a few of the more painful problem areas, and who was most affected.

Table 2

<i>Problem Area</i>	Web Designer	Web Manager	Web Publisher
♦ WHICH FILES WERE CHANGED?			X
♦ WHICH FILES WERE SYNCHED ON THE PRODUCTION SYSTEM?			X
♦ PRODUCTION ENVIRONMENT DIFFERS FROM TEST ENV.	X	X	
♦ DESIGNER MAY OVERWRITE CHANGES MADE TO TEST SYSTEM FILES.	X	X	X
♦ DESIGNER CAN'T EASILY GET UPDATED COPY OF FILES	X		X

The solution to this situation was to set-up a configuration management system that would serve as the "master" versioning archive for all files, and establish effective lightweight processes for coordinating the work.

Applying the CM Tool to Versioning and Archive

- Easy identification of files changed, added, deleted & the changes made
- Keep development, integration, production versions consistent (as needed)
- Update most recent versions that reflect latest standards
- Eliminate accidental overwrite of prior file modifications
- Previous versions of work are easily retrieved and viewed

Applying Process Standards

- Develop and follow procedures for planning, controlling and tracking items
- Maintain production baselines with strict change controls

Benefits of Combining Process Standards and Tools

Today the web project operates with a weekly asynchronous updating process to the web pages. The process is much less error prone. We have also found that it requires less effort to update than previously - we estimate that the same type of work requires about 50% less effort than before. This reduction is mostly due to the elimination of errors and rework, although having a central CM archive also reduces the effort to update the files.

When we made the decision to use a central CM archive and improved processes for our web development project, we also realized additional benefits:

- *More independence while working together* - we needed fewer phone calls and e-mails to coordinate file changes moves and updates.
- *Automatic list of what changed* - anyone on the project could quickly review the list of files that had changes and "diff" between prior versions or check the change header log for each file.
- *Fewer mistakes* -files that were changed are now always updated correctly on the production publishing system.

- *Accuracy doesn't depend on savvy and skills of the people* - files that are modified aren't "forgotten" during the update process, and the web page designer doesn't overwrite production environment changes.
- *Can test out of the archive* - we use a special capability to allow us to directly test web pages out of the CM archive. This capability provides a transparent link between the web server and our CM system.
- *Opportunity for more people to work on it* - as the project expands we can bring in additional web page designers and other departments, without compromising the quality or efficiency of our development and update process.

Evaluating Your Process Risks

A risk assessment is a useful first step in determining your exposure with no SCM (software configuration management) capability. Some basic requirements in SCM are needed for addressing the typical risk issues, and these are generally simpler than those imposed by complex software development situations.

CM Capability Area	Questions to Review
Branching	<input type="checkbox"/> Are multiple versions of files identified and tracked during development, test or production?
Versioning	<input type="checkbox"/> Can a prior version of a file always be identified and then retrieved?
Baselining	<input type="checkbox"/> Can a baseline production release be re-constructed by someone unfamiliar with the project?
Process Definition	<input type="checkbox"/> Are documented processes available for planning CM, item control and baseline updating of the systems?
Process Deployment	<input type="checkbox"/> Is a defined process used to keep track of design, test and production files that are added, modified or deleted?
Environment Control	<input type="checkbox"/> Are the sources, object libraries and tools available to easily re-construct a prior release of the software?

Summary

Web development and updating has risks similar to that of traditional software application development. The risks of product failure, expensive rework or just simple mistakes exist even for small team projects that typically happen around web applets.

Web applets and page designs are becoming more integrated into a company's business processes faster than the traditional software applications because they can be developed faster and more easily. Intranet applications for managing project data, for employee training and reimbursement are common in HP. Extranet applications for advertising products, providing sales tools, and product ordering are now becoming the norm for companies. These applications reflect basic processes for the organizations that rely on them and this means that some risk to the business exists if they suddenly stop working.

Configuration management tools and processes that mitigate the risks are highly beneficial. In our example web development project we found many problems and higher costs when we tried to operate without using a CM version control system and some simple processes.

If you are already doing web application development work or you plan to do so, you can use a simple assessment procedure that we propose to evaluate the areas of risk for your project.

Figures 1 - 4 are available in paper copy only, or at the presentation session.

If you wish to have a copy of the presentation materials, you can contact the author, Doug Lowe, by e-mail : lowe@fc.hp.com for a paper copy or electronic Powerpoint version.

