

In order to better understand the problems and issues of ROI analysis, the Center for Technology in Government decided to use a form of ROI analysis for one of its own IT investment projects. The Center's mission to build and disseminate knowledge about IT uses in government usually leads to projects with other organizations. But this case provided an excellent opportunity for learning and knowledge building as part of the Center's own internal work. The Center's Web site is a critically important tool in disseminating information and maintaining contact with colleagues and customers.

During the time covered by this case, the Web site consisted of approximately 3500 pages and experienced over 1000 visitors per day. The Web site was supported by a full time Webmaster and part time maintenance and development staff of two professionals and two graduate students who are part of the Center's Technology Unit. The Web site had been in operation for several years and there was a substantial baseline of experience with development and maintenance using the current architecture. Any investment in new Web site architecture would use internal staff and resources. Therefore, it presented a good opportunity to apply ROI analysis to a realistic problem in a situation where data collection, analysis methods, and the results can be examined at close range.

The purpose of the investment was to change the Center's Web site from a static, HTML-based architecture, to a dynamic, XML-based one. The problem to be solved was a real one, namely that the Center's Web site had grown in size and complexity to the point that maintenance and additional development put a serious strain on existing resources. In addition, the static site could not support new formats and capabilities the Center wanted to use on the site.

The Center faced the decision of whether to continue to invest additional human and technical resources in the existing HTML-based static architecture, or change the architecture into an XML-based one that had potentially much lower maintenance and development costs. The change would require a substantial initial investment in tools and staff time to learn how to use XML and related development applications. The ROI analysis would help answer the question of whether the potential returns to be obtained by the use of a dynamic (XML) Web site architecture would exceed the costs of the conversion. The expected returns would consist of savings compared to the costs of maintaining and continuing to develop the existing Web site. Returns would also include the expanded abilities of staff to create enhanced Web site capabilities and features using the new architecture and application tools.

Estimating the potential cost of expanding the staff to maintain the current Web site architecture was very straightforward. However, estimating the conversion costs to a dynamic Web site was a much more complicated problem. A complete ROI analysis should also include an estimate of the savings (if any) to be obtained by using the dynamic architecture. Estimating the savings would require comparison with the costs of maintaining and developing the existing system, so estimating these costs was part of the design.

### Advantages of a dynamic Web site

Web sites can be created with either a static or fully dynamic architecture. Static Web sites consist of HTML pages that combine content (words, numbers, and images), logic (how the content is manipulated), and presentation (colors, layout, fonts, and formatting). In HTML, there is only limited capability to manage or change the content separately from the way it is manipulated or presented. Managing and changing the material requires working with content and HTML tags directly within each of the pages. Static sites are usually easier to develop than dynamic Web sites, but can become much more costly to maintain and manage as they increase in size. Dynamic Web architecture, in this case based on XML, provides the ability to greatly simplify the management, evolution, and expansion of a Web site by providing control of the logic and presentation independent of the content. For example, users can be presented with information based upon their individual preferences, such as larger fonts for a visually impaired person. The site can also be made much more interactive. Dynamic Web sites are generally more expensive to develop than static ones, but are cheaper to manage and maintain up-to-date content.

### Data sources

The best source of data about the costs and returns for this new Web architecture came directly from the staff's experience. The Center staff needed an estimate of costs and returns for the XML conversion that they could trust as a valid representation of what they could expect in a full-fledged conversion, without actually implementing that conversion. To do this, they chose to develop a few relatively small new Web-based applications as pilot projects using the new methods and architecture. The first was a decision making guide to help organizations design electronic information access programs. The project was called "Gateways." The experience from those efforts

provided the necessary information about costs and returns to inform the larger decision.

Technology Unit staff members responsible for Web site development and maintenance consisted of three full-time personnel and two half-time graduate assistants. All were individually interviewed weekly for a period of four months. During the interviews, members were asked to recall the work they had done the previous week. They were asked to identify:

- whether tasks were done in HTML or in XML,
- the amount of time spent in production ("doing" the work) versus learning the new application in order to do the work,
- the benefits of working with XML, and
- the drawbacks.

The data for the study also included information about the operation of the Web site and related activities, such as impacts on routine site maintenance, changes in content development procedures, and coordinating work on this activity with other technical tasks.

Weekly interviews were a low cost and relatively unobtrusive data collection method. They provided enough data so that it was not necessary to observe work directly or have the staff record their work times in activity logs. This method fit one purpose of the case: to be a useful example for others contemplating ROI analyses, employing methods that would be useful in a wide range of settings without requiring highly specialized training or high costs. The staff involved in the work agreed to the interviews and prepared for them as a regular weekly activity.

### Cost estimation

One goal of the case study was to establish an estimate of baseline costs for Web site operations. This was complicated by the fact that the CTG Web site is continuously growing with the addition of new and different types of information. The estimated costs for maintenance of the current site (at the start of the study period) were approximately 75 percent of one full-time staff person's effort per week. This was treated as stable over the term of the case study.

The case data also included estimates of the effort devoted by staff members to researching XML (dynamic Web page design), which began several months prior to its initial use by the Technology Unit staff. The director of the Technology Unit had been contemplating moving to XML for quite awhile. This early deliberation was prompted by the rate of growth for maintenance and development efforts on the CTG Web site. The effort devoted to this period of informal research and deliberation was considered part of the initial investment.

**Baseline costs.** The Center's Web site is very extensive, consisting of approximately 3500 Web pages and receiving on average over 1000 visitors per day. Information is continuously updated and new products and reports are introduced frequently. Maintaining the Center's Web site is quite an undertaking. Under the old HTML architecture, a Webmaster was responsible for maintaining site links, updating current materials, adding announcements and new materials, and related tasks. A full-time staff person devoted approximately 75 percent of their effort to maintaining that site (considered nine person-months per year.)

**Investment costs.** The primary investment cost for this project was staff time to learn the software and develop the skills necessary to build the pilot projects. The software costs for the conversion might have been quite high, since initial research showed most commercial products to be very expensive. However, after a review of available tools the Technology Unit decided on the open source (free) application, **Cocoon**. This tool seemed to fit the needs of CTG and the staff began working with it.

While the software was free, it required the staff to learn new skills. The underlying logic of how the Web site operates was also considerably different under the new architecture. So the development team had to learn to think about its tasks in a new way as well. By contrast, there were zero learning costs for maintaining the Web site with the old architecture and methods.

The initial learning process was a slow one. It took approximately three months for the staff to reach a level of skill necessary to move from using HTML to using XML. At first, as much as 80 percent of their time was spent on learning the XML application and related skills, and 20 percent on production. After a 10-week period, the time devoted to learning dropped to approximately 40 percent of the total, with the remaining 60 percent used in production. Three months into the effort, 20 percent of staff time went to learning and 80 percent to production. The staff found that this latter 1:4 learning-to-production ratio is consistent with the learning-to-production ratio under HTML (prior to the switch to the new technology).

The learning took place in two stages. In the first stage, the team shifted its way of thinking about Web development from the HTML to the XML architecture. The second stage was to learn the technical details of working in XML. After the initial three-month learning stage, staff members considered themselves knowledgeable enough about the language of XML to deal with problems quickly. The second stage included learning to use the **Cocoon** development tool. Since documentation was limited, they used the **Cocoon** listserv for questioning others who use the program. Much of this learning came from applying the new tools to the tasks specific to the pilot projects. This included creating interactive tools that allow users to work with applications on the Web site. The team's learning process was focused and task-specific, the results of which could be applied immediately. The staff avoided exploring unneeded features and functionality of **Cocoon**.

Additional learning costs are reflected in the different amounts of time needed to revise the printed version versus the electronic version of the Gateways decision making guide. The first pilot application developed with the XML architecture was based on that guide. Initial changes to the electronic version of the guide took up to 50 percent longer than print-only revisions because of the time spent learning to apply the new technology in the Web application. Initially the staff found it difficult to use XML data structures. As a result it took considerable time to understand basic concepts and acquire basic skills for solutions that at first often appeared to be quite simple.

### Benefits

**More efficient development of new features and capabilities.** Benefits were seen almost immediately during the learning process. During the first three months the staff were able to see how XML would directly benefit their future work. For example, a separate file of interactive tools was integrated into the Gateways Guide using established style sheets taken from existing components of CTG's Web site. The new technology saved considerable amounts of time. This particular task required only four hours of work instead of the estimated 16 hours it would have taken using HTML.

**More efficient content management.** In HTML, the content of the page is tied to the format, which makes it difficult (if not impossible) to separate ongoing development and maintenance functions from content revisions. In order to update the content of a page, some knowledge of HTML is required. XML operates differently, separating "content" from "presentation" or "style." Since the content and style are not tied together, the ongoing development and maintenance functions are separable—allowing for division of labor or specialization. Content "owners" or "creators" work only with content while Web designers work with Web coding.

In XML, content changes can be managed by the content creator alone. Changes made to the XML source file will appear in all formats such as Netscape, Internet Explorer, text-only, and PDF. Under HTML changing one word of content typically requires changing that word in all formats. In comparison, XML saves considerable time and ensures consistency throughout the site. More importantly it allows for greater flexibility in managing content. Since changes are more easily made, it is more likely that the information will be updated and changed as needed.

When XML is used, the Web site can be managed quite differently from HTML-based methods. Work in XML and **Cocoon** played to the strengths of the staff members by allowing different components of the Web site (content, logic, presentation) and their related tasks to be kept separate. Individual staff members can concentrate on work at which they are more proficient. Dividing the labor saves time and allows staff members to become "experts" in specific tasks. In an HTML environment, all three components—content, logic, and presentation—are all combined in the individual Web pages. Anyone working on the page must understand and deal with the components together. Small changes in content could require extensive work with logic and presentation factors and involve much more interaction and coordination among content creators and HTML workers. In the XML environment, content revisions and expansion is separate from the other components, which can be manipulated largely independently. Content changes require very little coding or programming resources, and changing or developing new programming capabilities is not constrained by potential impacts on content.

**Less maintenance effort.** Current Web sites based on HTML use the Web page as the unit of composition; so a 50 page site can be thought of as having 50 separate units of work. These static Web pages combine both content and formatting. As a result, changes in one page have virtually no effect on the others. For a single Web page or a Web site with few pages or minimal updates, working on individual pages may be relatively simple and affordable. However, as the size and complexity of a site increases, the cost of ongoing development and maintenance increases dramatically. It thus becomes increasingly difficult to keep the site consistent and up to date. In addition, some functions, such as forms or databases, are difficult or impossible to do on static sites.

In XML, code and content are easier to maintain since code and content are stored in separate sources. XML

allows for the streamlining of ongoing development and maintenance functions because content is stored in one place but propagated in many places. Because of this architecture there are fewer files to maintain (a 50-page site may contain only one content file and 3–5 style files). Less time is required to maintain the fewer files. One of the benefits of employing this new approach, despite the time and resources spent on the learning process, comes from the potential "payback" in greater functionality, easier maintenance, and reduced ongoing costs.

**Greater browser support.** In XML, Web design is independent of browser capabilities. Changes made to an XML source file will appear in all formats such as Netscape, Internet Explorer, text-only, and PDF. Under HTML changing one word of content would require changing that word in all formats. XML saves considerable amounts of time. More importantly it allows for greater flexibility in managing content. This lowers the cost of adjusting formats for presentation in different browsers. It is difficult and time consuming to craft HTML documents to work consistently in all browsers, since each browser interprets HTML standards differently. However, when a page is created in XML using the **Cocoon** framework, it can easily be made viewable in all browsers. Staff no longer devote time testing to ensure a page is viewable in all browsers, since the inclusion of an XML browser parameter automatically adjusts formatting.

**Greater platform support.** XML standardizes and universalizes "content" or "data" across platforms. Currently, a majority of Web pages are mainly constructed using HTML. The "content" is tied to the HTML format. The page works on a Web browser, but may not work on other platforms such as wireless devices, cell phones, and PDAs. XML offers greater delivery functionality across a variety of platforms. It allows the Center's staff and customers the flexibility of viewing pages of the Web site on any Web server architecture.

**More efficient management of style and presentation.** Style and presentation of the Web site are more efficiently managed with XML than with HTML for many of the reasons discussed above. Since the page itself is not the unit of composition, it is easier to maintain consistency throughout the site. Banners and footers, for example, are maintained as single files and can be imported to all style sheets. The separation of "content" from "presentation" allows for division of labor. Since Web design is independent of browser capabilities, Web pages are viewable in all formats.

## Summary of costs and returns

The relationship between investment and returns in the Web site conversion is best expressed in terms of shifts in productivity and opportunity. The principal costs of the project are reflected in the opportunities sacrificed and the added personnel costs incurred in order to develop new skills and capabilities. The primary cost component is opportunity costs: staff were diverted from routine work to learn new skills and to begin to use the new techniques. Some additional staff resources, in the form of graduate students, were used as well. The magnitude of that opportunity cost is represented in the comparison between the two charts in figures 4 and 5.

If the project had not been started (Figure 4) there would be available resources during the first three months of the time-frame to maintain routine Web development and to pursue some new opportunities. In the same time period (Figure 5), by contrast, the project began with a much expanded Web development/learning effort that consumed all slack resources and some operational resources. No other opportunities were taken up and ordinary Web operations were reduced. That was a period of substantial investment with little return.

By months 4–5 into the project (Figure 5), sufficient learning had taken place and allowed for exploiting new opportunities. As experience and the stock of usable components grew, the cost of added development dropped rapidly and resources were freed up to undertake even more new opportunities.

### Figure 4. Cost Trends

### Figure 5. Cost Trends

The cost of ordinary Web operations did grow somewhat during that period due to the normal development of new content. However, the total cost of including new content and making routine maintenance changes in the site is expected to drop, despite increasing volume, due to the increased efficiency of the dynamic architecture. As efficiency in the development and operations of Web site work increase, the opportunities for exploiting new opportunities grow. This can be contrasted with the projections in Figure 4 that suggest the increasing cost of ordinary operations under the old architecture will squeeze out development opportunities and in time will likely exceed the budget constraint.

This kind of opportunity-based analysis can show positive results when relatively little financial data is available, and when the value of new products or capabilities are difficult to determine. There are no savings apparent from this effort unless the analysis takes into account what it would cost to develop these new capabilities and exploit new opportunities under the old technology. However, the expansion of development opportunities over time, in contrast to the baseline scenario, does give a reasonable basis for estimating the returns for an investment such as this in its early stages. Since the projections are based on the early stages of the work, it is possible that the unanticipated problems of increased scale could change the results over a longer term.

---