



## TenStep Supplemental Paper

---

7 April 2002

### Evaluating Legacy Application Technology

You might think of legacy business applications as those workhorses from the seventies and eighties that handle the routine, but critical, transaction processing for your company. There can be a negative connotation associated with these applications, and much of it justified. Most of them use non-standard technology. They usually don't give your business the competitive advantage you need today. In general, they require resources to support and enhance, resources that your company would rather utilize on new initiatives that would provide substantial value to the business.

A CIO that did not come up through the development ranks may find it odd to look at the various development languages and tools being utilized. How did you get into such a hodge-podge of technology? Were there no standards in the past?

#### The Problem with Standard Architecture

Yes, your developers probably had development architecture in the past and used a standard set of tools. The problem is that as technology moves forward, the standard development architecture changes as well. In the seventies, your applications were COBOL and CICS/VSAM on a mainframe. In the eighties, you moved to relational database technology and converted to DB2. When the client-server revolution hit, you built your applications in PowerBuilder, Visual Basic and Oracle. Today, your new applications are HTML, XML, VB Script and ASP. (Thank God Oracle is still around.)

How about the future? Are you looking at Linux? How about wireless technology? We will probably have tools and technology that do not even exist today.

#### What You Can Do

Once you see how your legacy environment evolved, you will probably struggle to find how to get out of it. You are not alone. There are no good, easy solutions. Options include:

- **Convert applications to the latest technology** - Usually this does not make business sense, unless the old application is no longer meeting the basic business need. It is hard to put together a business case in which you spend lots of money for new, standard technology, only to have the same basic business functionality you have today.
- **Outsource the legacy environment** – Usually, companies are reluctant to lose control of their critical legacy applications. However, if you do it, you can make legacy support someone else's problem.
- **Use a rolling long-term migration plan** – Over time, try to replace applications that are the oldest, require the most resources, and are the most difficult to support from a technology standpoint. Each year, determine what makes the most sense for the coming year and the next four. The following year, create the five-year plan again, this time pushing the end of the planning cycle one year further.



## TenStep Supplemental Paper

---

Don't ignore the problems of the legacy environment. Determine which technologies are being utilized and what problems are being encountered. You cannot do everything at once, but the legacy mess will not be any easier or cheaper in the future. See if it makes business sense to replace applications using a five-year plan.

Make sure your development organization has a comprehensive, standard development architecture, and that it is being followed. If you do not have a technology architecture, your problems will be worse, and new applications will be technically fragmented much sooner.

### Summary

- The legacy environment can be a technical Tower of Babel. If you utilized proper governance over your development architecture, you still got that way by making hundreds of valid technology decisions and compromises over time. If you did not follow a standard development architecture, you evolved to your current state haphazardly, and are much worse off.
- You cannot wave a magic wand and standardize the technology in the development environment. You may be able to outsource the headache. Otherwise, take a five-year view to continually migrate older applications to newer technology. You have to start somewhere.
- Update and enforce your development architecture. If you do not, you will surely become worse off.