



A Disciplined Approach to Process Improvement

Wed, May 03, 2006

[CIO](#) — By **Fernanda Young**

A few months after I joined the Export-Import Bank of the United States (Ex-Im) last year, the agency, which serves as the official U.S. export credit arm, embarked on an effort to update its loan-processing and other financial systems. As CIO, my goal was not only to respond quickly to demands made by top management, federal regulators and outside auditors, but also to establish a rigorous, repeatable process that would ensure the success of this, as well as future, development projects.

The successful update of Ex-Im's financial applications and reports has enabled the bank's analysts to gauge risks more precisely. The project also established new processes that have fundamentally changed the way the IT group works with the business users it develops applications for, and with outside consultants—who often get involved with such efforts. The project, which took about six months, transformed Ex-Im's IT group from an organization that, in the absence of formal structures, let programmers make changes on the fly, to one that adheres strictly to a highly disciplined approach.

Here's my advice for others heading down the process improvement path.

Zero In on Top-Down Drivers

Step one is identifying what you want to accomplish, from a business, not a process point of view. Three factors drove this project: First, to represent the bank's financial position more precisely, top management asked that the CFO's office present financial data not just in U.S. dollars, but also in the foreign currency of the country to which the money was loaned. That sounds like a small enough change, but from a programming point of view, it touched every system we have. Second, the bank had to comply with federal regulations recently enacted by the Financial Accounting Standards Board, which specified accounting guidelines for institutions that make loans to organizations that buy and then lease equipment. Complying required literally hundreds of changes to existing mainframe applications. Third, Ex-Im's auditors asked the bank to establish procedures that could prove the accuracy of the financial reports used by the bank's analysts to assess risk. Because such reports pull data from a wide array of applications and databases, and often require complex programming interactions, documenting how the numbers were arrived at was critical.

Build a Dream Team

To accomplish all three goals, we put together a "dream team" of 17 players to move beyond the spiraling costs and last-minute crunches that plague many development efforts. The team included Ex-Im's IT staff, the customer (in this case, the CFO and analysts who use the bank's financial

applications), and a professional services team including quality assurance and testers from Reston, Va.-based Software AG.

To foster communication and build cooperation among members, the dream team met every day. By creating an atmosphere that allowed the group collectively to understand what had to happen, we forged a true partnership among management, IT and the outside consultants.

Gather Input from Programmers

At the heart of any application development project are the programmers. Make sure you solicit their input. The Software AG team led that process, conducting a series of meetings with each of the nine programmers. Although this group had earlier been accustomed to working in a more informal environment, it quickly grasped the scope of this effort and saw the new, more rigorous approach as a way to get things right—to deliver in a timely fashion the application capabilities financial analysts needed to do their jobs. It helped that the programmers were strong proponents of configuration management systems and that they understood the concept of the application development life cycle.

Develop a Disciplined Approach

Changing the way things work takes time. To get the team to accept the new principles, you have to train and explain—enabling them to understand what happens when a team moves too quickly without enough analysis and quality. Success results from following the rules: All changes to an application require sign-off by the customer; changes should be deployed in releases, not daily; all requests made by the customer must be documented; risks and time frame must be explained to, and signed off by, the customer.

Expect Resistance

The customer may say, “Why can’t I get my change made tomorrow? That’s what we used to do.” It is a culture shock, but you have to get them to look at the big picture. Although things don’t appear to be progressing as fast as with earlier projects, adding rigor to the process results in a better product, and that ultimately saves time. Make sure, too, that the customer understands that you’re not developing the application for IT, but for them. Tell them: “You, as the owner of the application, tell us if it works and sign off. An error in the application is your responsibility.” Ex-Im’s CFO and deputy played a key role in enforcing that mindset. He would gather his group around the conference table, guiding them to reach consensus on which changes were priorities. Everyone had input and accepted the outcome.

Implement Checks and Balances

Programmers cannot be responsible for reviewing their own code. It’s not realistic to expect them to always find their own mistakes. Instead, implement a peer review process. Also key is separating the testing and quality-assurance processes from the actual coding effort. Prior to this initiative, Ex-Im had not established a separate quality assurance environment.

The project’s successful completion has given the bank’s analysts, executives and auditors a more precise picture of the financial risks associated with loans. And it provided Ex-Im’s IT group with a reliable, repeatable process—a clear road map to guide all future application development efforts. That’s the proper way to do things.

Fernanda Young is chief information officer of the Export-Import Bank of the United States.