HOW TO COMPLETE A SUCCESSFUL COBOL VERSION 5.2 OR 6.1 ROLLOUT

After you've worked through our suggested steps for migrating your COBOL programs to COBOL Version 5.2 or 6.1, it's time to begin an initial rollout. Here's how.

After you've migrated your COBOL programs to COBOL Version 5.2 or 6.1, it's time to begin an initial rollout. As always, be sure to refer to the IBM Migration Guide. Following those guidelines will generally make the migration go more smoothly.

Before you roll out your COBOL Version 5.2 or 6.1 programs, set up some comparative testing. Run the data from the previous COBOL release you were using. Later, when you run your new programs, you can compare them to the old programs. This will help you quantify the benefits of migrating to the new release to business leaders. Every machine cycle saved allows other applications to use those saved cycles for better performance.

Rolling Out Your Programs

Begin your initial COBOL Version 5.2 or 6.1 rollout with one business application and document the migration process. This will serve as a template for subsequent rollouts. During and after the initial rollout, seek feedback that will help you inform other business units on the success and encountered issues of the rollout. Discovering issues early on will make it easier to correct them and adjust your template for later rollouts.

After the initial COBOL Version 5.2 or 6.1 rollout, go back to that comparative testing we talked about

for an early idea of potential savings. See what worked and prepare to move forward, but don't begin rolling out everything.

Only convert those applications that can benefit from the new versions of COBOL. Start with your highly arithmetic, floating-point or complex mathematical applications. These application types will take advantage of the z architecture of COBOL Versions 5.2 and 6.1 and will benefit from higher levels of optimization. If applications do not take advantage of the z architecture extensions, or would not benefit from higher levels of optimization, you should leave them in their current versions of COBOL.

As a side note, for sites with 24/7 CICS regions, replacing PDS with PDSE won't be easy. What you can do is dynamically insert program objects ahead of your load libraries in the CICS RPL. This will provide a way for you to take advantage of new COBOL Version 5.2 and 6.1 programs for CICS.

Continue this COBOL Version 5.2 or 6.1 rollout path, releasing programs that will benefit from the new COBOL version you've decided to go with and testing those against their previous versions to calculate savings.

Reviewing Your COBOL Migration Strategy

Before you roll out, it may be useful to go back and review the steps that you've taken. For your convenience, here is a list of things to note, compiled from past blog posts on the subject:

1. Don't convert your COBOL programs to Java. This requires sacrificing efficiency for cost saving, and the cost saving you generate will be short-lived and evolve into a series of long-term pitfalls costing you more money.

1. The first step in a COBOL Version 5.2 or 6.1 migration strategy is roadmap planning. You need to understand your current programs, consider some potentially problematic areas you could happen upon when migrating them, understand your hardware machines, and determine which version of COBOL you're going to migrate to (there's an explanation for why you have a choice).

1. If you have a choice between COBOL versions, which is better? Starting at COBOL Version 6.1 eliminates duplicate compiler upgrades twice, once to 5.2 and again to 6.1. However, by migrating to COBOL Version 5.2 first, you can save costs by leveraging the IBM Enterprise COBOL trials for both 5.2 and 6.1 to gain maximum experience and application comfort before making a formal decision to upgrade.

1. A large percentage of these migration problems are data related, so thorough testing is crucial to discovering them. The testing you do now will save you from countless hours of debugging, potentially losing revenue and disappointing your customers.

1. There are several types of performance optimization available in COBOL Versions 5.2 and 6.1,

but you may need some guidance on how to identify and use these new COBOL optimization features.

Digital business is driving more mainframe activity. Mainframe shops should optimize their programs so they perform well against the modern demands they face from new and growing digital engagement. The best way to optimize your COBOL programs is to migrate them to COBOL Version 5.2 or 6.1. Good luck along your journey.