Getting started in software delivery automation with IBM Rational Build Forge

Tips to get up and running quickly

By Khurram Nizami
Worldwide Technical Empowerment
IBM Corporation

August 2007

In this article

Automating software delivery processes

Software delivery process automation and optimization is quite possibly the "final frontier" left for modernization in the software development lifecycle (SDLC). Today, many organizations have taken their first steps by automating their delivery processes with custom solutions. These solutions help organizations make due, but they often leave substantial value in coordinating, organizing, and automating delivery processes unanswered.

This article will describe best practices to help you quickly begin realizing value with a software delivery automation solution using practical examples with IBM® Rational® Build Forge™. Specifically, it will cover basic planning and groundwork, initial deployment and sizing considerations, instituting your existing processes, and where to go once you have started.

Plan for success

Proper planning and strategy can often make the difference between lasting value and fast failure. Your software delivery automation solution should not be a big bang deployment. Instead, it should be implemented in phases, and should focus on automating processes that will provide the greatest value with the lowest risk. This requires you to evaluate your existing software delivery processes and determine which of those will fit this profile.

Not all projects or software delivery processes can or should be automated initially. You may find it tempting to implement complex delivery processes right away, because of the high overhead and potential value that are associated with them. However, you should only implement them after you have successfully established simpler delivery processes.

Complex delivery processes often involve multiple workflows and user groups, handling multiple exceptions, requiring integration from multiple repositories and data sources, and operating in multiple environments. Simpler processes usually source and integrate from a single data source, with one primary group of users and one platform or environment type, and only require delivery automation across two or three environments.

One effective strategy is to **select a single project or process** from your list of existing software delivery processes, and then conduct a proof of value with the software delivery automation solution that you are considering. This will help you build a strong case for the value that software delivery automation can bring to your organization, as well as give you a jump start on quickly realizing those benefits. You should take care to ensure that your software delivery automation effort does not divert your selected

project's goals. You should also **ensure that the right people and resources are committed to the automation project**.

In addition, you should think about how you will **articulate the business value that your software delivery automation solution will provide for your organization**. Your business value analysis may include your improvements in productivity, cost savings, and time to market. You may also articulate how you can demonstrate the payback or break-even point for your solution.

For example, if your organization is mostly project-based, you may plan to deliver your software delivery automation solution as a shared service with an associated charge. This could help your software delivery automation solution pay for itself, as projects leverage the solution. You could also further articulate the added value that the solution delivers to your individual projects and how the solution has collectively benefited the organization as a whole. IBM Rational can help you with your software delivery process automation planning and value analysis.

Get up and running

The hardware required for a software delivery automation solution is highly dependent on your processes, the amount of parallelism of the process steps, and the number of projects and concurrent active users. You should implement your selected processes into your delivery automation solution as they exist today before making broad changes to them. This allows you to benchmark your process execution, and to make incremental improvements with a manageable amount of change.

Your initial software delivery automation deployment does not have to be overly hardware and resource intensive. Your hardware sizing will depend largely on the number of concurrent projects that will be executed, and then on the concurrent users. The following hardware profile is an example of a deployment supporting 100-150 concurrent users:

- **Operating system:** Microsoft® Windows®
- **Hardware:** Server-quality, dual processor
- **CPU:** 2 GHz
- **RAM:** 2 GB
- **HDD:** 200 GB

Using IBM Rational Build Forge as an example, your initial deployment would install the Build Forge engine and core on your selected software delivery automation server. You would then install Build Forge agents on the servers where Build Forge will construct and complete its work. For example, you may install the agents on your development, testing, pre-production, and production servers so that Build Forge can orchestrate your software delivery across these environments.

Your next step should be to make sure that Build Forge can access all the repositories and data sources that it will be interacting with. For instance, your delivery process may retrieve and integrate assets from IBM® Rational® ClearCase® as well as other repositories. Build Forge supports a wide array of adapters to interface with diverse data sources.

Build Forge supports both horizontal and vertical scalability. This means that Build Forge will take advantage of hardware enhancements to the server as you scale up, as well as allow you to cluster Build Forge across physical servers for flexible load balancing as you scale out. Transferring a Build Forge deployment to new servers is simple, and it allows you to transition a small server deployment to larger, multiple servers as your needs grow. Build Forge also supports server virtualization, enabling you to maximize the value of your hardware resources.

Implementing your processes

When you implement a software delivery automation solution, that doesn't imply that you should reinvent your delivery processes. In fact, your foremost goal should be for business as usual with your new software delivery automation solution. You should spend little time writing new scripts or streamlining your existing processes until you can get up and running first.

Organizations often make the mistake of concurrently implementing radically new or significantly updated delivery processes with a new software delivery automation system. Your primary goal should be to get your existing processes up and running, so that you can minimize the changes that your software delivery automation system will have on project results. Your processes will not be, and should not be expected to be, optimal from day one.

First, take the steps in your existing software delivery process, along with preexisting process scripts, and translate them into the steps for a Build Forge project. Large processes may actually be multiple, interrelated processes that work together. You can implement the different workflows of these interrelated processes in separate projects, and then create a master project that executes them inline or chains them together. Do not forget notifications and scheduling: your software delivery process is not limited to only things that run on a server.

After your delivery process is up and running, you can revisit the steps in your process and find opportunities for improvement. You should clone the Build Forge project that you are enhancing, and then use the clone as a test area for your enhancements. You may begin your process improvement by looking for parallel step execution, and then refactor or eliminate steps. You can also add alternative pass/fail executions to make your process automation more resilient and minimize intervention.

Build Forge makes it easy for you to quickly assess the health of your processes with comprehensive reporting that includes iteration speed, performance, optimization, and so on. Incrementally improving and demonstrating the value of your software delivery automation solution not only reduces your risk, but also strengthens your organizational support and lasting commitment to software delivery automation.

Now what?

The real power and value of Build Forge is realized as you build upon your successes. A software delivery automation solution based on Build Forge makes it easy and fast for other projects and users to adopt software delivery automation.

Software delivery automation is more than just build automation. The true power of a software delivery automation solution is agile governance and oversight of your software delivery initiatives. For example, with Build Forge you can automate the production and delivery of release reports to your stakeholders. These reports can take your requirements, changes, and test results, and then provide you with a comprehensive lifecycle view for software delivery.

One of the first things you can do is revisit your automated processes and look for reusable steps. For example, some common reusable steps to look for are the following:

- Retrieving source code from various repositories
- Application packaging
- Deploying to various environments

Next, you can take a look at the Bill of Materials (BOM) reporting and customize it to include the information that is important to you. Build Forge allows you to easily update the adapters used to interact with various systems and repositories in documenting BOM information.

You can also extend the user base for Build Forge to include other roles, and provide them with greater self-service options. For example, you may have offshore development teams that need control in the distributed software delivery process. With Build Forge, you can provide these teams with self-service access and give them the ability to influence and change the delivery process without intervention from other teams.

You can provide granular security and access control so that different roles and teams only have influence on areas that are deemed acceptable. This is just one example of how you can keep your teams agile while still maintaining necessary governance. Finally, you can break the boundaries of software delivery automation and derive even greater value from Build Forge. For example, customers have used Build Forge for many innovative applications, including regular system administration tasks, such as:

- Adding new users
- Password aging
- Automating high availability failovers

The value that software delivery automation provides goes far beyond build automation. Software delivery automation can provide you with significant benefits in time-to-market and agile flexibility. Getting up and running quickly, and demonstrating value, is not as difficult as you may think.

See the **Resources** section that follows to explore how Build Forge can help your organization quickly derive value from software delivery automation, and enable you to achieve your business goals.

Trademarks

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both. Other company, product, or service names may be trademarks or service marks of others.

Resources

Learn

- Take a look at a demo of Build Forge.
- Attend a live webcast demo of Build Forge.
- Attend a webcast on build and release management with Build Forge.
- Subscribe to The Rational Edge weekly newsletter.

Discuss

- Talk to an IBM Rational Build Forge specialist by phone or e-mail.
- Seeking answers or advice about good build and release management practices? Are you a Build Forge user looking to connect with others? Post in the Build and release Management/Build Forge forum on developerWorks.
- Post your ClearCase questions and comments, and share your thoughts, ideas, and solutions with other users on the Rational ClearCase forum on developerWorks.
- Check out developerWorks blogs and get involved in the developerWorks community.

Get products and technologies

- Find more resources for build and release engineers and managers in the Build Forge area of the developerWorks Rational zone, including articles and whitepapers, links to training, discussion forums, product documentation and support.
- Find more resources for ClearCase users and administrators in the ClearCase area of the developerWorks Rational zone, including articles and whitepapers, plug-ins, scripts and triggers; and links to training, discussion forums, product documentation and support.
- ClearQuest users and administrators can find more resources in the ClearQuest section of the developerWorks Rational zone, including hooks, Eclipse plug-ins, product documentation, articles, and white papers.
- Download IBM product evaluation versions and get your hands on application development tools and middleware products from DB2®, Lotus®, Rational, Tivoli®, and WebSphere®.

About the author

Khurram Nizami is responsible for worldwide enablement of the IBM Rational Software products. Khurram's expertise is in helping customers by applying practical solutions to challenges that the software delivery community faces today. His specialty and area of interest is global development and delivery.