bmc

solution white paper

# Workload Change: The 70 Percent of Your Business DevOps Forgot

# Table of Contents

## INTRODUCTION

It is almost universally accepted that applications built today must be delivered faster, updated faster, and most likely retired faster. The current ideal state of application development is continuous delivery, where change, and hopefully improvement, is constant.

The problem for enterprise IT is that change is the anathema of stability— but IT has to accommodate it fluidly. Power and budget reside with the business, and what the business wants, the business gets.

This background of constant change has led to the rise of DevOps and a focus on communication among the various groups and disciplines that make up the entire software development lifecycle.

One mystery, however, is the almost complete absence of workload automation or batch processing in the DevOps discussion.

Batch, you say? This is the 21st century. We are dealing with cutting edge social, mobile, analytics, and cloud technologies. What's the value of batch in this ever increasing real-time world?

Payroll, inventory, supply chain management, logistics, bank deposits, customer sentiment analysis, web store recommendation engine infrastructure, and big data are all predominantly, if not completely, batch. Do any of these sound relevant to your business?

The truth is that computing happens in one of two major categories. One is interactive, meaning there is a person (or perhaps another computer) waiting for a response. The other is batch, meaning you give the computer a whole batch of stuff to do and it goes off and does it in its own sweet time.

And if you think batch is going away, nothing could be further from the truth. Batch makes up the majority of business processing performed by computers, and that share is only going to get bigger. This is because "interactive" processing is getting more complex, requiring more computing power and leaving fewer resources for doing the stuff that doesn't need to be done in real time. For example, if you have ever booked an airline reservation "in real time" or bought something at a web store, you may have gotten an email some time later (with airlines up to 24 hours) with the actual confirmation. The reason is because there's a lot of processing that has to be completed before the actual confirmation can be made— and you wouldn't want to wait in front of your computer for that confirmation. Instead, it is completed in batch, and you're notified later.

The goal of this paper is to discuss why workload automation (occasionally referred to in this paper as WLA for convenience and also called batch or job scheduling by many) is largely missing from the DevOps discussion and to explore some possible ways toremedy that situation.

## WORKLOAD AUTOMATION IN THE ENTERPRISE

When examining the typical collection of technologies that make up an application in the production environment, WLA stands out as unique.

For the sake of discussion, let's say a complete application stack consists of an OS such as Linux®, an application server like Apache™, and a relational database—Oracle® MySQL® might preserve the LAMP stack concept, but in traditional IT, it would most likely be Oracle® RDBMS. Other components may comprise messaging infrastructure and perhaps a "plain" HTTP web server.

Developers develop in their language of choice. They then start testing. At this point, their test environments include pretty much the entire application stack. Perhaps in enterprise apps, testers are running only portions of the entire application and are therefore using only a portion of the application stack. But eventually, once unit testing is complete and system testing begins, the entire stack is deployed and utilized. This all happens as part of the software development life cycle (SDLC) and many if not most developers get pretty comfortable with that technology stack.

Eventually, system, user, and acceptance testing is all complete and it's time to promote to production. Once the application arrives in this new environment, some technology may be applied that has not touched this application previously. It's likely there is a monitoring solution that now watches for storage, processor, and memory consumption. DBAs may be watching the complexity and efficiency of SQL queries, network admins may be examining the load this new application places on network bandwidth, etc. Very generally, these are operational tools that watch the application in the context of the overall IT environment and monitor the health of the technical components. If any or all of these tools were to disappear for some reason, there would certainly be severe implications in terms of quality of service, maintainability, capacity management, etc., but the application would continue to run.

That is, unless it was a batch application (or had a batch component) and one of the solutions that disappeared was WLA. Ask any IT operations practitioner what happens when their "batch scheduler" stops and the universal answer is "our business stops!" The business of IT is inexorably linked to WLA. Clearly, workload automation is in some way a critical part of the application, yet it is usually perceived as exclusively an operations tool.

Developers are very familiar with the application technology stack I mentioned earlier, but they are usually completely unaware of WLA. Even though the success of an application will eventually depend on how the workflows are built and how efficiently they run, the connection between application code and WLA is not made until a relatively late point, often at the very end of the SDLC.

## DEVOPS

DevOps is a software development method that responds to the interdependence of software development and IT operations with a focus on communication, collaboration, and integration. The end goal is rapid, efficient production of software products and services. It's considered a culture or process upheaval that seeks to address the disconnect and the traditional "throw it over the wall" approach to application deployment.

It is not intended to significantly change roles or to disrupt processes but rather to focus on greater communication and transparency. So when part of the application deployment includes a web application server configuration or the need to build a database instance or update schema, DevOps seeks to automate, streamline, abstract, and communicate those activities.

However, if a step or activity is absent from the current manual process, it is unlikely to be introduced in the automated version of that process especially if it requires skill or knowledge absent in one domain (Dev) or the other (Ops).

### DEV

In this context, it doesn't really matter whether application development is done using waterfall or agile approaches. Developers write code and do some level of testing. At that stage, the workload automation challenges that will be faced later already make an appearance.

Let's say we are developing a web application that processes various transactions, and after 'n" number of those transactions, spawns or triggers batch processing against the accumulated data.

There's code that runs on an application server, code that transfers the collected data to a location for processing, code that publishes a message to a message queue to kick off the batch processing, and code that performs the batch processing itself.

All the code generates logs and for some or all of these steps, the log files have to be captured for subsequent debugging and analysis. Additionally, the success of one step must be verified before the next step is taken.

The very first time, or maybe even the first couple of times, the sequence is managed by a developer manually invoking one step, observing the result, and then going on to the next step.

However, pretty quickly, this becomes not only boring but inefficient, and scripting is usually applied to address the problem. This decision is what leads to the absence of WLA in the application developer domain.

Scripting comes into play, and as the application proceeds through testing and on to production, scripts may proliferate and become more complex, leading to problems when it comes time to "operationalize" in the production environment.

### OPS

On the operations side, WLA is viewed as an IT operations tool, so frequently developers are not given much support or access. It is common to hear of very large organizations that have no "test" environments for WLA. Imagine an application deployment in which a configuration change to some system component is required but has never been tested anywhere. You may shudder at the thought, but it happens with WLA all the time.

This view of WLA seems to be entrenched on both the Dev and Ops sides. This results in developers viewing WLA as outside their domain and their job responsibilities. Like monitoring or network management, it's something for IT folks to handle.

### So What?

Several problems arise from this approach.

The first is the amount of time spent on scripting. If one were to examine the shell scripts that become operational batch services, 60 to 100 percent of the code can be directly replaced by native capabilities provided by workload automation solutions.

Another problem is the quality of what gets deployed. No matter how good a programmer is, he or she is unlikely to be able to build functionality that will be as robust, mature, and flexible as what is provided out of the box with solutions that have been maturing in the marketplace for decades.

Yet another is decreased operational flexibility. Scripts tend to become multi-step, monolithic "black boxes." When something fails, only an on-call programmer can identify even the most basic problems. Once identified, the only recourse is usually complex manual cleanup and wholesale rerunning of successfully completed work. Changes to scripts become new development projects instead of simple configuration changes.

Finally, there is the impact of manual changes as part of the application deployment. This one problem may be the representative goal of DevOps: to eliminate manual changes that lead to failures in promotion—and worse, failure in operation

## IS THIS UNIQUE?

There are other technologies where the level of developer skills may be deficient even though the technology is commonly used and may even be considered fundamental.

One such common technology is relational databases. Many DBAs find the database schema and SQL structure submitted by developers to be woefully deficient. Basic mistakes abound where randomly-queried tables are not indexed, or a SQL query returns millions of rows when only a small subset is required. Such errors don't surface in testing where volume is small, but become massive bottlenecks in production with fully populated, large databases.

Although these problems do have significant impact, basic DBA skills are generally considered mandatory for developers. That major distinction is absent when it comes to workload automation.

## WHAT IS NEEDED?

So how hard is this problem to fix? As it turns out, pretty hard because it requires a change in attitude as well as functionality and tooling.

### The Attitude

It seems the view that workload automation belongs to IT operations is almost universal. Even organizations that make WLA available to developers do not expect them to produce production-ready workflows.

If the job definitions and WLA elements that are built by developers are not production ready, there is little point to attaching them as artifacts to the deployment process and no reason to expose them in DevOps.

To change this situation, WLA needs to be integrated with the set of developer tools. And in order for that to happen, developers must see a benefit to going out of their way to learn something new. That benefit could include a shorter development effort or a higher quality deliverable or an attractive skill to add to their resume—although it may take the industry a while to recognize the value of this new skill.

### The Functionality

Today, most WLA solutions are aimed at IT schedulers and operators. The architectures, user interfaces, and terminologies are all oriented towards expert or power users. Building workflows requires not only deep scheduling domain expertise and knowledge of the specific product used, but also the standards and conventions unique to the organization. The entire process is monolithic and doesn't even contemplate collaboration. In order to draw developers in, a different approach is required.

First, it must be acknowledged that different levels of engagement will be the norm. Just like some developers are whizzes with relational databases and others are novices, WLA must accommodate both power and casual users.

Second, the solution must accommodate organizational standards. This is particularly crucial in the world of outsourcing, offshoring, and extreme staff turnover. It's enough of a challenge to become comfortable with a WLA that may be broadly used throughout the developer community. It is outright optimism to expect highly mobile developers to know anything about the local standards of the organization, especially since it's common for complex environments to have equally complex standards and naming conventions.

Third, collaboration must be fundamental. No matter how tightly organizations embrace DevOps, there will remain for the foreseeable future two separate groups: Dev and Ops, which need to communicate and collaborate.

Fourth, and perhaps most obviously, the solution must be a high quality "management" application. That means it must be intuitive and easy to use, fully automate the managed function (in this case building workflows), and contain comprehensive audit and logging facilities to support high levels of governance.

## THE BRAVE NEW WORLD

Once developers have embraced workload automation as part of their world and have started generating workflows, the organization is ready to add WLA to DevOps.

Ideally, the DevOps solution is WLA-aware and can ingest workflows as a component of the application profile. If your DevOps flavor is not there, WLA can join as yet another artifact that is part of the deployed application.

**A possible series of steps follows:**

- Application is developed
- Workflows are developed
- Testing is performed
- The application code is checked into some source control (Apache™ Subversion®, IBM® Rational® ClearCase®, Git™, etc.)
- The workflows are exported into some format (usually XML), and that export is either checked in to the same repository or added to the application package just like any other type of config or properties file
- Your application release solution promotes the application package

This scenario can work well assuming the workflow definitions have already been built in the proper format and are fully formed (or close to it) and ready for subsequent environments.

## WHAT YOU GET

Organizations that have successfully integrated WLA into their software development lifecycle have seen substantial benefits all the way down the chain. Developers finish their development projects faster because WLA significantly reduces scripting and facilitates testing. In fact, the more complex the application, the greater the benefits of applying WLA to its testing.

The quality of the delivered application is higher because WLA provides comprehensive operations capabilities like monitoring for success or failure, automatic opening of incidents, ability to send email notifications, and codifying service levels that drive predictive run time analysis and early warning of potential breaches.

Once applications get into production, fewer errors occur because workflow creation has been automated to avoid the errors that result from manual procedures. Automated adherence to site standards eliminates failures resulting from security and similar administrative controls requiring a level of conformance difficult to achieve manually.

And finally, in the context of this paper, adding WLA early in the development process ensures that the benefits of DevOps accrue for all applications, including your batch services. Considering that your paycheck (or payroll deposit) is most likely generated by workload automation, you may find that reassuring.

### Author information

Joe Goldberg is an IT professional with over 35 years of experience in the design, development and implementation of enterprise solutions to Global 2000 organizations. Joe is a workload automation expert. He is very active in helping BMC products leverage new technology to deliver market-leading solutions.

You can reach him at:
BMC Communities: https://communities.bmc.com/people/jogoldbe/content
Twitter: https://twitter.com/goldbergjoe