Liquibase

MedImpact's Journey to Database Change Automation

MedImpact

CUSTOMER'S BUSINESS

MedImpact Healthcare Systems is the nation's largest independent, trend-focused pharmacy benefit manager (PBM). MedImpact works with health plans, self-funded employers, and government entities to help reduce prescription drug costs, improve member health, and increase medication adherence and safety. Their business model is unique, prioritizing member experience and inherent flexibility to align with plan sponsors in ways that other PBMs cannot.

OVERALL CHALLENGE

MedImpact's value proposition dictates unique and agile views of data to support the reporting and analytics behind their service offerings. In order to provide these views, data objects need to be created and modified quickly and reliably. With over 55 million consumers counting on their services, high availability is crucial. MedImpact also recently adopted Agile software development methodologies to be as responsive as possible to their customers' needs.

SOLUTION

Database change and deployment automation. With all of these challenges, MedImpact needed reliable database change management practices that eliminates errors.

For 30 years, MedImpact has had a single mission: To help make pharmacy benefits affordable, understandable, and honest. As the nation's largest privately held pharmacy benefit manager (PBM), MedImpact's technology serves more than 55 million consumers around the world.

In order for MedImpact to deliver on their mission, they found they needed to update their technology and software development methodology. "Many—if not most—organizations are guilty of treating database code differently than other code stacks," said Mabuti Ng'andu Director of IT, Data Management, Database & Middleware at MedImpact. "Many organizations are disciplined about the process of checking app code into Git and checking it out in an automated fashion as they push it through their various environments. Not so much with databases." There are a couple of reasons for this:

1. Very few organizations have dedicated database developers, unlike they do for other specialized code for Java, Python, Swift, etc.
2. Databases and database code have not traditionally been easily managed by commonly used software deployment processes or toolsets.

These differences can lead to occasional deployment mishaps resulting in slow deployments or costly mistakes that are difficult to roll back. Ironically, these mishaps, though costly, are simply viewed as process failures due to resource or team discipline challenges. This makes the case to justify an investment in database deployment automation a tough task.

Liquibase

Why database change and deployment automation is important to MedImpact

- MedImpact's value proposition dictates unique and agile views of data to support the reporting and analytics behind their service offerings
- Data objects need to be created and/or modified quickly, with reliable quality
- MedImpact guarantees high availability and quality-related performance to customers
- MedImpact began the process of adopting Agile software development methodologies to be even more responsive to market needs

All of the above points mandate reliable database change management practices that minimize or eliminate errors.

> "Our unique set of challenges requires us always to be on top of the management of change within our data landscape. We have to have reliable data change management practices."
>
> – Mabuti Ng'andu, Director of IT, Data Management, Database & Middleware

Database Change and Deployment Challenges

MedImpact's journey to improve database deployment throughput and quality required people, updated processes, and tooling changes. It also required a justification process to enable the investment. Here's an overview of the team's challenges and how they overcame them to get to implementation.

There were three types of challenges to overcome on their journey: process, team, and outcome challenges.

Process Challenges

Like most companies, MedImpact did not treat database deployments with the same rigor as software deployments. Although database source code lived in Git, the traceability of the code from checkout to deployment was manual and limited. Database deployment components were not part of the core build package. In fact, at times database changes were deployed days or even weeks before the software deployments.

Deployment processes were mostly manual, save for some scripted verifications/validations. This caused a number of problems:

- Long, labor-intensive deployment processes (not Agile!)
- Inconsistent deployments across the enterprise
- Incomplete deployments within databases
- Limited ability to audit when changes occurred in a given environment
- Limited visibility into, and limited ability to review, deployment failure points
- Labor-intensive, manual validation/verification of all of the above

Team Challenges

One major challenge, from a people-perspective, is that three sets of teams had the ability to deploy DML or DDL and each team had their own set of deployment processes:

- The DBA team deployed all DDL with the exception of application-specific PL/SQL
- Software engineering teams deployed application-specific DDL via configuration management processes
- The DBA team, software engineering teams and even some business teams could deploy DML through a manually controlled configuration management processes.

This variation of deployment processes MedImpact had limited the visibility and traceability of deployments.

Outcome Challenges

Even though MedImpact employed rigorous manual verifications (unit, QA, test and validation, as well as UAT), they had a few instances of deployment failure. In those cases, additional challenges were introduced, including:

- Long, arduous debugging processes for the DBA, Dev, QA and testing & validation (business) teams
- Unexpected post-deployment incidents/problems in QA, UAT and production environments
- Longer software delivery timelines
- More rework

The Solution: Database Change and Deployment Automation

There were two primary objectives for MedImpact's Database Deployment Automation initiative:

1. Create a process that provided a consistent framework for all database deployments including DDL, PL/SQL and DML to all non-COTS Oracle databases using the same rigor for DB deployments that was expected in software deployments
2. Identify and acquire a tool that would streamline database deployments with the following capabilities:
    - Enforce automated deployments from the source code repository (Git)
    - Deploy predictably, repeatably, and consistently to individual or group targets (one QA database, or all QA databases, etc.)
    - Log all deployment script outcomes for immediate or future review
    - Integrate with standard DevOps tools, including the Atlassian stack
    - Audit deployments

Getting Executive Buy-in

Now that the MedImpact team knew what they were looking for, it was time to present it to executives for approval. They thought, "this is the obvious answer to our problems. This is a no-brainer." Well, making this kind of investment isn't always easy to make to C-suite approvers.

> We thought it would be really easy and that it was self-explanatory. Benefits are clear and obvious; If we invest in this set of tools, we'll immediately improve deployment quality and we'll have a much faster, more agile turnaround. But Executive management doesn't work that way."
>
> **– Mabuti Ng'andu, Director of IT, Data Management, Database & Middleware**

First Attempt at Executive Justification

The first-round justification described the potential value of adding database deployment automation.

- Improved deployment quality
- More Agile turnaround

These points weren't enough to sway the executive team. Mabuti and his team were asked why this same value couldn't be achieved with more efficient processes and better 'people management'. It's not always inherently obvious to executive

decision-makers that you can only manage fatigue to a point. Manual processes always carry a certain error rate (to err is human, after all) and throwing more people at a broken process doesn't make it better.

> "This takes a significant amount of effort, but I strongly encourage anyone who is embarking on this journey to spend the time here to make the justification for the investment. If you do not do this, you will likely spend a lot more time answering questions instead of just being able to show the value."
>
> **– Mabuti Ng'andu, Director of IT, Data Management, Database & Middleware**

Second Attempt at Executive Justification

Mabuti had to go back to the drawing board to get executive buy-in. They needed to make the value clearer from an executive standpoint. His team spent two months documenting and analyzing their current process so they could put a number on the reduction in overall deployment costs so his team could demonstrate the potential value of an automated database change and deployment tool.

- Improved deployment quality
- More Agile turnaround
- Reduction in deployment costs
- Reduction in rework, associated costs, and schedule impact
- Seamless integration with planned CI/CD pipelines

MedImpact's second justification summary included the following components:

- An opportunity cost calculation including:
  - Labor cost of slow deployments through all environments (DEV, QA, PRD)
  - Labor cost of manual verification
  - Cost of deployment-related rework
- A roadmap to achieving automated deployment value, including:
  - Tool evaluation and selection process
  - Pilot (POV) of tool to prove it could deliver on value
  - Alignment on success criteria
  - Reference calls with enterprises who have implemented selected tool

Why MedImpact Chose Liquibase

MedImpact evaluated four tools and agreed that Liquibase was the best fit for their needs. After talking to the companies, downloading data sheets, attending demos, and talking to Gartner, they narrowed down their selections. The team executed a Proof of Value, meeting all success criteria. All executive leaders agreed to make the investment in Liquibase.

The Implementation Process with Liquibase

The MedImpact team decided to pilot the Liquibase implementation on two applications: one that was simple and one that was a bit more complex so that they could modify their database deployment processes and train people.

Today, MedImpact is integrating Liquibase into its broader continuous delivery ecosystem.

After successfully piloting the two applications, MedImpact updated their database deployment process. There

are currently 17 applications using Liquibase and they will have 60 applications total. They are also integrating Liquibase in their broader continuous deployment system with one team testing out a full CI/CD process that will roll out to the rest of the organization. In the end, they expect 100% of software deployments at MedImpact to be completely automated with Liquibase at the center of their automated CI/CD process for database change and deployment.

Mabuti Ng'andu, **Director of IT at MedImpact Healthcare Systems**, is responsible for the installation, configuration, availability and operational management of key database and middleware technologies. He is also responsible for the configuration and software delivery change management into databases, and their associated release automation capabilities. Starting his career as an Oracle developer, Mabuti transitioned into database administration and solutions architecture before finally taking on leadership roles. He has worked for a variety of private and public organizations, small startups and fortune 100 companies including Abbott Laboratories where he led database and infrastructure teams. Mabuti holds a bachelor's degree in Computer Information Systems from La Sierra University.

Key Takeaways from Mabuti Ng'andu Director of IT, Data Management, Database & Middleware at MedImpact

1. Be very clear on what your objectives are.
2. Invest time in articulating the value that automated database deployments provide. It may be obvious to the technical team, but it must still be well articulated to executive leaders.
3. If you are not already actively Agile, plan time to work through process questions (e.g., Who owns deployments in which environments? What is the process to trigger a deployment? Who needs to approve it? etc.)
4. Solicit input from and stay well aligned with all deployment stakeholders—DevOps, software engineering, database administration/data management, configuration management, release management, etc.
5. Never forget that this is a CHANGE MANAGEMENT activity.
    a) Identify the advocates and utilize them to drive change.
    b) Communicate 7 times, 7 ways.
    c) Prioritize those who understand the vision and the rest will follow.