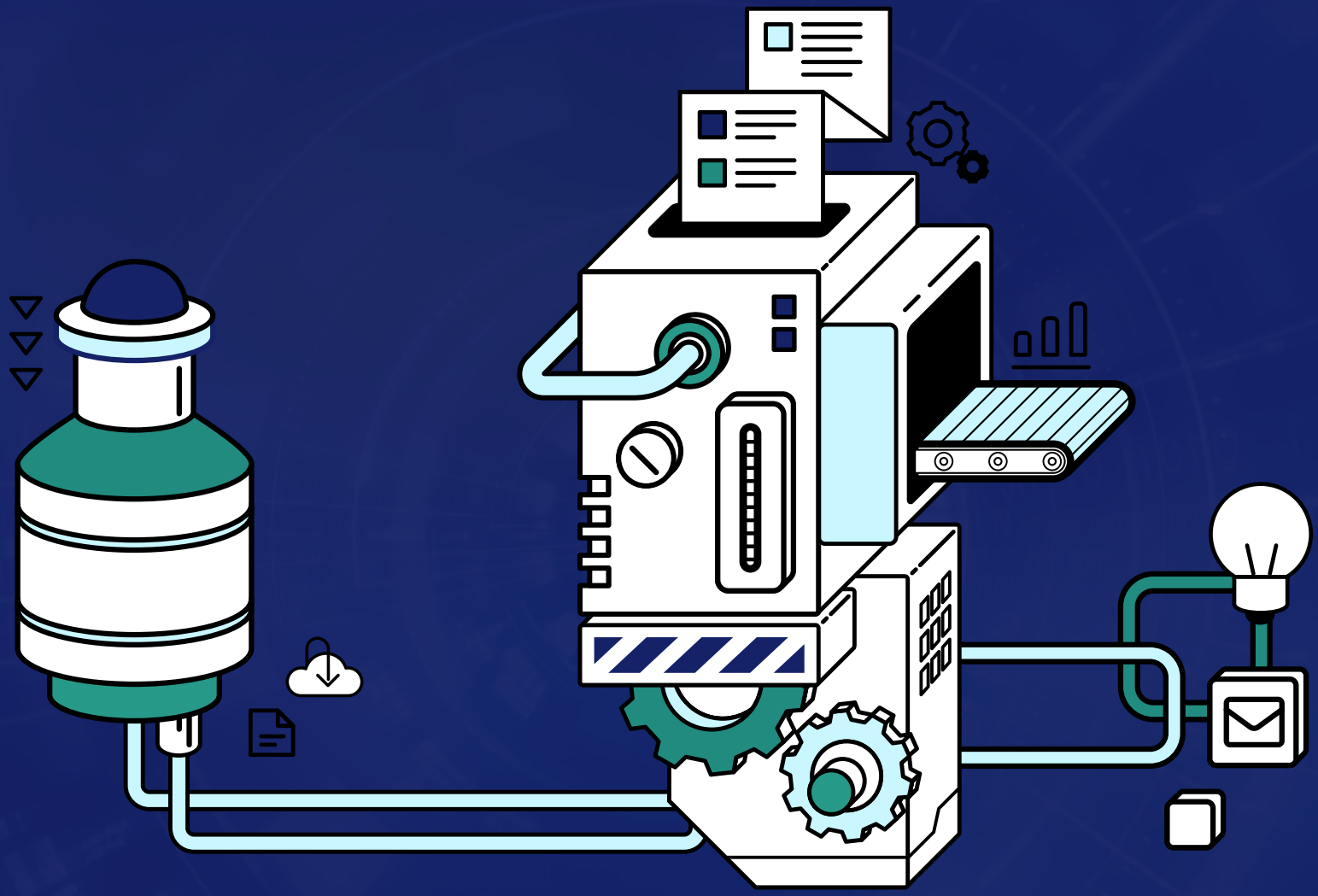
In Database Automation we Trust



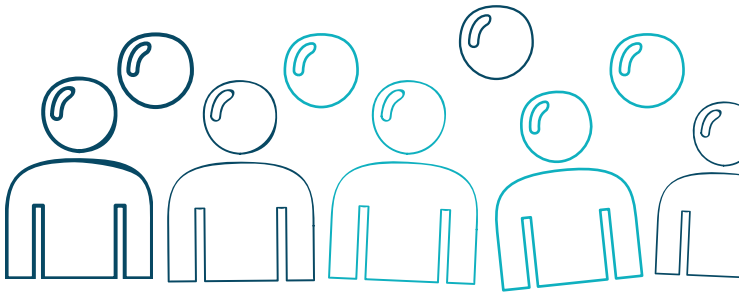
DBmaestro
DevOps for Database

Executive Summary

Figures quoted in this eBook are based on a survey of over 450 IT professionals. The survey found that, although companies are increasingly adopting Continuous Delivery (CD) for software development, adoption for the database is lagging behind. This is due primarily to a mistrust in, and lack of awareness of automation.

Continuous Delivery is strong and continues to grow. 67% of respondents replied that they use CD for their application. This shows growth over previous surveys, and is remarkable considering that the concept of CD was only first described in a 2010 book co-authored by Jez Humble and David Farley. The need to increase productivity, achieve quicker time to market, and reduce risk are the top reasons why organizations are adopting CD.

Mistrust of automation is holding back CD for database. When asked if they use continuous practices for their database changes, 34% answered in the affirmative. The largest group, 36%, cited mistrust in automation for the database as the number one barrier for adopting CD. The number two barrier listed was lack of awareness. Just 3% said management opposition was the biggest barrier.



Perception doesn't always meet reality. Despite 34% responding that they implement continuous processes for the database, when asked how their organization deploys database changes, only 13% had an automated process in place - a prerequisite for database continuous delivery.

Dealing with mistrust in Database Automation. Leveraging database enforced change management and baseline aware deployments can make the difference and help combat mistrust. Ensuring all database changes are properly managed, safely deploying them with no risk of code over-rides, and being able to issue automation-stopping red-flags if needed will make the database a safe player in the continuous delivery process.

Continuous Delivery is a cross-organizational effort: 46% of respondents said DBAs should be responsible for adoption, while 40% said it is the responsibility of management. This defies a commonly-held belief that either the developer or top executives alone are responsible for adopting new development methods.

Benefits of implementing the database into continuous processes: Organizations can boost overall productivity by 15%, reduce production incidents by 20%, and obtain the ability to make the database adherent to audit and compliance regulations.

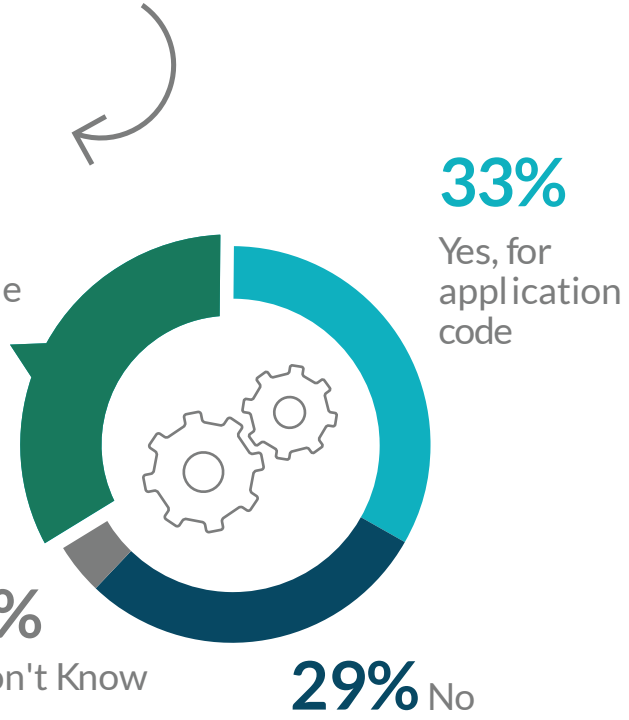
The Quest for Competitive Advantage Drives Organizations to Continuous Delivery

Business needs are the most significant driver of change. The ability to do more with less and deliver sooner is what differentiates leading and successful companies from the rest.

When a competitor delivers relevant features, faster and with higher quality, you're eventually going to lose market share. Investing in sales and marketing campaigns to compensate for your product is expensive and unreliable. You might find that customers are moving to the superior product anyway.

Continuous Delivery is a concept in software and database development that calls for a well-defined process, feedback loop, and automation. Through automation, you can dynamically perform the same task over and over again, without human shortcuts or errors, while ensuring you get the desired results every time. Automation allows you to break down the complex parts of a task into smaller, simpler parts. It also saves you time, because your team members will be free to work on other tasks.

Does your development organization use a continuous delivery process?



Database Automation: Gap Between Perception and Reality

Continuous delivery has continued to go mainstream, with 67% implementing automation in their development process. When asked about implementing continuous process in the database, that number halves to 34%.

One third of respondents practicing continuous delivery for the database may be lower than the 67% who practice CD for the application, but it would lead us to believe that a large and growing percentage of organizations are adopting the same best practices they already use for the application - to the database.

But when asked about the specifics of the automatic processes organizations have in place, a different picture emerged.

The survey asked participants to choose which of four options best described the way the respondents deploy database changes. 87% chose options that involve manual processes including 37% who replied that they build scripts by hand and then fix or tweak them regularly, 26% who use compare and sync tools to identify changes, but have to review and fix the results because they can't always trust them to automatically deploy correctly and 24% who build scripts manually.

Only 13% responded that they use compare & sync tools to identify changes between environments, and automatically deploy them with no issue.

Thus, despite 1 in 3 claiming that they have automated continuous process in place for the database, only 13% are actually leveraging automation.

Which of these Options Best Describes the Way you Deploy Database Changes ?

37% We build scripts by hand and must maintain, fix or tweak them regularly as part of a deployment process

26% We Use compare & Sync tools to identify changes, but have to review and fix the results as we can't always trust them to automatically deploy correctly

24% We build scripts manually

13% We use compare & Sync tools to identify changes between environments, and automatically deploy them with no issues

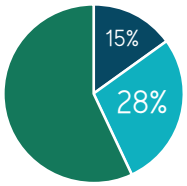
Developers See High Risk in Manual Processes

When asked to rate the risk level of five non-automated, database development and deployment processes, respondents found significant risks.

57% considered it high risk for the database to be out of sync with the version control repository (26% considered it medium risk), 57% said it was a high risk to override changes when integrating valid changes (28% said medium) and 44% said it was high risk to integrate changes directly into the QA (28% said medium).

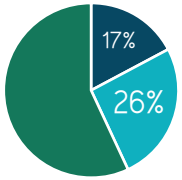
The inability to trust the validity of their version control repository, overriding other people's changes and breaking the deployment order are exactly the reasons why developers and DBAs have a hard time trusting their change management process. Out of date changes being deployed over more recent ones when automating the compare & sync tools results, is a common reason why these tools are not used in the continuous delivery pipeline.

Rate the Level of Risk Associated with the Following Database Deployment/ Development Practices



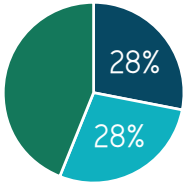
57%

Override others' Changes when Integrating Valid Changes



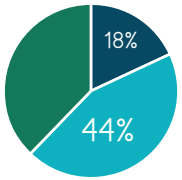
57%

Database is Out of Sync with the Version Control Repository



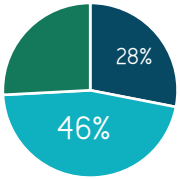
44%

Introduce Changes Directly to the QA Environment



38%

Test Urgent Hot-Fixes in Pre-Production



26%

Verify each Database Object vs. its Version Control Revision to Make Sure it is Up-to-Date

Why Adopt Continuous Delivery

Organizations recognize that developing better code is of critical importance to the organization, with an immediate reduction in costs being of lesser significance.

Why Companies adopt Continuous Delivery?

29%

Increase Productivity

24%

Reduce Risk

23%

Quicker Time to Market

17%

Increase Quality

7%

Reduce Cost

The needs to increase productivity (29%), achieve quicker time to market (23%), and reduce risk (24%) are the top reasons why organizations are adopting Continuous Delivery.

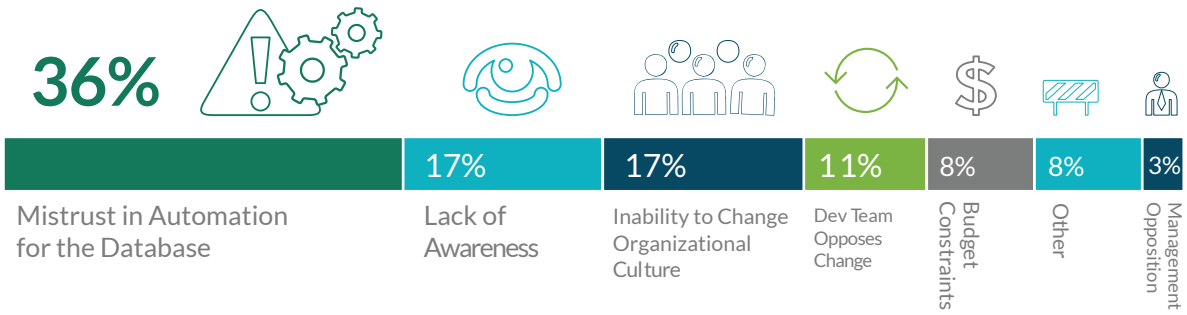
Increasing quality (17%) and reducing costs (7%) are secondary drivers.

Higher Concern about Automation for the Database

Organizations trust an automated process to accelerate the release of software. They are seeking minimum manual intervention. Automation is the key for implementation of Continuous Delivery. Of those who are practicing CD with their application, only a fraction also do so with their database.

When it comes to the database, there is **a lack of trust in automation, which is holding many back from implementing Continuous Delivery for the database.** Over one-third of companies (36%) list it as the biggest barrier to adoption.

What is the biggest barrier to adopting Continuous Delivery for the Database?



This mistrust in automation, along with a lack of awareness (17%) and an inability to change organizational culture (17%) are preventing many organizations from adopting CD for the database, as they do for the application.

The fact that organizations trust automation in the database less than they do in the application, highlights an important point. While enterprises recognize how critical the database is, and realize that any error in database development can result in massive loses, they also do not trust or lack awareness of the tools currently available to automate database deployments.

Reasons for Mistrust are Real

"It was **difficult to track who made a change to a database object** and what change they made."

(Working around and failing to enforce usage of file-based version control.)

Sr. DBA @ Large USA Bank

"We recently had **a disaster.** The script in the version control was not updated, and when executed in production, ran the **wrong revision.** That cost tens of **thousands of $s**"

(An out-of-process update to QA that was not properly tracked.)

DBA @ Algo Trading Company

"It took **hours to get releases working.** Some changes were not documented and left out. We actually preferred crashes in integration. It is much worse when something works, but works wrong in production."

(Manual and error-prone releases. Change control and change scope issues.)

Sr. R&D Manager @ Credit Card Company

"We had **multiple releases to production every day.** That is one release a week with multiple **follow up fixes**, and yet more fixes."

(Code overrides, partial versions, wrong versions - all pushed to production.)

CTO @ Credit Card Company

Dealing with Mistrust in Database Automation

In order to take a database into proper automation, and to practice safe Continuous Delivery, the following factors must be considered:

1. Proper database version control, dealing with each database's unique challenges (structure, code, and content), while enforcing a single work process. This prevents any out of process changes, code overrides, or incomplete updates.

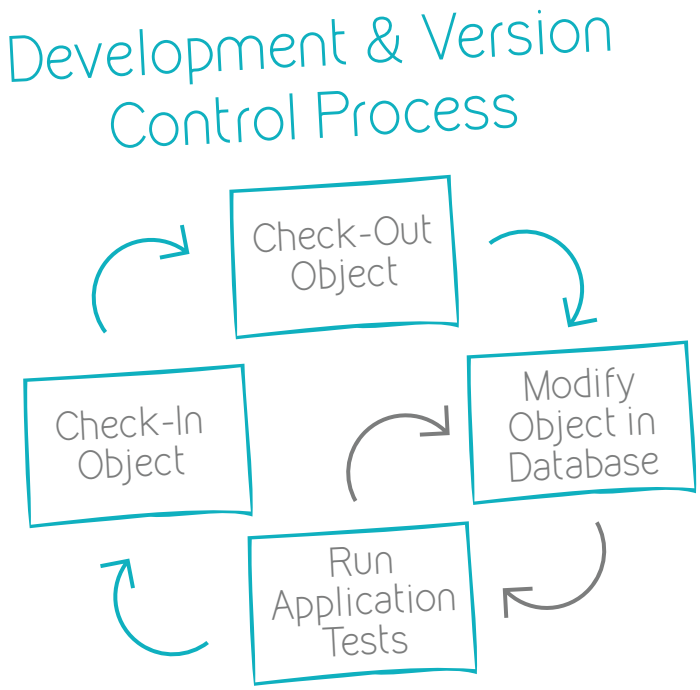
2. Leverage proven version control best practices (check-in and check-out changes, etc.) for complete information about who was doing what, when, and why. Making sure changes are perfectly documented is the basis for successfully deploying them later.

3. Synergy with task-based development enables correlation between each version control change and a change request or a trouble ticket. This coordinates task-based deployments, partial deployments, and last minute scope changes of code and database.

4. Ensure configuration management and consistency, so every development environment, branch, trunk, sandbox, and testing or production environment follows the same structure and matching status. Any deviation and difference must be well accounted for.

5. Use API interfaces to deal with automation of deployment processes, providing repeatable results every single time. Even the most sophisticated solution becomes cumbersome if you have to use the UI to perform the same task over and over again.

6. Utilize reliable impact analysis for deployment generators. It must leverage baseline-aware decisions based on version control knowledge and be capable of dealing with conflicts and merges of database code, as well as cross updates from other teams, while also ignoring wrong code overrides and dealing with out of process changes.



Simple Compare & Sync

Source vs. Target	Action
=	No Action
≠	?

Baseline Aware Deployment

Source vs. Baseline	Target vs. Baseline	Action
=	=	No Action
≠	=	Deploy Change
=	≠	Protect Target
≠	≠	Merge Changes

7. Provide an automatically-generated development package, on the fly, to deal with deploying projects of any scope, from multi-schema mega-updates, to a single task-based change and its dependent objects.

8. Leverage 'labels' for before and after deployment of changes to act as a safety net, so quick and easy roll-backs are always close at hand.

9. Ensure full integration with other systems (ALM, change management/ trouble tickets, build servers, and release automation solutions).

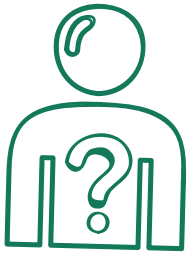
Implementing a solution to deal with these challenges, would enable a company to practice proper and safe database Continuous Delivery, which would be easy to integrate with the rest of change and release processes, to achieve complete end-to-end Continuous Delivery.

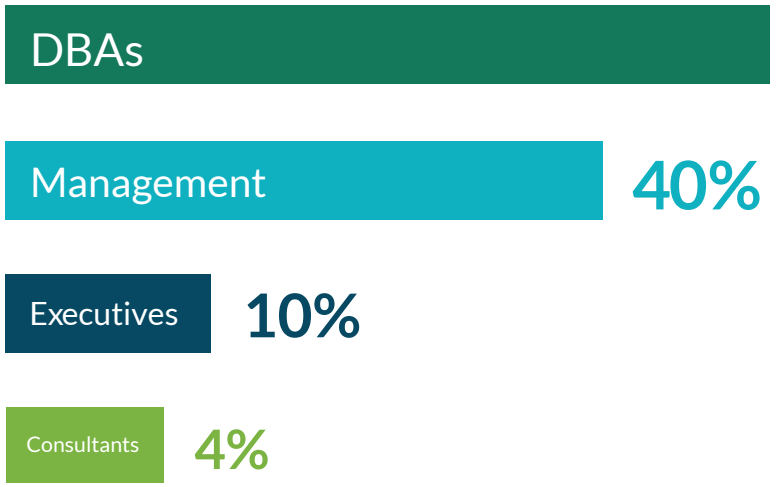
Continuous Delivery Must Come From All Levels

Continuous Delivery is a cross-organizational effort to be embraced by all parts of the organization. When it comes to the database, 46% said DBAs should be responsible for adopting CD for the database, while 40% said it is the responsibility of management. Therefore, in order for CD to be successful, it must be accepted throughout the organization.

As the entire industry is moving to Agile development and Continuous Delivery, DBAs should spearhead the adoption of database continuous processes, building these procedures with controls to their liking and leveraging their experience to achieve a trustworthy process.

Responsiblity for the Adoption of Continuous Delivery for the Database





Benefits of Implementing Continuous Processes for the Database

"Focusing on changes, rather than managing changes and dealing with re-work, boosted overall productivity of 200-300 developers. We estimate we were able to do 15% more with the same resources.

We went from several fix-centric deployments a day, to one feature-centric deployment a week.

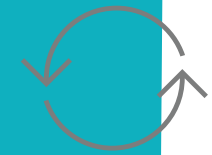
The amount of incidents in production has declined as well. We had 20% less incidents.

We have to follow regulatory requirements like SOX and ITIL, and a change management approach with change management enforcement can help. You have to do it anyway, so do it automatically and efficiently.

Change management in production itself is regulatory required (ITIL). But you cannot ensure it without managing the whole process starting at development.

We have to comply with regulations - but the business benefits from it."

CIO @ Credit Card Company



Conclusions

Continuous Delivery continues to grow, and over the past few years has become the norm for a majority of organizations. These organizations are implementing it for their code, and a large number are also adopting it for their database.

CD for the database is fully adopted at a fraction of the rate of CD for the application. This is because there is a lack of awareness about the relatively new ability to automate database changes with Database Enforced Change Management. This leads to a lack of trust in automating the database, which is such a critical infrastructure in the organization.

Database professionals recognize that database continuous delivery is possible, and vitally important, and as enterprises continue to adopt CD for the database, the need to increase productivity and reduce time to market will drive more companies to implement CD in their database. This implementation will be driven by all levels of the organization, including DBAs, management, and executives.

About DBmaestro

DBmaestro is a pioneer and a leading solution provider for database DevOps. Its flagship product, DBmaestro DevOps Platform, enables Agile development and Continuous Integration and Delivery for the database.

The Platform supports streamlining of development process management and enforcing change policy practices.

The solution empowers agile team collaboration while fostering regulatory compliance and governance.

With DBmaestro, organizations can facilitate DevOps for database by executing deployment automation, enhancing and reinforcing security as well as mitigating risk.

DBmaestro's solutions are deployed at many Fortune 500 companies.

Connect with DBmaestro

Read our blog: http://www.dbmaestro.com/blog/

Follow us on Twitter: https://twitter.com/dbMaestro

Follow us on LinkedIn: https://www.linkedin.com/company/dbmaestro

Headquarters 300 Baker Avenue, suite 300, Concord, MA 01742, USA +1 978.405.3368

EMEA Headquarters 59-60 Thames St, Windsor, Berkshire, SL4 1TX, UK +44 (0)1753 968 358

DBmaestro
DevOps for Database

Follow Us: