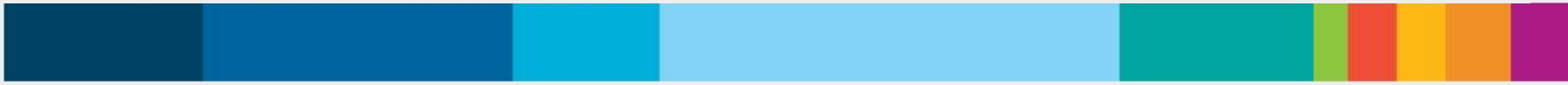
IBM Brighter Blue

Introduction to DevOps



IBM

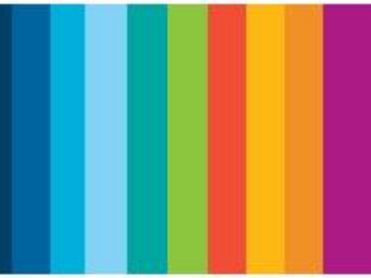
Module Overview

This module will introduce you to the key concepts of DevOps and should take about 90 minutes to complete.

At the end of this module, you should be able to:

- Narrate solution production using traditional development methods
- Describe DevOps
- Identify characteristics of DevOps team
- Outline the DevOps life cycle
- Give evidence of the impact of DevOps
- Explain the fours stages of adoption
  1. Steer
  2. Develop / Test
  3. Deploy
  4. Operate
- List the tools of DevOps and their purpose

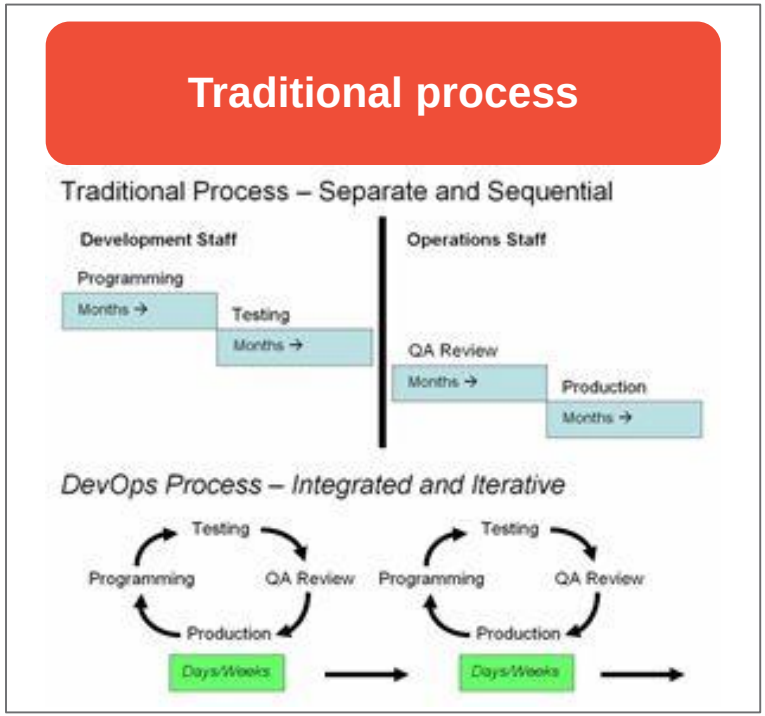
01 What is DevOps?



Solution production was earlier performed by traditional development methods

In the beginning, Development and Operations were typically performed by the same person. As time went on, that changed, and Development and Operations separated.

For the purposes of solution production, the traditional model was used. In this model, Development and Operations were separate, and did not interact until the end of the process.



A new project would take between 6 and 18 months before release 1.0 is in production!

Challenges with the traditional method

Apart from taking a very long time these are some of the other challenges you may face while working on projects using traditional development.

Your customers find major defects.
Major defects take a long time to fix.

You cannot do anything until you have everything.
Legacy model leads to unique infrastructure dependencies.

People do not talk to each other.
This leads to broken processes and overbearing governance.

Even if you are “agile,” the delivery process is still broken.
Going live still takes as long as ever.

Any problems lead to finger pointing.
Development and Operations unable to operate as one team.

Software failures often impact the bottom line

What the client's felt

We need to deliver our software faster and more often to meet today's needs of the business and our customers!

We must improve our quality to stay competitive.

We need to reduce our costs of software delivery,

We always deliver late when the requirements have lost their relevance.

41% clients experienced **development delays.**

34% clients experienced **deployment delays.**

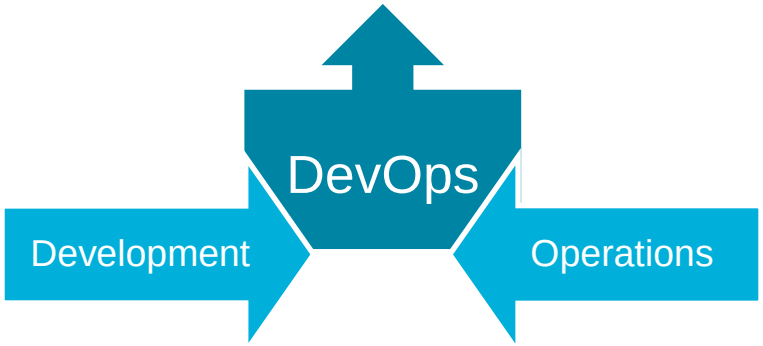
45% clients experienced **production delays.**

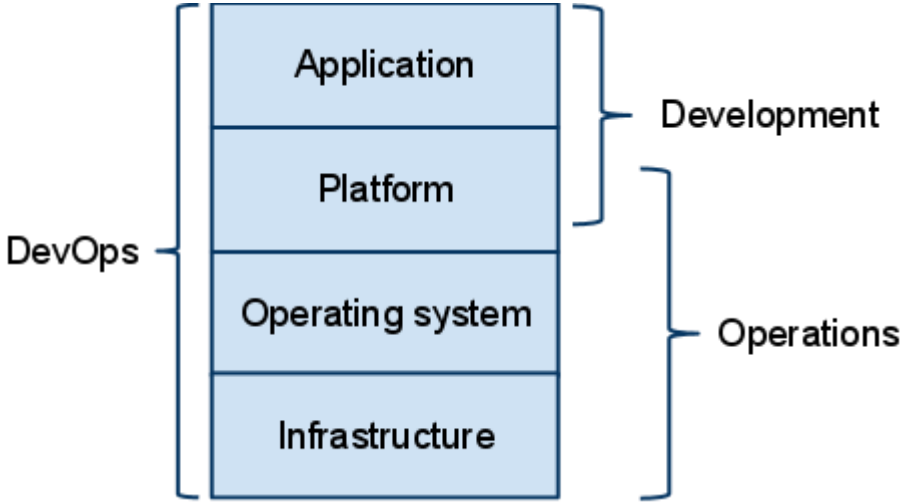
IBM Business Value Study 2013

DevOps was introduced

DevOps is a **software development method** that emphasizes on collaboration between Software Developers (Development) and Information Technology professionals (Operations).

The term “DevOps” is a contraction of the terms Development (Dev) and Operations (Ops) in the context of common tools used to deliver software solutions.





How does DevOps work?

Business function owners, developers, and operations personnel—all are required in the DevOps process.

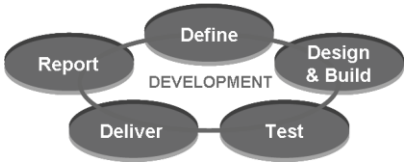
The process requires the team to:

- Develop and test against production-like systems
- Deploy with repeatable, reliable processes
- Monitor and validate operational quality

DevOps change is as much a **culture** change as it is a **technology** change.

DevOps is a set of processes, methods, and tools

Process



Management
Accountability
Integration

Technology

Development Tools

Automation
Integration
Visibility

Operations Tools

People

Experience
Expertise
Culture
Collaboration
Communication

DevOps brings together Development and Operations to create solutions that are:

- Rapid
- Repeatable
- Consistent
- Flexible
- Scalable

DevOps capabilities

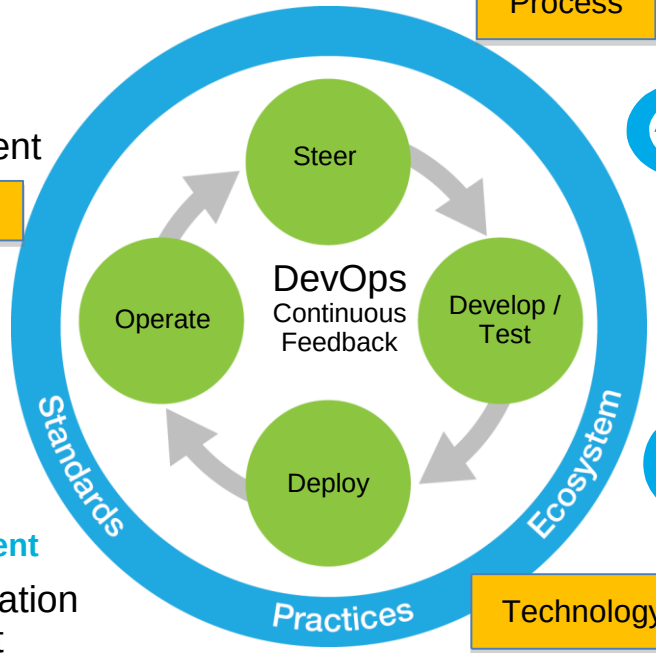
DevOps is an enterprise capability for continuous software delivery that enables clients to seize market opportunities and reduce time to customer feedback.

Continuous business planning

- Requirements management
- Project and portfolio management

Continuous monitoring

- IT operations analytics
- Application performance management
- Operational problem management



Collaborative development

- Collaborative development life cycle
- Omni-channel development
- Team ownership

Continuous customer feedback and optimization

Customer experience management

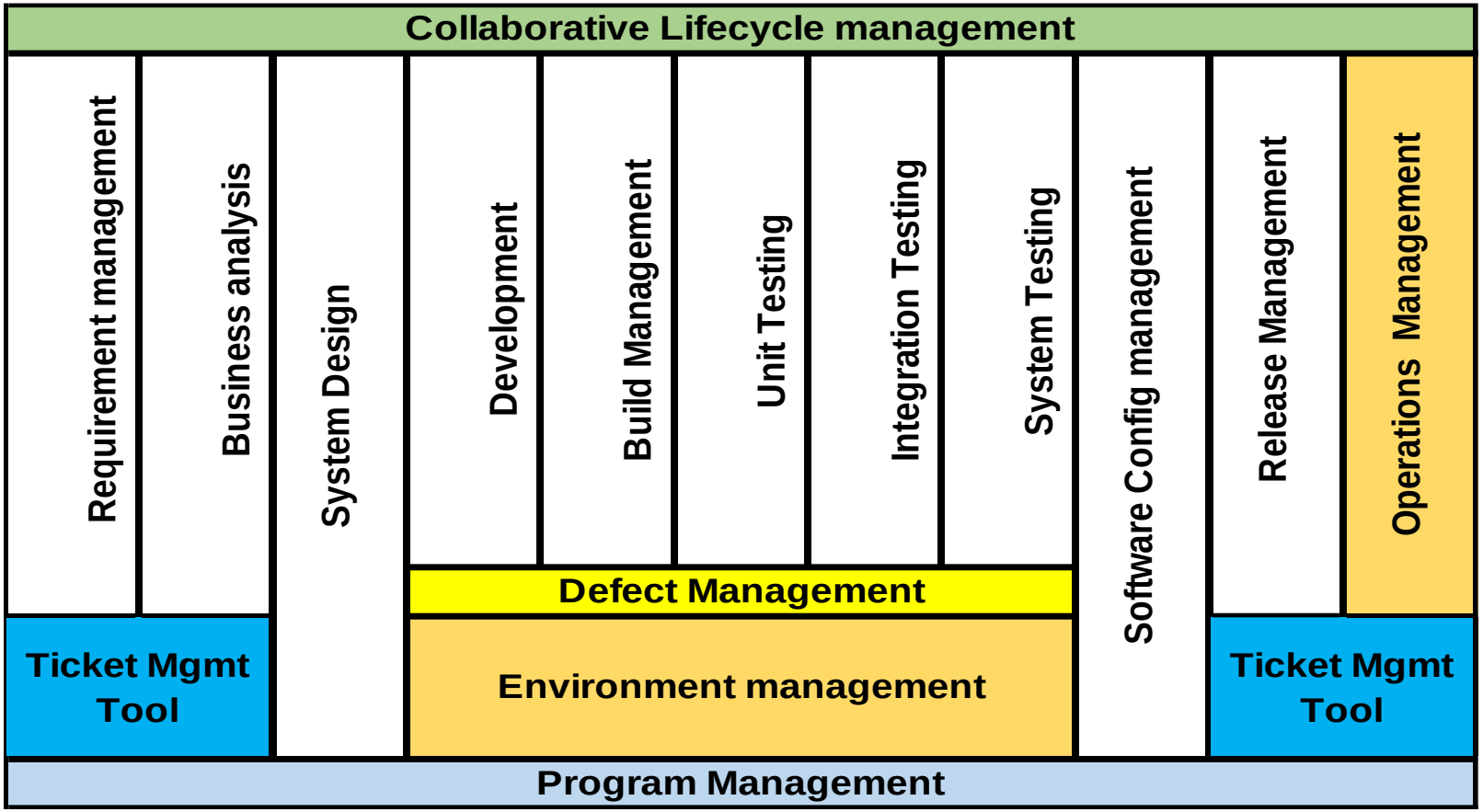
Continuous testing

- Service virtualization
- Software quality management
- Multi-platform automated testing
- Security vulnerability testing
- Test data management

Continuous release and deployment

- Application release automation
- Hybrid cloud management
- Hybrid cloud application integration
- Software life cycle patterns

DevOps is an end to end tooling blueprint



Understanding DevOps further

What DevOps is

DevOps is:

- Faster reach to market
- Applying Lean principles to software engineering
- Catching and fixing defects early in the cycle
- Helping continuous delivery
- Cultural change at every level in the organization
- Using a correct combination of tools in the SDLC
- Removing the wait period and delays caused by manual processes and reliance on historical knowledge
- Enabling the feedback loop for continuous learning from customer
- A blend of processes, methods, and tools that can be effectively adopted by the development and operations teams

What DevOps is not

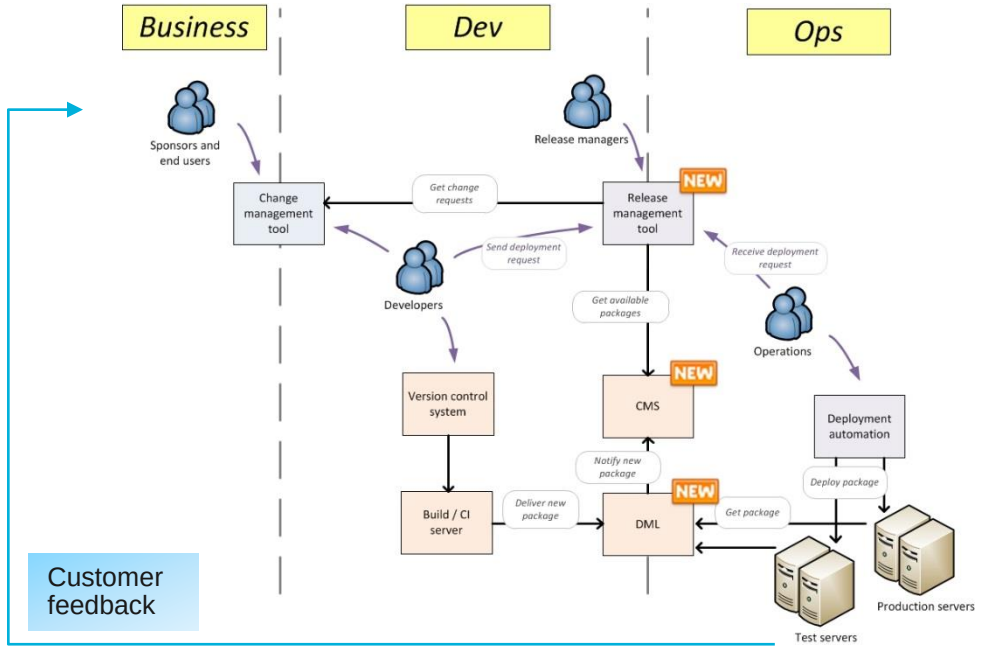
DevOps is not:

- A magic wand
- A product
- A formula/set of fixed steps
- That the DevOps team owns it and should deliver end to end
- Another process of delivery excellence
- Only limited to Build Release tools
- A set of tools to be deployed into the project

Characteristics of teams who practice DevOps

Teams can also amplify feedback loops. This means that:

- Development team can adjust its project plans or priorities
- Production team can enhance the production environments
- Business can modify its release plans



For DevOps teams, collaboration is the key

Teams who practice DevOps:

- Ensure better communication between those who create and those who operate—these could be the same people in some cases.
- Possess a reduced fear of breaking the build / deployment / environment
- Benefit by experiencing instances of “fail small” before “fail all”
- Focus heavily on experimentation and learning
- Improve the speed of their feedback loop to enable rapid evolution of ideas

Why DevOps?

DevOps, therefore, is an approach based on Lean and Agile principles in which business owners and the development, operations, and quality assurance departments collaborate to deliver software in a continuous manner.

This allows the business to more quickly seize market opportunities and reduce the time to include customer feedback.

DevOps provides significant return on investment in three areas:

- Enhanced customer experience
- Increased capacity to innovate
- Faster time to value

IBM DevOps Solution

When is DevOps used?

DevOps is not a magic bullet, it can not be used in every project. DevOps lends itself to situations requiring rapid changes in response to functionality and scale.

When to use DevOps

- Web sites (mobile, desktop)
- Mobile apps using back-ends as a server component for information resources
- Mobile application front-end development
- Startups, quick initiatives, exploring / prototyping where more systems of engagement are involved

When not to use DevOps

- Typical packaged / platform-based services (BI Systems, ERP systems)
- Back-end systems (ETL Systems, Systems of Record data sources)

What are Systems of Record and Systems of Engagement?

Systems of Record (SOR)

SOR are the ERP type systems that you rely upon to run a business—financials, manufacturing, CRM, and HR. These have to be **correct** and **integrated** so all data is consistent. These were traditionally designed for people who had no choice but to use these. These systems are basically, used for old style applications that have huge transactional databases backing up all the data.

Systems of Engagement (SOE)

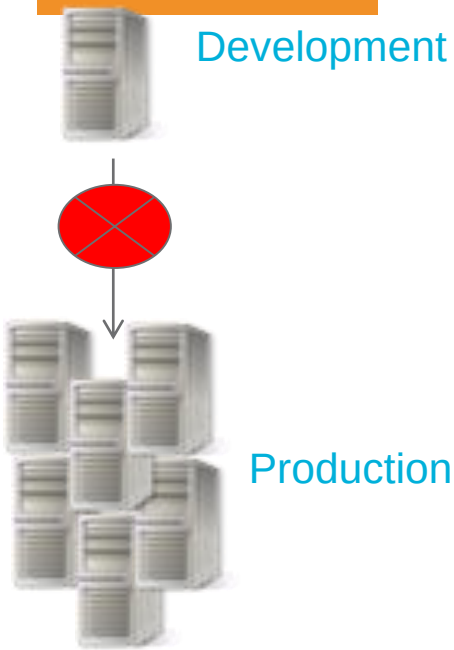
SOE are systems that are used directly by employees for **sticky uses** such as email, collaboration systems, and new social networking and learning systems. These systems **engage** employees. Mobile applications are a good example.

In the current world scenario, the two complement each other. The SOE allows customers to interact with the SOR conveniently through devices such as mobiles.

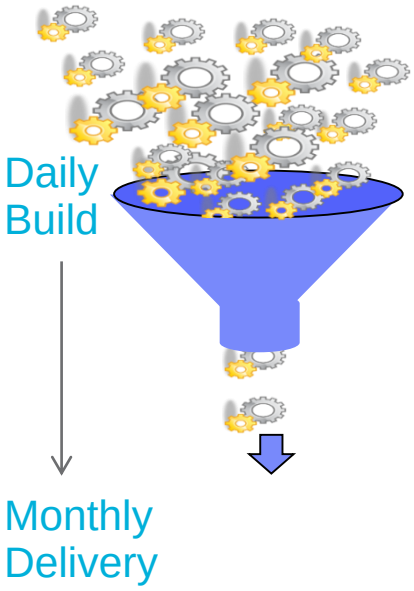
Click here to learn more.

Challenges tackled by DevOps

Failures due to inconsistent development and production environments



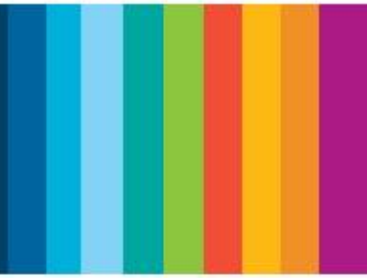
Bottlenecks trying to deliver more frequent releases to meet market demands



Complex, manual, processes for release lack repeatability and speed

Poor visibility into dependencies across releases, resources, and teams

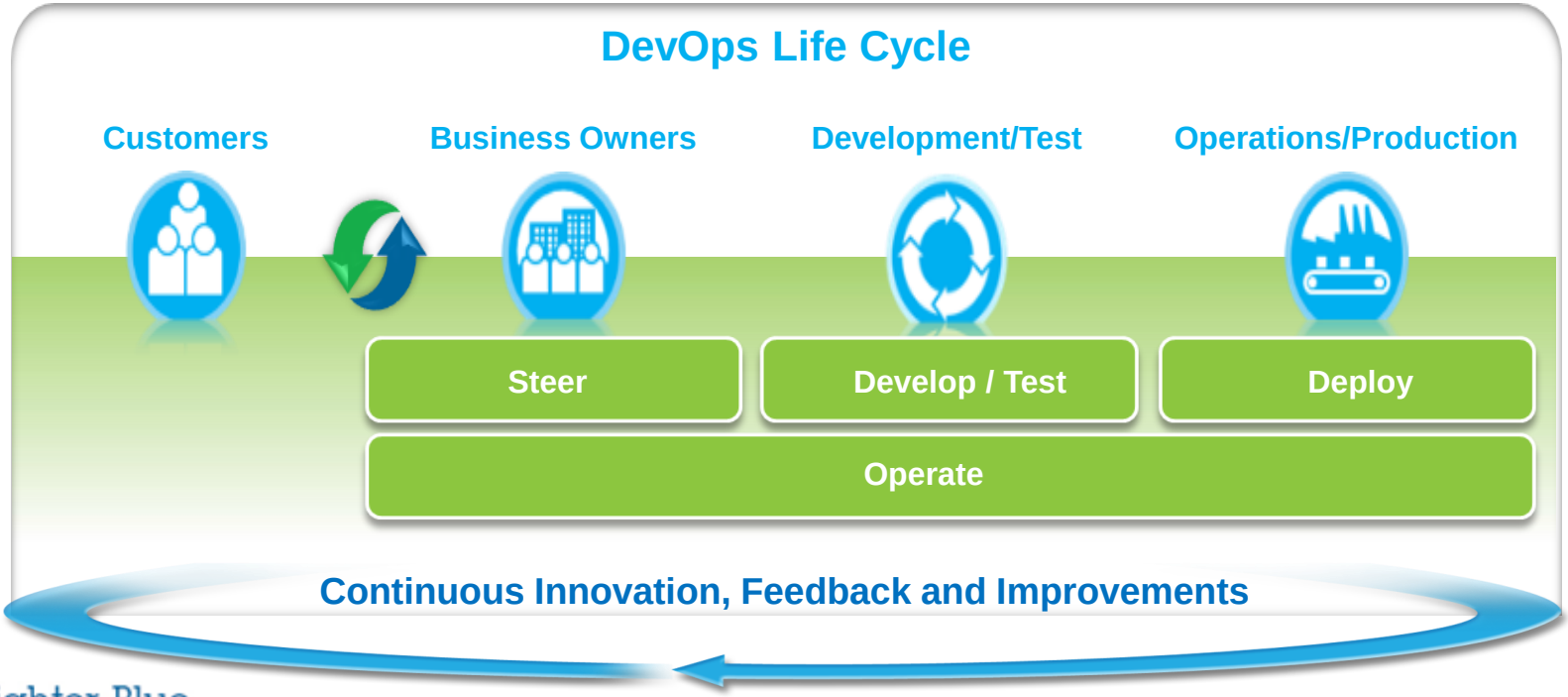
The DevOps Life Cycle and Its Impact



The DevOps life cycle

If you are on a software development team, you may feel the pressure to deliver faster. But you will still need to deliver value. And, you must keep the costs down too.

How do teams deliver at the speed of business? DevOps is an enterprise capability for continuous service delivery that enables clients to seize market opportunities and reduce time to customer feedback. DevOps is a continuous process of innovation, feedback and improvements.



The DevOps life cycle (continued)

DevOps is a continuous process of innovation, feedback, and improvements.

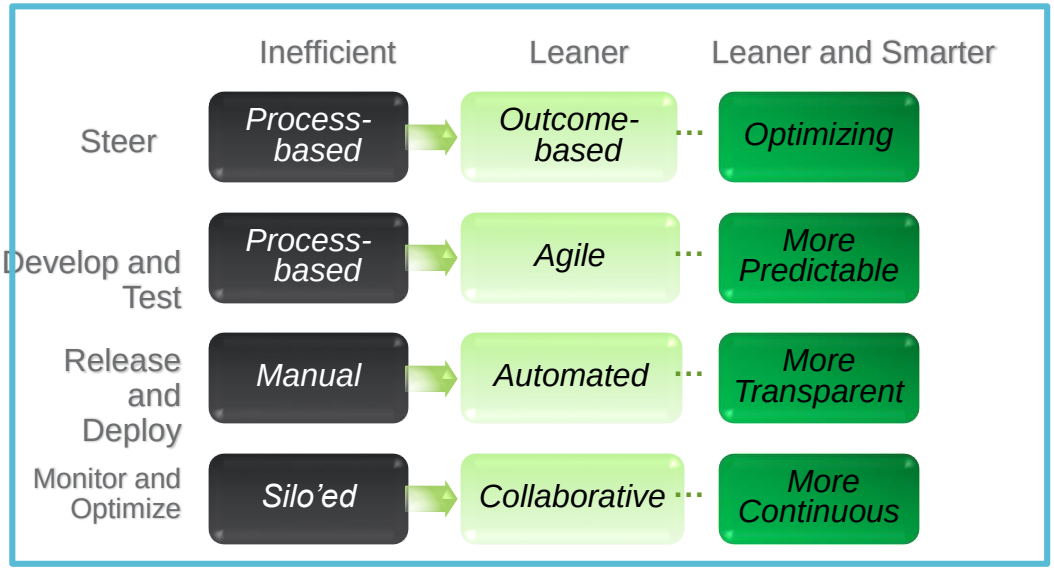
- Accelerates software delivery
- Balances speed, cost, quality, and risk
- Reduces time to customer feedback

Click each bullet point to learn more.

Accelerate Software Delivery

- Expanding collaboration to include customers, LOB and others to eliminate organization silos

The graphic depicts the maturity levels of an organization adopting DevOps, moving from “Inefficient” to “Leaner” to “Leaner and Smarter”. The characteristics of the three kinds of organizations are mentioned in the boxes below the three kinds of organizations.



The DevOps life cycle (continued)

DevOps is a continuous process of innovation, feedback, and improvements.

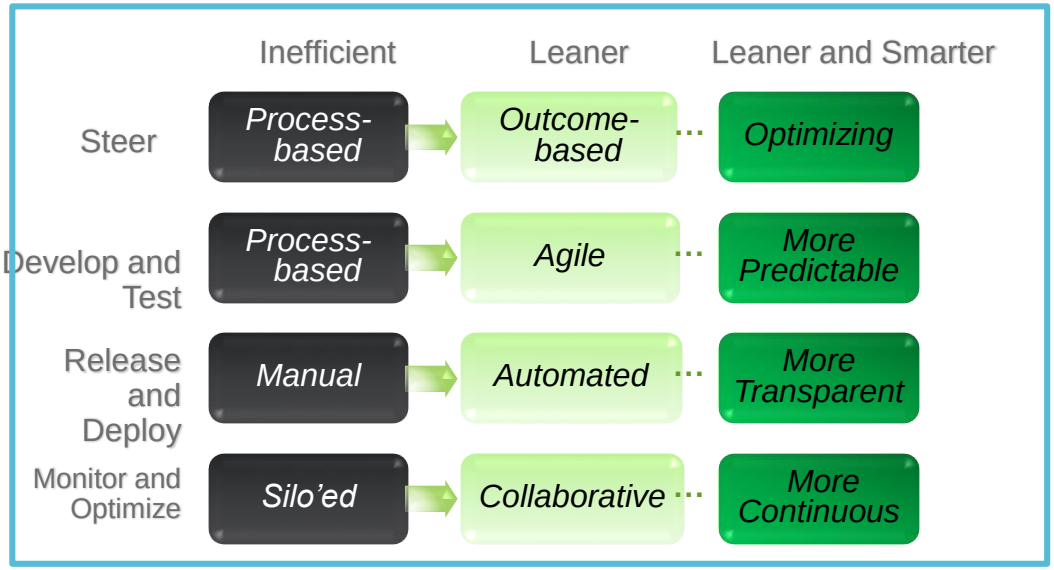
- Accelerates software delivery
- Balances speed, cost, quality, and risk
- Reduces time to customer feedback

Click each bullet point to learn more.

Balance speed, cost, quality and risk

- Automating manual processes across delivery life cycle to eliminate waste/delays and compliance tracking

The graphic depicts the maturity levels of an organization adopting DevOps, moving from “Inefficient” to “Leaner” to “Leaner and Smarter”. The characteristics of the three kinds of organizations are mentioned in the boxes below the three kinds of organizations.



The DevOps life cycle (continued)

DevOps is a continuous process of innovation, feedback, and improvements.

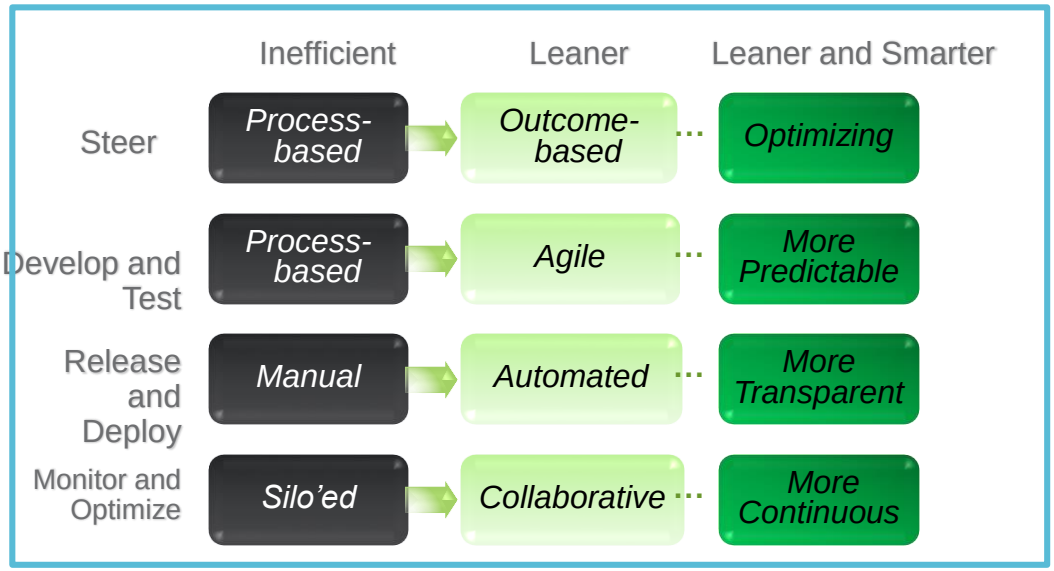
- Accelerates software delivery
- Balances speed, cost, quality, and risk
- Reduces time to customer feedback

Click each bullet point to learn more.

Reduce time to customer feedback

- Enabling a customer feedback loop for continuous improvement

The graphic depicts the maturity levels of an organization adopting DevOps, moving from “Inefficient” to “Leaner” to “Leaner and Smarter”. The characteristics of the three kinds of organizations are mentioned in the boxes below the three kinds of organizations.

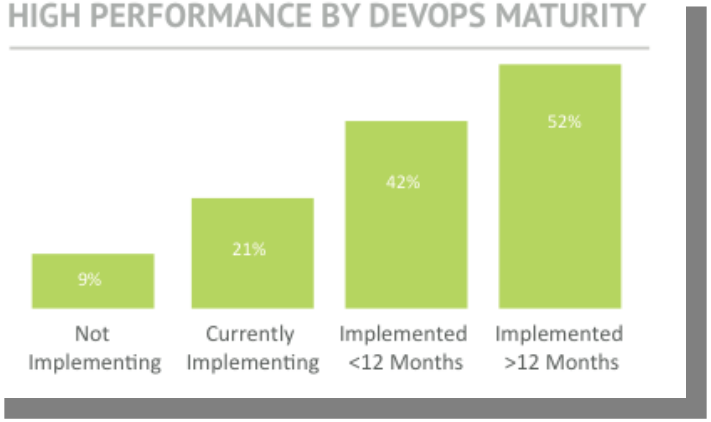


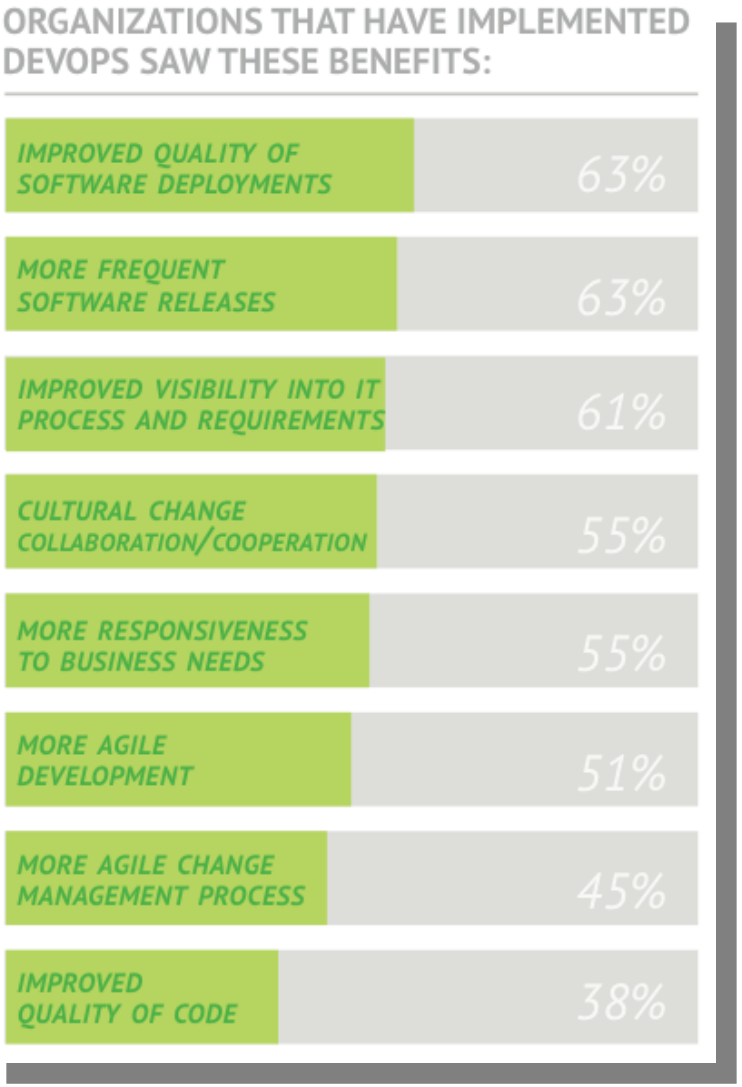
The impact of DevOps

High performing teams adopt DevOps.

Ship code 30x faster
and complete those deployments 8,000 times faster than their peers.

Have 50% fewer failures
and restore service 12 times faster than their peers.





Reference: 2013 State of DevOps Report by PuppetLabs

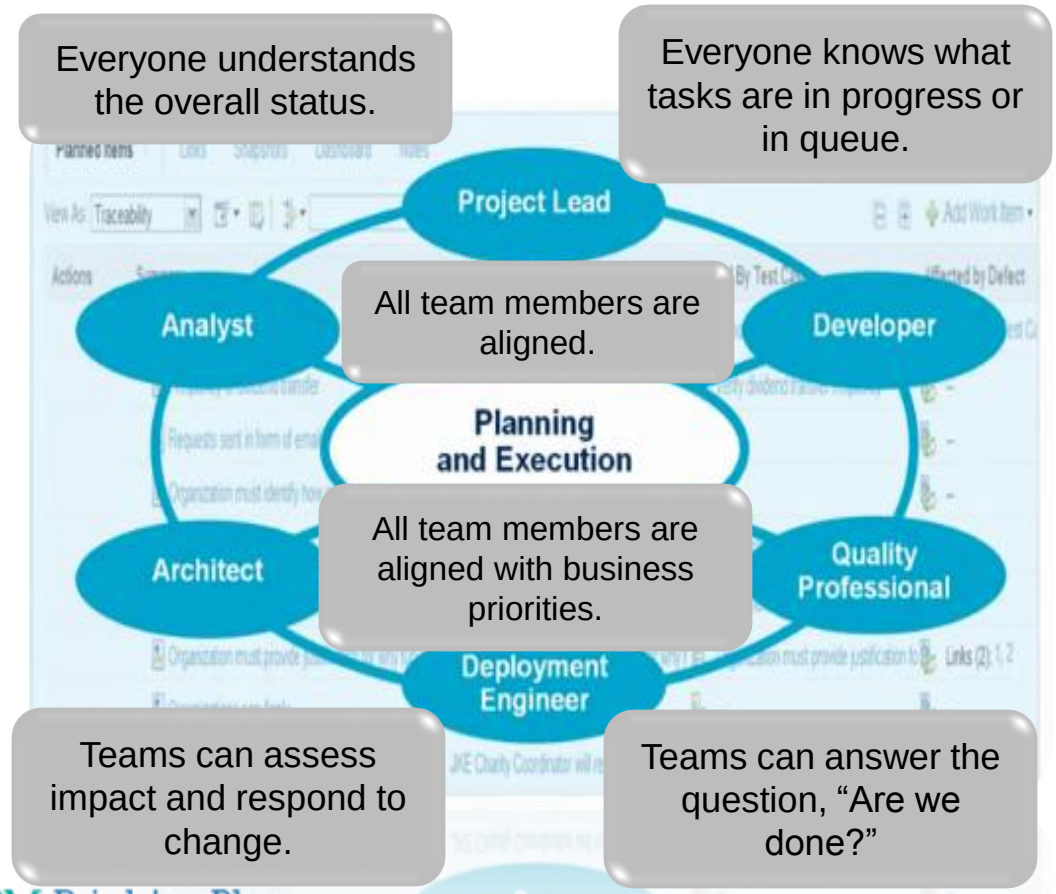
DevOps adoption and improvement in metrics

Lifecycle Measurements	2008	2010	2012 – 2013	Total Improvement
Project Initiation	30 days	10 days	2 days	**28 days**
Groomed Backlog	90 days	45 days	On-going	**89 days**
Overall Time To Development	120 days	55 days	3 days	**117 days**
Iteration Length	6 weeks	4 weeks	4 weeks	**2 weeks**
Number of Iterations	6	8	3	**N / A**
Composite Build Time	36 hours	12 hours	8 hours	**400 %**
BVT Availability	N / A	18 hours	< 1hour	**17 hours**
Iteration Test Time	5 days	2 days	4 hours	**4 days**
Total Deployment Time	2 days	8 hours	2 hours	**2 days**
Overall Time To Production	9 days	3 days	15 hours	**8 days**
Time Between Releases	12 Months	12 Months	3 Months	**9 Months**

Stage 1 of Adoption: Steer

Teams effectively plan and measure to deliver at the speed of business

Steer encompasses planning and measuring. Planning is done for particular requirement or a change. Requirements are the foundation of systems and software development. Project teams must understand the various attributes of a requirement.



Why does change happen?

- Business Reasons
  - Urgent customer requirement
  - Market shift
  - Business shift
- Execution Issues
  - Resource / capacity changes
  - Scope change

Planning and measuring provides relief to client's pain points

Business pain points	DevOps ensures "continuous steering"
▪ There is an inability to quickly respond to business needs and changes because of complex interdependent applications, processes, technologies, and tooling. ▪ Unremarkable Customer Experience: Value stream is fragmented and diluted because needs are lost in translation from business planning to IT deployment. ▪ Unintended consequences in business release delivery due to inability to manage complexities. ▪ There is lack of understanding on what to deliver.	▪ Plan & Execute: Get all teams moving in the same direction to deliver value (not feature/function) ▪ Explore & Adjust: Effectively re-plan when change occurs, exploring trade-offs to maximize the delivery of value while reducing risk ▪ Monitor & Measure: Provide visibility into health, status, risk of value delivery at the right level for the consumer

Key areas of the Steer stage

Requirements are the foundation of systems and software development. There are nine attributes of a good requirement.

Correct	Complete	Clear
Consistent	Verifiable	Traceable
Feasible	Modular	Design-independent

Let us now study the five key areas of the Steer stage.

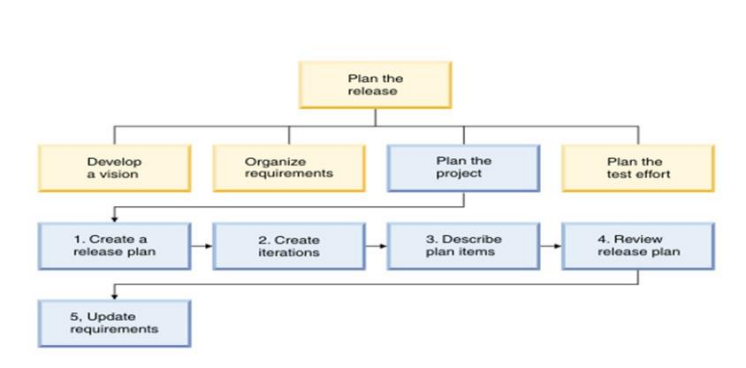
Click each key area tab to learn more.

- Project Planning
- Release Planning
- Metrics
- Traceability
- Dashboards

Key areas of the Steer stage

Planning a project includes creating a release plan, iterations, creating plan items from high-level requirements, describing the plan items, reviewing the plan with stakeholders, updating requirements to reflect the team's decisions. The project manager completes the initial steps and the business analyst completes the updating requirements step.

Click the image to view an enlarged version.



Good release planning helps you to more accurately predict the:

- Required resources
- Scope of the project
- Delivery dates

Click each key area tab to learn more.

Project Planning

Release Planning

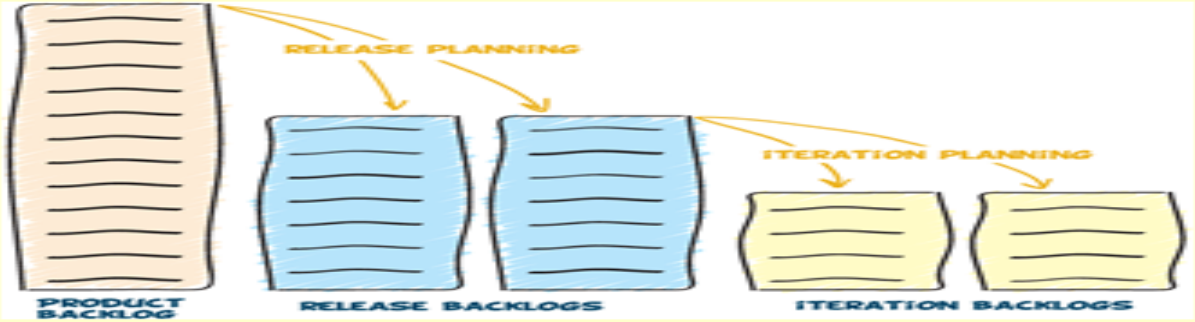
Metrics

Traceability

Dashboards

Key areas of the Steer stage

A Release Plan determines how much can be accomplished by a certain date. It is beneficial for the project that the release plan is revisited after each iteration. This helps refresh objectives and the overall release plan. The quicker the date of release, the better it is for the organization as it begins earning a return on its investment.



Click each key area tab to learn more.

Project Planning

Release Planning

Metrics

Traceability

Dashboards

Key areas of the Steer stage

In development, there are hard, quantifiable technical, and financial metrics that you can track. DevOps has two primary categorizations.

Process Metrics
Here, you will look at:

- Requirements elicitation and management
- Agile development
- Build
- Release and deployment
- Unit testing
- User acceptance testing
- Quality assurance
- Application performance monitoring
- Cloud

Culture Metrics
Indicators include:

- Cross-skilling, knowledge sharing, and pairing
- Fluid, focused work
- Multidisciplinary team work
- Organizing teams around projects not skill-sets
- Benefitting from failure
- Position on business demand
- Extraneous lines of code
- Continuous improvement
- Obsession with metrics
- Technological experimentation
- Team autonomy
- Team features:
  - Rewards and feelings of success
  - Hierarchical and political obstacles and annoyances
  - Inspiring and fostering creativity

Project Planning

Release Planning

Metrics

Traceability

Dashboards

Click each key area tab to learn more.

Key areas of the Steer stage

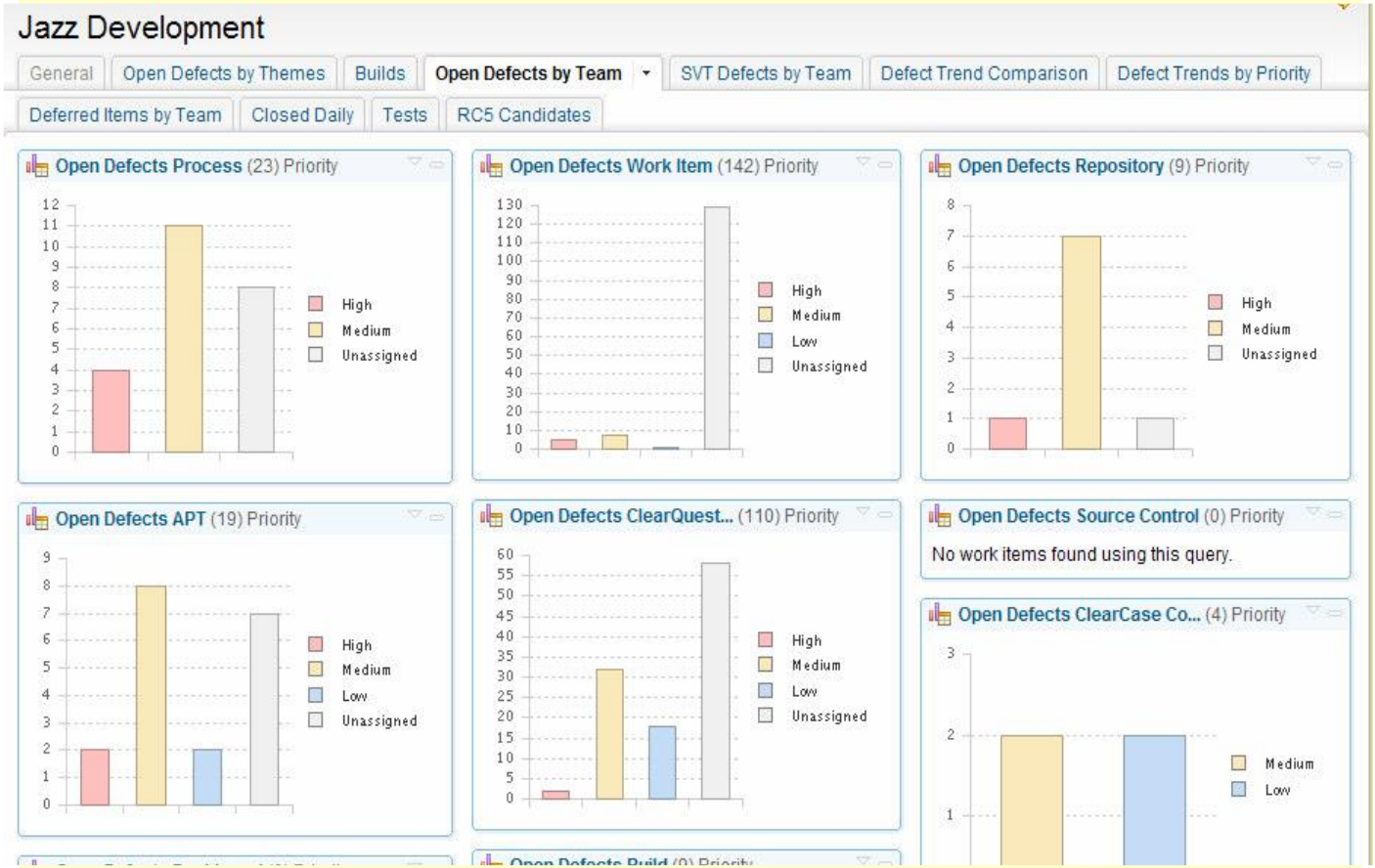
Linking in and across project areas enables real-time life cycle traceability. Traceability helps teams answer difficult questions.

Click each key area tab to learn more.

- Project Planning
- Release Planning
- Metrics
- **Traceability**
- Dashboards

Key areas of the Steer stage

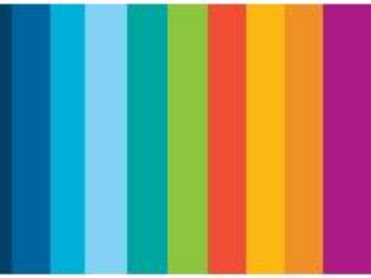
Dashboards include open defects by the team



- Project Planning
- Release Planning
- Metrics
- Traceability
- **Dashboards**

Click each key area tab to learn more.

Stage 2 of Adoption: Develop / Test



Integration through collaborative development and continuous testing

Each integration is verified by an automated build to detect integration errors quickly. This stage forms the core of development and quality assurance (QA) capabilities. DevOps practices can be applied here as it has much of friction, blockages, and pain points. This approach leads to significantly reduced integration problems and allows cohesive software development more rapidly.

Benefits:

- Accelerates software delivery
- Balances speed, cost, quality, and risk
- Reduces time to customer feedback

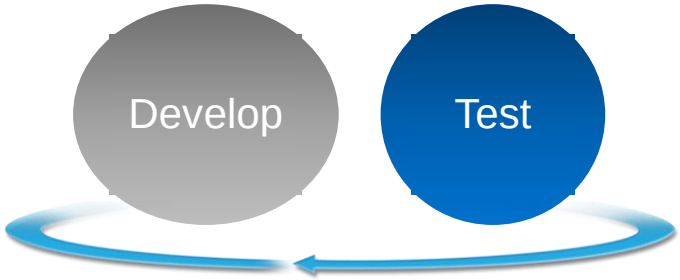
Benefits for the IT team:

- Customers using new apps pass requirements to the business, which clarifies customer need.
- Business constantly passes requirements to development team, which builds, tests, and demos code.
- Development continuously releases new apps to operations, which continuously monitor them for effectiveness.

How do continuous integration and testing work?

Integration is when developers and testers collaborate to generate quality code, which is a deployable unit. When done in a continuous manner, this becomes continuous Integration. To do so:

- **Embrace lean thinking and agile methodologies**
  - Eliminate unessential activities
  - Emphasize fast short iterations
- **Automate, automate, automate**
  - Establish traceability
  - Virtualization
  - Deployment
  - Testing
- **Optimize Develop / Test cycle**
  - “Shift testing left”: help developers catch defects earlier
  - Practice continuous integration
  - Blur the silos



Continuous Integration and Testing

Integration is challenging and takes time

Integration is challenging:

- It requires project-wide communication and coordination
- It frequently uncovers unanticipated problems

Integration takes time:

- It is difficult to predict the time it will take
- A problem in one area can block progress in another

Integration typically happens at the end of a project, the worst time to introduce risk and delay.

Critical aspects of continuous integration (CI)

- The project has a single main stream of development—the integration stream.
- The integration stream builds successfully periodically.
- Automated tests run as part of the build.
- Project status is transparent through build results.

Collaborative Development is an implicit part of CI

Collaborative Development means that developers and operations collaborate and work together to churn out a cleaner code which is a deployable unit.

Collaborative development helps to:

- Minimize delays, quality issues, and budget overruns
- Provide life cycle traceability across disciplines
- Allows real-time planning and development.
- Facilitates frequent team integrations and automatic builds.
- Integration issues are identified and fixed early.

What is Configuration Management?

It is the process of defining, monitoring, and controlling the status and versions of the components of a system, which together constitute a consistent whole for a period of time.

Collaborative Development integrates automated code review and unit testing

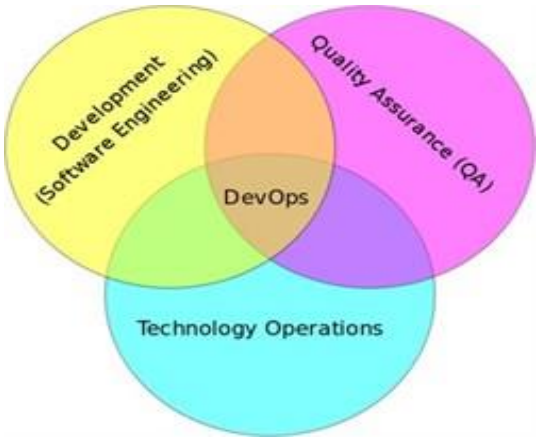
Code Review

The traditional way of code review was manual peer reviewing of the code. Automating code review saves the waiting time in peer review. A code review ensures that the code:

- Complies with recognized coding standards
- Finds unwanted dependencies
- Ensures that code development follows the intended structural design.

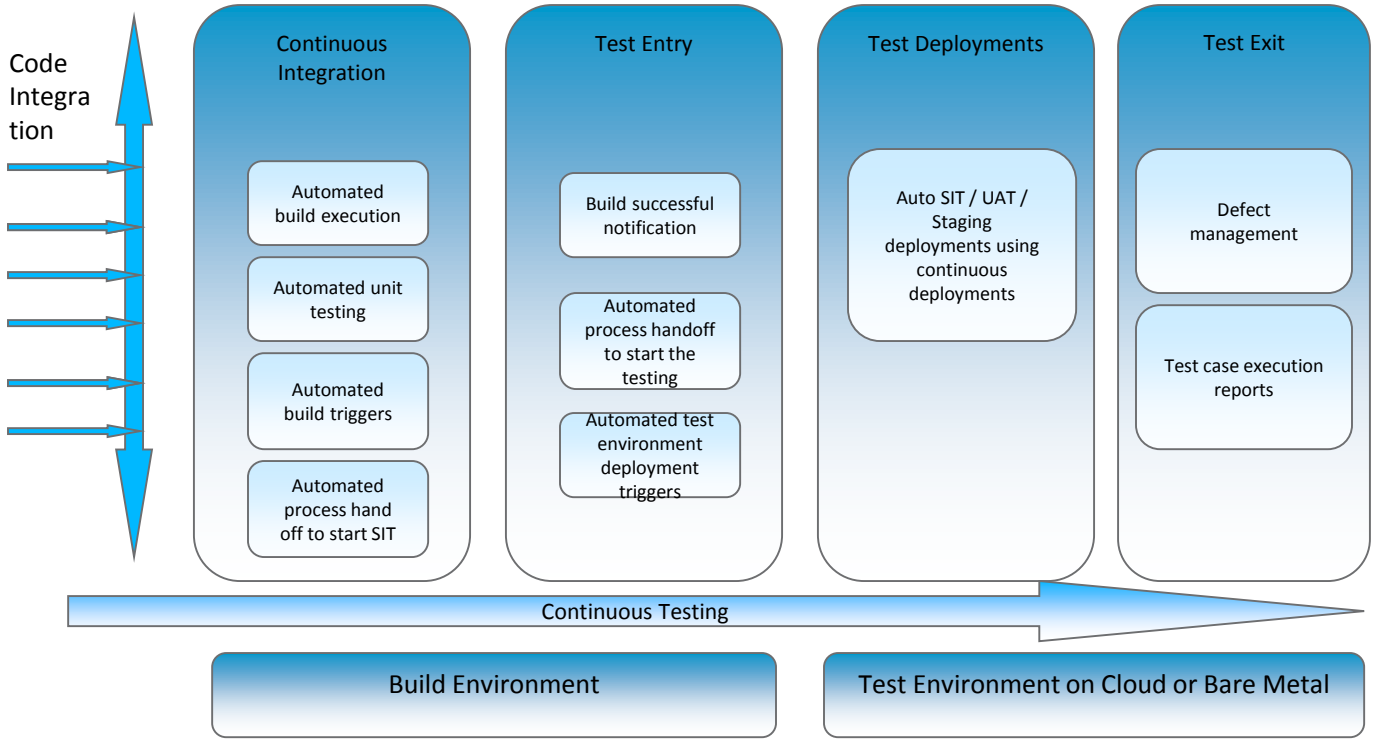
Unit Testing

Unit testing is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures are tested to determine if they are fit for use.

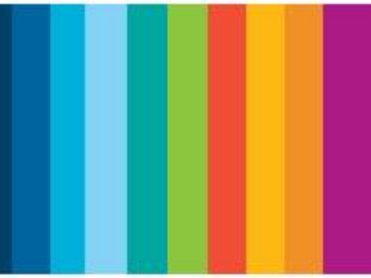


Collaborative integration requires continuous testing

Continuous testing is a critical part of the handoff between development and IT operations. You divide your requirement in small chunks and develop and test them in continuous manner. Continuous testing capabilities reduce the cost of provisioning and maintaining test environments, and shorten test cycle times by allowing integration testing early in the life cycle.



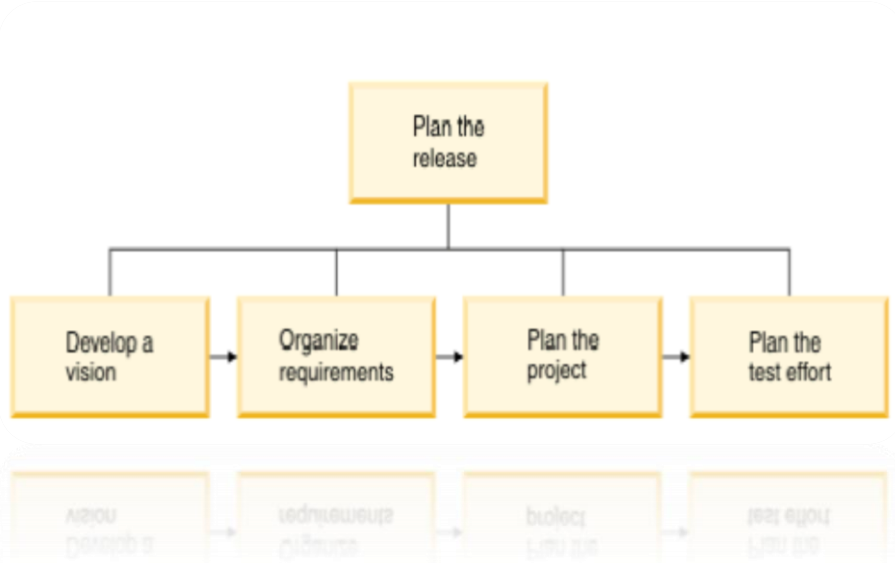
Stage 3 of Adoption: Deploy



Deploy stage includes release and deploy

Release is a workable software product labeled or named with some number or name. It is produced to deliver specific requirements. It is normally incremental and produced out of software development life cycle (SDLC) phases.

Deployment is the activity responsible for movement of approved releases of hardware, software, documentation, and processes to any environment.



Release and deployment activities

- Plan release
- Prepare for build, test, and deployment
- Build and verify
- Test
- Plan and prepare for production deployment
- Perform production deployment
- Verify production deployment
- Provide early life cycle support
- Review and close release

Challenges of Release and Deployment

Releases encompass more than application deployment

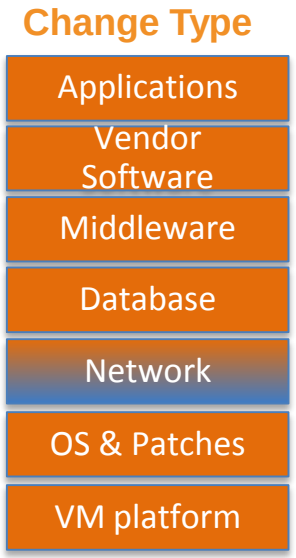
- Examples are middleware, network, hardware changes in addition to application changes
- Steps known in development and integration, but missed in production

Interaction between applications in a release

- Ordering application deployment steps fails to account for dependencies between applications or deployment steps
- Required artifacts or applications missed or wrong application versions deployed

Difficulty coordinating dozens of participants

- Late breaking changes to deployment instructions or targeted artifacts are not communicated
- Work product quality and process check lists scattered about many tools and not digested for at-a-glance status



Business impact of challenges

Costly delays and missed deadlines for application releases / updates

- Customer dissatisfaction, Lost revenue & credibility

Systems that are hard to support, troubleshoot and maintain

- High Risk with knowledge held by few, Need headcount for increased volume

Unpredictable product release cycles

- With limited repeatability and portability

Unpredictable results

- Unauthorized changes to systems

Costlier tracking and auditing throughout the application life cycle

- Information gathered manually across disparate toolsets

Common Release problems are:

- Long conference calls
- High costs
- Difficulty in getting status
- Hard to resolve issues

Application Deployment problems include:

- Manual hand-offs
- Costly delays
- Disconnected environments
- Lack of repeatable processes

What is Continuous Release and Deployment

Continuous Release and Deployment practices help organizations relieve bottlenecks for accelerated application delivery and help to decrease feedback time. It also:

- Provides a continuous delivery pipeline that automates deployments to test on production like environments
- Reduces the amount of manual labor, resource wait-time, and rework by means of push-button deployments that allow higher frequency of releases, reduced errors, and end-to-end transparency for compliance

The continuous release and deployment practice within DevOps offers following benefits :

- Speed time to market
- Stable and predictable releases
- Increased visibility
- Fewer outages and efficient rollbacks, if required
- Release better software more often

Continuous Deployment is closely related to Continuous Integration

Continuous Deployment is closely related to Continuous Integration and refers to the release into production of software that passes the automated tests. Essentially, it is the practice of releasing every good build to users.

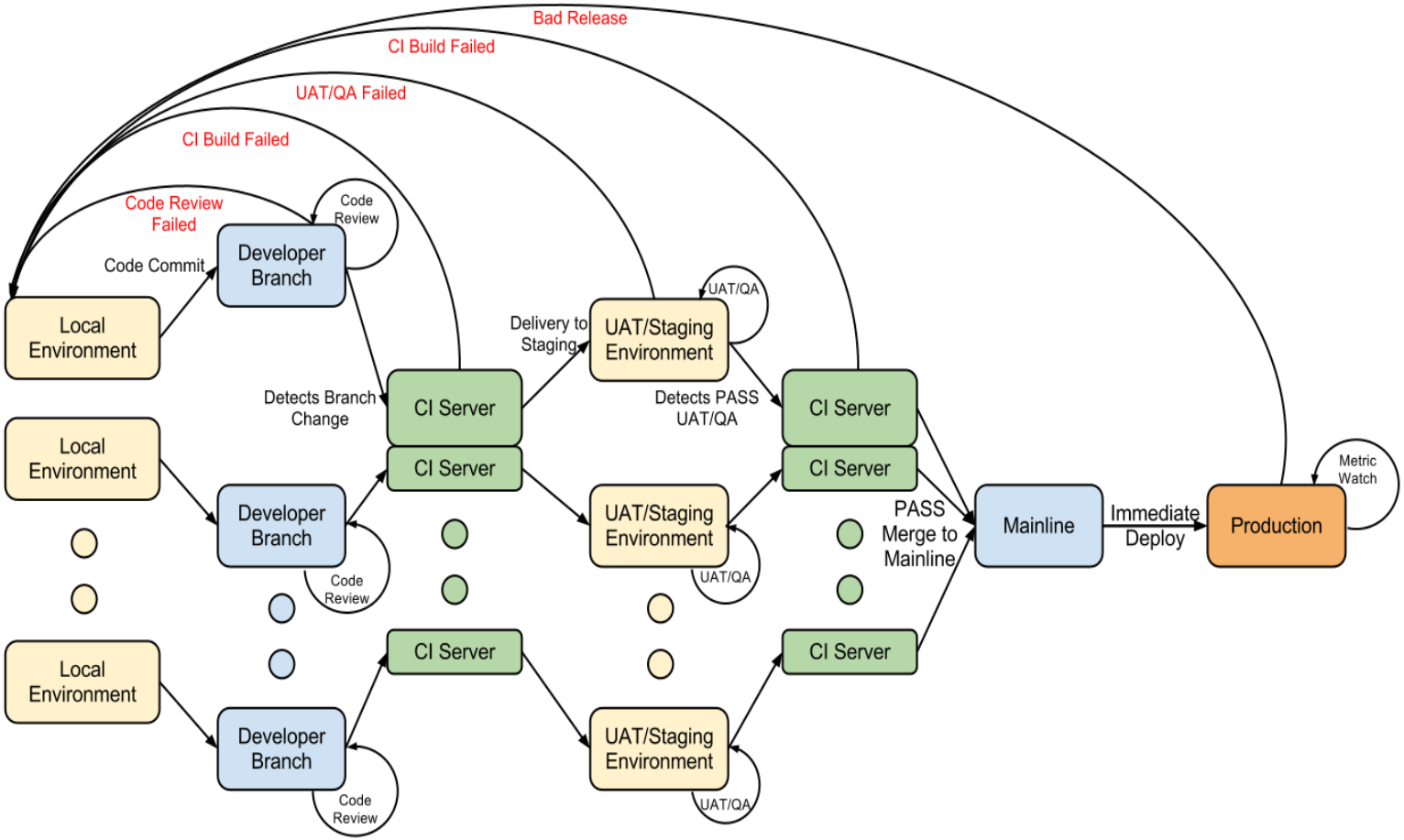
Continuous Deploy:

- Promotes multi-tiered applications through to Production
- Versions deployment artifacts
- Manages incremental deployment changes
- Deploys to middleware environments
- Facilitates database change deployments
- Provides Deployment snapshots
- Facilitates rollbacks

Continuous Release:

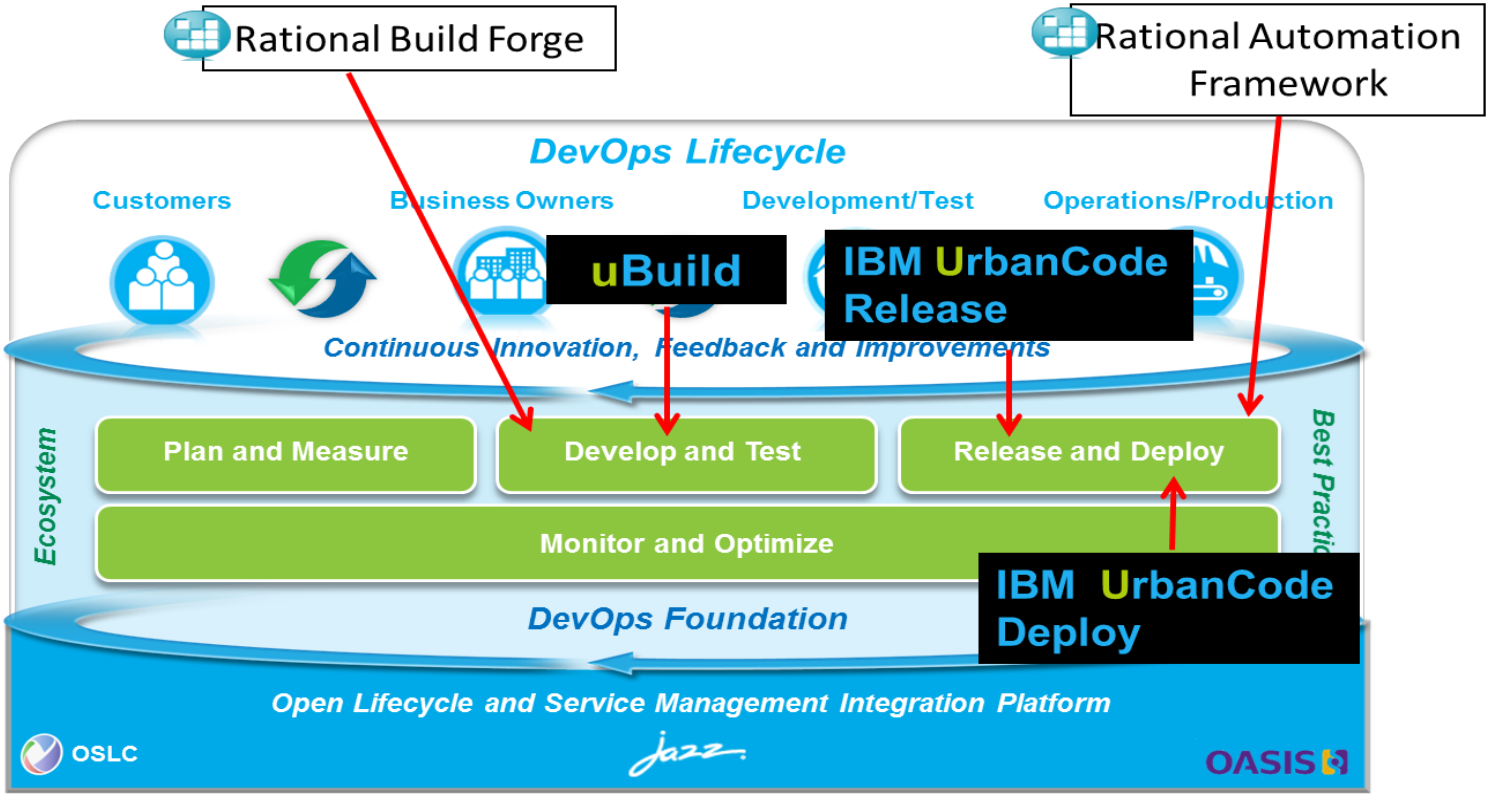
- Manages environment changes in release events
- Tracks infrastructure and application changes through a release
- Orchestrates releases of inter-dependent applications
- Facilitates release collaborations

DevOps drives quickly to customer feedback



Use of UrbanCode in the IBM DevOps Strategy

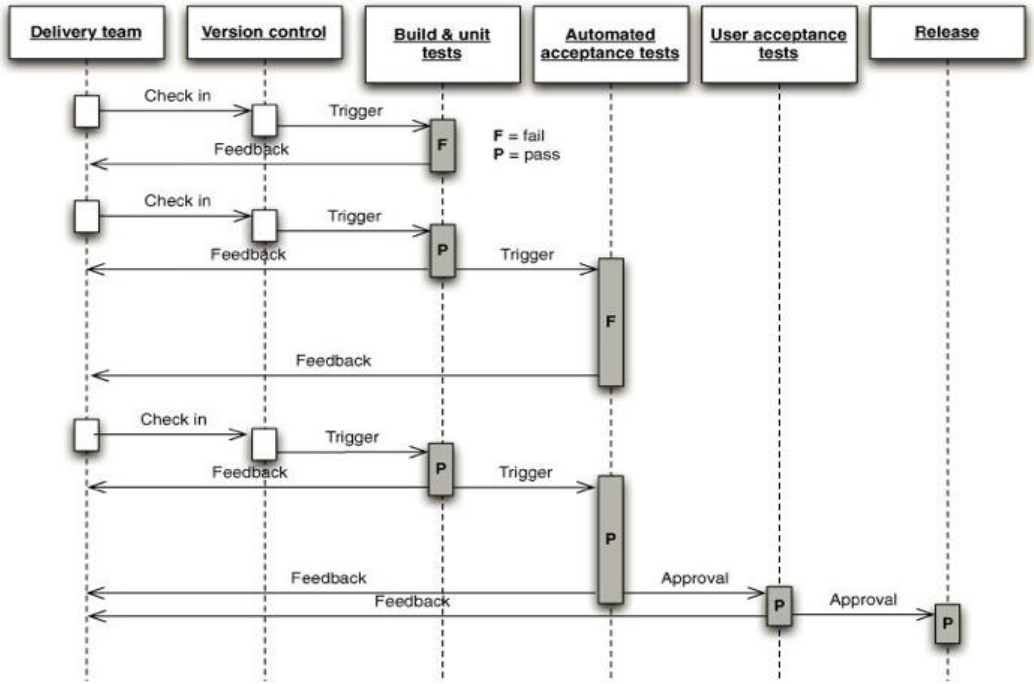
IBM® UrbanCode Release and IBM UrbanCode Deploy solutions provide Continuous Release and Deployment capabilities, with the successful implementation of continuous release and deployment practices.



What is Continuous Delivery?

Continuous Delivery is:

- An automated flow from Build to Ready to Deploy to Production
- Push-button deployment to production
- The execution of many types of tests
- Cultural emphasis on constant shipability



Stage 4 of Adoption: Operate

Operate includes monitoring and optimizing

Monitoring and optimizing encompasses the monitoring, customer feedback, and optimization of a software solution. These capabilities include two specific practices:

1. Continuous monitoring

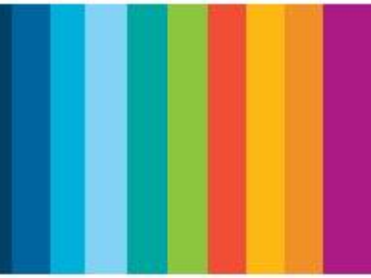
2. Continuous feedback and optimization

These capabilities help stakeholders understand the performance, customer behavior, and pain points of the software in both pre-production and post-production environments.

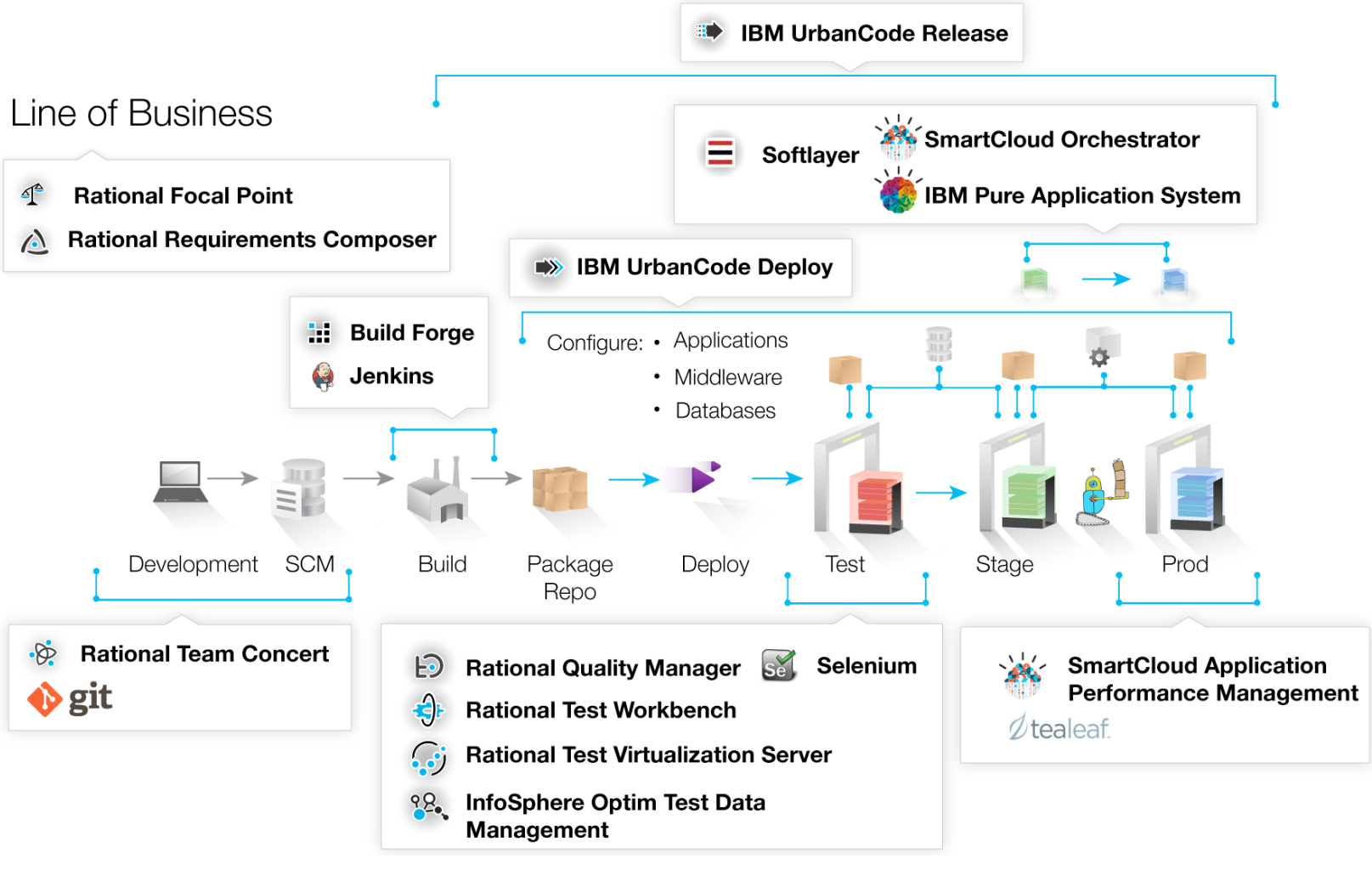
There is value for customers in tying the four adoption paths together

A good example is tying Release and Deploy to Monitor and Optimize. A customer interested in automating deployment with our UrbanCode solutions in speeding delivery, reducing deployment costs and bridging the Dev-Ops divide, will logically be interested in automation solutions for Monitor and Optimize as further complimentary automation that help the business and development teams understand their customer behavior and performance of their applications so they can continually improve them and stay competitive.

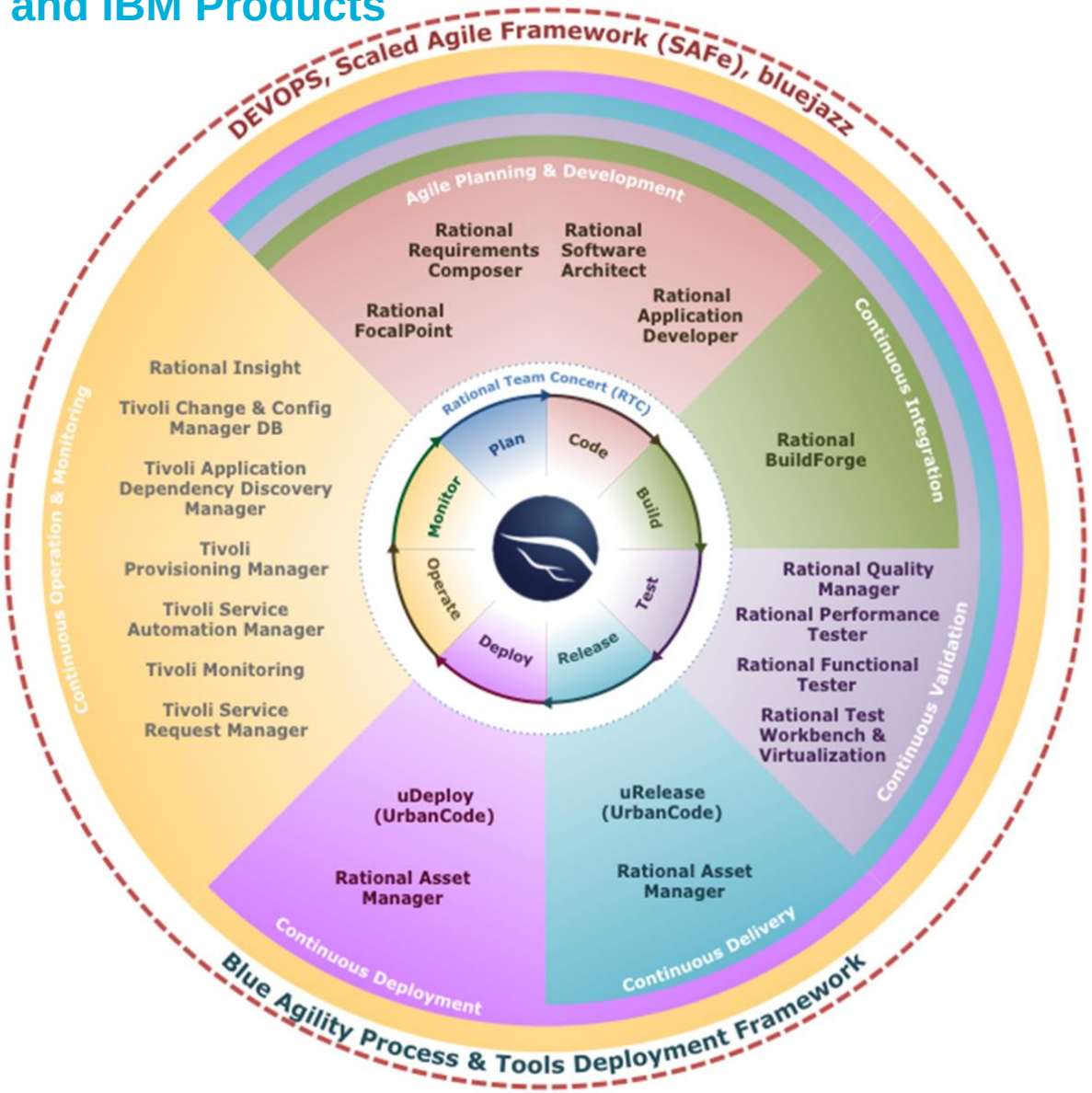
Tools and Impact of DevOps



Typical pipeline using DevOps



DevOps Tools and IBM Products



Module summary

Congratulations! You have completed this module on Introduction to DevOps.

You should now be able to:

- Narrate solution production using traditional development methods
- Describe DevOps
- Identify characteristics of DevOps team
- Outline the DevOps life cycle
- Give evidence of the impact of DevOps
- Explain the fours stages of adoption
  1. Steer
  2. Develop / Test
  3. Deploy
  4. Operate
- List the tools of DevOps and their purpose