DevOps Toolkit
Our Perspective and Our Approach

Business driven challenges

SCALING

Inefficient on-demand scale due to changing customer needs

HIGH CYCLE TIME

Manually intensive & uncoordinated activities between the varied teams

DASHBOARD

No Standardized dashboards are available for DevOps solutions

RETURN ON INVESTMENT

Tools & resources that are underutilized due to 'Siloed Operation'.

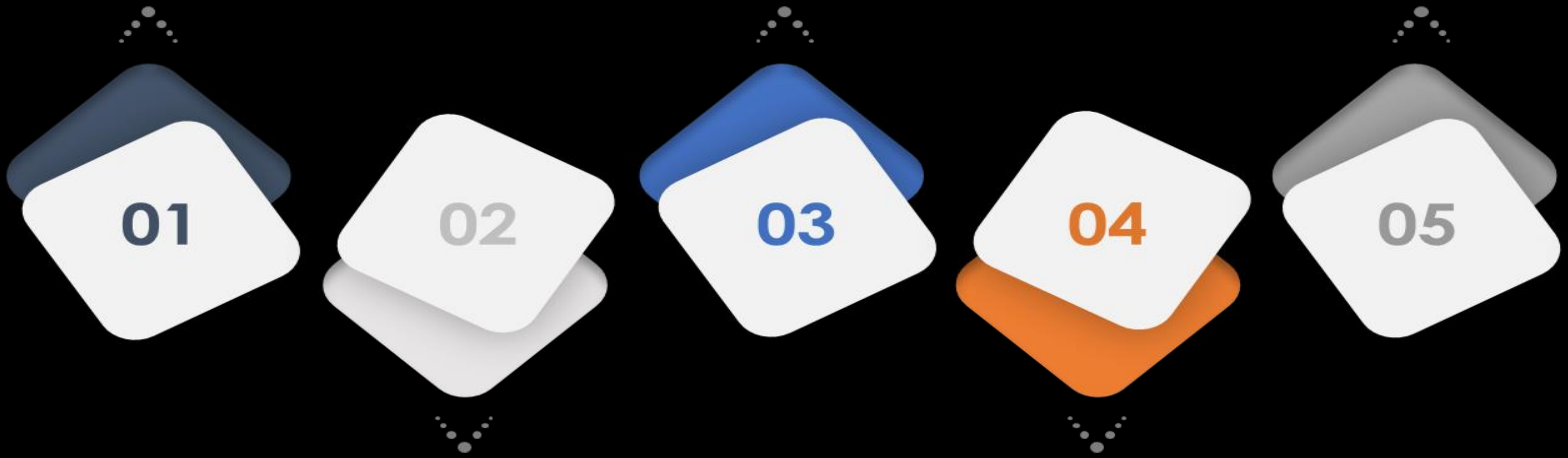
TOOL CHOICE

Wide variety of tools that may not always suit all requirements

FOCUS AREA

Dev teams are spending more time focusing on complex infrastructure

What problems are we solving



01 Limited visibility of process and practices

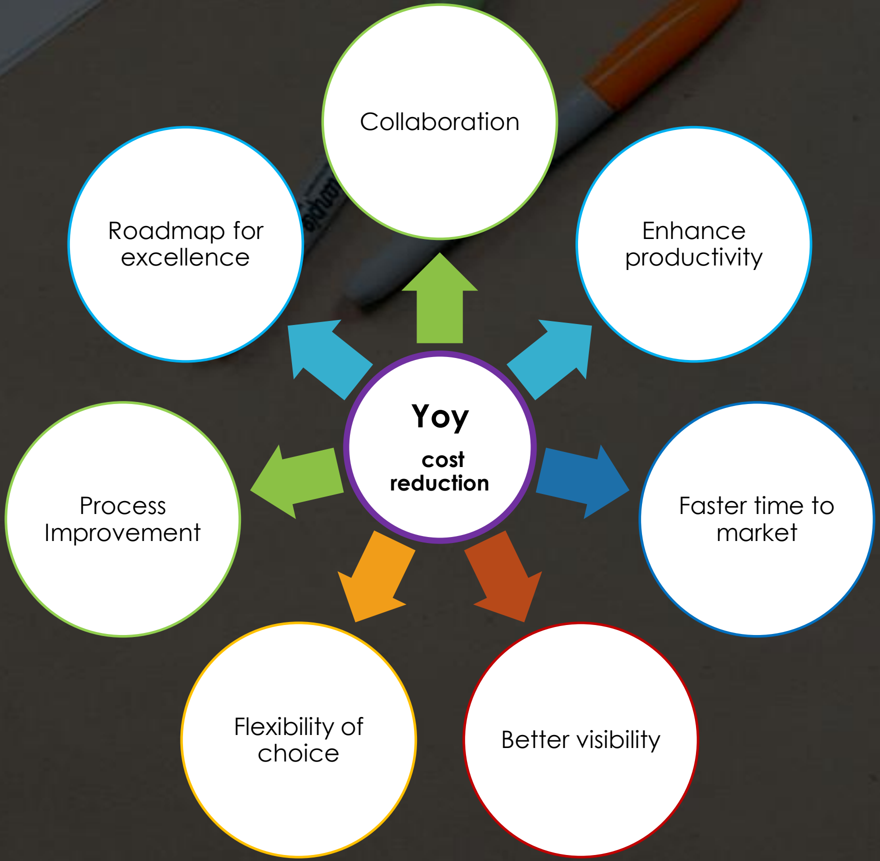
02 Absence of insights platform to make informed decisions

03 Identifying the right tool

04 Bringing consistency in practice and speeding up delivery

05 Being relevant with ever changing application architecture

Why use DevOps toolkit



Yoy cost reduction

Collaboration

Enhance productivity

Faster time to market

Better visibility

Flexibility of choice

Process Improvement

Roadmap for excellence

High Performance Engineering with DevOps

Faster to market

Being relevant

Accelerating
ahead
of the
curve through
a streamlined
software delivery

A cultural shift

A mind-set change

A united
and a
shared identity
between
development ,
testing
and operations

Richness of tools

Adopting Automation

A blend
of plethora
of tools
and
plugins
that eliminates
friction

Powering with insights

Making informed decisions

Harnessing
data silos
to take
actionable
insights
and
predictable
delivery

brillio

How are we provisioned to help ?

Brillio BOLT^{TM} Insights

- ❖ Project Health
- ❖ Actionable Insights
- ❖ Release Predictability
- ❖ Team Performance

New age Relevance

- ❖ Containerization Framework
- ❖ DevOps for New Age
- ❖ Cloud DevOps

Jumpstart Kits

- ❖ Pre-built scripts
- ❖ Templates
- ❖ Frameworks
- ❖ Process & Methodologies

Infrastructure assessment

- ❖ Cloud services and Infrastructure ROI
- ❖ Cloud DevOps service options

Tools assessment

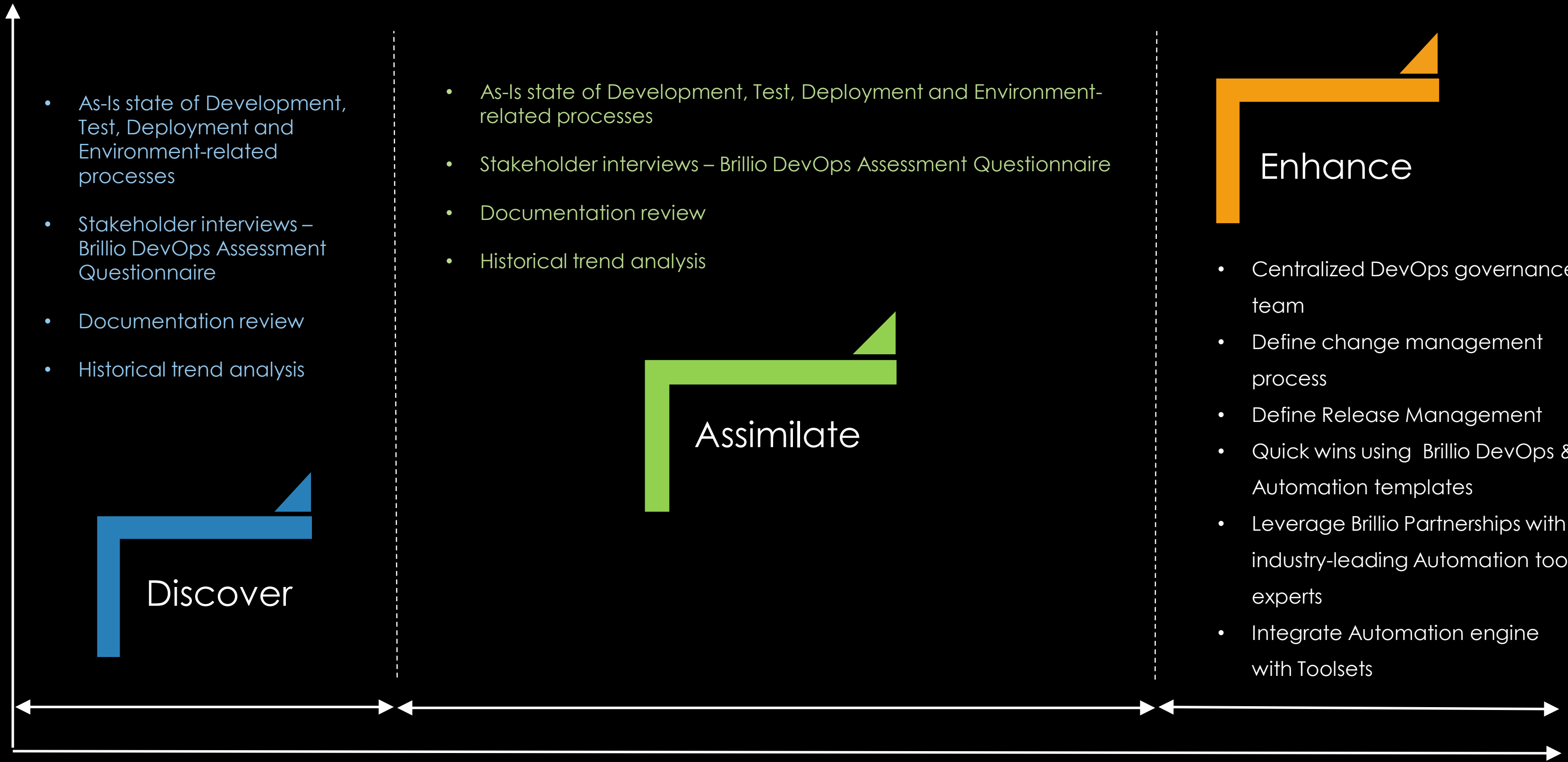
- ❖ Assessment and Comparison
- ❖ Recommendations

Maturity assessment

- ❖ True state of practice
- ❖ Guidance and roadmap for future

About DevOps Toolkit

Execution Overview



- As-Is state of Development, Test, Deployment and Environment-related processes
- Stakeholder interviews – Brillio DevOps Assessment Questionnaire
- Documentation review
- Historical trend analysis

Discover

- As-Is state of Development, Test, Deployment and Environment-related processes
- Stakeholder interviews – Brillio DevOps Assessment Questionnaire
- Documentation review
- Historical trend analysis

Assimilate

Enhance

- Centralized DevOps governance team
- Define change management process
- Define Release Management
- Quick wins using Brillio DevOps & Automation templates
- Leverage Brillio Partnerships with industry-leading Automation tool experts
- Integrate Automation engine with Toolsets

Solution Mapping

Business Problem	Our Solution(s)
Limited visibility of process and practices being followed as part of engineering excellence	DevOps Maturity Assessment Framework
Plethora of toolsets poses challenges in identifying the right tool choice	Tools Assessment
How do we bring in uniformity and consistency of practices and speed up delivery	Jump Start Kits
Absence of insights platform results in overhead time consumed gathering metrices and taking informed decisions	Brillio BOLT, Brillio CLIP
How do we stay relevant with ever changing application architecture	Containerization Framework, Brillio CLIP

Activities & Deliverables

	Discovery & Assessment	Design	Build & Deploy	Run
Activities	• As-Is state of Dev, Test, Deployment and Environment-related processes • Architecture overview • Documentation review • DevOps Maturity Assessment • Gap Analysis • ROI Analysis	• Validation of Findings and define roadmap for mature DevOps process • Define release management and change management process • Finalize tools for each phase of application lifecycle • Setup validation through POC • Identify infrastructure and environment landscape • Bill of Materials for cloud services or tools usage	• Environment setup and configuration • Branching strategy for projects • Tool setup and configuration • Application re-configuration • Integration with automation frameworks • Integration with monitoring system • Cloud based service configuration • Insights platform setup and configuration	• DevOps maturity assessment • Release management process setup • Integration with security and performance management tools • Application monitoring • Environment monitoring • Onboarding new projects
Deliverables	• Assessment Report • AS-IS/TO-BE architecture • Maturity Roadmap • ROI report	• Release management process • Tools Stack • Pilot projects • BOM for Cloud services • Delivery milestones	• Automation Pipeline setup • Metrics & Insights	• Governance and Change management • Release management

How do we do?

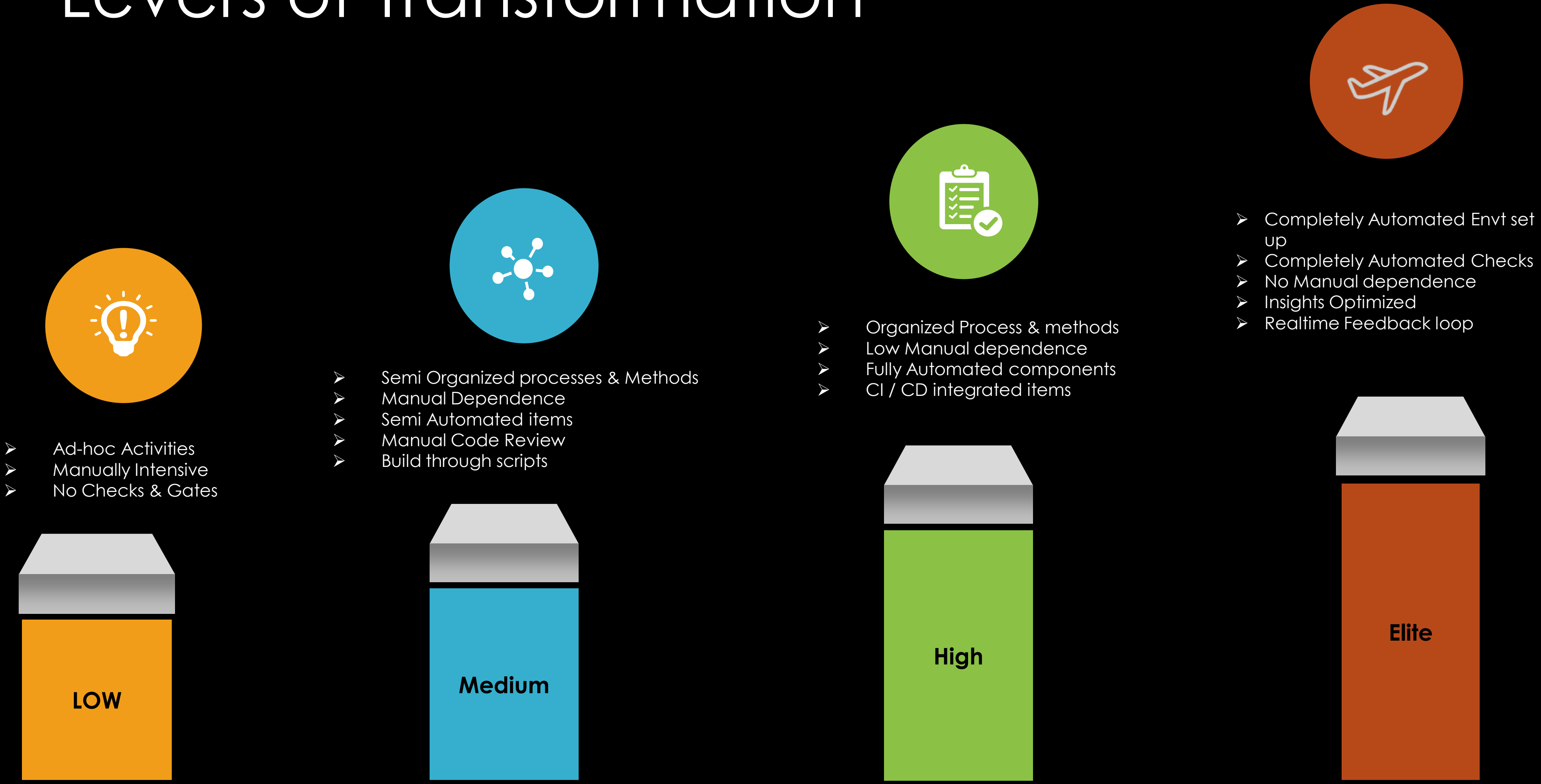
DevOps Maturity Model

brillio

● Requirements management ● Release and config management ● Product quality control ● Build & deploy efficiency ● Collaboration

Low	Medium	High	Elite
• No clarity on requirements exists, team takes requirements based on customer or SPOC inputs. • Roadmap not clear • Requirements are maintained ad-hoc	• Requirements are documented with scenarios , estimated with scenarios & is trackable. • Acceptance criteria are defined for each requirement	• Requirements are tracked through ALM with prioritization. • Acceptance Criteria are mapped to requirements in the tool. • Requirements elaboration and clarifications	• Requirements are integrated with other tools in the lifecycle that exhibits real time traceability from requirements to code to tests • Automated functional testing and test coverage tracked. • Working software is an integrated Product
• Code is maintained in local machines, no branching and release strategy	• Version control tool used for source Defined Branching mechanism. • Deployment instructions and release notes maintained and version controlled.	• Version control tool is integrated with other tools from CI pipeline. • DDL and DML scripts are versioned controlled • Environment variables are versioned controlled. • Frequent check-in and commits are tracked via tool.	• Automated merge practiced • Smart commits are in practice that integrates the with ALM/CI pipeline • Feature toggling to switch on/off.
• Informal review process. • Ad-hoc Unit/functional testing. • Defects are not captured at any stage in development process	• Manual Code review. • Verification coding using excel based checklist • Manual Unit testing. • Capture and track using tool.	• Automated code review. • Static Code Analysis tools integrated with CI. • Automated Unit/functional testing. • Defects tracking and Code coverage captured through integrated CI pipeline. • Metrics defined to measure effectiveness	• Load and performance testing done in indented environment. • Security vulnerability and performance optimization. • TDD / BDD practices inbuilt into delivery
• Informal review process. • Ad-hoc Unit/functional testing. • Defects are not captured at any stage in development	• Automated build using scripts or tool. • Data Migration and deployment scripts Elementary notification mechanism. • Manual release notes and deployment instructions as practice.	• Automated CI build trigger (schedule/polling). • Deployment scripts integrated with CI. • Data Migration scripts integrated with CI. • CI/CD pipeline setup.	• Environment provisioning through CI/CD. • Single touch deployment to next environment. • Integrated BOLT dashboard. • Automated Rollback on failure. • Application monitoring
• Unstructured team meetings and status Updates. • Norming or storming stage of team formation.	• Project meetings or feedbacks are planned are actions are tracked.	• Project team uses ALM tools to plan and track their tasks & actions • Team constantly retrospect's and enables continuous improvement in the way they work • Team solicits periodic feedback from customer stakeholders	• Real-time dashboards in ALM that reflect the status Team leverages quantitative data to retrospect & course correct. • Project team members engages with customers that impacts product value. • Success stories that demonstrate Brillio capabilities.

Levers of Transformation



- ➢ Ad-hoc Activities
- ➢ Manually Intensive
- ➢ No Checks & Gates

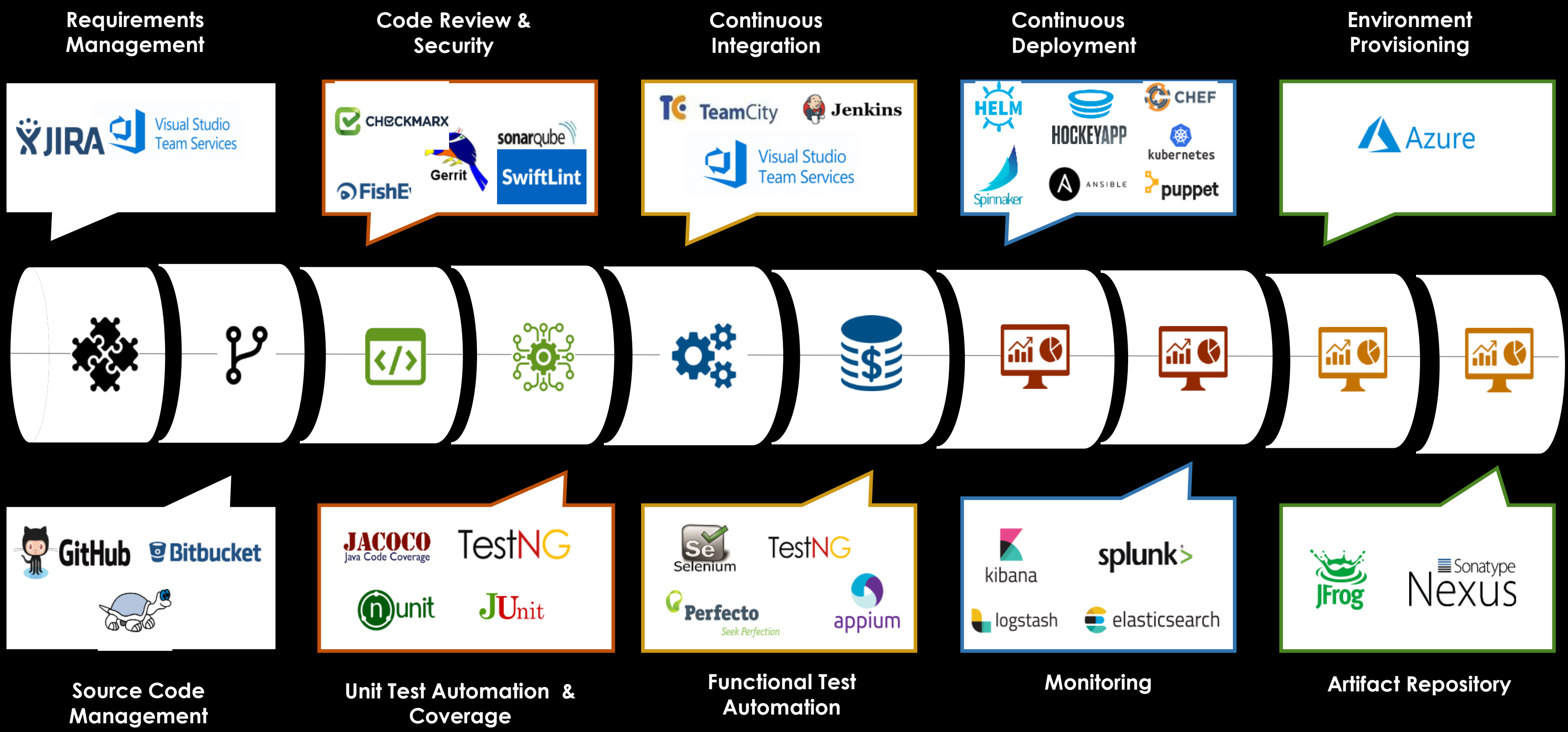
- ➢ Semi Organized processes & Methods
- ➢ Manual Dependence
- ➢ Semi Automated items
- ➢ Manual Code Review
- ➢ Build through scripts

- ➢ Organized Process & methods
- ➢ Low Manual dependence
- ➢ Fully Automated components
- ➢ CI / CD integrated items

- ➢ Completely Automated Envt set up
- ➢ Completely Automated Checks
- ➢ No Manual dependence
- ➢ Insights Optimized
- ➢ Realtime Feedback loop

brillio

Tools Ecosystem – Choose Right Do Right



Requirements Management
- JIRA
- Visual Studio Team Services

Code Review & Security
- CHECKMARX
- sonarqube
- Gerrit
- FishEye
- SwiftLint

Continuous Integration
- TeamCity
- Jenkins
- Visual Studio Team Services

Continuous Deployment
- HELM
- HOCKEYAPP
- CHEF
- kubernetes
- Spinnaker
- ANSIBLE
- puppet

Environment Provisioning
- Azure

Source Code Management
- GitHub
- Bitbucket

Unit Test Automation & Coverage
- JACOCO Java Code Coverage
- TestNG
- nunit
- JUnit

Functional Test Automation
- Selenium
- TestNG
- Perfecto Seek Perfection
- appium

Monitoring
- kibana
- splunk>
- logstash
- elasticsearch

Artifact Repository
- JFrog
- Sonatype Nexus

Brillio BOLT^{TM} – Actionable Insights

01 REQUIREMENTS MANAGEMENT

02 VERSION CONTROL

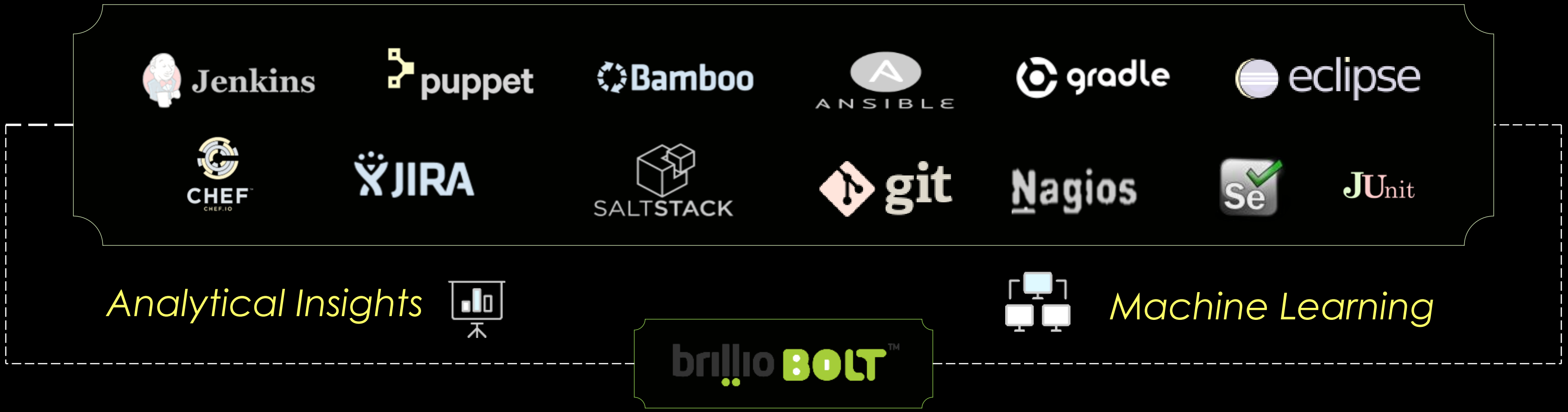
03 CODE REVIEW & ANALYSIS

04 CONTINUOUS INTEGRATION

05 TEST AUTOMATION

06 ENVIRONMENT PROVISIONING

07 CONTINUOUS DEPLOYMENT



Jenkins, puppet, Bamboo, ANSIBLE, gradle, eclipse, CHEF, JIRA, SALTSTACK, git, Nagios, Se, JUnit

Analytical Insights

Machine Learning

brillio BOLT™

Project Health

Release Predictability

Actionable Insights

Team Performance

Cost Optimization

Better Decision Making

Accurate Planning

Increased Efficiency

Techstack Tool View

	Requirements Management	Source Code Management	Code Review & Security Compliance	Unit Test Automation & Coverage	Continuous Integration	Functional Test Automation	Continuous Deployment	Monitoring	Environment Provisioning	Artifact Repository
Java	○ Azure Boards	○ Azure Repos	○ SonarQube ○ Gerrit	○ Junit ○ Jacoco	○ Azure Pipelines	○ TestNG ○ Selenium	○ Ansible ○ Kubernetes ○ Spinnaker ○ Helm ○ Azure Pipelines	○ Azure Monitor	○ Azure	○ Azure Artifacts
.Net			○ FXCop/Resharper	○ Nunit ○ NCover		○ MTM ○ Selenium	○ Ansible ○ Kubernetes ○ Spinnaker ○ Helm ○ Azure Pipelines			
Mobile			○ SonarQube ○ Gerrit	○ XCTest ○ Fastlane ○ Junit		○ Perfecto ○ Selenium ○ Appium	○ HockeyApp			
Integration			○ SonarQube ○ Gerrit	○ Munit ○ Munit ○ Coverage		○ Selenium	○ Ansible ○ Kubernetes ○ Spinnaker ○ Helm ○ Azure Pipelines			

Our Philosophy

Continuous Planning:

- Identify Key stakeholders and their roles and responsibilities
- Define process and strategies for various aspects of application lifecycle.

Continuous Integration:

- Provides tools to automate build & testing environment, similar to development
- Provides feedback early in the lifecycle to correct code
- Guarantees quality code with min defects and max coverage

Continuous Testing:

- Helps team to analyze code fails and provide quick fix
- Automation is leveraged across all the phases of testing
- SmarTEST Automation Accelerator frameworks

Continuous Deployment:

- Minimizes lead and cycle time from coding to deployment and production
- Quicker feedback ensures deployment ready quality deliverables

Continuous Insight:

- Provides insight into progress during development using metrics and analytics
- Serves as continuous quality assurance mechanism, transparent to the customer

Continuous Improvement:

- Identify gaps in the existing framework and improve
- Tailor existing process to achieve faster cycle times, minimize wastage and maximize productivity

Containerized Application

On-Prem

Cloud

Hybrid

Analyze

- Application Analysis
- Architecture Analysis
- Target Deployment Envt.
- Component
- Access control
- Availability & Usage
- Service Requirements
- Mandated toolsets

Recommend

- Assessment & Recommendation of Cluster components
- Cluster Architecture & resource Recommendations
- Tool Recommendation
- Decoupling of services

Design

- Guide team on Micro Service Arch
- Containerization
- Tools & Methodology
- Deployment Intricacies
- Scaling
- Dev Ops Processes

Configure

- Cluster Set up on Target Envt
- Access control
- Service Accounts
- Cluster Monitoring
- Image Registry
- Infrastructure Automation

Build

- Docker File Creation
- Docker Image Build
- Docker Image Storage
- Image Versioning
- Deployment Yamls Creation
- Component Yamls
- Service Yamls
- Config maps

Deploy

- Build Processes
- Deployment of Components on Cluster
- Deployment of Application
- Deployment Fine Tuning
- Orchestrator Configuration
- Build Pipeline Automation

Monitor

- Build Process Finetune
- Pipeline Fine tuning
- Container Testing
- Cluster health monitoring

Thank You!