DevOps Landscape

APM

Jenkins, TeamCity

DevOps Tool Orchestration

End-To-End CD Pipeline Management

DevOps Landscape

SVN		
GIT		Perforce
	TFS	

CHEF		
MAVEN		DOCKER
	PUPPET	

JUnit		
SONAR		Selenium
	Gerrit	

SERENA		
uDeploy		CollabNet
	Release	

Azure		
Docker		AWS
	vmware	

SCM – Code & Commit   Build and Config   Scan and test   Release management   Deploy

Workflow management   feedback   feedback
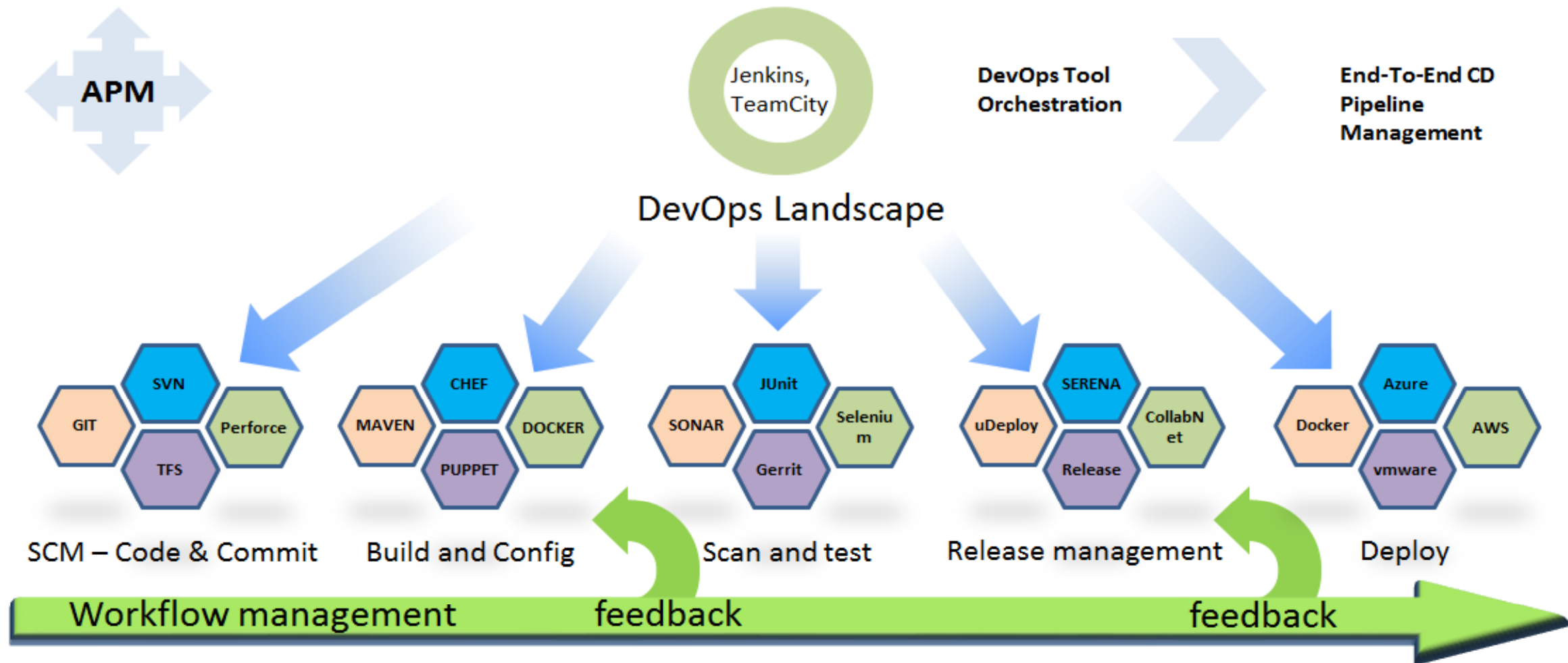
Industry leading DevOps tools

- Version Control system : SVN, GIT, TFS
- Code quality and test validation: Sonar, IntelliJ
- Build automation: Maven
- Continuous Integration and delivery: Jenkins, TeamCity, Bamboo, TFS
- Configuration management: Chef, Puppet, Ansible, Docker
- Application performance and monitoring (APM): Nagios, Zabbix
- Log analysis: Logstash, Elasticsearch

The deployment pipeline concept
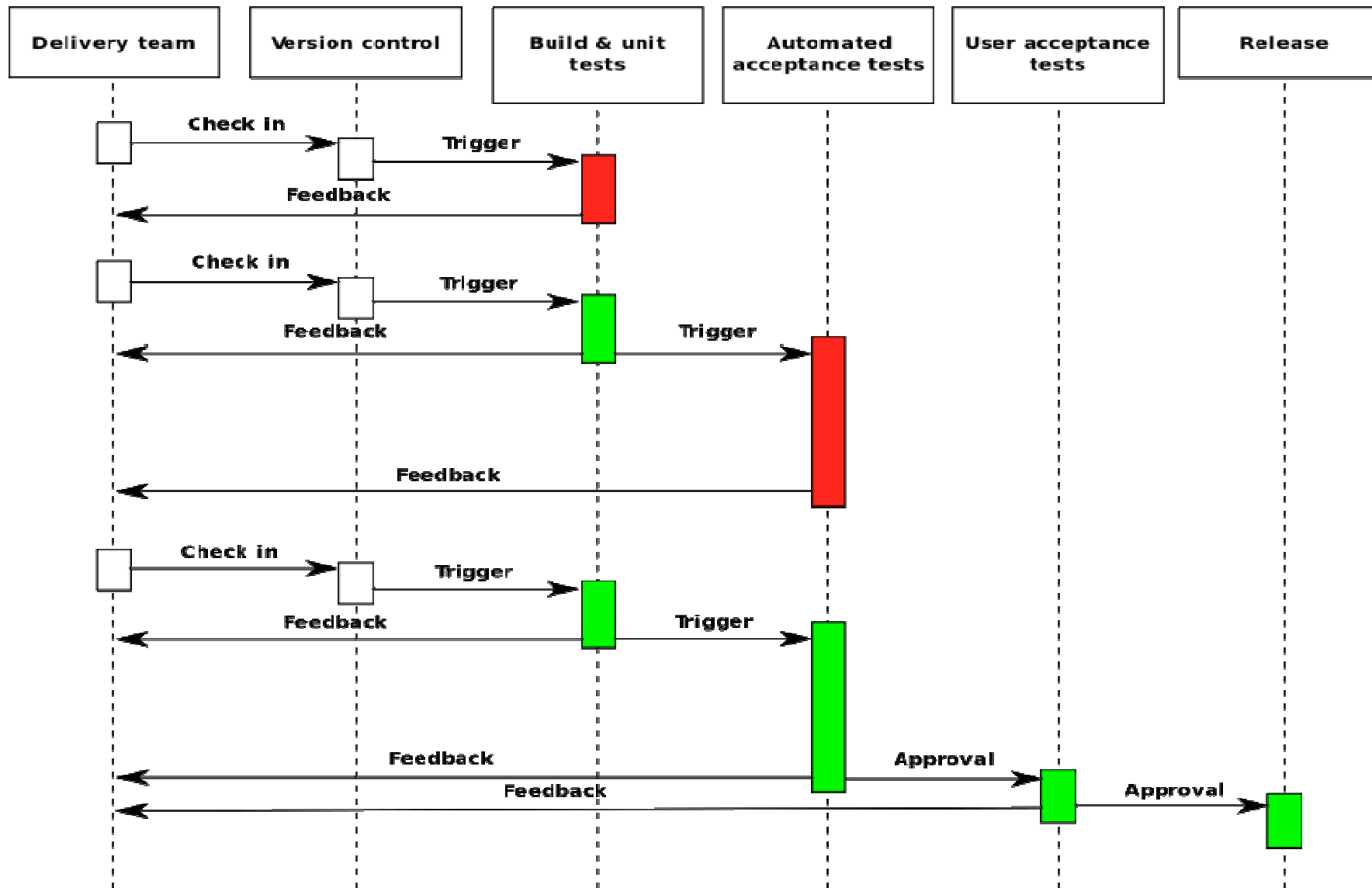
The deployment pipeline concept

Continuous Delivery Maturity Matrix

	Novice	Beginner	Intermediary	Advanced	Expert
Build	Verification before commit Nightly build	Ci server triggered by commit Artifacts are managed	No build scripts - only configurations Dependencies are managed	Distributed builds Staged build sequence	Build on VM CI server orchestrate VMs
Test + QA	Unit Test Code Coverage	Additional code metrics Mock-ups & proxies	Peer-reviews Automated Functional Test	Automated Deployment Maintain test data Test in target	Automated Acceptance Test
SCM	"Early Branching" Branches used for releases, merges are rare	"Late branching" Branches used for work isolation, merges are common	All commits are tied to tasks Individual history rewrites In DVCS	Pre-tested Commits Integration branch is pristine	Automatically generated release notes and audit trails
Visibility	Build status is notified to committer	Latest build status is available to all stakeholders	Trend reports Build status can be subscribed to (pull vs push)	Monitors in work areas show real-time status	Build reports and statistics are shared with customer

Benefits

- Innovate faster

- Fast to Market - More responsive to business needs

- Better collaboration

- Better software quality

- More frequent releases – new features added.

Best Practices in DevOps

- Active partnership and close coordination among the stake holders in establishing DevOps culture.
- Implement DevOps in totality. Avoid partial implementation, can become a reason for failure.
- Choose right tool for each phase in DevOps implementation.
- Options of substituting a exiting tools should be taken solicitously. No Fancy ideas.
- Give equal importance to log analysis, report generation and circulation.
- Mindset to adapt to changes.

Thank You

Email us – support@intellipaat.com

Visit us - https://intellipaat.com