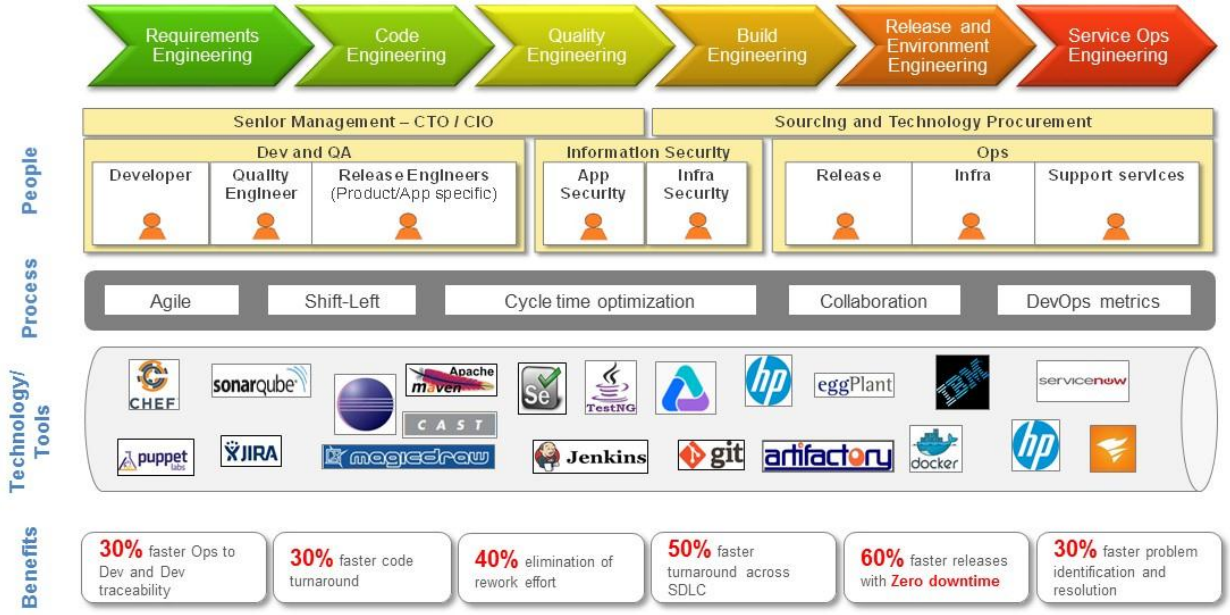
TCS DevOps

DevOps is a way of doing IT where the developers, the QA and the operations work as a single unit using collaboration, automation on the basis of Agile principles and practices. This is of foremost importance in terms of achieving **enterprise IT agility** for the customer organization. While agility and collaboration is quite common in developer and QA world, and the Agile practices, tools and methodologies have matured to a large extent, Agile operations which include infrastructure and service management is typically not mature. Hence, DevOps focuses on first on Agile operations and links it with Agile development (incl. QA embedded).

DevOps = Agile Dev + Agile Ops

(Where, Dev = development + QA; Ops = infrastructure & environment + release + IT support services teams)

The TCS point of view for DevOps consists of DevOps being realized through a set of six engineering principles and associated practices. Each of these principles have a bearing on and across the people, process and technology dimensions as depicted below:



At an implementation level, an organization may start with release and environment engineering which consists of creating the basic technical platform (other than associated people and process changes) for faster release of code to production. However, given the propensity of corresponding teams (Dev, QA or Ops) to move towards higher agility and seamless IT, the journey may start anywhere on the aforementioned flow. Based on whether agility is sought first in pre-prod/ SDLC or in prod (for say, support services) a shift-left or a shift-right approach may be taken.

For some of TCS' customers who have matured into an end-to-end DevOps practice, primarily in pre-prod till rollout for some of their IT systems, following are the pillars based on which implementations have been executed:

1. **On-demand infrastructure provisioning**: Blueprint-based infrastructure provisioning either on-premise or on cloud infrastructure, where the developer or tester can view infrastructure simply as a compute resource.
2. **Continuous-X** (where, X = integration / delivery & deployment / monitoring / testing): Using various tools to facilitate continuous automation pipelines such as Jenkins & Hudson (for integration), GoCD & XLDeploy (for delivery & deployment), Zabbix & SolarWinds (for monitoring) and SoapUI, HP LoadRunner, Selenium, Perfecto, et al (for testing).
3. **In-sprint Automation**: To ensure that testing and development happens in the same Sprint, thereby supporting concepts such as TDD/BDD, in a typical Agile Scrum SDLC.
4. **Zero Downtime Deployment**: Zone-based deployment to ensure working software availability at all times in production.

How TCS strategizes DevOps for customer's IT

As aforementioned, TCS proposes a progressive approach to adopting DevOps across customer's enterprise IT based on the current state of IT agility, and requirement for adopting DevOps across its departments.

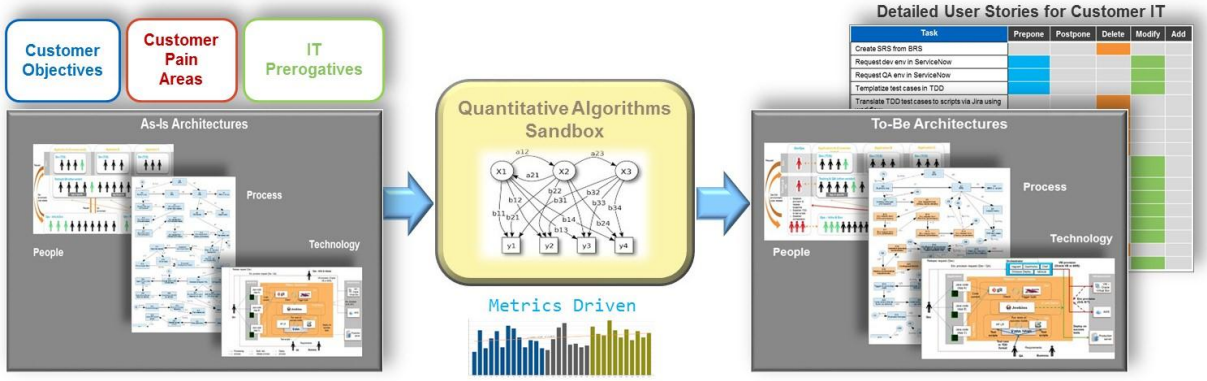
Nevertheless, given that DevOps caters across the dimensions of people, process and technology/tools, TCS proposes a **DevOps workshop** at the beginning of proposing any degree of DevOps roadmap to understand customer's IT landscape and status of DevOps adoption, and current pain areas; and come out with the recommended approach including the roadmap with timelines, working prototypes (optionally – based on scope and time of workshop), metrics, and identification of pilot projects to validate the approach and key benefits' realization.

Day 0	Days 1 and 2	Day 3	Day 4	Day 5
Pre-engagement Readiness through data gathering and provisioning system access	**Kick-off** Identify focus areas & Provide right knowledge to define DevOps 'Future State'	**Discovery** Initial identification of opportunity approaches for improvement	**Assessment Preparation** Create a tailor-made roadmap, & preliminary review with customer	**Finalized Plan** Presentation and next steps for phase-wise DevOps implementation

(**Note** – the workshop duration would be based on customer needs, priorities and scope for DevOps; a 5-day workshop is minimum recommended with both the Dev and Ops teams)

TCS' DevOps CoE (Center of Excellence) has a **Quantitative DevOps assessment framework** that enables drawing out the detailed implementation stories, To-be architectures (people, process and technology) and metrics; thereby bridging the gap between strategy and actual

implementation on-ground. This works as the launch pad for further programs to realize a given state of DevOps.

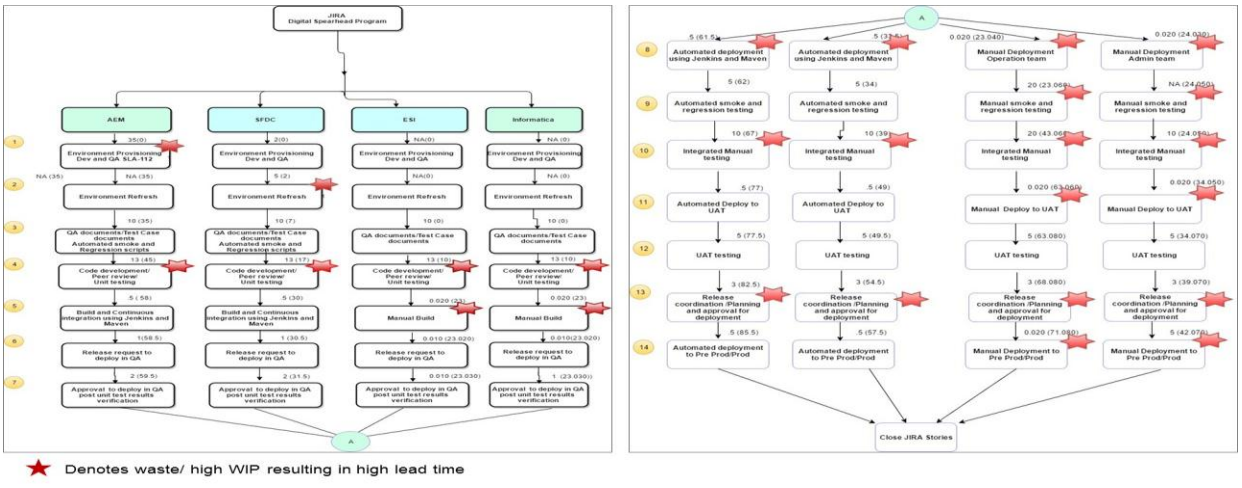


TCS follows a metrics driven approach to realize DevOps. Following are the four key metrics, apart from a full-blown metrication framework, that is determined for the As-is state (vis-à-vis the envisaged desired state) for any implementation:

- Release frequency for changes
- Lead time to change (from pre-prod to prod)
- Change success ratio (prod v/s pre-prod)
- Mean-time-to-failure (MTBF) / Mean-time-to-repair (MTTR) in prod

Following is a high level representative process analysis, as a subset of applying the aforesaid framework, for a global publishing customer:

(The customer needs to establish a one-click deployment automation pipeline catering to multiple technology stacks; each application platform encompasses such stacks with dependencies that need to be arduously taken care of during every release, resulting in high slippage of effort and schedules for both Dev and Ops teams)



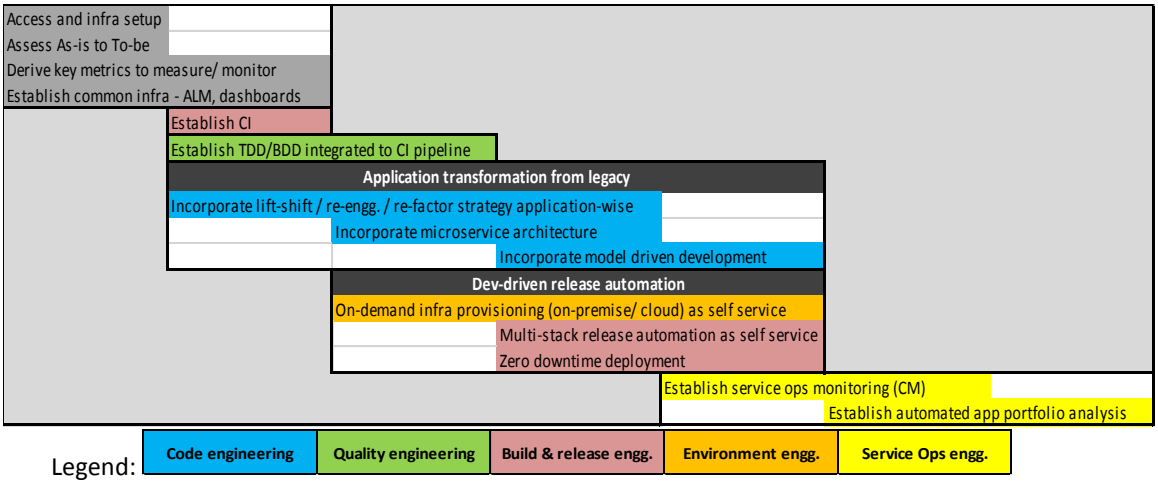
How TCS operationalizes DevOps based transformation for customer's IT

Following are representative approaches to progressive transformation based on As-is methodologies:

1. For customers who solely or primarily operate projects using Waterfall methodology, TCS proposes a bi-modal approach to start with, where select projects (based on criteria such as complexity, need for IT agility and business impact) are piloted in Agile. The transformation from a primarily-Waterfall to primarily-Agile project execution approach is recommended over time with gradual adoption of continuous integration, continuous delivery and so on.

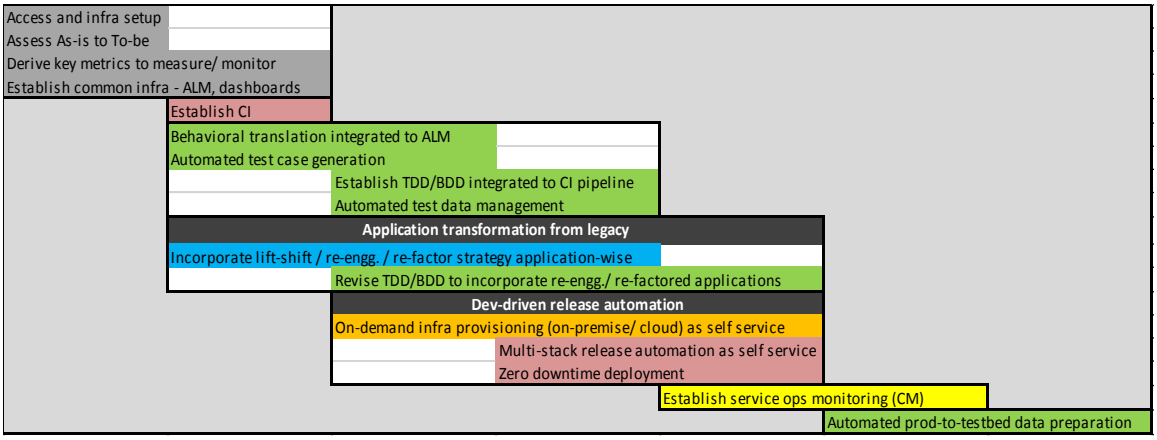
2. For customers already into Agile (or in advanced stages of automation and IT processes' and systems' integration), based on customer's IT life cycle pain points with respect to (a) how Dev and Ops teams work together, and (b) automation footprint; the roadmap would typically start with:

    - Dev team if primary pain point indicates problem areas on (a) how code is managed and engineered vis-à-vis IT agility, (b) effort spent by the Dev team in carrying out non-coding activities. This would lead towards gradual adoption of practices such as Dev-driven Ops, microservices, model driven development, in-sprint test automation and continuous integration.
        - From an engineering perspective, following would be the high level stories with indicative implementation sequence:

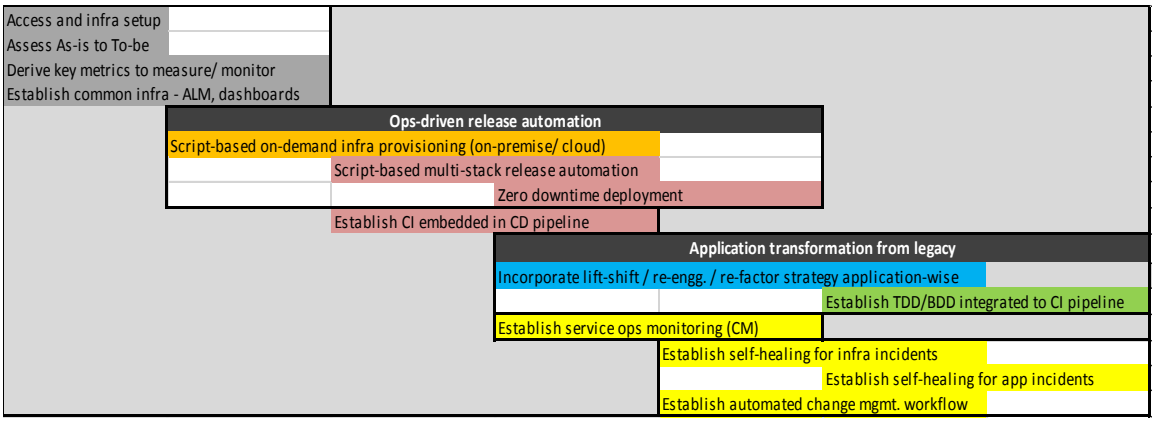


    - QA team if primary pain point indicates problem areas on how overall assurance services are delivered vis-à-vis IT agility. This would lead towards gradual adoption of practices such as test driven development (TDD) or behavior driven development (BDD), continuous testing and continuous integration.

- From an engineering perspective, following would be the high level stories with indicative implementation sequence:



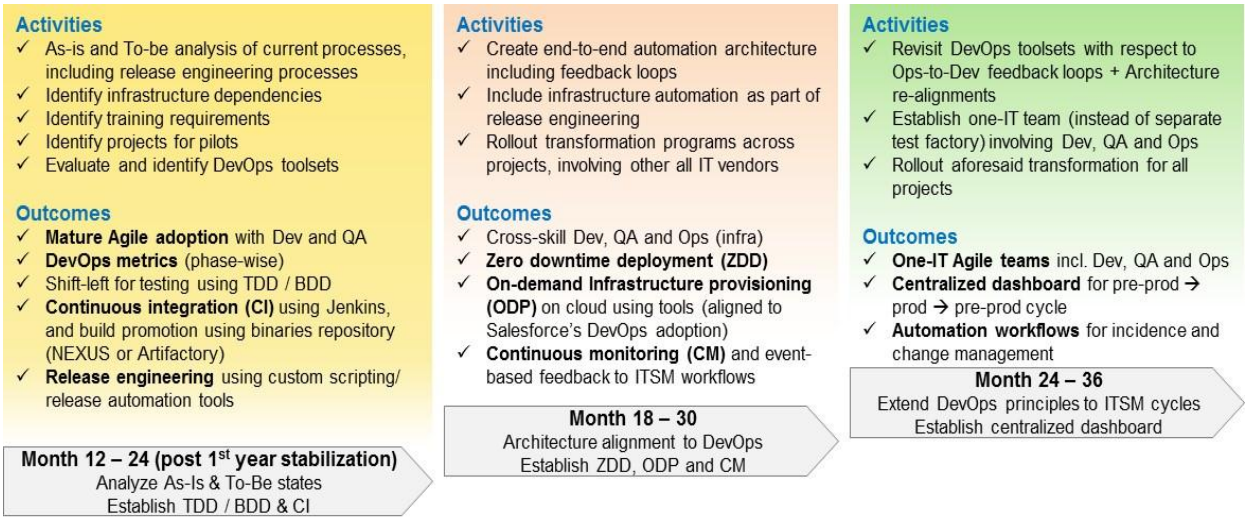
- Ops team if primary pain point indicates problem areas on how infrastructure, releases and support services are delivered vis-à-vis IT agility requirements. This would lead towards gradual adoption of practices such as release engineering incorporating one-click deployment automation possibly incorporating zero downtime deployment, Ops-driven Dev incl. PaaS enabled IT delivery over continuous delivery pipelines, and automated ITSM incl. self-healing systems.
  - From an engineering perspective, following would be the high level stories with indicative implementation sequence:



A representative roadmap for a customer with the following profile, wanting to start on the DevOps journey, would be thus:

- Understands Agile and pursues quite a few programs using Agile methodology
- Primary champion for driving IT agility is Dev team (incl. QA); pain points for the team includes spending substantial time and effort doing non-coding activities, especially helping out Ops during releases

- Technology stack includes Java, .NET, SFDC; thereby requiring an unified deployment automation approach



Key Benefits with associated metrics

1. Reduction in overall application change cycle time by as much as **70%** (key metric: lead time to change)
2. Faster time to market for applications by as much as **50%** (key metric: release frequency, application change success ratio)
3. Faster incident resolution, including permanent fixes, by as much as **30%** (key metric: mean time to repair [MTTR], mean time before failure [MTBF], reduction in problems by RCA]
    - bringing in robotic automation for self-healing may result in human person-effort saving of **70%**
4. Higher on-demand infrastructure/ environment availability and reliability (key metric: infrastructure/ environment %availability, %utilization, change success ratio]
5. Reduction in effort by **20%** in service management through incident and change life cycle automation (Ops to Dev feedback)
    - A gross saving of at least **30%** effort in IT service management in production (including savings from SDLC due to higher quality and reduction in cycle time)
    - Higher delivery accuracy and predictability; more time for developers to focus on actual coding activities through self-service led automation

Leveraging TCS' DevOps CoE Solution Assets

Apart from the quantitative assessment framework, TCS' DevOps Center-of-Excellence owns and manages a set of solution assets (which is constantly and gradually being built upon as a catalog of services) that can accelerate given DevOps implementation from a technology automation standpoint. **The solutions can drive an effort saving of 30% or more for typical DevOps-led automation implementations.** The solutions are extendable with respect to people and technology considerations for a given customer engagement; hence, the base tools or platforms that implement any given solution can be replaced by alternative tools (or, platforms) based on customer requirements.

Following are some of the solutions available with the CoE, with a brief note on the same:

Immutable Blue-Green deployment on cloud for .NET application (incl. automated DB deployment)*	The solution outlines an immutable infrastructure based zero downtime deployment on AWS cloud (using Packer and Terraform to provision using AMI) for a .NET application and its associated database
DevOps using robotic process automation (RPA) [DevOps 2.0]*	The solution outlines a typical automation workflow starting with provisioning -> deploy -> test -> monitor, for an application using adaptive software bots, on AWS
Deployment automation using XL Deploy*	The solution is to showcase a typical visual configuration led deployment automation on XL Deploy, that can hence enable a unified deployment approach for multiple stacks
Deploying a package to SFDC environment using XL Deploy	Same as above, however specific to SFDC stack
Deployment using XL Deploy with REST API to Oracle WebLogic environment	Same as above, however to showcase extensibility of the XL Deploy platform for stacks supporting REST API based deployments
CI / CD for SFDC using Jenkins and Gearset*	CI / CD for SFDC stack, in case a heavy solution (say, using XL Deploy) may be an overkill
Puppet and Python (script) based deployment automation	The solution outlines script based deployment automation, thus enabling fine-grained control on the parameters for deployment that can also support multiple stacks

***** Working demos available with CoE

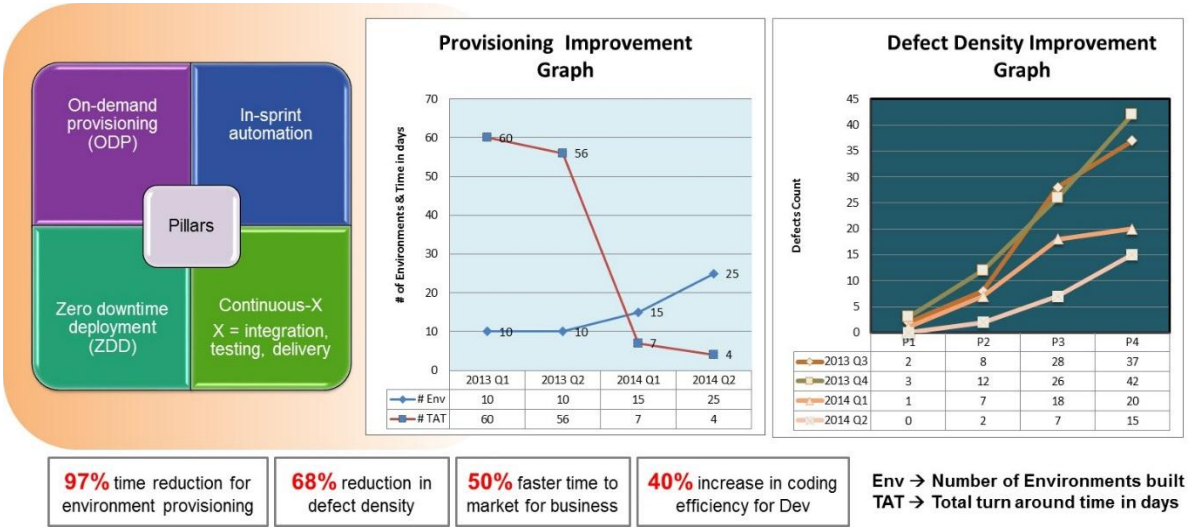
The TCS delivery team engages with DevOps CoE for asset based accelerated solution delivery. DevOps CoE provides required technical expertise through professional services for this approach by (a) assessing customer landscape for solution fitment, (b) supporting the delivery team and customer FTEs in PoCs for closing the fitment to the given requirement, (c) supporting actual implementation and configuration of the solution assets, with relevant extensions, and (d) providing secondary support with relevant training, if required, during handover-takeover phase between the CoE and TCS delivery and customer teams.

In case the customer and TCS (with other vendors, if applicable) sets up a DevOps group in the customer's organization to drive DevOps initiatives across the enterprise, TCS' DevOps CoE can

be leveraged by the TCS delivery team as outlined above, however subject to relevant information security policies with respect to the solution assets being deployed.

Success Stories

- A global publishing house required to better manage its global customer services platform – its core customer engagement system for education and publishing domain – in terms of agility, security and integration to multiple third party vendor ecosystems. To achieve this, TCS has provided the solution over a multi-year program in terms of creating a custom DevOps framework comprising of tools such as Jenkins (CI), Cucumber (BDD), Artifactory and NEXUS (binary repositories), Puppet (infrastructure configuration), Selenium, JUnit, SoapUI (test automation), Liquibase (database migration) and Maven (build), et al. Once they reached a given state of maturity, they have moved towards microservices architecture by leveraging Netflix Microservices platform. Substantial savings have been realized by the customer throughout the IT life cycle service chain from environment provisioning till service management optimization.



- A large US-based research firm is using full-fledged Agile practices for SAP implementations using people and process integration. They are currently working towards establishing Ops agility for faster infrastructure provisioning, and integrating such infrastructure with their Dev part; having started with faster problem resolution in production through substantial Agile-driven collaboration. The customer has realized considerable savings in terms of productivity increase and release time, including **5x productivity increase of the team**.

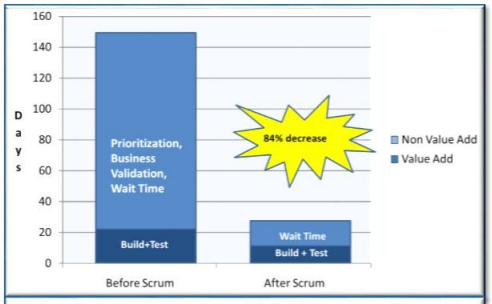
Client profile & TCS Engagement

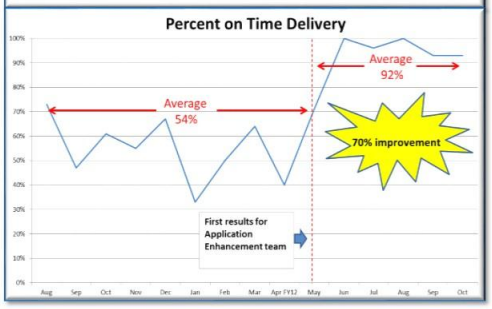
A Privately held Research company in the US, founded in 1964, with four main divisions across North America, Europe & Asia Pacific
Global Single Instance:
- ECC 6.0, SCM 7.0, SRM 7.0, CRM 7.0, BI 7.0 on Hana, BPC on Hana, IBP On Hana, PI, EP, MDM, etc.

9000+ users across the globe
TCS Runs their Global Production Support

The Context
- 5 times productivity increase with same team size
- Reduce Lead time for minor requests to < 1 month

Solution: Implement Agile across SAP teams





- TCS has assessed DevOps state and recommended a DevOps journey based on which a large consulting firm has implemented DevOps and realized substantial savings downstream **(>40% saving in release cycle time supporting multiple technology stacks; >30% increase of efficiency in handling production issues**) in overall IT from achieving agility and faster time-to-market for its' customized solutions.

Client profile

Leading management company based in the US having a diverse technology stack including Ruby, .Net, Java, Drupal, Oracle, SQL Server, MySQL and Oracle EBS
Implemented Agile Scrum since 2007

Business Context

- Ops and Dev teams working in silos
- Rigid enterprise processes
- 1 day lead time needed for QA deployment
- 3 days lead time needed for production deployment
- Frustrated sponsors due to high turnaround time
- No ownership of application issues

Key Benefits

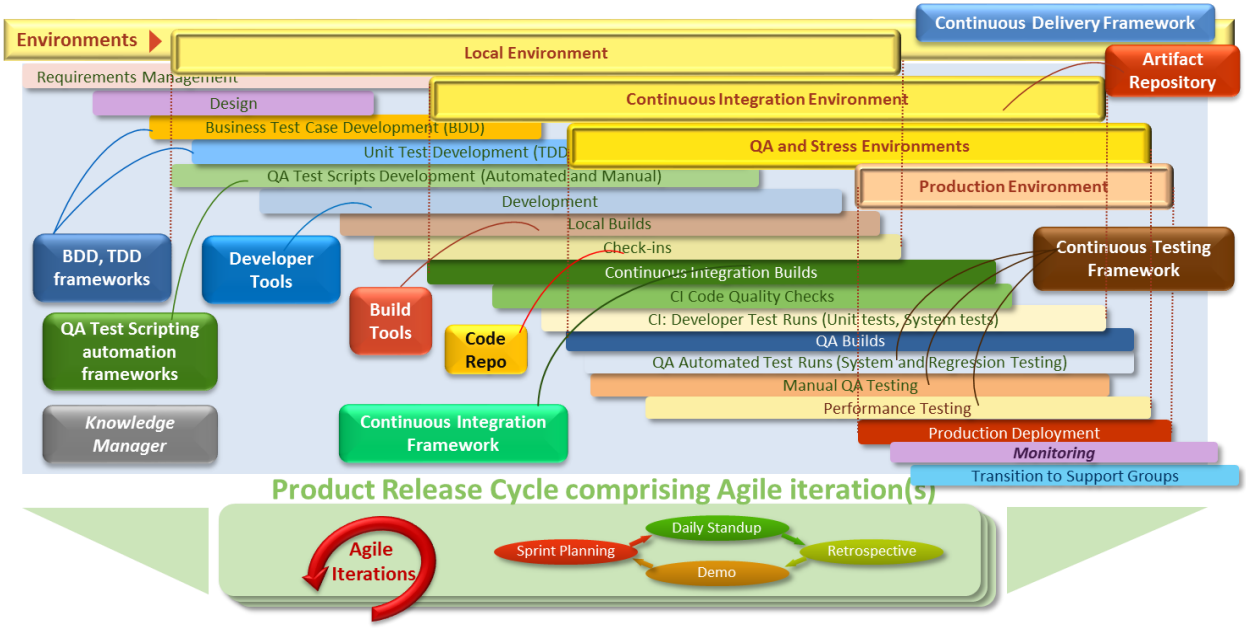
- ✓ **90%** reduction in end to end deployment time from start of SDLC
- ✓ **50%** reduction in resolution time for critical production issues
- ✓ **30%** reduction in occurrence of production issues
- ✓ High collaboration between Dev and Ops teams
- ✓ Ops team as an integrated part of application journey

TCS Solution

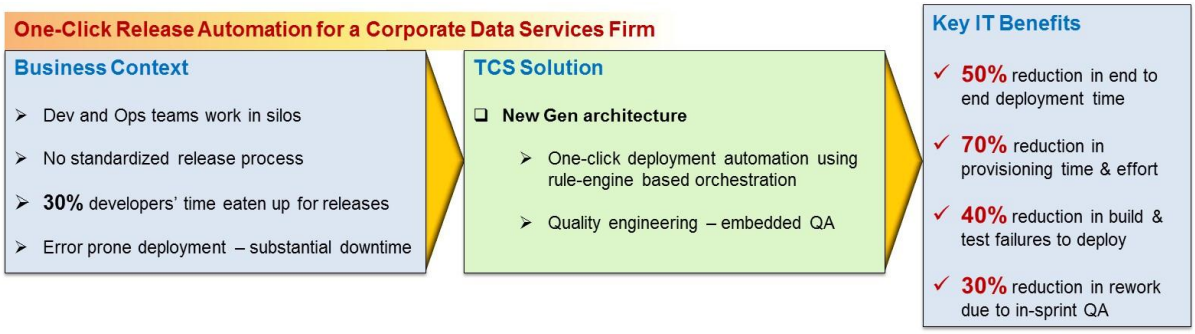
- TCS partnered with the customer to adopt DevOps
- Integrate Dev and Ops teams through SDLC
- One-click deployment in Ruby across all environments incl. production
- Dev team access to QA and prod servers and access to error logs for faster defect resolution
- Continual knowledge sharing across Dev and Ops
- Integrated QA – adoption of shift-left

- A global publishing firm is using XebiaLabs (XL Deploy) and Atlassian (Jira, Confluence) products to achieve a given state of IT agility (in release cycle time to production) through adopting an unified deployment model for multiple stacks, and high degree of collaboration

and end-to-end SDLC traceability. Given that the customer CTO wanted to separate out its product releases function through a dedicated 'product rollout' group, TCS has provided a detailed roadmap towards achieving DevOps for the new group implementation. Implementing the architecture (people, process, and technology) has resulted in substantially faster release cycles and high delivery quality across product lines in its portfolio.



- A US-based corporate services firm had a problem statement of (a) silo'ed Dev and Ops teams resulting in high amount of rework pre-release, (b) no standardization of release processes that further involve substantial manual effort, (c) substantial time to deploy (with high downtime) and error-prone deployments for core applications in .NET stack due to its in-house built legacy deployment platform. TCS DevOps CoE defined a new architecture to address the aforesaid problems by applying the assessment framework, and further implemented the architecture over a cloud based infrastructure (instead of existing on-premise, and decommissioning the proprietary in-house deployment platform). This resulted in **(a) 50% reduction in end-to-end deployment time, (b) 70% reduction in infrastructure provisioning time, (c) average 30% reduction in rework across build-to-test and build-to-deploy cycles**.



- For a large US bank, TCS has implemented **continuous integration for its mainframe platform primarily using IBM tool stack**, to speed up release cycle times. The pain points for the customer included (a) longer release cycles impacting business' time to market, especially for its credit card division, (b) people issues in migrating from one-stop text-based "blue-green" mainframe console to toolchain based distributed systems with GUI capabilities. TCS adopted IBM RDz Explorer (with IBM RTC) platform to integrate code, connecting backend mainframe to Windows based systems; code development and integration happened in Windows systems, whereas the actual deployment was done in mainframe using ChangeMan tool. Key benefits includes **(a) 6-8% effort savings in the development phase itself, (b) enhanced time to market from 9 months down to 6 months project-wise release cycles**.

- For a large Australian bank, TCS has implemented a **container-based (using Puppet, Ansible and Docker) infrastructure paradigm to support integration of multiple data systems to a single view of customer data, supporting Big Data analytics** by creating a data lake. This was done as a 3-year program involving (a) Big Data based development and infrastructure integration, (b) platform tuning and production support. This enabled the customer to help reduce costs, respond faster to customer requirements through established personalized customer service, and provide effective customer insights.

Infrastructure

- **Production:** 100 node : 750 TB (usable)
- **Analytics:** 120 node : 1.2 PB
- **Development:** 15 node
- Offline DR

Technology Stack	Tools
Hadoop Platform	Cloudera – CDH 5.3.3
Discovery	SAS, Qlikview
Data Access	Maestro
Data On-boarding	Sqoop. Kafka, Maestro
Data Processing	Hive, Scala, Scalding
Analytics	Python. R
Workflow Automation & Monitor	Puppet, Ansible, Docker
Security	MS AD. Kerberos, Sentry

- For a large UK based financial institution having problems in managing its massive legacy IT estate resulting in error-prone and erratic deployments with SLA slippages (hence, high penalties in both regulatory non-compliance, and meeting business expectations), TCS implemented a streamlined **unified deployment automation infrastructure using IBM UrbanCode Deploy across environments.** The solution implementation resulted in key benefits in terms of **(a) 50% reduction in end-to-end application deployment time, (b) increased process visibility and flexibility in configuring new disparate deployments, (c) self-service based deployments freeing up developer's time from non-coding activities, (d) increased RoI of 200%**.

Challenges

- Slow and error-prone manual deployment; adverse impact to accuracy and SLA violations
- Difficult to track deployments in different environments resulting in poor visibility and control
- Chances of alteration of artifacts communicated over mails
- Changing regulatory requirements and audit compliance, and inefficient risk management process
- Plugins unavailability for rare technology stacks (e.g. TIBCO, Mesh, Control-M, IBM Cognos, etc.)
- Dependency on IBM UCD support team for service requests/incidents

Solution

- Deployment of over **115 applications** with team size of 15
- The program has L1 and L2 Support team for IBM UCD
- Heterogeneous mix of mostly Unix and few Mainframe applications
- **Application environment comprise of multitude of technology stacks** e.g. Java, JBoss, .NET, DWeb, SQL, SQL+, ORACLE, Sybase
- Development of the process steps/scripts for deploying the artefact
- Created Project Management Framework for the customer for smooth UAT
- Use of Application Acceptance Testing for quality checks
- Perform hand over for the newly established deployment workflow

- For a large US based financial institution having problems in managing its deployments that were error-prone and slow; given the multiple technology stacks to be supported, manual and semi-manual processes in substantial fragmentation, thereby causing issues with resolving multi-stack deploy dependencies. TCS implemented a **unified deployment automation infrastructure using IBM UrbanCode Deploy.** The solution implementation resulted in key benefits in terms of **(a) Eliminating manual effort for deployments resulting in 70% reduction in deploy cycle time, (b) increased visibility of deployment process, (c) flexibility in configuring blueprints for new and/or complex deployments.**

Challenges	Solution
▪ Slow and error-prone manual deployment ▪ No proper view of dashboard for keeping track of environment and application version ▪ Template creation of process steps for IPaS component deployment	▪ Deployment of over 324 application with team size of 3 ▪ Applications based on IPaS, Java, J2EE, .NET (.war, .ear) ▪ Deployment of patches/incremental updates to the applications ▪ Manage DB deployments aligned to application deployments ▪ Property file deployment of App Servers e.g. Weblogic, Jboss ▪ Target App and Web Servers in use – Weblogic, Tomcat, Jboss ▪ Dashboard: environments and application wise deployment usage

There are many other TCS customers who have embarked on the journey of DevOps with TCS as partner and are at different stages of maturity where TCS supports them.

For customers who are thinking to move towards DevOps and are still moving in an iterative or waterfall software delivery model, TCS DevOps CoE proposes guidance through a structured assessment and downstream implementation for DevOps. Consultants and engineers from the TCS DevOps CoE have the relevant skills and experience to guide the customer through the journey of DevOps and achieves a state of IT agility.