TESCO

IT Delivery Model

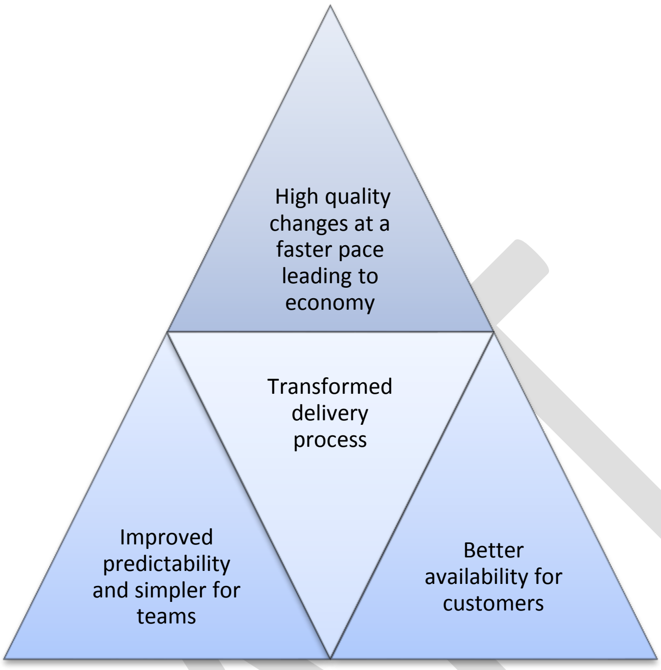
Strategy | Methodology | Implementation | Support | Evolution | Capability

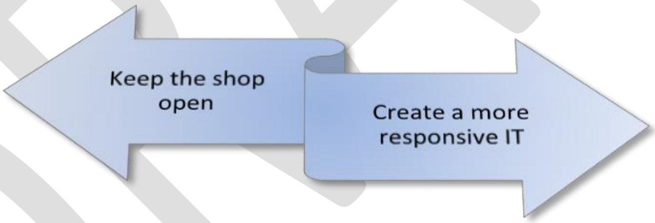
11/15/2013

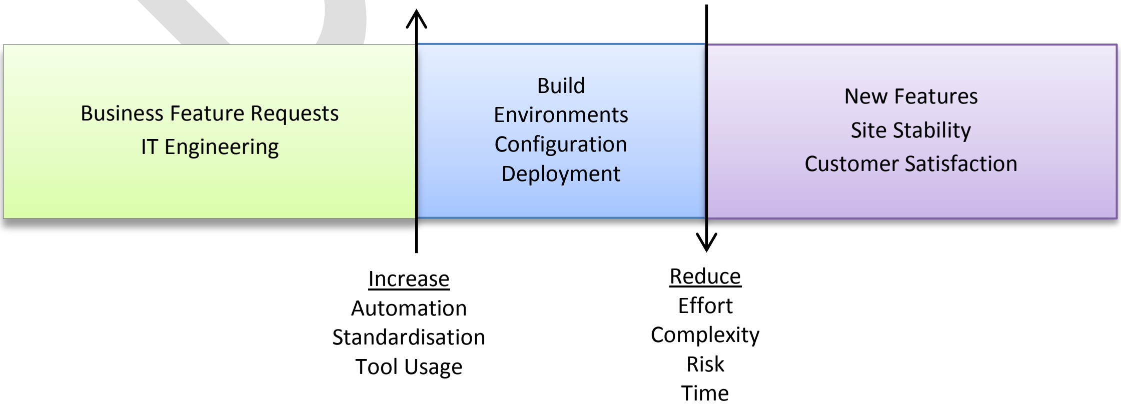
Table of Contents

Strategy

Vision


High quality changes at a faster pace leading to economy
Transformed delivery process
Improved predictability and simpler for teams
Better availability for customers





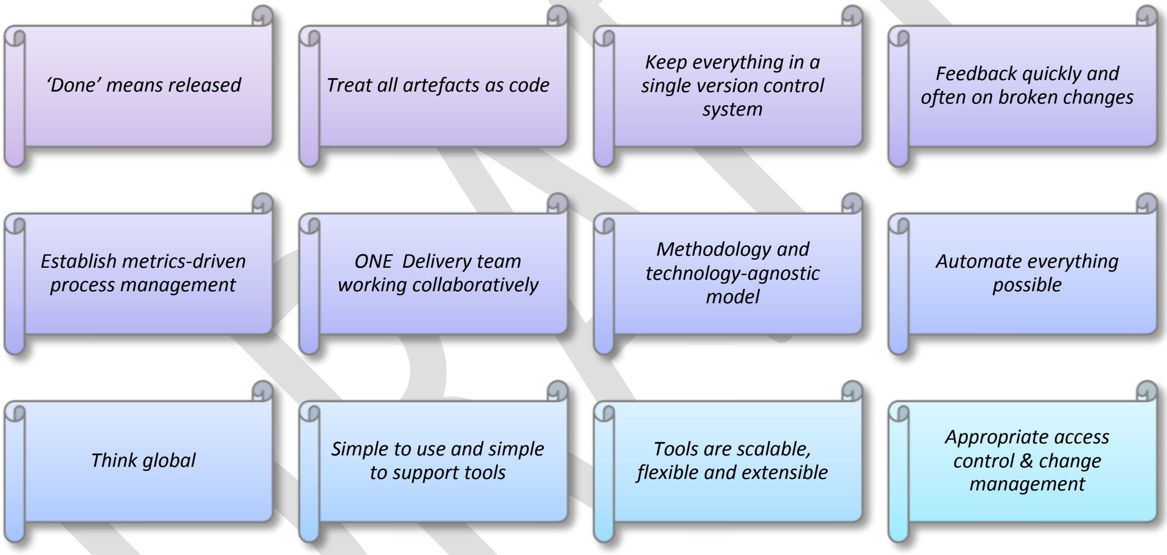
Model Overview

The IT Delivery Model is based around the concept of Continuous Delivery, which is focused on releasing software rapidly through build, test and deployment automation. This creates a reliable and repeatable mechanism that allows IT to be more responsive to business needs. DevOps, ITIL, Agile and Six Sigma elements are also incorporated to provide a holistic approach, tailored specifically for Tesco.

Continuous Delivery is a combination of continuous integration and a delivery pipeline:

- **Continuous Integration**: A method for ensuring applications are working and, if not, who and what broke it.
- **Delivery Pipeline**: An extension of continuous integration and models the process for getting applications built, deployed, tested and released.

Guiding Principles



'Done' means released

Treat all artefacts as code

Keep everything in a single version control system

Feedback quickly and often on broken changes

Establish metrics-driven process management

ONE Delivery team working collaboratively

Methodology and technology-agnostic model

Automate everything possible

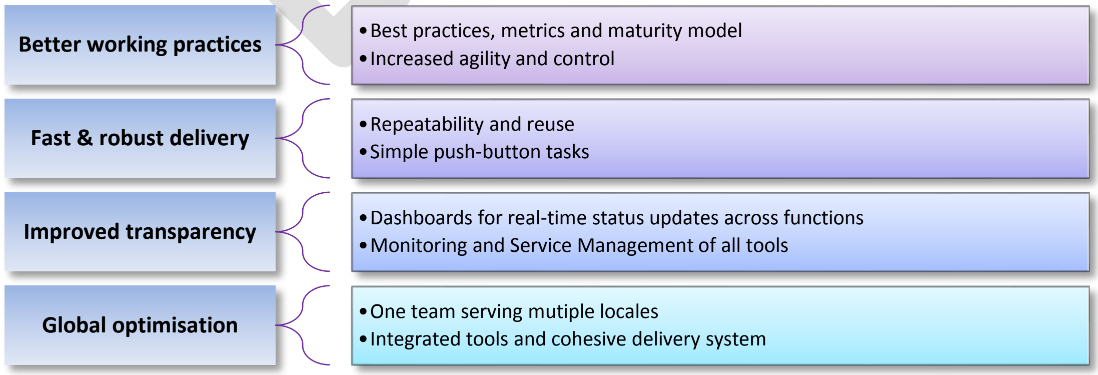
Think global

Simple to use and simple to support tools

Tools are scalable, flexible and extensible

Appropriate access control & change management

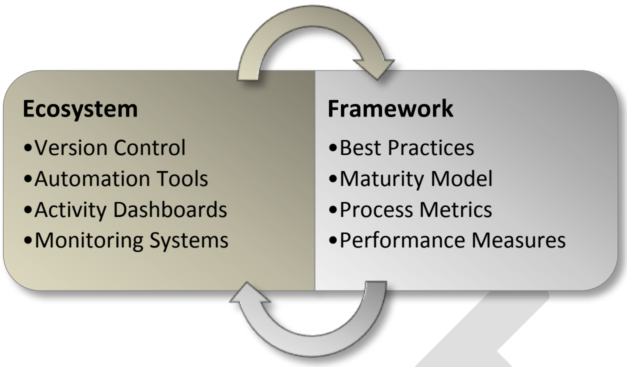
Outcomes of Transformation



Better working practices	• Best practices, metrics and maturity model • Increased agility and control
Fast & robust delivery	• Repeatability and reuse • Simple push-button tasks
Improved transparency	• Dashboards for real-time status updates across functions • Monitoring and Service Management of all tools
Global optimisation	• One team serving mutiple locales • Integrated tools and cohesive delivery system

Methodology

The delivery model is comprised of two overarching parts - an **Ecosystem** and a **Framework**:-



The Ecosystem and Framework provide the basis for re-engineering build, configuration, environment, deployment and testing tasks. The consolidation and integration of tools creates a global delivery model, leading to better resource utilisation, simpler processes and cheaper licencing costs.

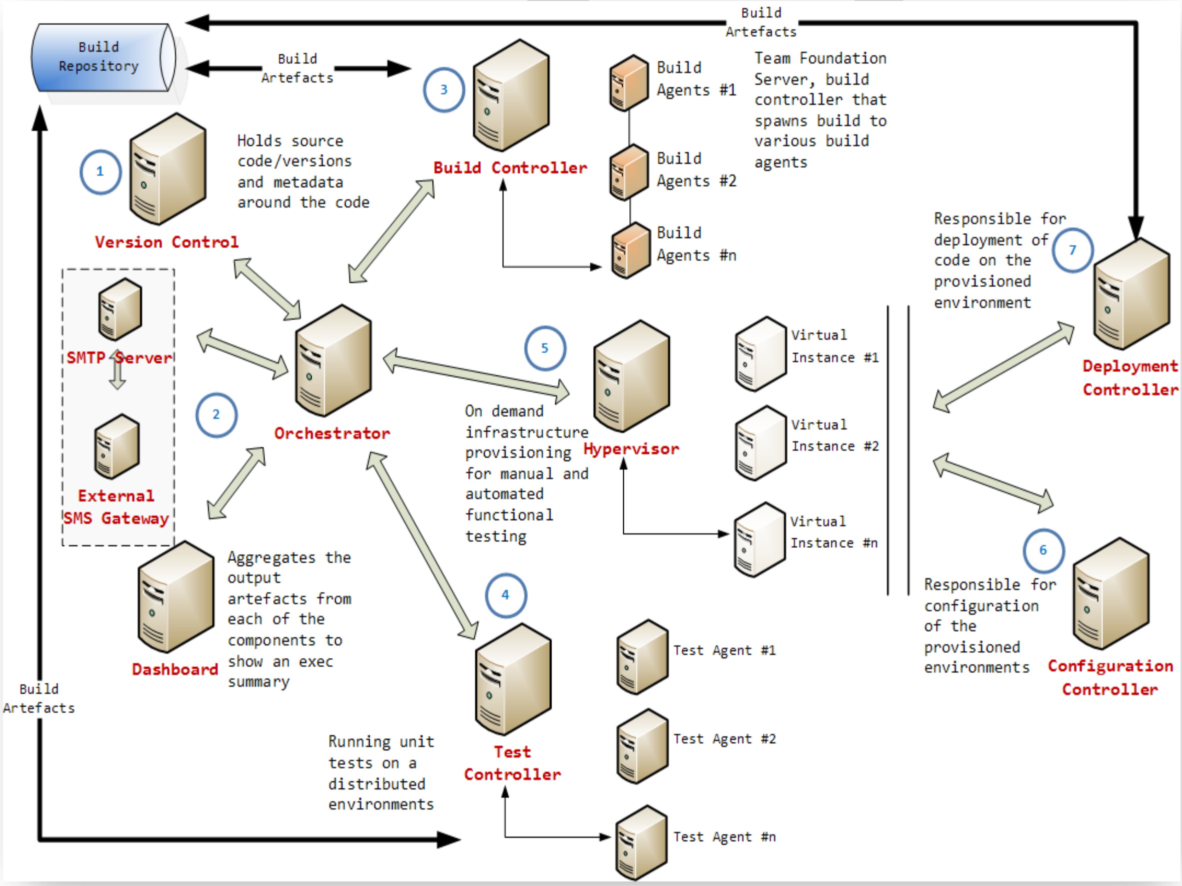
Process	Challenge	Solution	Outcome
Configuration	Artefacts stored in different locations	Everything in a single place	**Better:** Increased confidence in product quality
	Inconsistent environments due to untracked changes	All environments configured, monitored and healed via tools	**Cheaper:** Less project delays due to environment problems
Build	Human intervention required	Automatically trigger deployment upon completion	**Cheaper:** Faster time-to-release with less manpower
	Code quality unknown	Code quality measured	**Better:** Product quality
Environments	Hand-crafted – long time to build, prone to human error	Built automatically and right first time	**Simpler:** Easier to scale to meet business demands
	Static, congested, get polluted and degrade	Dynamic, can be rebuilt cleanly on-demand, each project has their own	**Cheaper:** Less project delays as queuing eradicated
Deployment	Manual, time-consuming, prone to human error	Automatic – right product, right place, right time	**Simpler:** Tasks made easier for staff
	Engineering and Operations use different tools and methods	Same process from start to finish	**Cheaper:** Less risk to systems and faster time-to-release
Testing	Resource intensive and time consuming	Faster execution with less overhead	**Better:** Product quality improved as staff freed up for exploratory testing
	Not thorough	Comprehensive and repeatable	**Cheaper:** Waste reduction

Ecosystem

The Ecosystem is essentially the tool chain that supports the delivery of business changes. While ownership may be split across Infrastructure, Engineering and primarily the Platform team, all tools are fully integrated to eliminate the process fragmentation that typically exists in traditional IT silos. Products have been selected or developed following a detailed evaluation and factored in usability, functional and non-functional requirements from Engineering, Service, Architecture and individual project teams. A centralised dashboard provides a real-time end-to-end view of project release activities and encourages collaboration.

The Ecosystem is designed to be neither too restrictive (teams forced into using sub optimal tools, thereby impeding quality, speed and morale) nor chaotic (unnecessary tool proliferation, integration problems, increased complexity and costs).

Logical Architecture


Build Repository
Build Artefacts
Build Artefacts
3
Build Controller
Build Agents #1
Build Agents #2
Build Agents #n
Team Foundation Server, build controller that spawns build to various build agents
1
Holds source code/versions and metadata around the code
Version Control
Responsible for deployment of code on the provisioned environment
7
Deployment Controller
SMTP Server
External SMS Gateway
2
Orchestrator
5
On demand infrastructure provisioning for manual and automated functional testing
Hypervisor
Virtual Instance #1
Virtual Instance #2
Virtual Instance #n
Dashboard
Aggregates the output artefacts from each of the components to show an exec summary
4
Test Controller
Running unit tests on a distributed environments
Test Agent #1
Test Agent #2
Test Agent #n
6
Responsible for configuration of the provisioned environments
Configuration Controller
Build Artefacts

Note:

- *This currently represents the Customer view only and is based on a tactical solution*

Toolbox

Dashboard & Orchestration

Jenkins

- Triggers automation tools in the Ecosystem
- Manages the top-level release dashboard and provides real-time status of continuous integration and the delivery pipeline
- Invokes virtual machines to provision virtual machine hosts
- Centralised end-to-end view brings Development and Operations staff closer together
- Open source

Test Controller

Se Cucumber

- Functional test automation suite
- Multi-platform, language and browser compatibility
- Behaviour Driven Development approach to create greater cohesion and transparency between testers and non-technical stakeholders to pass stories and acceptance criteria
- User-friendly, colour-coded HTML test report outputs which can be archived
- Can execute tests faster in a parallel processing architecture using Selenium GRID
- Open source

Configuration Controller

Puppet

- Configures the 'grey area' between the infrastructure and application layer
- Monitors servers and self-heals in the event of unauthorised or misconfigured change
- Designed to manage configuration in simple text files
- Provides the configuration catalogue for virtual machines
- Open source

Code Quality

sonarqube

- Reports on duplicated code, coding standards, unit tests, code coverage, complex code, potential bugs, comments and design and architecture
- Can show and compare historical data snapshots and coding trends
- Each project represented by high level matrix offering static & dynamic analysis
- Drill down into the project at specific levels (in terms of modules, package, class) and view the coding issues via HTML viewer
- Supports multiple languages including Java, C#
- Open source

DB Automation

Flyway

- Java, agile database migration tool
- Each time there is a need to evolve the database (structure or data), it creates a new migration with a version number higher than the current one
- On start up it will find the metadata table and upgrade the database accordingly
- Zero-required dependencies, only Java v5+ and jdbc required
- Supports multiple DBMS technologies
- Open source

Infrastructure as a Service

- Automatic server and network setup
- Building block for dynamic, uncongested environments
- Provides the basis for high availability and resource optimisation
- Infrastructure can be self-serviced by projects and billed accordingly
- Vendor & product tbc

Deployment Dashboard

A Bespoke Design Ltd

- A visual representation of components and installers that make up a project
- Shows the versions installed across various environment
- Retains the state of the dashboard which can be viewed via the achieves link
- Accessible from Jenkins
- Developed in-house

Deployment Controller

NOLIO

- Responsible for deployment of application code on the provisioned environment
- Visual drag-and-drop application release design giving the ability to build multi-step, multi-tier application release processes
- Connects the different toolsets and resources to allow project applications to be delivered through the environment promotion path
- CA tool

Version Control & Build Controller

Microsoft® Visual Studio™ Team Foundation Server

- Central repository for all IT artefacts
- Runs the build, test processes and stores metadata at various stages of the delivery workflow
- Supports both Java and .Net applications
- Distributed build environment already available
- Microsoft tool

Ecosystem Benefits

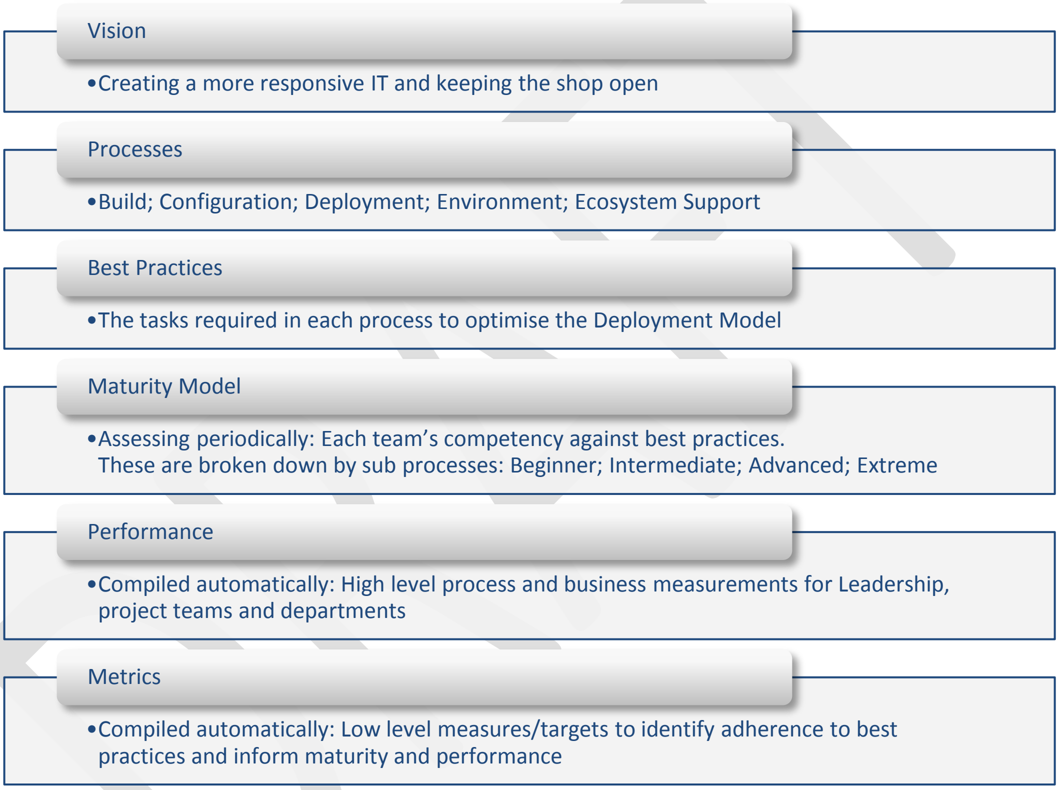
- ✓ Fast feedback on code quality even before builds are compiled
- ✓ Every type of change triggered via tools and tracked through a version control system
- ✓ Environments no longer congested
- ✓ Zero-touch up to integration testing and push-button thereafter
- ✓ Releases are pulled rather than pushed into production
- ✓ Improved transparency leads to increased confidence and trust
- ✓ Increased innovation as automation provides space and opportunity to try out new ideas

Framework

The objective of the Framework is to guide projects towards creating a more responsive IT and keeping the shop open. Tesco business areas adopt many different techniques and tools for software delivery - the Framework provides a more consistent way of working and measuring performance across the enterprise. A maturity model shows projects the journey path to reach an optimised state.

Components

The Framework is composed of six aligned components, linking what needs to be done, how it is done, why it needs to be done, and how it is measured.



Vision

- Creating a more responsive IT and keeping the shop open

Processes

- Build; Configuration; Deployment; Environment; Ecosystem Support

Best Practices

- The tasks required in each process to optimise the Deployment Model

Maturity Model

- Assessing periodically: Each team's competency against best practices. These are broken down by sub processes: Beginner; Intermediate; Advanced; Extreme

Performance

- Compiled automatically: High level process and business measurements for Leadership, project teams and departments

Metrics

- Compiled automatically: Low level measures/targets to identify adherence to best practices and inform maturity and performance

Assessments

Using Framework components, projects can begin to understand their current delivery maturity and identify key areas for improvement. This information can be used to help prioritise technical requirements in sprints or project plans.

Maturity Scores By Process

- Assessment ratings and desirable characterisitics broken down by individual process. Measuered by: Beginner; Intermediate; Advanced; Extreme.

Detailed Maturity Scores

- Assessment ratings ordered by process and broken down by individual sub-process. Each sub-process rating is shown on a graph between a beginner and target maximum score.

Top 3 Score Improvement Opportunities by Process

- For each process, the top 3 sub-processes with the greatest opportunities to achieve a higher score and influence the overall rating.

Assessment Metrics Availability

- Percentage of metrics stated in the Framework that were available to inform the assessment. Enables scoring to be more objective and accurate.

Recommended Actions

- Shows the top three sub-process actions to take for each process. These can be treated as epics or milestones for easier integration into project planning .

Overall Rating

- Overall Framework assessment rating reported on a scale: Beginner; Intermediate; Advanced; Extreme.

The assessment process is a joint exercise between the coordinating Quality team, appointed Framework champions in each project and process owners. Projects will be expected to self-score but, occasionally, audits will be carried out by process specialists. At the end of each assessment period, a review and recommendation workshop will take place for each project. Summary reports are presented to the IT Leadership team to show project, process, domain, and enterprise maturity.

Framework Benefits

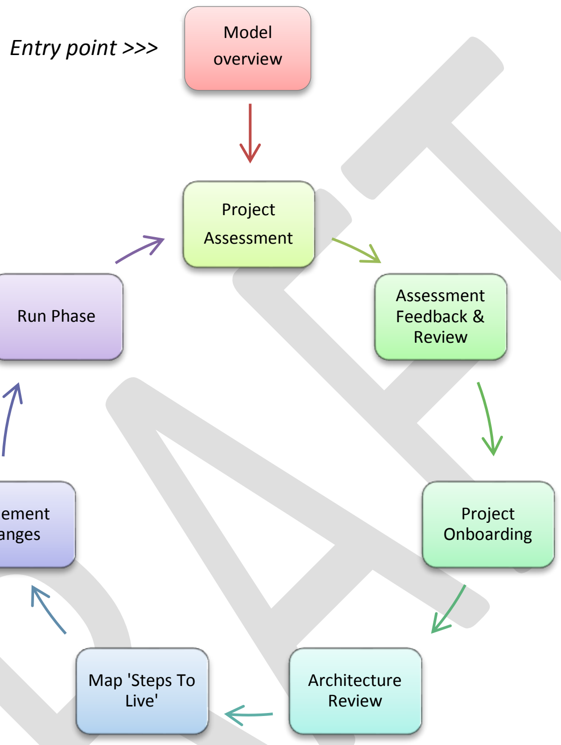
- ✓ Consistent delivery process across the enterprise
- ✓ Easier to onboard new people and projects
- ✓ Organisational effectiveness through project teams sharing knowledge and data
- ✓ A resource for continuous improvement and to justify technical requirements to business product owners
- ✓ A common standard for comparing project capability against performance
- ✓ A centralised resource where everyone can contribute to capture proven best practices

Implementation

This section describes the implementation of the Ecosystem and Framework at a project and enterprise level.

Engagement Model

The engagement model helps projects understand how to get from their current state to adopting the Framework and Ecosystem.



- **Model Overview**: Explanation of the delivery model, how the Framework links into the Ecosystem and how the Ecosystem provides metrics back for the Framework assessment.
- **Project Assessment**: How to score the current maturity of the project.
- **Assessment Feedback & Review**: Outlines the journey path for Ecosystem onboarding and reaching an optimised state.
- **Project Onboarding**: In the event of constraints, projects will be prioritised against a number of criteria (see p.14).
- **Architecture Review**: Understand the Ecosystem architecture and make recommendations on how application pipelines should be segmented.
- **Map 'Steps To Live'**: Understand current tasks that take a project from Dev to Live and identify areas that need automation, can be retired or are out of scope.
- **Implement Changes**: Automate processes, construct the pipeline and plug in required tasks.
- **Run Phase**: How to manage the pipeline and enable continuous improvement through periodical maturity re-assessments.

Rollout Approaches

There are two approaches to scaling the delivery model:

- **Depth**: Take a project and extend the delivery pipeline all the way to production. Upon a successful 'spike', re-engineer back into other projects.
- **Breadth**: Create delivery pipelines for multiple projects simultaneously. Once validated in a lower test environment, extend the respective pipelines to production.

Implementation RACI

The table below illustrates the key activities for scaling the delivery model. A combined view is required as dependencies exist between the two approaches. This table can also be used to calculate the resource requirements for a preferred approach.

Breadth & Depth

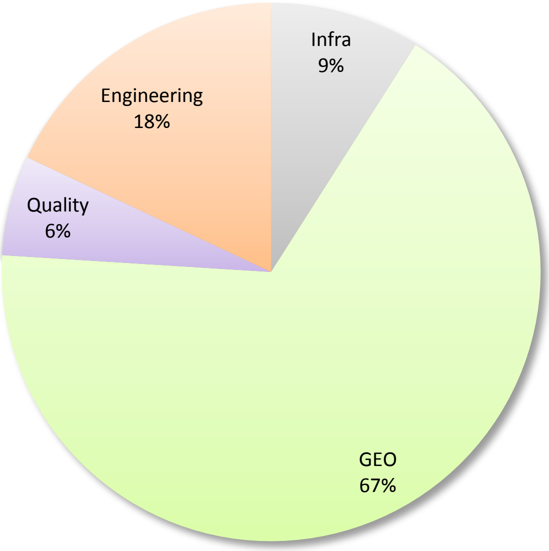
Depth only

			Internal functions			External functions				
		Deliverable	Platform Engineering	Platform Services	QA	Infra	GEO	Engineering	Service	Quality
Deployment Dashboard	1.	Dashboard built	R	C	I		C, A	C	C	
	2.	Nolio workflows updated	R	C	I		A	C	I	
	3.	Demo created	R	I	I	I	A	I	I	I
	4.	Scripts managed	R	R	I		A	I	I	
	5.	Change Impact Assessment completed	R	R	I	R	A	R		I
	6.	Integrated with Jenkins	C	R	I	I	A	I	I	I
FlyWay	1.	Ways of working agreed	R	C	I		A	C	C	
	2.	DB scripts refactored	C	I	I		C	R, A	C	
	3.	Demo created	R	I	I	I	A	I	I	I
	4.	Support level agreed	C	R	I		A	C	I	
	5.	Change Impact Assessment completed	R	R	I	R	A	R	I	
	6.	Oracle supported	R	I	I		A	C	C	
	7.	Nolio DB workflow created	C	R	I		A	C	C	
	8.	Enhancements agreed	R	C	C		A	C	C	
	9.	Enhancements delivered	R	I	I	I	A	I	I	I
	10.	Integrated with Jenkins	C	R	I	I	A	I	I	
	11.	Adopted by projects	I	R	I	I	R	A	R	I
Jenkins	1.	Installation complete	R	I	I	C	A	I	I	
	2.	Pipeline designed	R	R	I		A	C	C	
	3.	Pipeline extended to production	R	R	I	C	A	I	C	
	4.	Integration pipeline created	R	R	I	I	A	C	C	
	5.	Ways of working agreed	R	C	I	C	A	C	C	I
	6.	Access and security defined	R	C	I	R	A	C	C	
	7.	Demo created	R	I	I	I	A	I	I	I
	8.	Platform managed	C	R	I	C	A	I	C	I
	9.	High availability set up	R	I	I	R, A	R	I	C	I
	10.	Change Impact Assessment completed	R	R	I	R	A	R	I	
	11.	Integrated with Nolio	C	R	I	I	A	C	C	I

	12.	Adopted by projects	I	R	I	I	R	R, A	I	I
	13.	Disaster Recovery in place	R	I	I	R	A	I	C	I
	14.	Integrated with ICCM	R	R	I	I	A	I	C	I
	15.	Enhancements agreed	R	C	C	I	A	C	C	I
	16.	Enhancements delivered	R	I	I	I	A	I	I	I
	17.	Migrated to strategic solution	R	R	I	R, A	R	I	C	I
Puppet	1.	Ways of working agreed	R	C	I	C	A	C	C	
	2.	Installation complete	R	I	I	I	A	I	I	
	3.	Puppet scripts created	R	R	I	C	A	C	C	
	4.	Application server builds standardised	C	C		C	C	R, A	I	
	5.	Master & agents installed on all envs	I	R	I	C	A	I	I	
	6.	Compliance reports created	R	R	I	C	A	C	C	
	7.	Demo created	R	I	I	I	A	I	I	I
	8.	Platform managed	C	R	I	I	A	I	I	
	9.	Change Impact Assessment completed	R	R	I	R	A	R	I	
	10.	Adopted by projects	I	R	I	I	R	R, A	I	I
	11.	Migrated to strategic solution	R	I	I	R, A	R	I	I	I
Selenium / Cucumber	1.	Installation complete	R	I	I	I	A	C	I	
	2.	Demo created	R	I	I	I	A	R	I	I
	3.	Platform managed	C	R	I	I	A	C		
	4.	GRID platform upgraded	C	R	I	R	A	C		
	5.	Docs created for other languages	C	I	I		C	R, A		
	6.	Change Impact Assessment completed	R	R	I	R	A	R	I	
	7.	Integrated with Jenkins	R	R	I	I	A	C	I	I
	8.	Adopted by projects	I	C	I	I	C	R, A	I	I
SonarQube	1.	Ways of working agreed	R	I	I		R	R, A		
	2.	Coding standards agreed	C		I		C	R, A		
	3.	Demo created	R	I	I	I	A	I	I	I
	4.	Platform managed	C	R	I	I	A	C		I
	5.	Change Impact Assessment completed	R	R	I	R	A	R	I	
	6.	Adopted by projects	I	I	I	I	I	R, A		I
	7.	Governance process agreed	I	I	I		I	R, A		I
	8.	Enhancements agreed	I	C	C		C	R, A		
	9.	Enhancements delivered	I	C	I	I	I	R, A		I
	10.	Migrated to strategic solution	C	R	I	R	A	C		I
TFS	1.	Time-to-market report validated & adopted	C	R	C	I	R, A	C	C	I
	2.	Framework metrics developed	I	R	C		R, A	C	C	I
Framework	1.	Recalibration complete	C	C	R	C	C	C	C	A
	2.	Assessment complete	I	I	R	I	R	R	R	A
	3.	Workshop review complete	R	R	R	I	R	R	R	A
	4.	Summary reports submitted	I	I	R	I	I	I	I	A
	5.	Feedback provided for recalibration	C	C	R	C	C	C	C	A
IaaS	1.	Cont. Delivery requirements submitted	R	C	I	C, I	A	C	C	
	2.	Ways of working agreed	C	C	I	R, A	C	C	C	
	3.	Supplier selected	I	I	I	R, A	I	I	I	

	4.	Delivery Plan communicated	I	I	I	R, A	I	I	I	
	5.	Integrated with Ecosystem	R	I	I	R, A	R	I	I	I
	6.	Adopted by projects/departments	C	R	I	R	I	R, A	C	I
Nolio	1.	Ecosystem requirements submitted	R	R	I	C	A	C	C	
	2.	Delivery Plan communicated	R	R	I	I	A	I	I	
	3.	Platform upgraded	R	R	I	R	A	I	I	I

Implementation accountabilities by Function



Project Onboarding

Formation Phase

While the centralised Platform capability is being built, a basic assessment will be carried out to determine whether a project can onboard early.

Evaluation	
Project has a backup delivery method	Yes / No
Project accepted risk that Ecosystem is not managed and high availability is not guaranteed	Yes / No
Project agreed to provide their own resources to adopt new tools	Yes / No
Project onboarding approved by GESO Platform Manager	Yes / No
Project likely to benefit from being given access to the Ecosystem – based on above and: - Skills availability - Capacity in sprints/project plans - Feasibility in relation to target dates - Opportunity in relation to architecture and process complexity	Yes / No

Maintenance Phase

In the event of constraints once the Platform team has been formed, projects will be assessed against a number of criteria to determine the order in which they can onboard the Ecosystem.

Criteria	
Value	Size of additional business opportunity enabled by the project realising the direct benefits of Continuous Delivery (realised in the first year)
	Revenue size of the business area affected by the project
	Longevity of the project and impacted products
Variety	Ecosystem changes required
	Number of project components proposed for moving to Continuous Delivery
	SonarQube: Project already has a culture of continuous integration, i.e. keeping the check-in build green
	Puppet: Project has recorded instances of server configuration defects
	Puppet: Project has evidence of long server creation lead times
	Puppet: Project has servers that have a high complexity to build which need to be built regularly
	Puppet: Project has a large number of servers which require common re-usable components
	IaaS: Project has a large volume of upcoming infrastructure requirements (including to production)
	IaaS: Project scores well in Puppet eligibility criteria
	Pipeline Dashboard: Project has high complexity in terms of number of components and number of environments
Velocity	Number of branches
	Feature delivery rate
Volume	*Re-usable benefits*: Amount of project-specific technologies already supported in the Ecosystem
	Re-usable benefits: Amount of project specific technologies that have a wide scope for re-use by other projects
Pre-Conditions	*Skills & Attitude*: Skills available on the project to provide required inputs to adopting Continuous Delivery
	Skills & Attitude: Appetite for adopting Continuous Delivery from both the Development and Operations stakeholders on the project
	Past Progress: Initial Framework maturity score
	Past Progress: Framework score improvement since initial assessment

Tools Adoption Priority

This shows the sequence for introducing Ecosystem tools to projects, against four known delivery scenarios. Priorities are based on tooling dependencies, tooling complexity and value proposition.

Tesco in-house	1. Framework	Maturity assessment
	2. TFS	Version Control Build automation
	=3. Selenium/Cucumber/QTP/VSTS	Test automation
	=3. Nolio	Deployment automation
	4. Flyway	DB automation
	5. Puppet	Automated configuration
	6. Jenkins	Orchestration Pipeline dashboard
	7. SonarQube	Static code analysis
	8. Deployment Dashboard	Application tracking
	9. Infrastructure as a Service (IaaS)	Autonomic environments
Tesco in-house + Distributed deployments	1. Framework	Maturity assessment
	2. TFS	Version Control Build automation
	=3. Selenium/Cucumber/QTP/VSTS	Test automation
	=3. Nolio	Deployment automation
	=3. Deployment Dashboard	Application tracking
	4. Flyway	DB automation
	5. Puppet	Automated configuration
	6. Jenkins	Orchestration Pipeline dashboard
	7. SonarQube	Static code analysis
	8. Infrastructure as a Service (IaaS)	Autonomic environments
Tesco + Third party	1. Framework	Maturity assessment
	2. TFS	Version Control Build automation
	=3. Selenium/Cucumber/QTP/VSTS	Test automation
	=3. Nolio	Deployment automation
	4. Flyway	DB automation
	5. Puppet	Automated configuration
	6. Jenkins	Orchestration Pipeline dashboard
	7. SonarQube	Static code analysis
	8. Deployment Dashboard	Application tracking
	9. Infrastructure as a Service (IaaS)	Autonomic environments
Tesco + Third party + Distributed deployments	1. Framework	Maturity assessment
	2. TFS	Version Control Build automation
	=3. Selenium/Cucumber/QTP/VSTS	Test automation
	=3. Nolio	Deployment automation
	=3. Deployment Dashboard	Application tracking
	4. Flyway	DB automation
	5. Puppet	Automated configuration
	6. Jenkins	Orchestration Pipeline dashboard
	7. SonarQube	Static code analysis
	8. Infrastructure as a Service (IaaS)	Autonomic environments

All tools can be used independently apart from Jenkins

Test tools for specific projects will be selected by Engineering teams

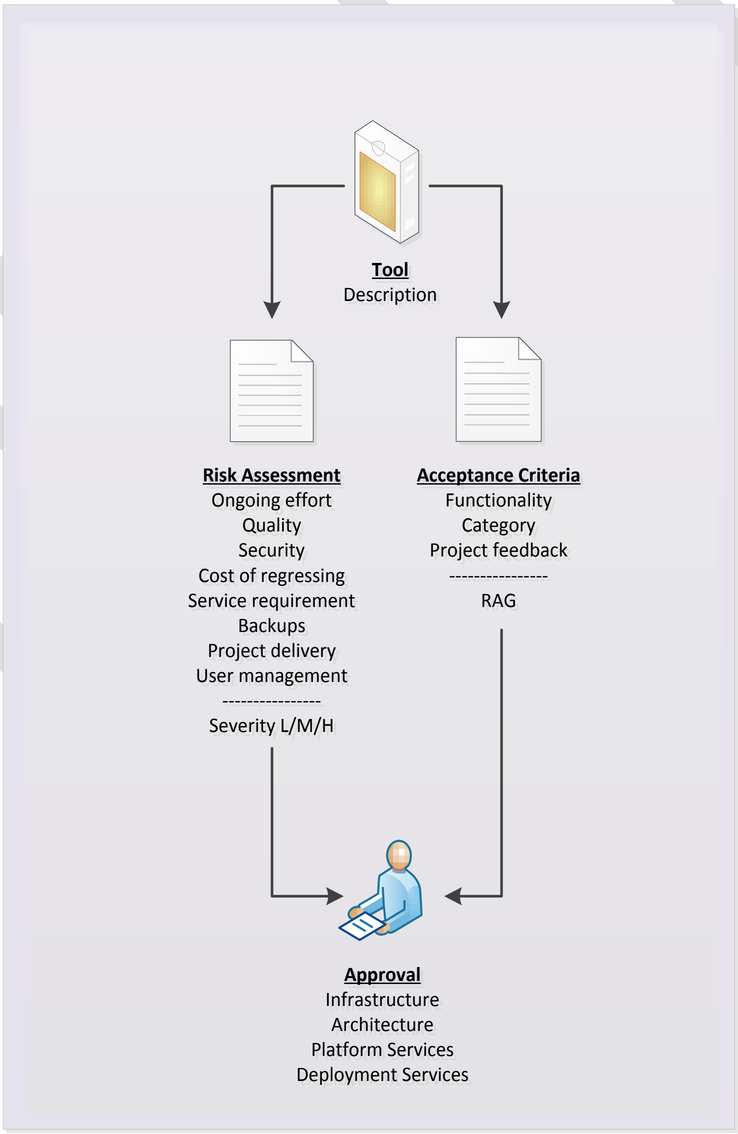
Support

Every tool in the Ecosystem needs managing to:

- Prevent unplanned downtime
- Prevent unauthorised access
- Assess the risk and impact of changes
- Ensure fitness for purpose and use
- Recover from system failure in a timely fashion

Change Impact Assessment

A lightweight process coordinated by the Platform team to evaluate new tools on a designated pilot project before they are integrated into the Ecosystem. A number of criteria are used to guide key stakeholder groups responsible for approval. This process is designed to encourage joint responsibility across internal and external functions for end-to-end tooling.



Tool Support

Recommended support levels are shown for each tool based on an assessment of potential scenarios, impact severity and software complexity.

Deployment Dashboard		
Category	Description	% of scenarios/plugin severity or complexity
Internal resolve	Issue(s) to be resolved internally, within the support structure of Tesco	**95%**
Non-commercial external resolve	Issue(s) to be resolved using external resources that are not contracted commercially: E.g. Online resources such as Forums, Community support, Free defect resolve systems, Wiki, IRC , etc.	
Commercial resolve	Commercial support such as: Subject matter expert contracted to resolve; Product owner support in the form of physical presence consultancy; Remote consultancy; Sponsored big/feature fix	**5%**
Complexity assessment	**Low risk of severe issues occurring as the Deployment Dashboard:** • In the event of the tool failing completely, the underlying deployment information should still be accessible • Should be recognised that there is a dependency on Jenkins connection as the tool is served off the web server on Jenkins • Small codebase and developed with clean structure to ensure greater ease for future enhancements and maintenance • Tool is developed in-house therefore internal training and handover can be managed • Tool is developed using common technologies; Perl, HTML and CSS therefore resourcing SMEs for fixes or enhancements should be less challenging	
Recommendation	• Primary instance, issues to be resolved internally using existing capability and in-house resources • Second instance, to use online resources such as online forums, community support, wiki and IRC channels • For critical resolves, be it in terms of complexity or quick turnaround time for resolve, commercial support in the form of a SMEs, (Depending on the chosen programming language of the adopting project) should be hired on a short term basis • Further analysis and discussions required with Service Management to decide the most appropriate model both in terms of business need and costing.	

Flyway		
Category	Description	% of scenarios/plugin severity or complexity
Internal resolve	Issue(s) to be resolved internally, within the support structure of Tesco	**98%**
Non-commercial external resolve	Issue(s) to be resolved using external resources that are not contracted commercially: E.g. Online resources such as Forums, Community support, Free defect resolve systems, Wiki, IRC, etc.	
Commercial resolve	Commercial support such as: Subject matter expert contracted to resolve; Product owner support in the form of physical presence consultancy; Remote consultancy; Sponsored big/feature fix	**2%**
Complexity assessment	**Low risk of severe issues occurring as Flyway;** • Would be running with no active services • Would not persist any data • Does not have its own database • The only logging that it generates are deployment output logs • Would not have any third party software integration, (does not have any dependencies apart from requiring Java v1.6) • Does not need to execute any commands as a root user • Would be used in all environments, (consistent usage of the tool from development through to production) • Low likelihood that an error would occur during the check in process however, if an error/failure did occur at this stage it would be more efficient to resolve internally • In the worst case scenario of loss or damage of production database, external user support would not be considered • With limited integration and low functional complexity there is a high percentage chance of most issues being user related Builds and triggers can be executed manually in the event of an automated process failing	
Recommendation	• Primary instance, issues to be resolved internally using existing capability and in-house resources	

	• Second instance, to use online resources such as online forums, community support, wiki and IRC channels • For critical resolves be it in terms of complexity or quick turnaround time for resolve, commercial support in the form of Java J2SE SMEs should be hired on a short term basis • For advanced features Flyway sponsored development can be considered, Cost of which is dependent on scope of work • Further analysis and discussions required with Service Management to determine the most appropriate model both in terms of business need and costing.

Jenkins		
Category	Description	% of scenarios/plugin severity or complexity
Internal resolve	Issue(s) to be resolved internally, within the support structure of Tesco	**80%**
Non-commercial external resolve	Issue(s) to be resolved using external resources that are not contracted commercially: E.g. Online resources such as Forums, Community support, Free defect resolve systems, Wiki, IRC, etc.	
Commercial resolve	Commercial support such as: Subject matter expert contracted to resolve; Product owner support in the form of physical presence consultancy; Remote consultancy; Sponsored big/feature fix	**20%**
Complexity assessment	**Low risk of severe issues occurring as Jenkins:** • Builds and triggers can be executed manually in the event of an automated process failing	
Recommendation	• Primary instance, issues to be resolved internally using existing capability and in-house resources • Second instance, to use online resources such as online forums, community support, wiki and IRC channels • For severe or critical resolves be it in terms of complexity or quick turnaround time for resolve, commercial support should be considered such as CloudBees. (CloudBees support covers generic support as well as plugins support) • For advanced features/plugin development, CloudBees sponsored development should be considered, (Cost of which is dependent on scope of work) • Pricing is calculated based on the number and size of the Jenkins Master/Slave architecture setup • Response times depend on the level of support selected; Silver, Gold or Platinum; http://wiki.cloudbees.com/bin/view/Jenkins+Enterprise/Support%2C+Severity+Level+and+Response+Time+Descriptions The confirmed level of which would need to be determined by Service Management. • Analysis and identification of scenarios show that a 24/7 based model may not be necessary. Instead negotiated models to be considered with commercial support offerings to support lower amount of support requests with faster resolve SLAs. Further analysis and discussions with Service Management required and decide the most appropriate model both in terms of business need and costing.	

Puppet		
Category	Description	% of scenarios/plugin severity or complexity
Internal resolve	Issue(s) to be resolved internally, within the support structure of Tesco	**80%**
Non-commercial external resolve	Issue(s) to be resolved using external resources that are not contracted commercially: E.g. Online resources such as Forums, Community support, Free defect resolve systems, Wiki, IRC, etc.	
Commercial resolve	Commercial support such as: Subject matter expert contracted to resolve; Product owner support in the form of physical presence consultancy; Remote consultancy; Sponsored big/feature fix	**20%**
Complexity assessment	**Medium to high risk of severe issues occurring as Puppet:** • Current projects do not have the practical experience and familiarity of using the tool • Highly dependent integration with many components of the ecosystem • Integral component as used for configuring the catalogue for virtual machines • Pre-configuration setups for both machines and applications will be complex and will require high accuracy	
Recommendation	• Primary instance, issues to be resolved internally using existing capability and in-house resources • Second instance, to use the commercial support from Puppet Labs • Enterprise level includes; escalated bug fixes, 24/7/365 time coverage, email support, phone support, access to private and public forums, feature priorities, access to all updates and upgrades of Puppet enterprise, unlimited number of technician contacts, unlimited number of cases/month and	

	public training. Standard level covers working days only with limited cases raised per month and does not include phone support. Further analysis and discussions required with Service Management to decide the most appropriate support coverage required. Thereafter, to negotiate directly with Puppet Labs to adopt a model befitting to the business needs and costing. • Suggest preliminary training for key administration users of the tool via Puppet Labs, (Enterprise level of offers four free places for training per year)

Selenium & Cucumber		
Category	Description	% of scenarios/plugin severity or complexity
Internal resolve	Issue(s) to be resolved internally, I.e. within the support structure of Tesco	**95%**
Non-commercial external resolve	Issue(s) to be resolved using external resources that are not contracted commercially: E.g. Online resources such as Forums, Community support, Free defect resolve systems, Wiki, IRC, etc.	
Commercial resolve	Commercial support such as: Subject matter expert contracted to resolve; Product owner support in the form of physical presence consultancy; Remote consultancy; Sponsored big/feature fix	**5%**
Complexity assessment	**Low risk of severe issues occurring as Selenium and Cucumber:** • Development and testing is carried out and signed off prior to reaching production environment • Is used for test automation therefore worst case scenario or critical failures would mean executing tests manually • Can involve complex test case development that may require programming language expertise • Includes additional features compared to that of other test suite offerings such as HTML test report outputs and the parallel processing using selenium GRID. Configuration of which may require further expertise or due diligence	
Recommendation	• Primary instance, issues to be resolved internally using existing capability and in-house resources • Second instance, to use online resources such as online forums, community support, wiki and IRC channels • For critical resolves, be it in terms of complexity or quick turnaround time for resolve, commercial support in the form of Java/C#/Ruby etc. SMEs, (Depending on the chosen programming language of the adopting project) should be hired on a short term basis	

SonarQube		
Category	Description	% of scenarios/plugin severity or complexity
Internal resolve	Issue(s) to be resolved internally: i.e. within the support structure of Tesco	95%
Non-commercial external resolve	Issue(s) to be resolved using external resources that are not contracted commercially: E.g. Online resources such as Forums, Community support, Free defect resolve systems, Wiki, IRC, etc.	
Commercial resolve	Commercial support such as; Subject matter expert contracted to resolve; Product owner support in the form of physical presence consultancy; Remote consultancy; Sponsored big/feature fix	5%
Complexity assessment	**Low risk of severe issues occurring as SonarQube:** • Coding analysis and code quality assurance will be carried out before the code is promoted to higher environments such as production	
Recommendation	• Primary instance, issues to be resolved internally using existing capability and in-house resources • Second instance, to use online resources such as online forums, community support, wiki and IRC channels • For critical resolves, be it in terms of complexity or quick turnaround time for resolve, commercial support in the form of a SMEs, (Depending on the chosen programming language of the adopting project) should be hired on a short term basis • Further analysis and discussions with Service Management required and decide the most appropriate model both in terms of business need and costing.	

Note:

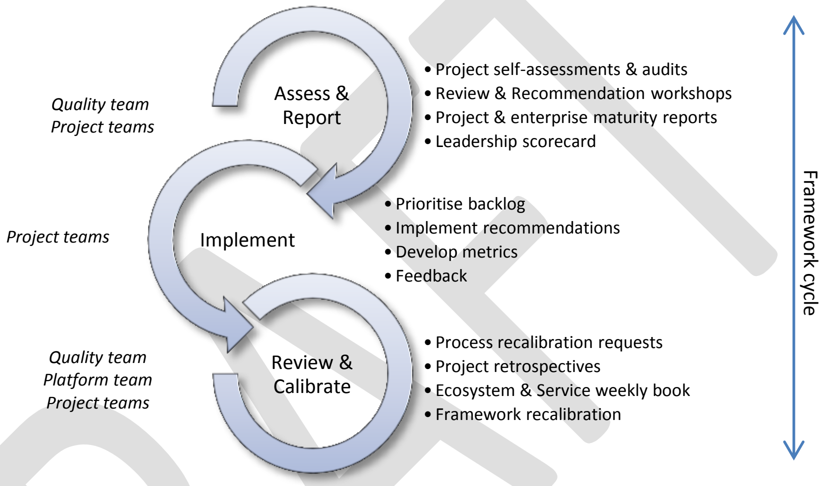
- *The maintenance and upkeep of SonarQube and test tools will be handled by the Platform Services team but process definitions and day-to-day use will be carried out by Engineering*

Evolution

The Framework and Ecosystem must be responsive to changing technology and business conditions. The aim is to strike the correct balance between having a consistent and flexible model. People are encouraged to share ideas and, along with quantitative analysis, this creates an effective feedback loop for product development.

Framework Recalibration

The Framework is recalibrated periodically to ensure ways of working reflect IT community developments, issue root-cause analyses and latest advancements in the industry. Feedback is provided internally via QA reports and process specialists, and externally through market research and crowdsourcing.


Quality team
Project teams
Assess & Report
• Project self-assessments & audits
• Review & Recommendation workshops
• Project & enterprise maturity reports
• Leadership scorecard
Project teams
Implement
• Prioritise backlog
• Implement recommendations
• Develop metrics
• Feedback
Quality team
Platform team
Project teams
Review & Calibrate
• Process recalibration requests
• Project retrospectives
• Ecosystem & Service weekly book
• Framework recalibration
Framework cycle

Ecosystem Performance

Weak links in the tool chain have an adverse effect on the wider Ecosystem and impact project velocity. As such, all tools are tightly monitored by the Platform team to measure performance and reliability. Key statistics, such as tool availability, are published in an Ecosystem weekly book and assessed by the Platform team.

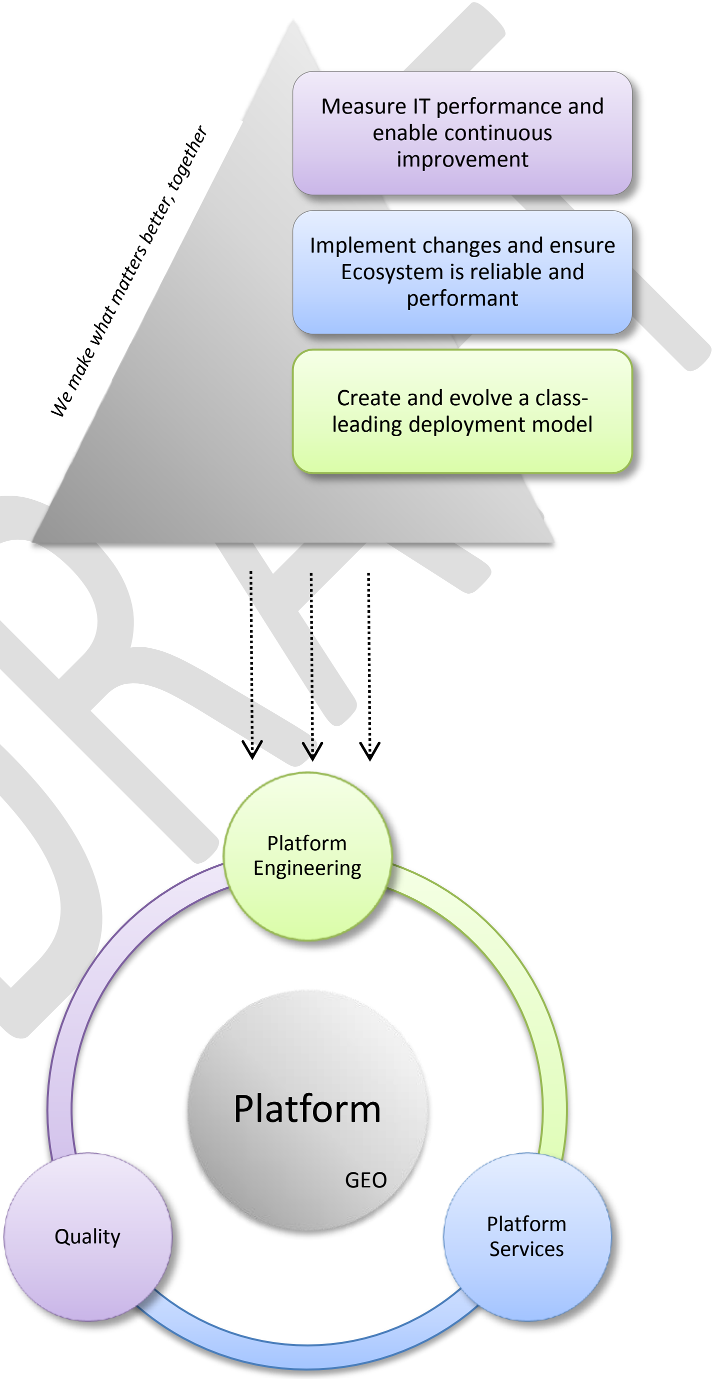
Product Backlog

Changes and enhancements to the Ecosystem and Framework are ratified and prioritised in backlog planning sessions. The Product Manager and Platform team assess each item against the core principles of the model.

Capability

People and work culture are crucial to sustaining the delivery model. Collaboration is encouraged under the new structure and this supports the creation of a cohesive delivery system. Process specialists are centralised and, using Framework assessment scores for guidance, their expertise is leveraged for projects most in need of help, thus raising IT performance for the organisation as a whole. There is greater emphasis on improving processes and tools by assessing QA reports and the latest technological developments in the industry.

Organisational Shape

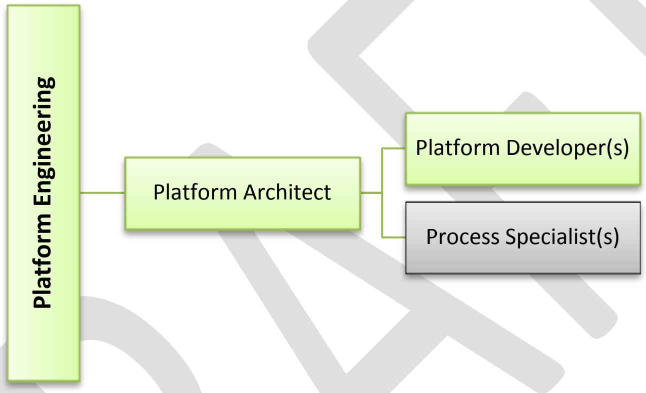

We make what matters better, together
Measure IT performance and enable continuous improvement
Implement changes and ensure Ecosystem is reliable and performant
Create and evolve a class-leading deployment model
Platform Engineering
Platform
GEO
Quality
Platform Services

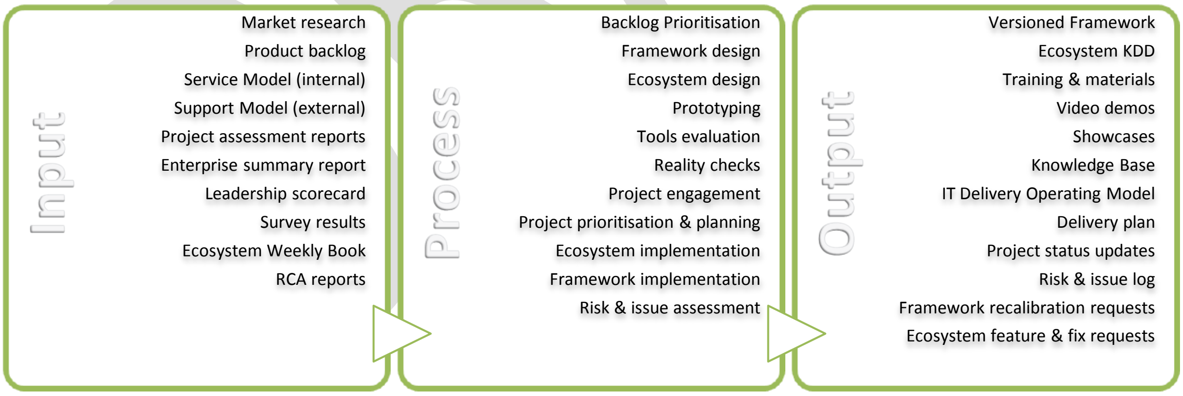
Team Structures & Responsibilities

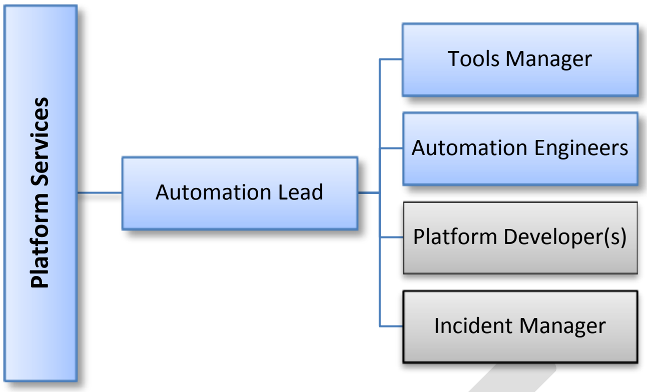
Three key functions:

- **Platform Engineering**: Responsible for Ecosystem architecture, tools development and process refinement/reengineering.
- **Platform Services**: Implement and rollout new tools , monitor Ecosystem performance, provide tool support and manage licencing and supplier contracts.
- **Quality**: Report on the performance of IT projects and collate information which enables the Ecosystem and Framework to be continually improved.

Roles are not aligned to specific tools as this creates cross-process expertise and more rounded development of the Ecosystem. Boxes marked in grey are dotted line roles, lead to greater cohesion between different functions and incentivise staff to take wider responsibility.







Platform Services

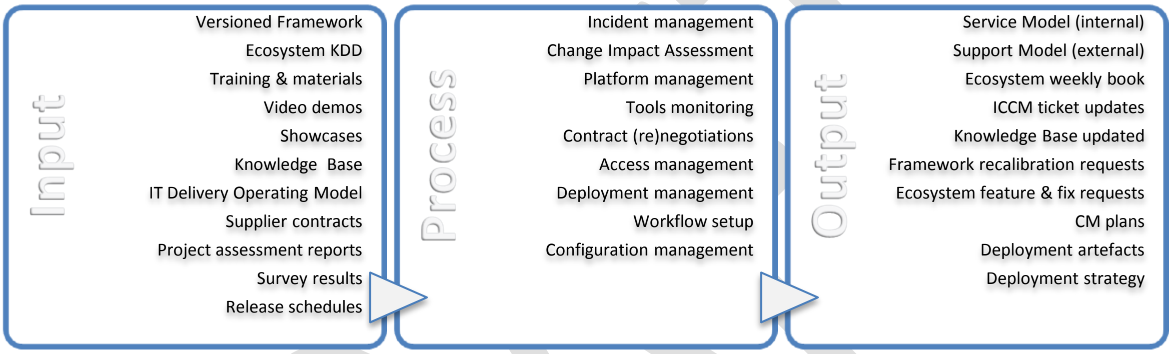
Automation Lead

Tools Manager

Automation Engineers

Platform Developer(s)

Incident Manager



Input

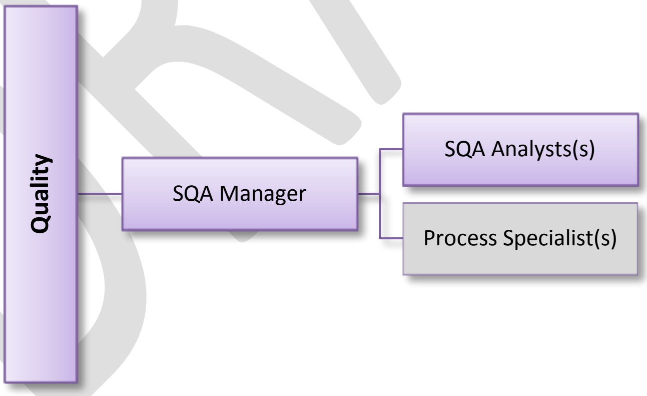
Versioned Framework
Ecosystem KDD
Training & materials
Video demos
Showcases
Knowledge Base
IT Delivery Operating Model
Supplier contracts
Project assessment reports
Survey results
Release schedules

Process

Incident management
Change Impact Assessment
Platform management
Tools monitoring
Contract (re)negotiations
Access management
Deployment management
Workflow setup
Configuration management

Output

Service Model (internal)
Support Model (external)
Ecosystem weekly book
ICCM ticket updates
Knowledge Base updated
Framework recalibration requests
Ecosystem feature & fix requests
CM plans
Deployment artefacts
Deployment strategy

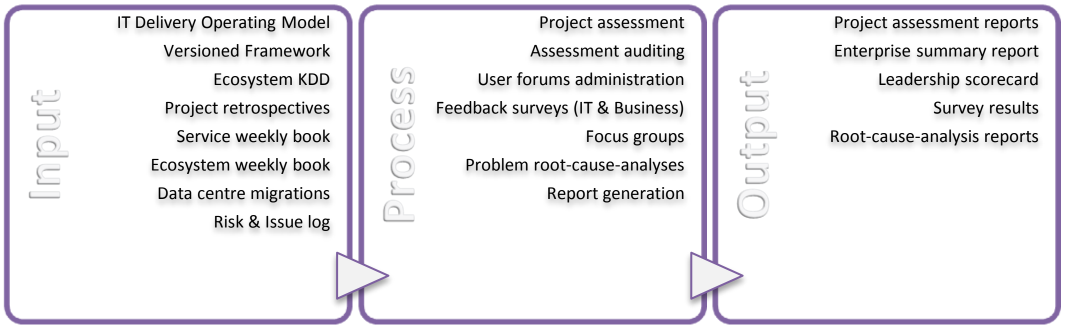


Quality

SQA Manager

SQA Analysts(s)

Process Specialist(s)



Input

IT Delivery Operating Model
Versioned Framework
Ecosystem KDD
Project retrospectives
Service weekly book
Ecosystem weekly book
Data centre migrations
Risk & Issue log

Process

Project assessment
Assessment auditing
User forums administration
Feedback surveys (IT & Business)
Focus groups
Problem root-cause-analyses
Report generation

Output

Project assessment reports
Enterprise summary report
Leadership scorecard
Survey results
Root-cause-analysis reports

Skillset Requirements

Model Component		Tasks	Skillsets
Version Control	TFS	Integration with version control events	**TFS command line utilities** **Team Explorer command line utilities & scripting**
Automated Build	TFS	Script triggering and manual builds Build definition customisation	**TFS command line utilities** **Team Explorer command line utilities & scripting**
Static Code Analysis	SonarQube	Configuration Integration with build process	**SonarQube** **Build manifest**
Orchestration & Pipeline Dashboard	Jenkins	Configuration Custom plugin development Customising existing plugins General troubleshooting	**Jenkins** **Jenkins API** **Java** **HTML** **CSS**
Deployment Dashboard	Perl scripts	Scripting in Perl	**Perl** **General programming** **HTML** **CSS**
Infrastructure as a Service (IaaS)	Tbc	API integration	**Java** **SI experience** **General understanding of IaaS concepts** **Hypervisors**
Automated Application Deployment	Nolio	Scripting Workflow development Modifying release manifests	**General scripting** **Nolio CLI** **Nolio** **Release management experience**
DB Deployment Automation	FlyWay	Tool customisation General usage	**Java** **SI experience**
Automated Configuration	Puppet	Writing puppet manifests Managing puppet master General troubleshooting	**Puppet DSL** **Unix administration** **SysAdmin experience**
Automated functional testing	Selenium Cucumber	Writing user stories in plain English Writing steps files for each of the tests Managing execution & reporting of the tests Managing selenium and browsers grid Writing scripts to build test harness	**.net** **Ruby** **Java** **Basic functional testing experience** **XPath and CSS** **Scripting skills**
Framework	Excel	Framework calibration Assessment coordination Reporting setup	**VB** **Macro** **XLS skills** **Cross-functional process expertise**

Organisational Benefits

The introduction of integrated tools and robust processes has a positive impact on the IT work culture.

- ✓ Joined up teams
  - All areas of IT working via centralised dashboards and tools
  - Feedback from Deployment, Environment, Build and Quality teams to Platform team
- ✓ Improvement and innovation
  - More emphasis on market and technology research
  - Process specialists work at an enterprise level and help projects most in need
- ✓ Consistency
  - Knowledge sharing
  - Cross-fertilisation
- ✓ Risk prevention
  - Constant analysis of IT performance
  - Monitoring-driven Ecosystem development

Useful Links

Video demos

Latest Framework

Ecosystem KDD

Tooling Analysis & Selection

Tool Support Matrix