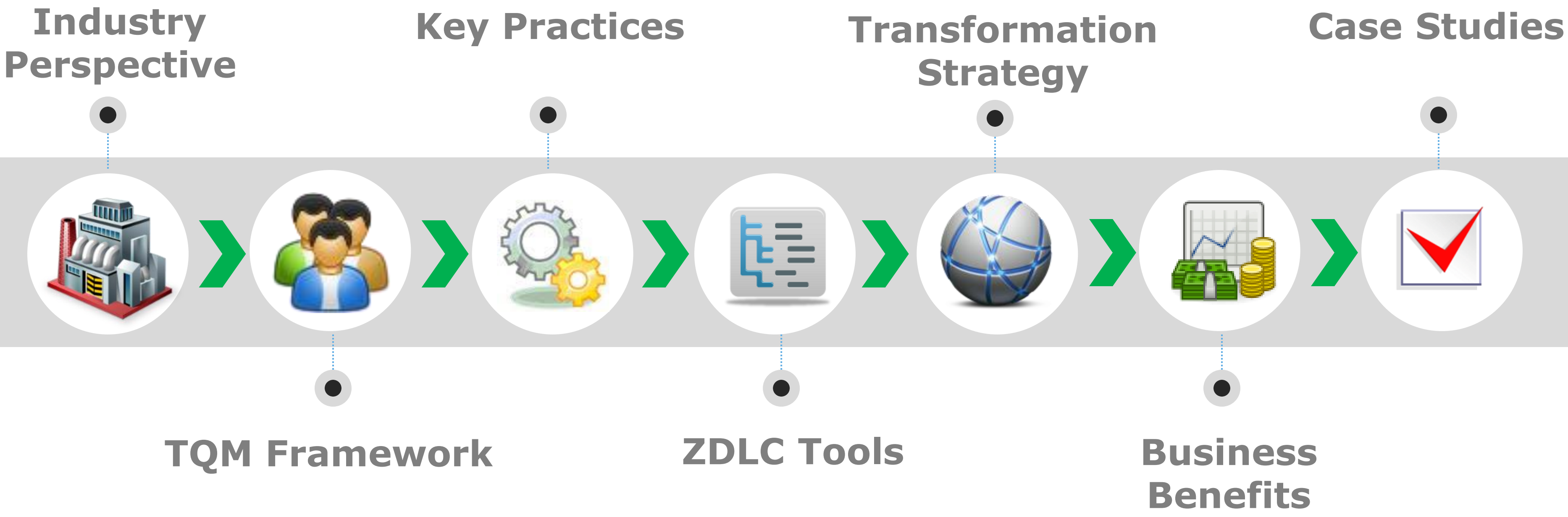
Quality Engineering and Management

CQAA Presentation
March 19, 2014

Jim Mondi – Global Practice Leader – Process & Quality Consulting

James.Mondi@Cognizant.com 630-235-3656

Agenda



Industry Perspective

TQM Framework

Key Practices

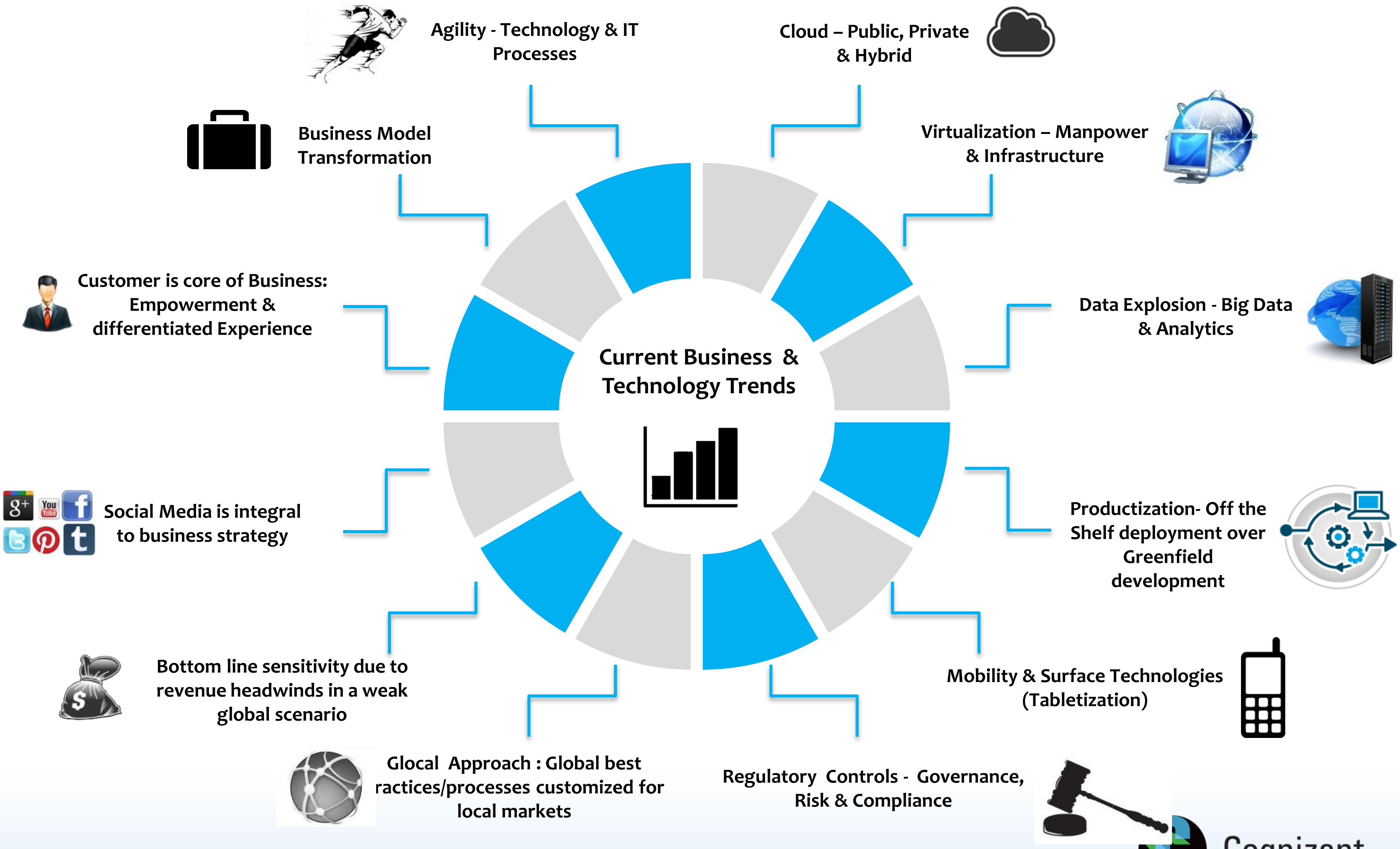
ZDLC Tools

Transformation Strategy

Business Benefits

Case Studies

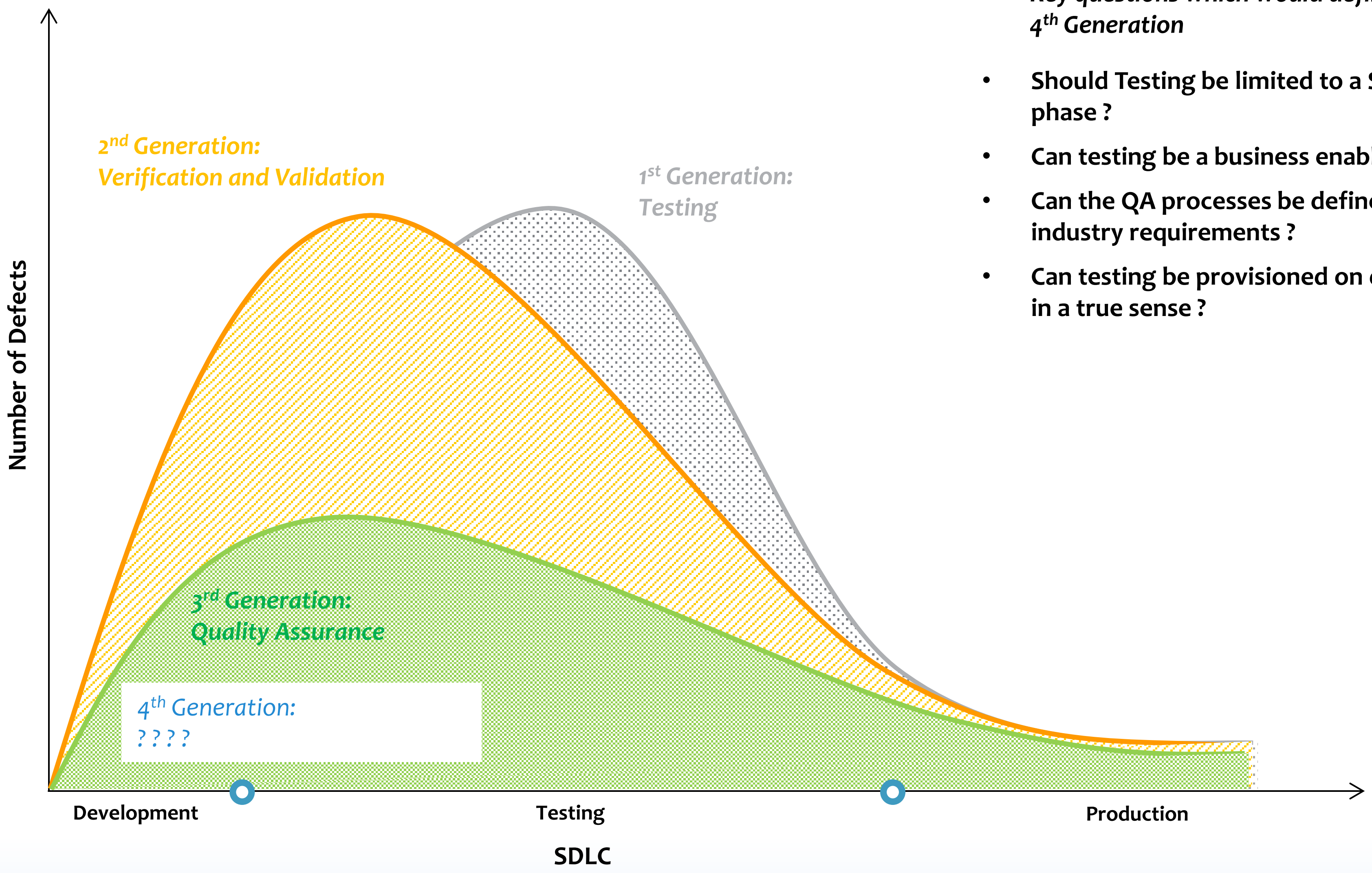
Business and Technology are changing


Current Business & Technology Trends
Agility - Technology & IT Processes
Cloud – Public, Private & Hybrid
Virtualization – Manpower & Infrastructure
Data Explosion - Big Data & Analytics
Productization- Off the Shelf deployment over Greenfield development
Mobility & Surface Technologies (Tabletization)
Regulatory Controls - Governance, Risk & Compliance
Glocal Approach : Global best ractices/processes customized for local markets
Bottom line sensitivity due to revenue headwinds in a weak global scenario
Social Media is integral to business strategy
Customer is core of Business: Empowerment & differentiated Experience
Business Model Transformation

And so is Quality Assurance

“Past”		“Present”
Software Quality Defect Detection	Goal	Business Assurance Defect Prevention
SDLC aligned Exhaustive Test Coverage	Strategy & Methodology	Domain contextualization Risk Managed Test Coverage
Defect removal efficiency Productivity	Measure of success	Enhanced user experience Agility, flexibility & innovation
Reporting	Outcome	Advisory
Testing effort based Capex + opex	Financials	Outcome Based Minimizing Capex
Fixed teams, limited use of Core flex model, shared services	Operating model	Evolution of Testing as a service on demand resource provisioning

But, what is next for QA?



Key questions which would define the 4th Generation

- **Should Testing be limited to a SDLC phase ?**
- **Can testing be a business enabler ?**
- **Can the QA processes be defined by industry requirements ?**
- **Can testing be provisioned on demand in a true sense ?**

Next Gen QA???

Testing as business enabler

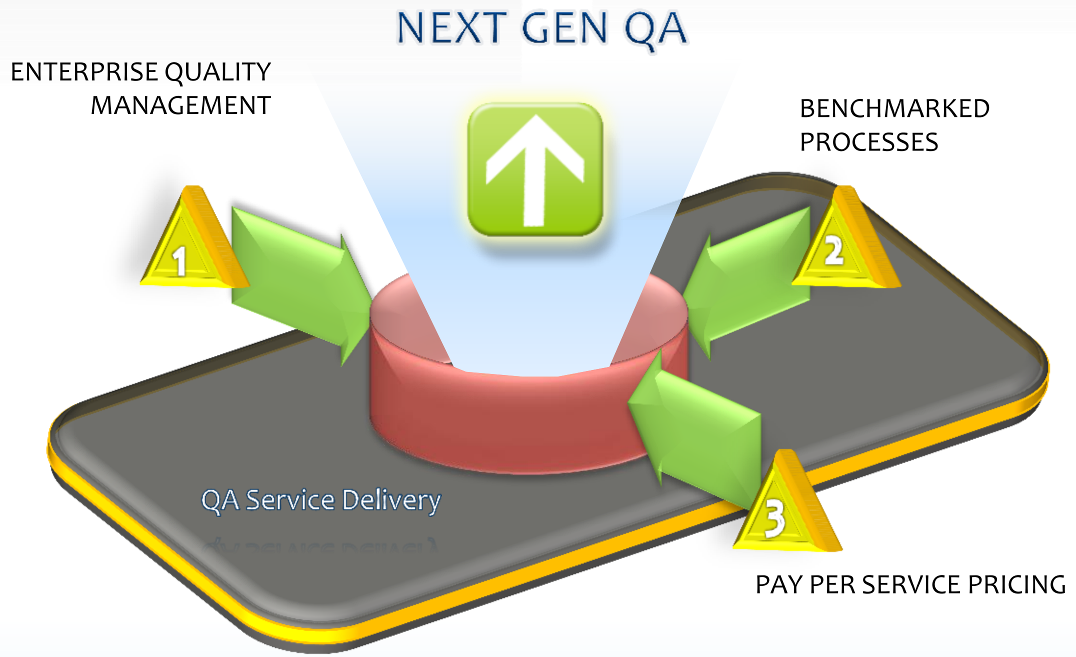
- **Business Assurance : End to End Ownership of Testing from Requirements to Release**
- **Tester as a user' - from 'requirements' validation to 'experience' validation**
- **Business release aligned process orchestration**

Testing as an integrated function

- 'Shift Left' to help seed quality while building applications
- **'Shift Right' to assure production readiness**

Industrialized Methods

- **Industry benchmarked processes and measures**
- **Scalability with predictability : Higher release volumes with same level of production quality**



Testing as a Service

- "Crowd on Cloud" model for variablized testing requirements
- SaaS based Tools and IaaS based environments

Cognizant

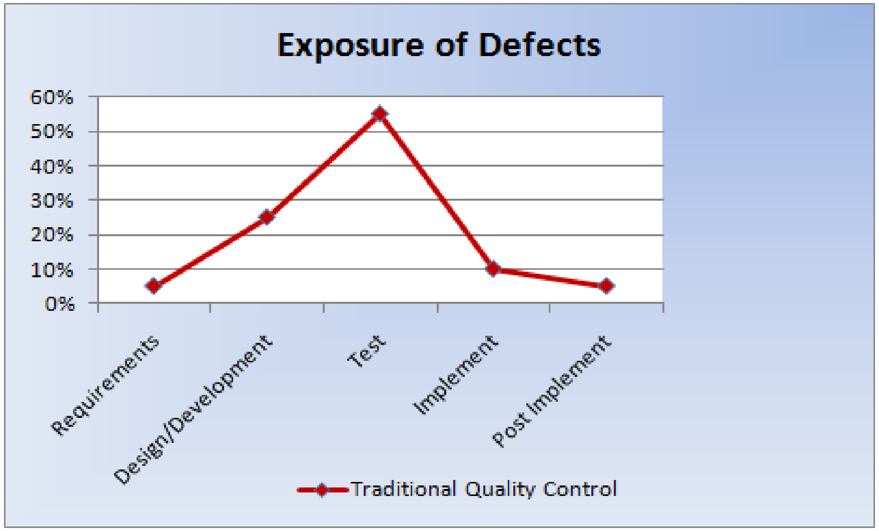
Challenges and Expectations

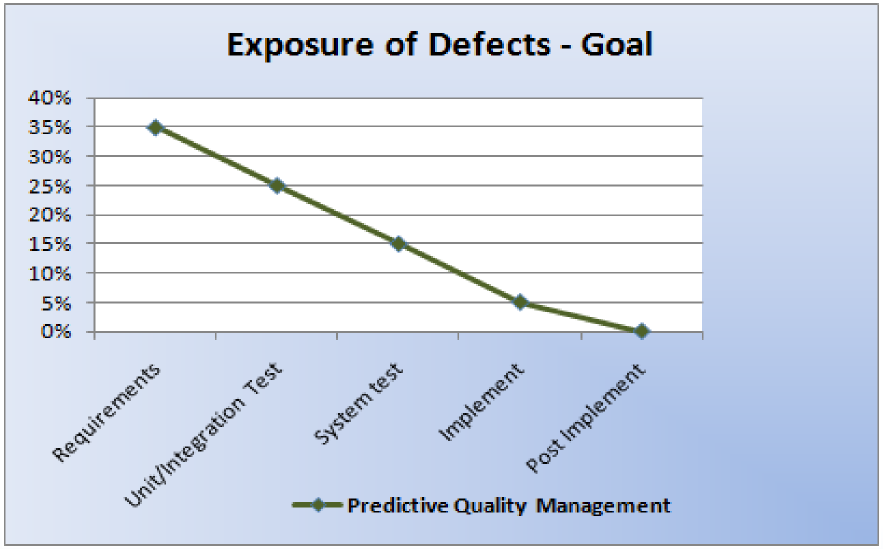
Challenges

- Unpredictable quality and schedule
- Cost and schedule overrun
- High cost of quality
- Low credibility with Business

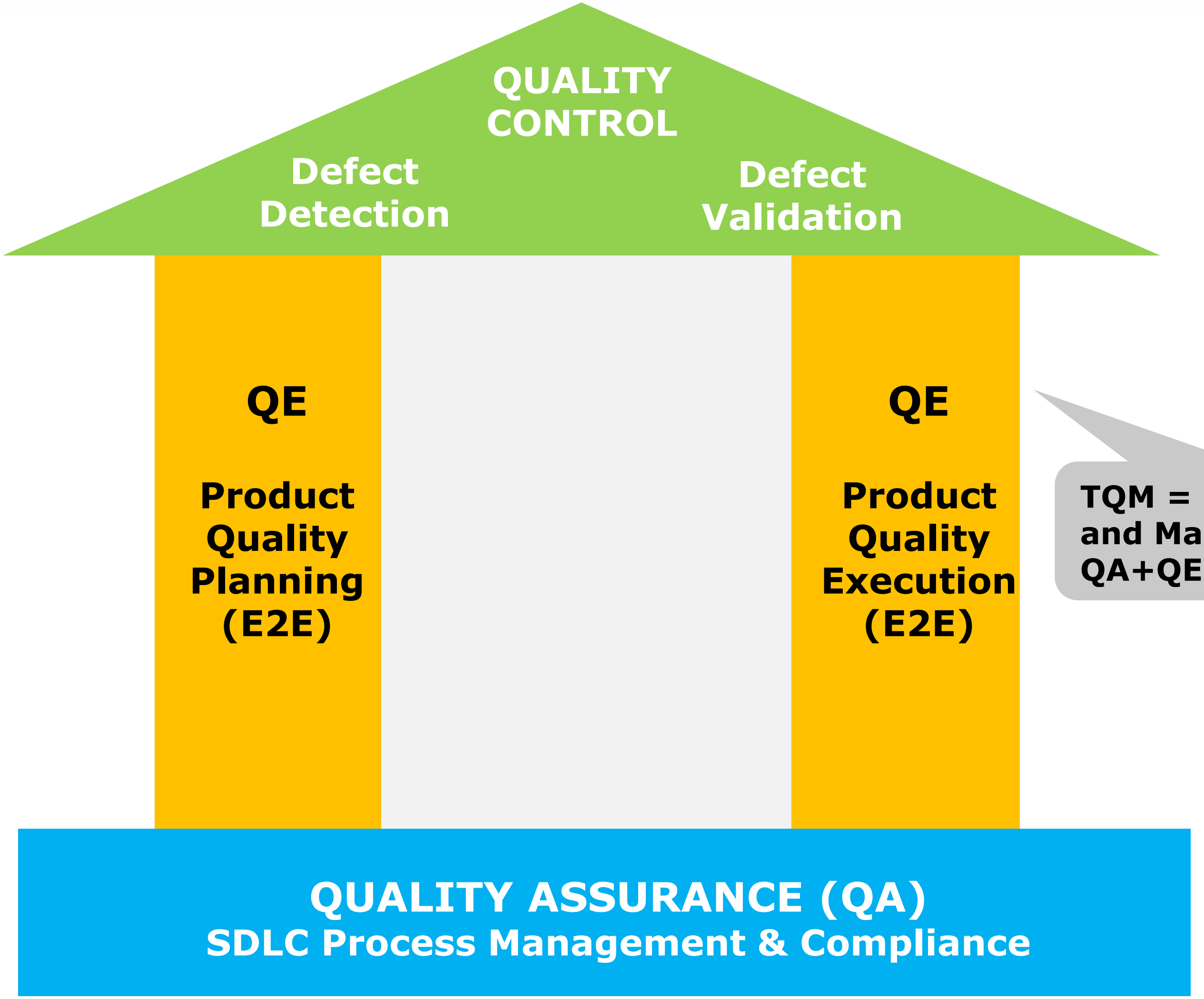
Customer expectations

- Predictive quality management
- System reliability
- Reduction in cycle time
- Reduction in Total Cost of Ownership





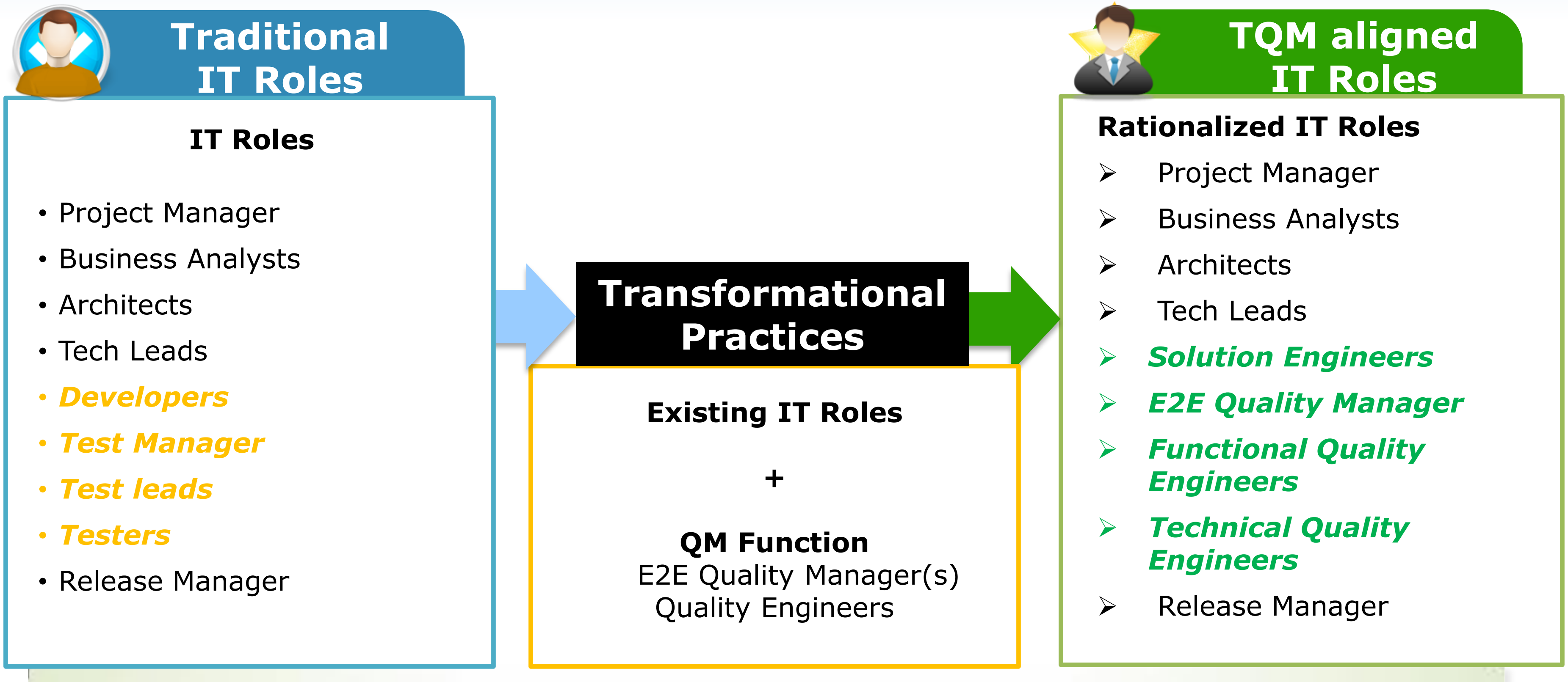
TQM Framework


QUALITY CONTROL
Defect Detection
Defect Validation
QE
Product Quality Planning (E2E)
QE
Product Quality Execution (E2E)
QUALITY ASSURANCE (QA)
SDLC Process Management & Compliance

TQM = Orchestration and Management of QA+QE+QC

QA – Quality Assurance **QE** – Quality Engineering **QC** – Quality Control aka Testing **TQM** – Total Quality Management

Approach to TQM



Traditional IT Roles

IT Roles

- Project Manager
- Business Analysts
- Architects
- Tech Leads
- ***Developers***
- ***Test Manager***
- ***Test leads***
- ***Testers***
- Release Manager

Transformational Practices

Existing IT Roles

+

QM Function
E2E Quality Manager(s)
Quality Engineers

TQM aligned IT Roles

Rationalized IT Roles

- Project Manager
- Business Analysts
- Architects
- Tech Leads
- ***Solution Engineers***
- ***E2E Quality Manager***
- ***Functional Quality Engineers***
- ***Technical Quality Engineers***
- Release Manager

Salient Points

- Expansion of responsibilities for all IT roles to engineer quality right the first time
- QA Head / QA Directors to drive quality across lifecycle (not to confine as Testing owners)
- Detailed competency profiling for fitment and training programs
- Follow Edward Deming's quality principles for sustenance
- Quality Circles – Empowered cross functional teams for innovation/improvements
- Right level of collaboration (not little, not too much)
- Recognition / Awards (employees and partners)

Quality Management Function

Facilitate, Coach and Govern Quality across the lifecycle

Quality Management Governance

- Executive Sponsor
- Owner of Quality Management
- QM Partner

QMO
(Tightly integrated with LoB/Program)

- IT Leadership
- Program Manager(s)
- Business Leaders
- End to End Quality Manager

Quality Champions
(Project level)

- BA Leads
- Quality Engineers
- Architects
- Tech Leads
- Test Leads
- Business SME

Quality Management Function

Roles & Responsibilities

- E2E Quality Coach/ Manager
  - Quality Manager for the entire lifecycle
- Functional Quality Engineer
  - Quality expert for Requirements and Functional design works closely with Business Analysts and Business Leads
- Technical Quality Engineer
  - Quality expert for Technical/ Structural design and build works closely with Tech Leads, Architects and Release Managers)

Tasks

- SDLC Quality planning/ roadmap
- SDLC Quality metrics management
- Work product quality assessments (Requirements to Deployment)
- End to end bi-directional traceability
- Quality Engineering tools (ZDLC)
- SDLC Cost of Quality
- Quality Forecasting
- Governance and Risks Management

Key Practices
Advanced Quality Planning

Advanced Quality Planning: Control checkpoints to build & measure quality across the project lifecycle defined along with the project plan

Tasks

- Collaborative sessions with PM, BA Leads, Tech Leads, Architects and Test Leads
- Identification of business functions at which quality is measured – involves Business Leader of the project
- Define key metrics to measure quality depending on tech stack and business criticality
- Tools and rule sets to support static code analysis, complexity and code coverage
- Schedule of work product quality assessments

Outcome

- Control checkpoints by work products
- PM to update project plan with tasks, resources and dates as agreed in the quality plan
- Base Quality Performance Index (QPI) model with metrics and checklist items
- Communication of quality plan to project team members
- Conduct QPI assessments as per project plan schedule throughout project

Key Practices

Quality Performance Index (adaptable to Agile/Waterfall methods)

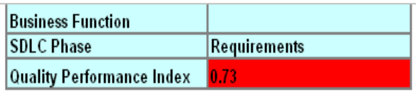
QPI: Objective and quantitative measure of quality across the project lifecycle. It complements CPI and SPI metrics.

Tasks

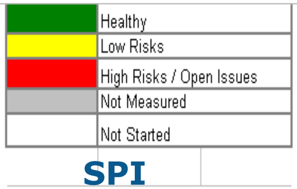
- Collaborative assessment of work product quality with BA Leads, Tech Leads, Architects and Test Leads
- Conducted as part of TL scrum & sprint closure meetings (Build and Test phase)
- Open items and issues are discussed and actions are followed daily or weekly basis

Outcome

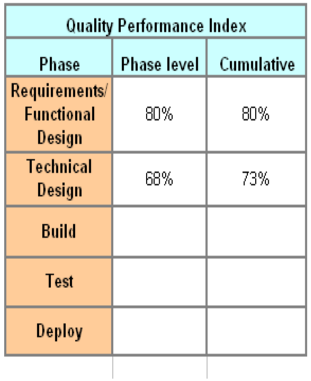
- QPI on a weekly basis published to Leadership team
- Detailed action items
- Risks/Issues mitigation/resolution



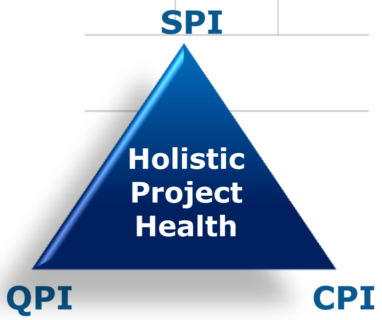
Business Function	
SDLC Phase	Requirements
Quality Performance Index	0.73

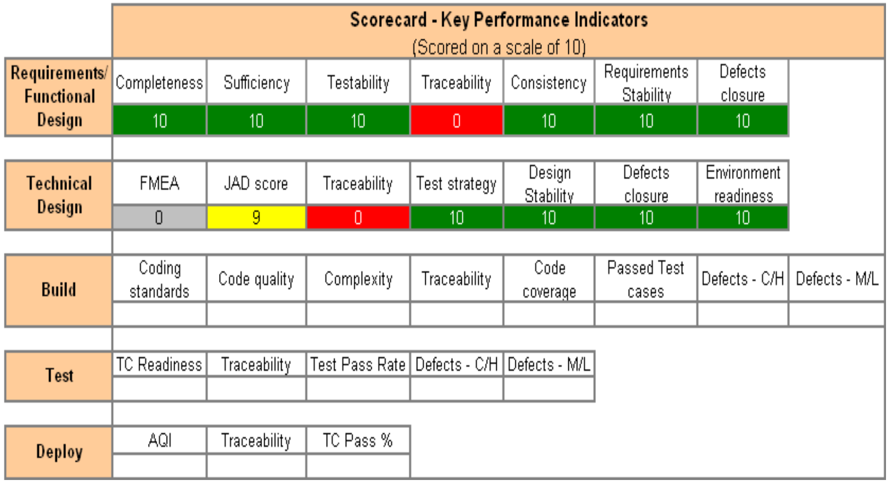


🟩	Healthy
🟨	Low Risks
🟥	High Risks / Open Issues
▨	Not Measured
	Not Started



Quality Performance Index		
Phase	Phase level	Cumulative
Requirements/ Functional Design	80%	80%
Technical Design	68%	73%
Build		
Test		
Deploy		





	Scorecard - Key Performance Indicators (Scored on a scale of 10)							
Requirements/ Functional Design	Completeness	Sufficiency	Testability	Traceability	Consistency	Requirements Stability	Defects closure	
	10	10	10	0	10	10	10	
Technical Design	FMEA	JAD score	Traceability	Test strategy	Design Stability	Defects closure	Environment readiness	
	0	9	0	10	10	10	10	
Build	Coding standards	Code quality	Complexity	Traceability	Code coverage	Passed Test cases	Defects - C/H	Defects - M/L
Test	TC Readiness	Traceability	Test Pass Rate	Defects - C/H	Defects - M/L			
Deploy	AQI	Traceability	TC Pass %					

Key Practices

Requirements Management

Requirements Classification

- Classify requirements -functional, & non-functional requirement etc.
- Build reusable business rules database and link functional requirements accordingly

Parent Child Relationship

- Establish the relationship between functionalities (Req, Specs, Rules)
- Assists in impact analysis, improves Business and IT communication

Bi-directional Traceability

- Requirements Coverage in a systematic manner
- End to end traceability with robust impact analysis and effort quantification

Metric Management

- Requirements Stability Index by Effort
- Requirements Change Control Board with approvers and dollar value cap
- Requirements cost as % of project cost

Requirements Versioning

- Eliminate references to obsolete documentation
- Build stringent Sign – Off process

Prototyping and Visualization

- Limit assumptions & provide functionality clarifications
- Results in early involvement of IT SMEs
- Encourage code reuse, build business and IT collaboration

Requirements Prioritization

- Helps BA and other teams to focus on critical requirements
- Top down view of requirements

Automation

- Automate test case generation by providing detailed information in requirements and reducing redundancy
- Encourage tool based approval management, version control, tagging & tracing

Key Practices

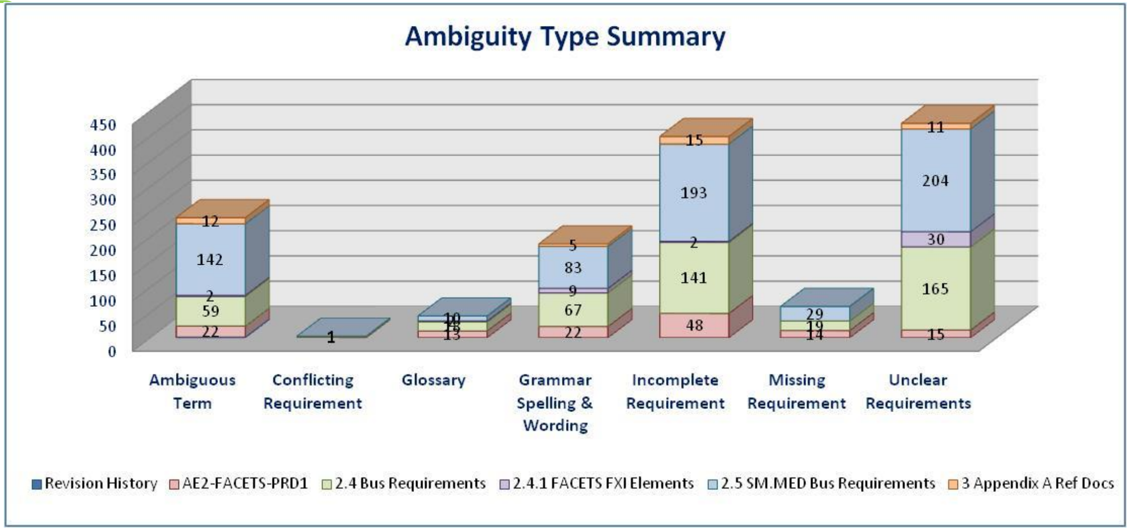
Automated Specifications Reviewer

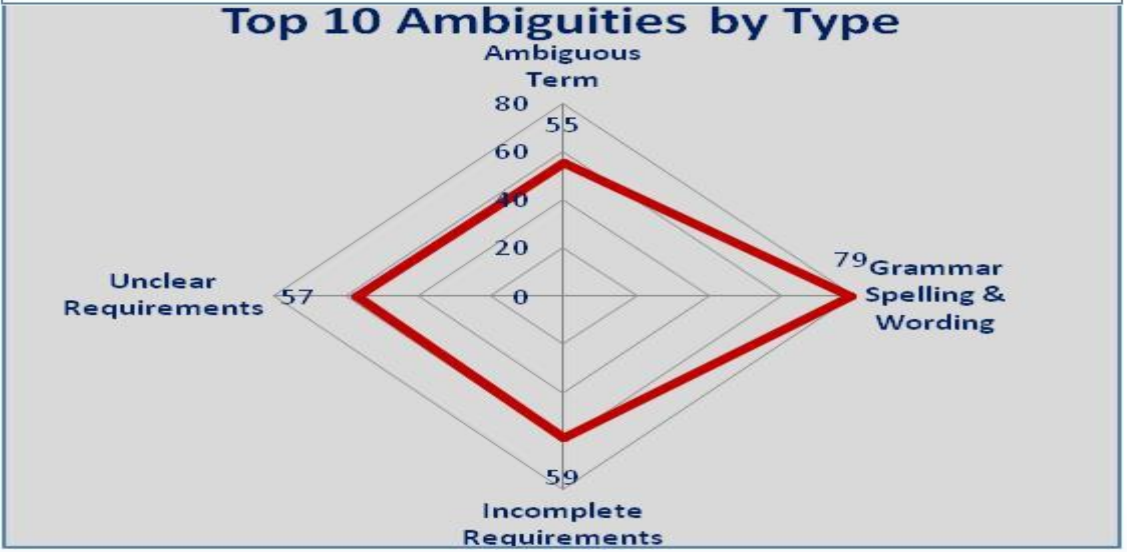
What is Proofer ?

- Proofer is Cognizant (IP) tool for automated static review of requirements and specifications.
- Brings out ambiguities and missing/ incomplete/unclear requirements.

Features

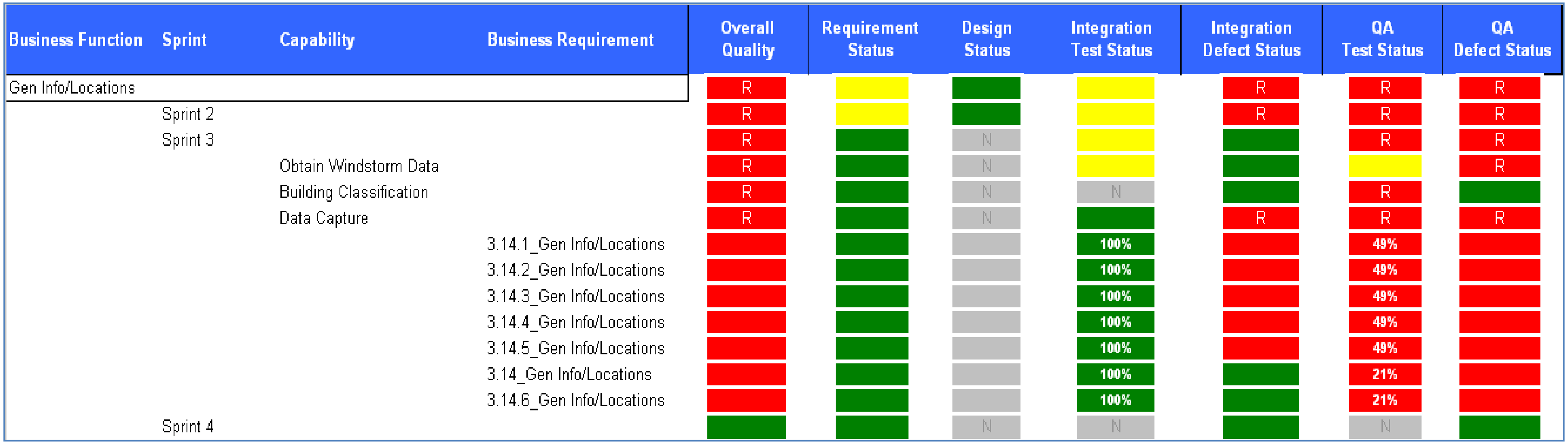
- Easy to use interface with standard filters.
- Automated tool scans and provide report in a faster manner.
- Dictionary can be customized to align with industry domain terminologies.
- Actionable and detailed report is generated.
- Supports documents in .doc, .docx, .rtf and .xls formats





Key Practices

Business Readiness Dashboard



Business Function	Sprint	Capability	Business Requirement	Overall Quality	Requirement Status	Design Status	Integration Test Status	Integration Defect Status	QA Test Status	QA Defect Status
Gen Info/Locations				🟥 R	🟨	🟩	🟨	🟥 R	🟥 R	🟥 R
	Sprint 2			🟥 R	🟨	🟩	🟨	🟥 R	🟥 R	🟥 R
	Sprint 3			🟥 R	🟩	⬜ N	🟨	🟩	🟥 R	🟥 R
		Obtain Windstorm Data		🟥 R	🟩	⬜ N	🟨	🟩	🟨	🟥 R
		Building Classification		🟥 R	🟩	⬜ N	⬜ N	🟩	🟥 R	🟩
		Data Capture		🟥 R	🟩	⬜ N	🟩	🟥 R	🟥 R	🟥 R
			3.14.1_Gen Info/Locations	🟥	🟩	⬜	🟩 100%	🟥	🟥 49%	🟥
			3.14.2_Gen Info/Locations	🟥	🟩	⬜	🟩 100%	🟥	🟥 49%	🟥
			3.14.3_Gen Info/Locations	🟥	🟩	⬜	🟩 100%	🟥	🟥 49%	🟥
			3.14.4_Gen Info/Locations	🟥	🟩	⬜	🟩 100%	🟥	🟥 49%	🟥
			3.14.5_Gen Info/Locations	🟥	🟩	⬜	🟩 100%	🟥	🟥 49%	🟥
			3.14_Gen Info/Locations	🟥	🟩	⬜	🟩 100%	🟩	🟥 21%	🟥
			3.14.6_Gen Info/Locations	🟥	🟩	⬜	🟩 100%	🟩	🟥 21%	🟥
	Sprint 4			🟩	🟩	⬜ N	⬜ N	🟩	⬜ N	🟩

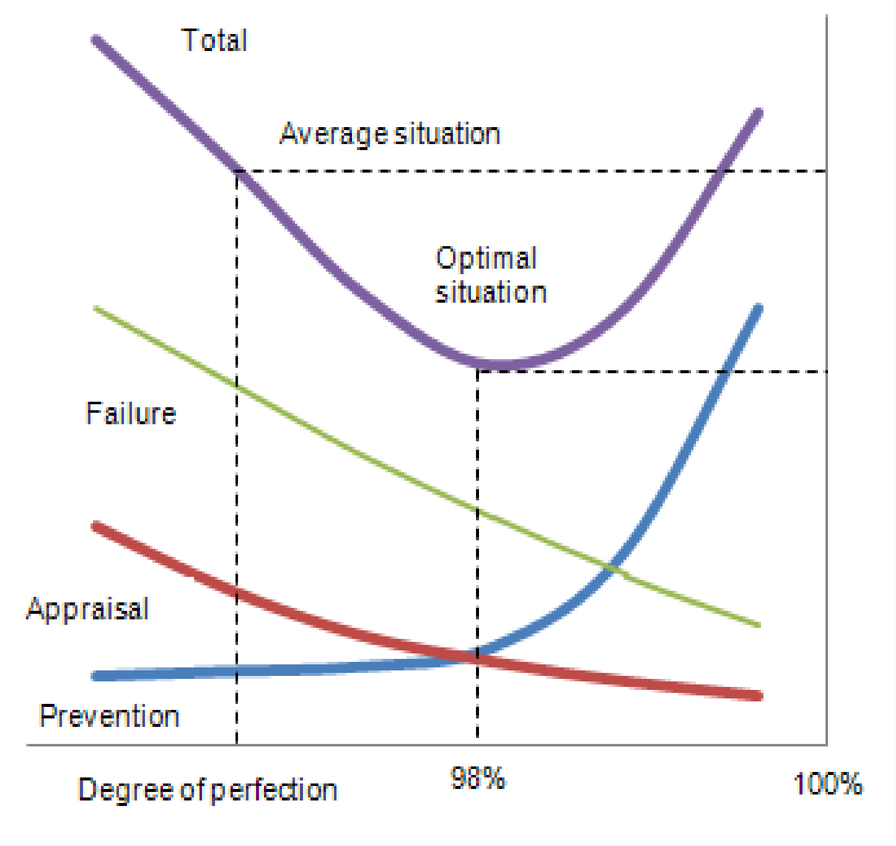
- ✓ Simple, easy and effective measure of overall quality
- ✓ Requirements based standard reporting of overall quality throughout the lifecycle
- ✓ Data / metrics from single source of truth (QC)
- ✓ Bidirectional traceability of requirements to tests to defects
- ✓ Drill down capabilities provides high level status for leadership and detailed views for developers/testers/BAs
- ✓ No manual effort to collect/generate the report

Key Practices
Cost of Quality (CoQ)

Cost of Quality - Categories

- **Work**
- **Prevention**
  - Training
  - Project/ Quality planning
  - Fail proof mechanisms
- **Appraisal**
  - Walkthroughs
  - Reviews/ Inspections
  - Testing (all types)
- **Rework**
  - Failures
  - Defects fixing
  - Defects testing/closure

CoQ = (Efforts spent on Prevention, Appraisal, Rework) / (Total project efforts)



"Quality is free, but not a gift." **- Philip B. Crosby**

Cost of Quality (CoQ) Metrics

Creating meaningful analysis

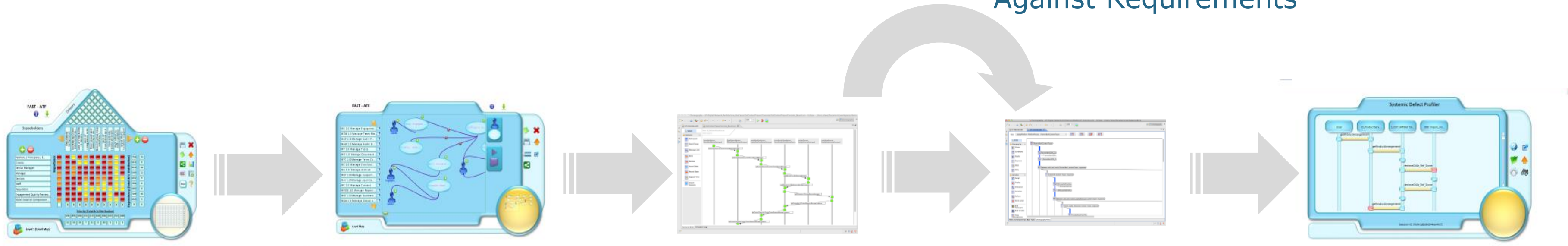
- SDLC Cost of Quality (CoQ) as a % of Project efforts
- Estimated CoQ and Actual to date CoQ
- Rework % (and distribution of rework efforts – estimate & actual)
- CoQ % Vs. Defects (by business function & work stream)
- Prevention & Appraisal Vs. Rework
- Defect density Vs. CoQ

*Use **CoQ** as a leading indicator from estimate to project closure and drive meaningful actions for current and future projects!*

ZDLC Tools - How can they help?

Zero Deviation Lifecycle is a set of engineering quality tools used in the end-to-end lifecycle of systems (*including by not limited to traditional SDLC*). It drives down cost and accelerates delivery through automation and <u>improved quality</u>. It compliments in place ALM platforms. It does this by;

- Structuring the process of requirements capture
- Employing statistical methods to validate the consistency of requirements attributes.
- Employing simulation techniques to test and validate design artifacts against the requirements.
- Reducing leakage of defects between SDLC phases



Test Architecture Against Requirements

Gather Requirements

Create User Stories

Generate Use cases

Create Architecture

Ensure Code meets Design

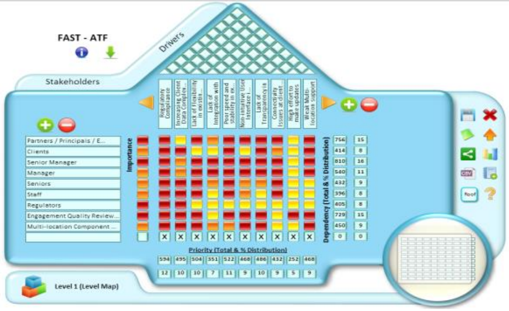
ZDLC Tools - How can they help?

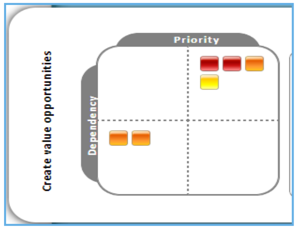
ZDLC to implement the 3 key aspects identified to accelerate the on-boarding process of the UCard Program:

- From a Requirement Perspective
  - Employ the ZDLC tools called **House of Quality (HoQ)** and the **Requirement Modelling Solution (RMS)** to reduce ambiguity and ensure consistency of the requirement specification.
- From an Architectural Perspective
  - Employ the ZDLC tool called **Testable Integration Architecture (TiA)** to model and validate the architecture and design of the Ucard platform so that the configurability and reusability of the components can be constantly tested and improved for new changes in requirements.
- From a Quality of Service (QoS) Perspective
  - Employ the ZDLC tool called **Testable Integration Architecture (TiA)** to simulate the architecture of Platform against key NFRs and QoS measures to continuously optimize the platform.

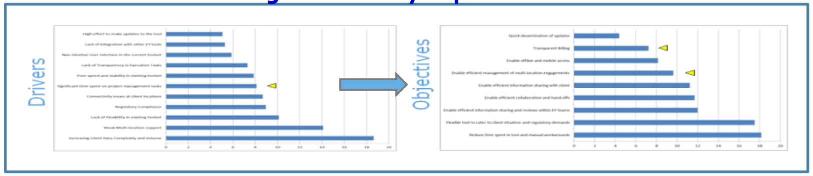
ZDLC Tools – House of Quality (HoQ)

Enables requirements to be collected, validated, prioritized and used for transparent decisions resulting in higher quality and rapid outcomes. Based on a proven methodology used in the manufacturing domain as part of Quality Function Deployment (precursor to Lean and Six Sigma)





Priority & Co-dependency objectively quantified



- Use of the latest technologies (e.g. HTML5 and Mobile) makes UI intuitive and easily accessible.
- New approaches to traditional tasks (e.g. Gamification) encourages usage and adoption.
- Prioritized and dependency-aware traceability Matrix.

Where to use HoQ?

Requirements elicitation & analysis, decision support, business case definition, consensus building

Benefits:

- Faster consensus-building &decision making across multiple stakeholders
- Significantly reduced risk of requirements churn in downstream phases
- Generates objective, unambiguous requirements aligned to stakeholder concerns.
- Reduced effort and timelines in the Requirements.
- Incremental sign-off allows better buy-in and alignment between business and IT.
- Graphical representation allows rapid and effective quality control and PM governance
- Facilitates Change Management by allowing rapid Scope and Change impact analysis

ZDLC Tools – Requirements Modeling Solution

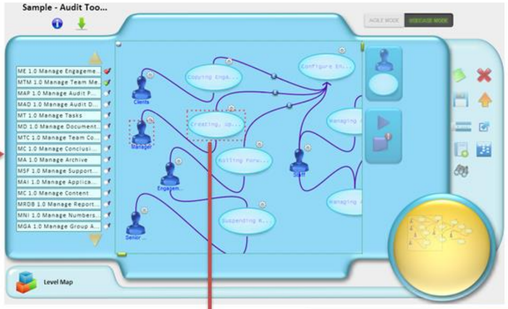
Enables system requirements to be defined and modelled using industry-standard UML notations. Fully integrated with HoQ allowing requirement/stakeholder traceability to use-cases.

Where to use RMS?

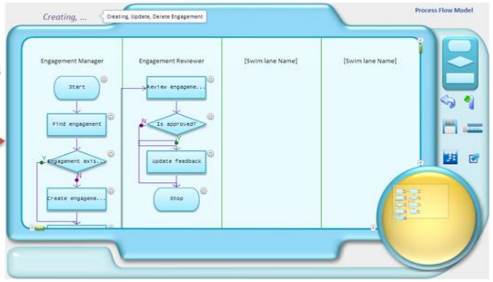
Systems Analysis, Modelling, UML generation

Attributes:

- Enforces structured approach for the capture and representation of system requirements as Use Cases and process models.
- Architects and Analysts can view the full decision process underlying requirements before modelling system behaviour.
- Scenarios are generated using advanced graph theory to ensure complete coverage of requirements and can be used for Test Planning.


Sample - Audit Too...
AGILE MODE
USECASE MODE
ME 1.0 Manage Engageme...
MTM 1.0 Manage Team Me...
MAP 1.0 Manage Audit P...
MAD 1.0 Manage Audit D...
MT 1.0 Manage Tasks
MD 1.0 Manage Document...
MTC 1.0 Manage Team Co...
MC 1.0 Manage Conclusi...
MA 1.0 Manage Archive
MSF 1.0 Manage Support...
MAI 1.0 Manage Applica...
MC 1.0 Manage Content
MRDB 1.0 Manage Report...
MNI 1.0 Manage Numbers...
MGA 1.0 Manage Group A...
Clients
Manager
Engagem...
Senior...
Staff
Copying Enga...
Creating, Up...
Rolling Forw...
Suspending A...
Configure En...
Managing
Managing
Level Map

Each User Story has a **Process Flow Diagram**



Benefits:

- Ease of use enables organizations to adopt model-based software specs as opposed to textual description.
- Eliminate ambiguity and risk of future rework through precise notation and modelling
- Automation in generation of models and design artefacts results in reduced effort and time.
- Accelerate IT project planning by preserving prioritization and dependencies from HoQ.

ZDLC Tools – Testable Integration Architecture

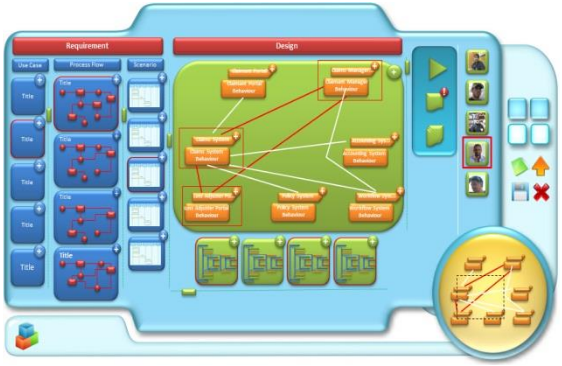
Enables software design using industry-standard UML notations to be tested and validated against the user requirements.

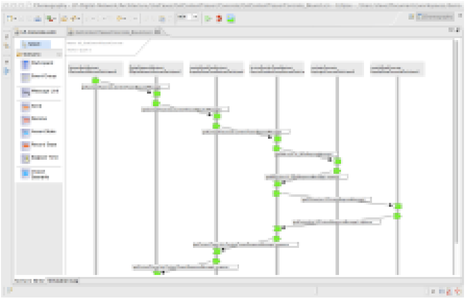
Attributes:

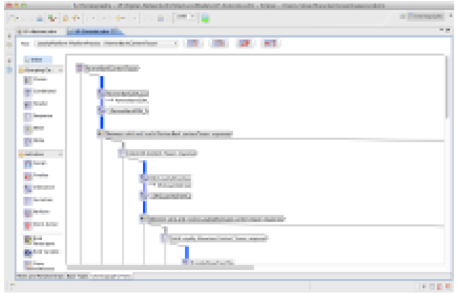
- TiA enables designers to model the integration architecture of software systems and platforms.
- TiA allows UML models to be validated against pre-defined business requirements using formal methods
- TiA identifies Re-usable Design components of software systems
- TiA is fully integrated with RMS, allowing traceability from Business requirements down to granular designs and specifications of the communicating systems.
- Business scenarios in the shape of sequence diagrams are used to generate the integration architecture.
- System designs and models artefacts can be exported into Word format for dissemination and review

Where to use TiA?

Systems Analysis, Modelling, UML generation, Formal Technical Review





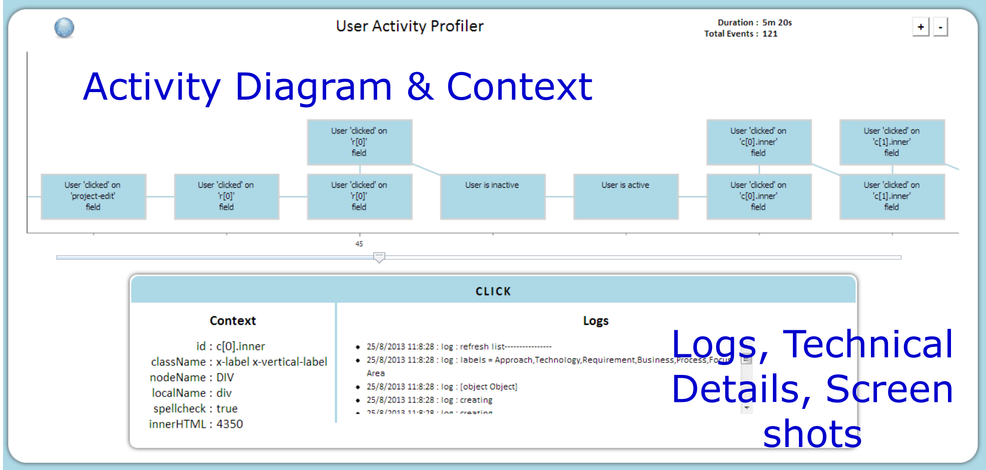


Benefits:

- Early detection of design defects.
- Reduces the amount of design defects leaking to coding
- Reduced effort and timelines due to automation in generation and validation of design artefacts.
- Reduces the defect density count in SDLC

ZDLC Tools – User Activity Profiler (UAP)

Enables monitoring & recording of client side User activities from a web based browser and represents them in the form of an Activity Diagram which is fully enriched with Context, Screen shots, Error details, Logs & other technical details to help the development & support teams reduce the effort in Testing, Root cause analysis & fixing the issues.


User Activity Profiler
Duration : 5m 20s
Total Events : 121
+
-
User 'clicked' on 'project-edit' field
User 'clicked' on 'r[0]' field
User 'clicked' on 'r[0]' field
User 'clicked' on 'r[0]' field
User is inactive
User is active
User 'clicked' on 'c[0].inner' field
User 'clicked' on 'c[0].inner' field
User 'clicked' on 'c[1].inner' field
User 'clicked' on 'c[1].inner' field
45
CLICK

Activity Diagram & Context

Context

id : c[0].inner
className : x-label x-vertical-label
nodeName : DIV
localName : div
spellcheck : true
innerHTML : 4350

Logs

```
• 25/8/2013 11:8:28 : log : refresh list---------------
• 25/8/2013 11:8:28 : log : labels = Approach,Technology,Requirement,Business,Process,Focus
  Area
• 25/8/2013 11:8:28 : log : [object Object]
• 25/8/2013 11:8:28 : log : creating
• 25/8/2013 11:8:28 : log : creating
```

Logs, Technical Details, Screen shots

Attributes:

- Use of the latest technologies (e.g. HTML5 and Mobile) makes UI intuitive and easily accessible.
- New way of visualizing User activities in the form of an Activity Diagram.
- Enriched context with Screen shots & logs
- Plugin model to easily inject the framework on target application

Where to use UAP?

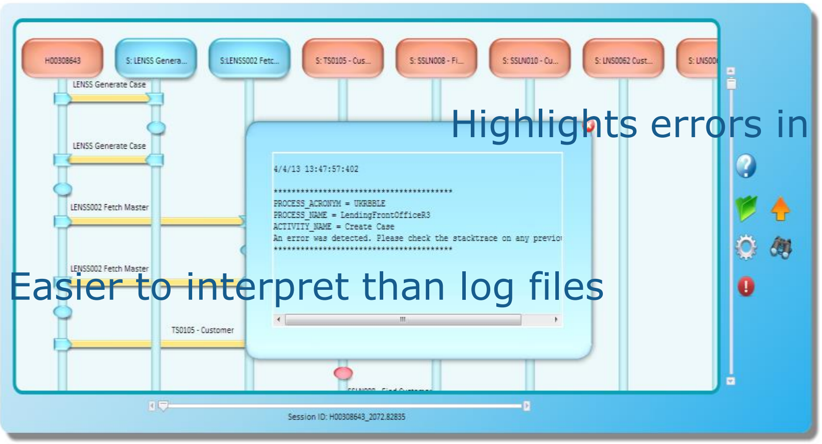
Systems Analysis, Root cause Analysis, Reverse Engineering, UI Analysis, Testing & Quality Control

Benefits:

- User activity is visualized as a model (Activity Diagram) and is annotated with context, screen shots, logs & technical details
- Simplifies the process involved in reporting issues for testers
- End users can report issues without the trouble of calling Customer care
- No communication gap in reporting the issues to developers
- Reduced effort of the testers
- Supports UI Analysis & Debugging
- Improved Quality Control as errors are automatically recorded in production

ZDLC Tools – Systemic Defect Profiler(SDP)

Reverse engineers System, Application & Network logs into Sequential diagrams. Uses the sequence diagrams to systematically identify Root cause of problems. Generates scenario models to form as input to create Architecture models of a software system. Delivers a reliable & interpretable form of "VOICE OF THE MACHINE".



```
4/4/13 13:47:57:402

****************************************
PROCESS_ACRONYM = UKRBBLE
PROCESS_NAME = LendingFrontOfficeR3
ACTIVITY_NAME = Create Case
An error was detected. Please check the stacktrace on any previo
****************************************
```

Highlights errors in RED

Easier to interpret than log files

Attributes:

- Use of the latest technologies (e.g. HTML5 and Mobile) makes UI intuitive and easily accessible.
- Eliminates learning curve for new starters
- Automated log parsers eliminate human errors during defect fixing.
- Automates Root Cause Analysis, reduces defect fix effort

<u>Where to use SDP?</u>

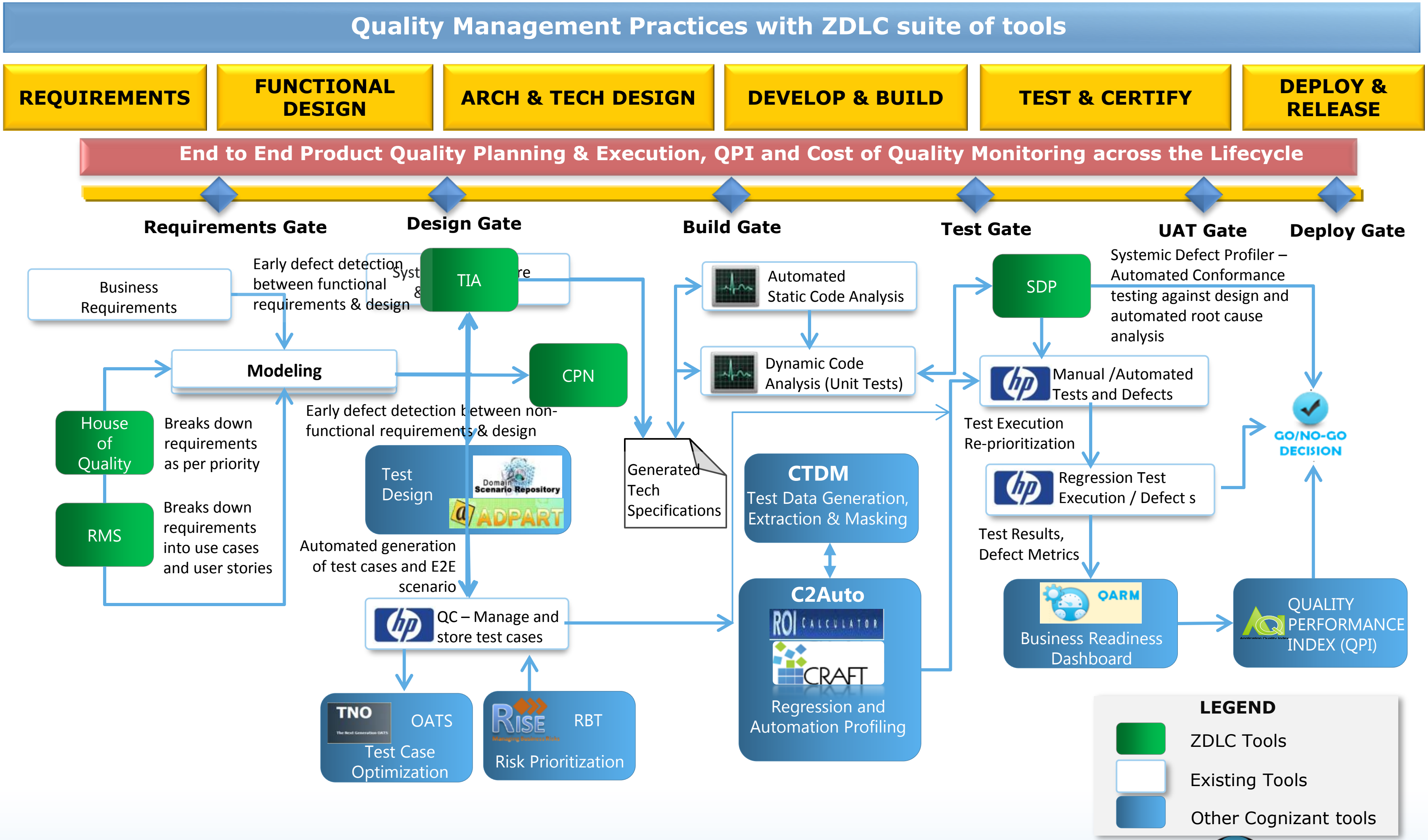
Systems Analysis, Root Cause Analysis, Scenario Models, Architecture Models, Reverse Engineering

Benefits:

- The VOICE OF THE MACHINE is never wrong. SDP report exemplifies Trust & Reliability.
- Developers easily understand Scenario models. Defect fixing is easier with the SDP report.
- Non-intrusive method of reverse engineering without needing the knowledge of underlying source code.
- Gathers multiple logs into one scenario providing an end to end picture of business scenario.
- Reduction in effort of Test & Development teams.
- Faster time to market!

Workflow

Zero Deviation Life-Cycle (ZDLC) for seamless integration of SDLC work products to build and measure End to End Quality



Quality Management Practices with ZDLC suite of tools

REQUIREMENTS | **FUNCTIONAL DESIGN** | **ARCH & TECH DESIGN** | **DEVELOP & BUILD** | **TEST & CERTIFY** | **DEPLOY & RELEASE**

End to End Product Quality Planning & Execution, QPI and Cost of Quality Monitoring across the Lifecycle

Requirements Gate | **Design Gate** | **Build Gate** | **Test Gate** | **UAT Gate** | **Deploy Gate**

Business Requirements

Early defect detection between functional requirements & design

Syst re &

TIA

Modeling

CPN

House of Quality

Breaks down requirements as per priority

RMS

Breaks down requirements into use cases and user stories

Early defect detection between non-functional requirements & design

Test Design

Domain Scenario Repository

ADPART

Automated generation of test cases and E2E scenario

QC – Manage and store test cases

TNO

The Next Generation OATS

OATS

Test Case Optimization

RISE

RBT

Risk Prioritization

Generated Tech Specifications

Automated Static Code Analysis

Dynamic Code Analysis (Unit Tests)

CTDM

Test Data Generation, Extraction & Masking

C2Auto

ROI CALCULATOR

CRAFT

Regression and Automation Profiling

SDP

Systemic Defect Profiler – Automated Conformance testing against design and automated root cause analysis

Manual /Automated Tests and Defects

Test Execution Re-prioritization

Regression Test Execution / Defect s

Test Results, Defect Metrics

QARM

Business Readiness Dashboard

QUALITY PERFORMANCE INDEX (QPI)

GO/NO-GO DECISION

LEGEND

🟩	ZDLC Tools
□	Existing Tools
🟦	Other Cognizant tools

Cognizant

TQM - Lifecycle View of Outcome

Business	Business Need Analysis Business Roadmap and CBA	Conduct / Participate in Reviews & Provide clarifications to project team	UAT	User Training/ Support	Enhancements / Defects Prioritization

IT Project Mgmt.	Initiation & Planning	Sprint Planning, Change Control & Management, Issue Tracking, Risk Management, Metrics, Status Reporting & Deliverable Management	Closure

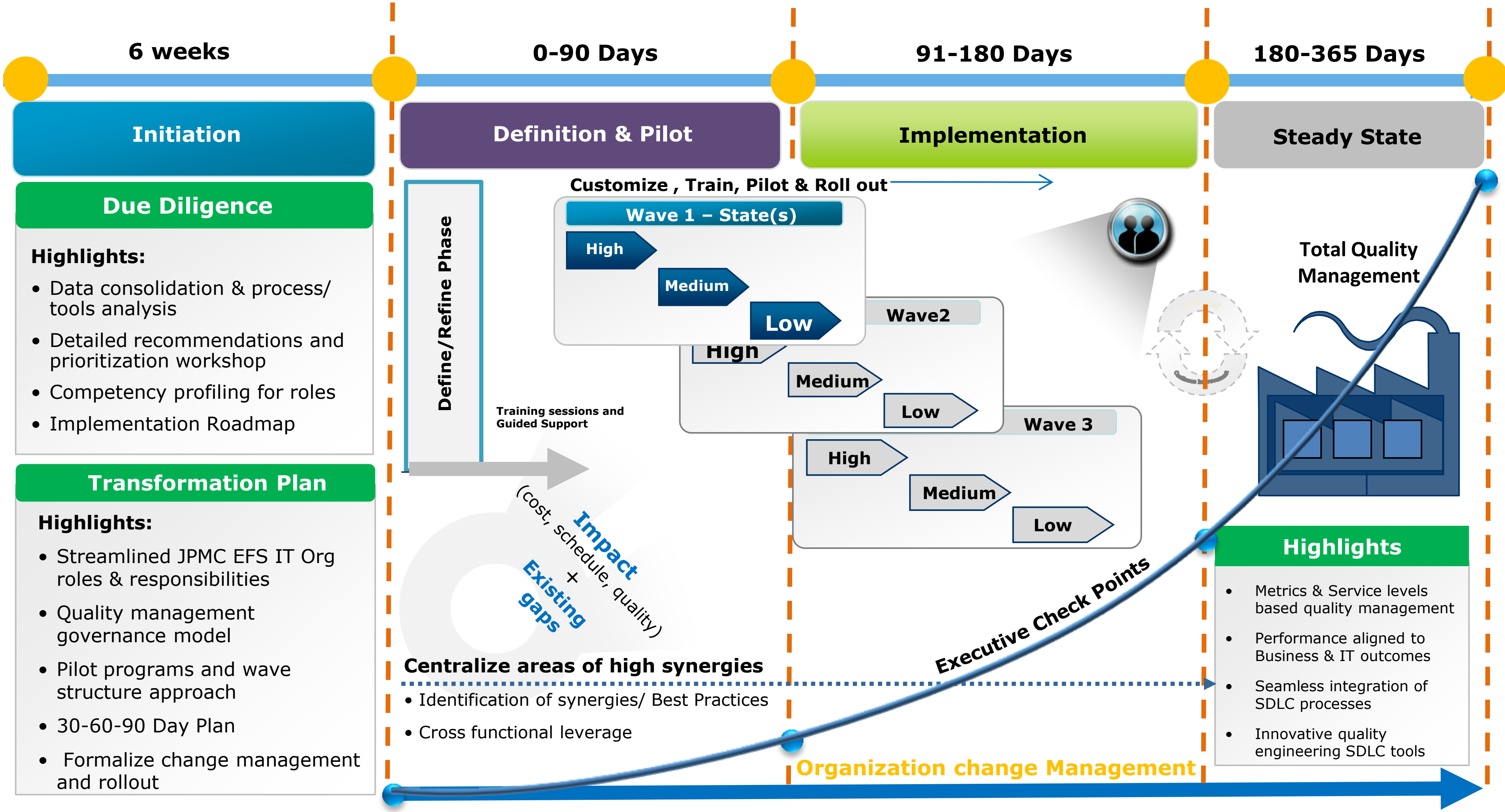
End to End Quality Mgmt.	Program Leadership meetings, Weekly project status meetings, Scrum meetings, War room sessions, Business demo sessions, Change Control meetings, SDLC Quality metrics reporting and governance

	Requirements	Design	Build*	Test*	Deploy
QM Tasks	✓ Advanced Quality planning ✓ Product Quality metrics / service levels ✓ QPI model / checkpoints ✓ Traceability (Requirements to Spec) ✓ Requirements/ Spec QPI assessment (by Business function) ✓ SDLC Cost of Quality - Project plan tasks classification	✓ QPI assessment – Architecture, HLD/LLD, Infrastructure and test strategy readiness ✓ JAD participation ✓ Traceability (Spec to design) ✓ Validation of tools setup/ rule sets for code scans and static analysis ✓ Automatic test case generation from spec ✓ Reporting CoQ (estimate/ATD)	✓ Validation of code scan reviews / analysis ✓ Validation of structural quality results ✓ Traceability (Spec/design to unit test/assembly test) ✓ QPI assessment – sprint closure ✓ Traceability (Spec to system tests) ✓ Reporting CoQ (estimate/ATD)	✓ Code coverage review / analysis ✓ Business Readiness dashboard ✓ QPI assessment – test execution ✓ Traceability (Tests to defects) ✓ Release & deployment readiness ✓ Reporting CoQ (estimate/ATD)	✓ QPI assessment – release mgmt. ✓ Business Readiness dashboard ✓ Release quality reporting ✓ Final review of product quality & process quality metrics ✓ Lessons learnt / quality introspection ✓ Finalized CoQ / Benefits review

* Build - Development, Unit test, Assembly test, Technical integration/ Features test, Component performance tests

* Test – Business Process Test, SIT, Workflow performance test and UAT

Transformation Strategy



6 weeks	0-90 Days	91-180 Days	180-365 Days
Initiation	Definition & Pilot	Implementation	Steady State

Due Diligence

Highlights:

- Data consolidation & process/ tools analysis
- Detailed recommendations and prioritization workshop
- Competency profiling for roles
- Implementation Roadmap

Transformation Plan

Highlights:

- Streamlined JPMC EFS IT Org roles & responsibilities
- Quality management governance model
- Pilot programs and wave structure approach
- 30-60-90 Day Plan
- Formalize change management and rollout

Define/Refine Phase

Customize , Train, Pilot & Roll out

Wave 1 – State(s)

High

Medium

Low

Wave2

High

Medium

Low

Wave 3

High

Medium

Low

Training sessions and Guided Support

Impact (cost, schedule, quality) + Existing gaps

Centralize areas of high synergies

- Identification of synergies/ Best Practices
- Cross functional leverage

Executive Check Points

Organization change Management

Total Quality Management

Highlights

- Metrics & Service levels based quality management
- Performance aligned to Business & IT outcomes
- Seamless integration of SDLC processes
- Innovative quality engineering SDLC tools

Transformation – Rollout Options

Implementation Options	Scope of Work	Key Outcomes
FULL IMPLEMENTATION - - **End to End accountability** of QM (Development & Execution) - **Quality management Tools & Solutions** evaluation and Implementation	- All QM practices will be implemented - Suitable for all Application Development work - Supports Agile, Waterfall and hybrid methodologies	- Ability to baseline, monitor and report **Cost of Quality** - Establish and Enforce **End to End Quality governance & gates** across SDLC - **Quality Performance Index** (QPI) - Objective and quantitative measure of quality (SDLC) - Effective mitigation of **work product's Quality risks** (interdependencies) - **SQALE** – Source code quality and technical debt evaluation - **Business Readiness Dashboard** - Use of **ZDLC suite of tool(s)**
SELECTIVE IMPLEMENTATION – - Definition and Execution of selected Quality Management practices based on **business priorities and current process maturity**	- Select practices from all levels and packages will be implemented - Suitable for Major releases of Application Maintenance work with Agile or waterfall methods	- Key outcomes listed under full implementation depending on the selected practices
LEAN IMPLEMENTATION – - Definition and Execution of selected Quality Management practices aimed at **achieving quick results**	- Thin versions of selected practices / solutions will be implemented - Suitable for minor releases of Application maintenance work	- Key outcomes listed under full implementation depending on the selected practices

Benefits & Business Value

- **50% reduction in SDLC defect density** measured in terms of defects to SLDC efforts
- **10% - 15% reduction in SDLC** cost of quality (CoQ)
- **Transparency and visibility into the quality** of SDLC work products in a timely manner
- **Provides confidence to business** and reduces the dependence on UAT for quality control
- **Faster go-live to market** due to defect prevention in requirements, design & build phases
- Harmonization of SDLC processes to deliver the expected quality in a predictable manner

Benefits & Business Value

Key Performance Metrics and Guidelines

Metrics	GREEN	YELLOW	RED	Dependencies
SDLC Cost of Quality as % of project efforts	<=40%	>40% and <=50%	>50%	• Adherence to quality plan • Accurate data collection process/ monitoring • Project management/ governance
Cost Variance (QM work)	<1%	>=1% and <=2%	>2%	• Change Controls • Adherence to quality plan
QPI effectiveness [measured in terms of defect density by effort*]	<=0.15	>0.15 and <0.25	>0.25	• Adherence to quality plan • Transparency/ visibility into quality of work products across the lifecycle
Defect leakage [Production]	No critical or high severity defects	No critical defects and <1% High severity defects	>0 Critical defects Or >1% High severity defects	• Effectiveness of QPI assessment reporting and governance • Business SME involvement in the project
Schedule variance (QM work)	0%	>0% and <=1%	>1%	•Change Controls • Adherence to quality plan

*** Ratio of valid defects monitored during independent testing (System Test, UAT) and Warranty to the project efforts**
Note: Service levels need to be established at enterprise level based on current process maturity and business goals

Case Studies

Cost of Quality across the lifecycle	Before TQM	After Year 1 of TQM	Industry Benchmarks
CoQ (as % of project efforts)	55%	45%	34%

Benchmark source: *Mckinsey, SPIN and ISBSG for Application development projects.*

Note: Total project efforts include efforts of Requirements, Design, Build, Test , Deployment/Implementation and Warranty. It also includes PM and QM efforts.

Case Study 1 (Large Financial Services client)

- Estimated CoQ at beginning of Quality Management implementation was 55%.
- Actual CoQ at end of warranty was 45%: 10% savings!
- Robust requirements & design (JAD) with RSI by effort at healthy levels (1.05).
- Rigorous code review, unit test/assembly test and technical integration tests.
- 96% pass rate from day 1 of System test to UAT closure! No critical defects!
- Zero post production defects!!

Case Study 2 (E-commerce program for an Insurance client)

- 50% reduction in SDLC defect density
- 15% reduction in the overall SDLC Cost of Quality (CoQ).
- Reduction in system testing defects by 75%.
- 95% Pass Rate realized day within the first days of system testing.
- Project went live 2 weeks ahead of schedule!
- Return on Investment directly from TQM initiatives: 4:1 (measured using CoQ savings).

Case Studies

Smart Technology Migration with ZDLC for Global Financial Services Firm

- <u>Context/Problem</u>:

  Client had several BPM Technologies (unplanned) and wanted to rationalize by consolidating onto a single product suite, but some platforms were undocumented

  - 10+year old platform without any proper documentation
  - License renewal date upcoming and Client facing significant unwanted financial outlay if not off platform by then
  - Operations team no longer knew ALL of underpinning functionality with significant ad hoc components added
  - Fear of loss of service if process was 'rushed' <u>but</u> limited budget and SME 'face time' available

- <u>Approach</u>:

  Smart Technology Migration with ZDLC's *Systemic Defect Profiler (SDP) & Testable Integration Architecture (TiA)*

  - Semi-automated reverse-engineering of legacy system from Log Files with minimal demand on SMEs' time
  - Transcribed legacy system design into industry-standard testable models and performed simulations
  - Fully automated generation of notationally-correct BPMN2 specifications for direct upload into IBM WS BPM 7+

- <u>Outcomes/Values</u>:

  - Systematically inferred complete behavior of legacy system otherwise impossible through conventional means
  - Re-instated complete documentation set and simulation capability for system
  - Achieved SAFE lift and Shift within timeframes allowing Client to NOT need to renew License Agreement
  - Project delivered at 41% of original time/cost projections

Cognizant

Appendix

Quality Engineering Tools

Tool	Capability Summary
ADPART™ The Ultimate Test Design Machine	• ADPART is a revolutionary product that redefines the Future of Model Based Testing. • Automated Test case Optimization, Prioritization, Impact Analysis, Risk Analysis, Regression Analysis and Requirements traceability can be performed with better quality at a lesser cost and time.
sonarqube	• SonarQube is an open source platform to manage code quality. It covers 7 axes of quality such as Architecture & Design, Duplications, Unit Tests, Complexity, Potential bugs, Coding rules and Comments. It is a web based application that can be extended with Commercial plugins.
FitNesse	• FitNesse is an open source collaboration test framework. It runs on a dedicated wiki server that can be accessed over a web browser by developers, testers and customers to create automated test cases integrated with narrative requirements usually for acceptance criteria testing.
jbehave	• JBehave is a framework for Behavior-Driven Development (BDD) • It shifts the vocabulary from being test-based to behavior-based, and positions itself as a design philosophy
HUDSON Extensible continuous integration server	• Hudson is powerful and widely used web-based Open Source "Continuous Integration" server sourced by Eclipse Foundation • Provides development teams with a reliable way to monitor changes in source control and trigger a variety of builds • Integrates easily with most version control systems and bug databases
Jenkins	Jenkins is an award-winning application that monitors executions of repeated jobs, such as building a software project or jobs run by cron. Among those things, current Jenkins focuses on the following two jobs: • Building/testing software projects continuously • Monitoring executions of externally-run jobs.
PROOFER	• Proofer is Cognizant (IP) tool used for static testing in requirement phase of SDLC. • It is used to identify issues related to clarity and ambiguity in requirement documents.

What is aLite?

- Automation Lite (aLite),helps you to reduce manual interventions in the process of quality engineering and test script execution.
- It enables complete automation of build deployment, automation test execution and continuous integration.

Key Features

- Enables continuous integration – automated build deployment and unit tests scheduling along with QE metrics reporting.
- Effective scheduling, reassignment and load balancing of test automation scripts across various VDIs/Machines.
- Technology Independent- works on top of all automation tools/scripts like QTP, Selenium, VSTS, Soap UI and others.
- Test Case Effectiveness Analysis based on line coverage which is an ideal indicator for automation candidature.
- Requirement based execution of TCs and Auto rerun of failed TCs
- Auto/Manual reassign of TCs to different systems
- Web based user interface which enables automation engineer to access easily from anywhere to monitor/reschedule
- SMS and email alerts and notifications so that continuous monitoring needed

Benefits

- Increased and effective system utilization.
- effectively implement test automation by managing the end to end process
- Reduction in manual intervention and monitoring
- Flexibility to schedule automation scripts, and subsequent unattended execution
- Reduction in cycle time by reducing the downtime on script execution.

QE Technique	Applicable Technology	Applicable SDLC Phase
Continuous Execution	Java / Dot Net / Mainframe / SAP	Build / Test automation

ADPART™
The Ultimate Test Design Machine

What is Adpart ?

- ADPART is a Revolutionary Product that Redefines the Future of Model Based Testing.
- Automated Test case Optimization, Prioritization, Impact Analysis, Risk Analysis, Regression Analysis and Requirements Traceability

can be performed with better quality at a lesser cost and time, With a single click.

Key Features

- Model business flows, embed business flows to any level
- Automated Test scenario ,Test case Generation, Test case prioritization and optimization, Requirement traceability and Regression Analysis
- Automated Impact Analysis for Defects and change in Requirements.
- Rule Based Test case generation
- Create Smart test suites based on Requirements, Defects and scenarios
- Shared Work space and effective configuration management of Test Artifacts.
- Import/Export business models and Test Artifacts
- Enables modeling in multiple languages
- Seamless Integration with HP ALM

Benefits

Faster:

- Automated Test Creation
- Test Selection
- Change Management
- Regression Analysis
- Knowledge Transfer

Effective:

- Requirement Analysis
- Requirement to Test Transformation
- Requirement Traceability
- User Friendly
- Better Coverage
- Change Management
- Implemented in all Domains

Smarter:

- Test Optimization
- Rule Based Test Creation
- Better Test Coverage
- Risk Based Testing
- Testing Business Criticality
- Categorization of Test Cases

QE Technique	Applicable Technology	Applicable SDLC Phase
Model Based Testing	Java / Dot Net / Mainframe / SAP	Requirement / Test Design

What is SonarQube ?

- SonarQube is an open source platform to manage code quality. It covers 7 axes of quality such as Architecture & Design, Duplications, Unit Tests, Complexity, Potential bugs, Coding rules and Comments. It is a web based application that can be extended with Commercial plugins.

Key Features

- Open Source
- Continuous Inspection
- Multidimensional Analysis
- Actionable Reporting
- Centralized Portfolio Management
- Rule-Based Defect Identification
- Recent Quality Issues Monitoring
- Customizable Dashboards
- Developer Perspective
- Technical Debt Evaluation
- Application Lifecycle Management
- Multi-Technology Support
- Teamwork and Collaboration
- Extensibility
- Security

Benefits

Process Benefits:

- Decrease Risk
- Increase Sustainability
- Improve Productivity
- Raise Quality

Product Benefits:

- Shorten Learning Curve
- Increase Developer Skills
- Benefit from Bottom-Up Adoption
- Scale with Business Needs

- Bring Technical Debt under Control
- Enable Continuous Code Quality Management
- Define and Implement Requirements Efficiently

Ecosystem Benefits:

- Foster Innovation
- Plan with Confidence
- Reduce Risk with Vendor Support and Services

QE Technique	Applicable Technology	Applicable SDLC Phase
Quality Gate	Java, Free plugins available C#, Flex, Groovy, PHP, Web, JavaScript, Python, XML	Development/ Build/ Testing Automation

SMARTEST Methodology

What is SmarTest Methodology ?

- Products which evolve over years, face the challenge of exponentially increasing regression testing cycles due to the steep increase in test-suites for new content , dependency in test execution sequence and sunk-effort due to non-removal of redundant content. Cognizant’s solution to the above problem is the proven “SMARTEST” methodology which leverages

- Industry-standard techniques
  - ✓ OAT
  - ✓ MC/DC
  - ✓ Boundary Value Analysis
  - ✓ Equivalence Partitioning
- Multi-level functional analysis
- Automation Analysis
- Optimal Test Planning

Key Features

- Baseline Test Case Repository
- Prioritize the features
- 2-phased approach for redundancy removal
- Rationalize Test Cases
- Effective automation strategy
- Execution optimization

Benefits

- 25% reduction in regression
- Increased coverage with optimized test suites
- Effective methodology for test management
- Better baseline for planning and decision making for future product releases
- Scalable, generic methodology extensible to other areas and products

QE Technique	Applicable Technology	Applicable SDLC Phase
Model Based Testing	Java / Dot Net / Mainframe / SAP	Requirement / Test Design

What is FitNesse ?

- FitNesse is an open source collaboration test framework. It runs on a dedicated wiki server that can be accessed web browser by developers, testers and customers to create automated test cases integrated with narrative requirements usually for acceptance criteria testing.

Key Features

- Light weight, open source framework
- Dedicated wiki web server
- It provides a simple way to run tests (Fit tables) and suits.
- It Supports sub Wikis for managing multiple projects
- allows you to validate those requirements with the actual software implementation

Benefits

- Shorter learning curve
- No configuration set-up required
- Ease of maintenance
- More coordination and communication between developers , testers and customers
- Superb documentation

QE Technique	Applicable Technology	Applicable SDLC Phase
Collaborative Test Design	Java, .Net, Ruby, Python, C, and PHP	Testing Design

What is JBehave?

- JBehave is a framework for Behavior-Driven Development (BDD)
- It shifts the vocabulary from being test-based to behavior-based, and positions itself as a design philosophy

Key Features

- Pure Java implementation, which plays well with Java-based enterprises or when interfacing to any environment that exposes a Java API.
- Users can specify and run text-based user stories, which allows "out-in" development.
- User stories can be written in JBehave syntax or Gherkin syntax.
- User stories can be specified as classpath resources or external URL-based resources.
- User stories can be executed concurrently, specifying the number of concurrent threads.
- User stories can be documented via generic user-defined meta information that allows easy story filtering and organization into story maps.
- Annotation-based binding of textual steps to Java methods, with auto-conversion of string arguments to any parameter type (including generic types) via custom parameter converters.
- Annotation-based configuration and Steps class specifications
- Dependency Injection support allowing both configuration and Steps instances composed via your favorite container (Guice, PicoContainer, Spring, Weld).
- Groovy scripting supported for writing configuration and Steps instances
- Extensible story reporting: outputs stories executed in different human-readable file-based formats (HTML, TXT, XML). Fully style-able view.
- Story cross reference report format in JSON and XML, consumable by external applications.
- Auto-generation of pending steps so the build is not broken by a missing step, but has option to configure breaking build for pending steps.
- Pluggable step prioritizing strategy. Strategies bundled in core include: by priority field and by Levenshtein Distance.
- Localization of user stories, allowing them to be written in any language.
- IDE integration: stories can be run as JUnit tests or other annotation-based unit test frameworks, providing easy integration with your favorite IDE.
- Ant integration: allows stories to be run via Ant task
- Maven integration: allows stories to be run via Maven plugin at given build phase

QE Technique	Applicable Technology	Applicable SDLC Phase
Behavior-Driven Development	Java	Development/Test Design/Testing-Automation

What is Hudson?

- Hudson is powerful and widely used web-based Open Source “Continuous Integration” server sourced by Eclipse Foundation
- Provides development teams with a reliable way to monitor changes in source control and trigger a variety of builds
- Integrates easily with most version control systems and bug databases

Key Features

- Easy installation
- Easy configuration
- Cross-platform tool
- Web based interface
- Distributed builds
- Unit test reporting
- File fingerprinting
- Build Status Notification
- Extendable with numerous plugins
- Supports SCM tools like CVS, Subversion, Git and Clearcase

Benefits

Extensibility

- Supports software releases, documentation, monitoring
- Extend functionalities with over 250+ **plugins** available
- Can be combined with Apache Maven, Apache Ant or other Build Automation Tools

Open Source Software

- Released under MIT License
- Comprehensive manuals and detailed documentation
- Active plugin development community

Other Benefits

- Easily traceable project relationship
- Generate Test Reports & trends
- Interactive Project & Build Dashboard
- Faster notification to stakeholders via email, SMS, IRC and Skype
- Faster creation & configuration of jobs
- Scalable for other languages

QE Technique	Applicable Technology	Applicable SDLC Phase
Continuous Execution	Java	Build/Testing Automation

Jenkins

What is Jenkins ?

Jenkins is an award-winning application that monitors executions of repeated jobs, such as building a software project or jobs run by cron. Among those things, current Jenkins focuses on the following two jobs:

- Building/testing software projects continuously
- Monitoring executions of externally-run jobs.

Key Features

- Easy installation
- Easy configuration
- Change set support
- Permanent links:
- RSS/E-mail/IM Integration
- After-the-fact tagging
- JUnit/TestNG test reporting
- Distributed builds
- File fingerprinting
- Plugin Support

Benefits

Flexibility

- Highly Configurable System
- Various plugins have been developed by additional Communities
- Can be combined with Ant , Gradle , or other Build Automation Tools

Free/OSS

- Released under MIT License
- Large support community and thorugh documentation
- Easy to write plugins
- Bugs found can be fixed by any end user

Other Benefits

- Generate Test Reports
- Integrate with many different Version Control Systems
- Push to various artifact repositories
- Deploys directly to production or test environments
- Notify stakeholders of build status`

QE Technique	Applicable Technology	Applicable SDLC Phase
Continuous Execution	Java	Build/Testing Automation

Proofer

What is Proofer ?

- Proofer is Cognizant (IP) tool used for static testing in requirement phase of SDLC.
- It is used to identify issues related to clarity and ambiguity in requirement documents.

Features

- Checks for weak and incomplete phrase in the document.
- Easy to use - User friendly GUI with filters
- Dictionary can be customized to align with client industry/domain specific terminology` .
- Detailed report is generated immediately which is easy to interpret.
- Supports documents in .doc, .docx,.rtf

Requirement Register

What is Requirement Register ?

- Requirement Register is an Excel Checklist which performs a 5 point check on every requirement at the start of SDLC phase.

Features

- This could be used to check if all requirements satisfy the following attributes – Singular, Unambiguous , Measurable, Complete and Testable
- Parameters like cohesiveness, completeness and feasibility could be implemented on a case to case basis as required.
- Requirement Register helps to bring out the overall priority to the existing requirements by:
  - ✓ Displaying the requirement status as "Pass/ Fail" which in turn helps to analyze the quality of requirement
  - ✓ Calculating the Risk Score (Impact * Probability) for each of the requirements

QE Technique	Applicable Technology	Applicable SDLC Phase
Static Testing	Java / Dot Net / Mainframe / SAP	Requirement

Thank You