
Test Automation

Keys for *Smart Starts*

Bob Galen

RGalen Consulting Group, LLC

www.rgalen.com bob@rgalen.com

Introduction

Outline

- Based on 3 part series in Software Test & Performance magazine, Nov 2006 – Jan 2007
- Start by pulling together a thoughtful and relevant **Business Case**
- Develop a view to your **Automation-SDLC** (software development life-cycle) and connect it to your Product SDLC
- Finally, establish **Selection Criteria** for how you pick good automation candidates from your test cases.

Developing Your Business Case

Climate Assessment

- Application Characteristics
 - Domain, complexity, lab (hardware, software, data) requirements

- Team Capabilities
 - Ratio or capacity, skill-sets, current projects, leadership

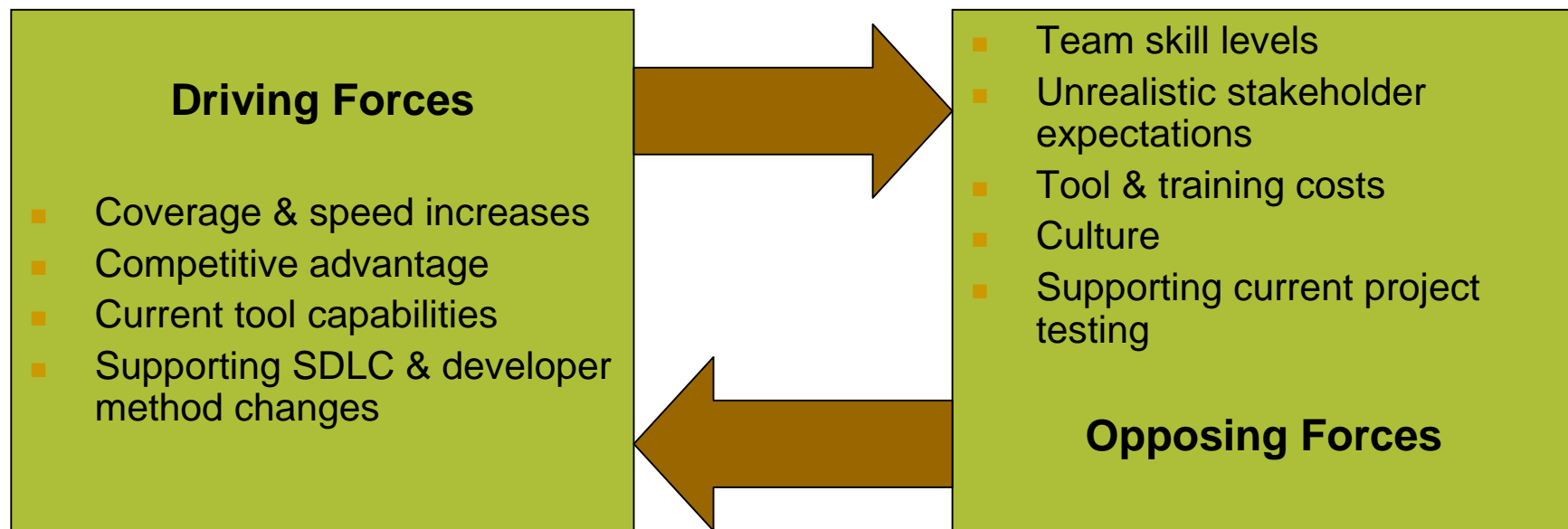
- Budget Constraints
 - Tools, training, lab, ongoing maintenance, return rates

- Leadership Understanding & Expectations
 - Automation lifecycle, techniques, trade-offs, long term investments

Developing Your Business Case



Force Field Analysis

1. Given a clear GOAL for automation
2. List driving forces – specific driving rational for automation
3. List opposing forces – impediments against automation implementation
4. Strategy – leverages driving forces to overcome resistance from opposing forces to implement your automation program



Developing Your Business Case

Force Field Analysis

Automation Assessment — Force Field Analysis	
Forces Supporting Test Automation Development 	Forces Opposing Test Automation Development 
<ul style="list-style-type: none"> • Cycle time reduction • Long term reduced costs • Increased investment in team performance 	<ul style="list-style-type: none"> • Overall team skill set • Overall management “understanding” – leading to short time expectations • Costs beyond the tools
Strategies for Amplifying Support	Strategies for Overcoming Resistance
<ul style="list-style-type: none"> • Running a “pilot” to illustrate applicability of approaches and speed potential 	<ul style="list-style-type: none"> • Training executive team members in aspects of automation

Developing Your Business Case

Developing Your Strategy

- Use FFA to craft an overall view to your landscape and to create realistic goals & plans
- Define your strategy iteratively – connecting towards multiple, focused goals
- Deliver automation in iterations; show progress often, connected to value
 - Business & Development
- Get help!
 - Training, consulting, hire / develop expertise; don't underestimate the effort
- Pilot your efforts; infrastructure first, but not too much!

Developing Your Business Case

Hard ROI

- Improved Test Coverage
 - Increased coverage levels – risk mitigation and compliance support

- Reduced Testing Execution Time
 - Faster testing cycles; more iterations & faster feedback

- Decrease in Product Test Escapes
 - Embarrassment and customer impact costs

- Improved Test Repeatability
 - Elimination of human error

Developing Your Business Case

“Soft” ROI

- Raw Incremental Speed
 - Product deployment competitive advantages

- Developer Confidence
 - Embracing necessary customer changes & late delivery of features

- Continuous Investment
 - Savings reinvestment into the SQA team capabilities – leading towards continuous improvement

Developing Your Business Case Checklist

<input checked="" type="checkbox"/> Driving forces for automation
<input checked="" type="checkbox"/> Key Challenges opposing automation
<input checked="" type="checkbox"/> Exhaustive automation costs
<input checked="" type="checkbox"/> Primary Hard ROI factors & measurable goals
<input checked="" type="checkbox"/> Primary Soft ROI factors and goals
<input checked="" type="checkbox"/> Implementation plan with a 3-12 month view – including frequent milestones
<input checked="" type="checkbox"/> Establish organization (prior) and (future) states to illustrate overall direction and level of challenge
<input checked="" type="checkbox"/> Develop and overall architectural strategy (iterative) with early pilot / proof of concept phases



Automation-SDLC

Success Criteria

- **Architecture** – develop a goal, select tools, decide on your model and map it to your product & domain needs. Models might include –
 - Simple tool driven record & playback
 - Data, Keyword, or Action-word driven
 - Structured, modular, or programmatic engine based

- **Requirements** – establish them with cross-functional feedback. Make them visible.

- **Implementation Strategy** – consider it your product delivery roadmap – phasing, early pilots, managing risk, incremental delivery, and mapping towards business value.

- **Project Synchronization** – prime directive is to support project delivery!

Automation-SDLC

Extensions to your Product-SDLC

- Given that Automation is a development project,

- I like to extend the Development SDLC forward towards my Automation-SDLC. It fosters –
 - Consistent terminology
 - Improved understanding
 - Collaboration between development and testing

- There are 7 key extensions that I recommend

Automation-SDLC

Extensions to your Product-SDLC

1. Make Automation Decisions from completed Manual Test Cases
 - ❖ Design complete, debugged execution, and documented – virtually “done”.
2. Review from an Automation Perspective
 - ❖ Early on, when you review product requirements – do it from the perspective of automation & testability.
3. Establish a Highly Iterative Automation Model
 - ❖ Deliver in small, goal-driven, time-boxed increments; continuously adding your automated coverage.

Automation-SDLC

Extensions to your Product-SDLC

4. Establish Clear Hand-offs for Infrastructure, Automation, and Execution
 - ❖ Entry criteria for quality, documentation, training and verification.

5. Extend Traditional Tools & Processes for Automation Support
 - ❖ Leverage your development tools & processes as much as possible; particularly CM, Defect Management, standards, and PM.

Automation-SDLC

Extensions to your Product-SDLC

6. Look to Analyze, Increase & Communicate Automation Run-time Efficiency

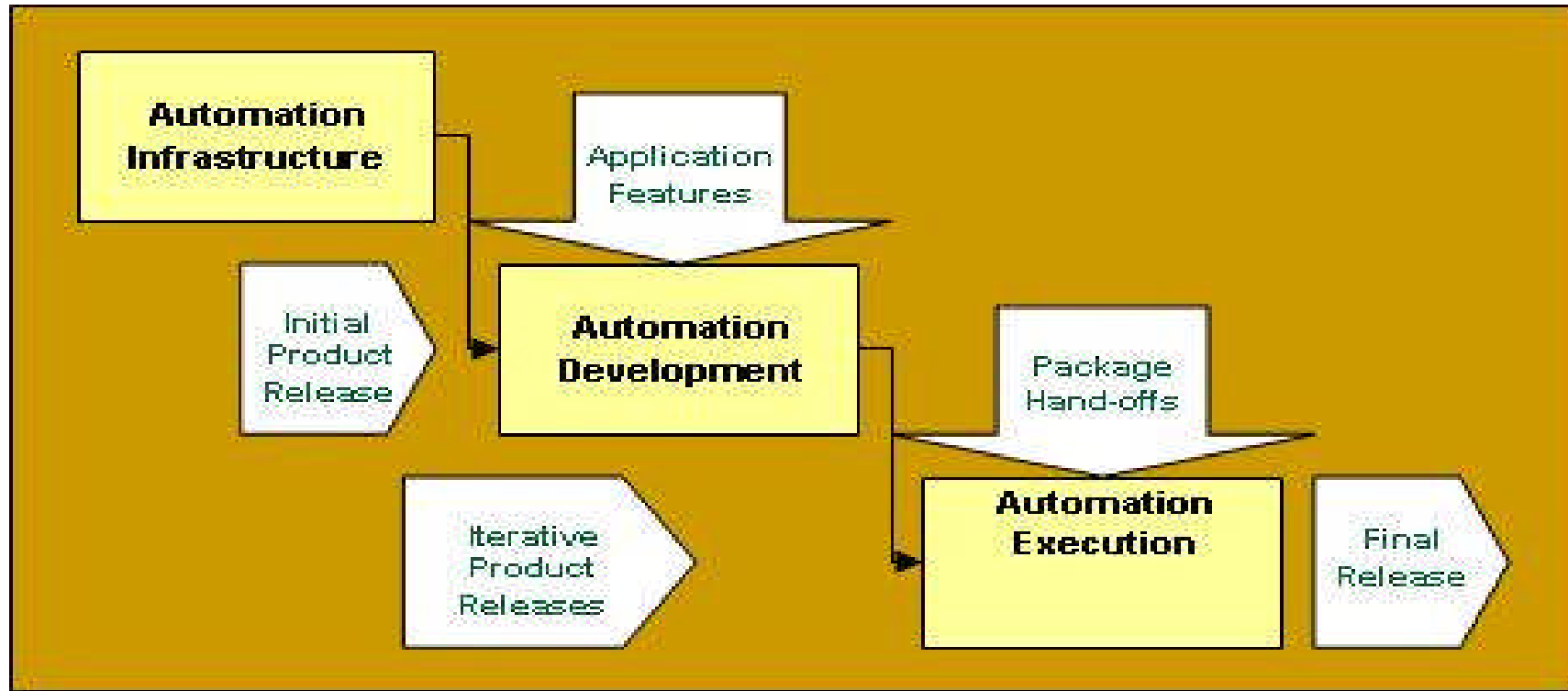
- ❖ PR your automation execution improvements; but also have goals for increasing speed and efficiency.

7. Properly “Connect” Automation to Your Project Phasing

- ❖ Create integration milestones, clearly articulate dependencies and interrelated risks, and stay off the project critical path!

Automation-SDLC

Typical 3-Tiered Development Model



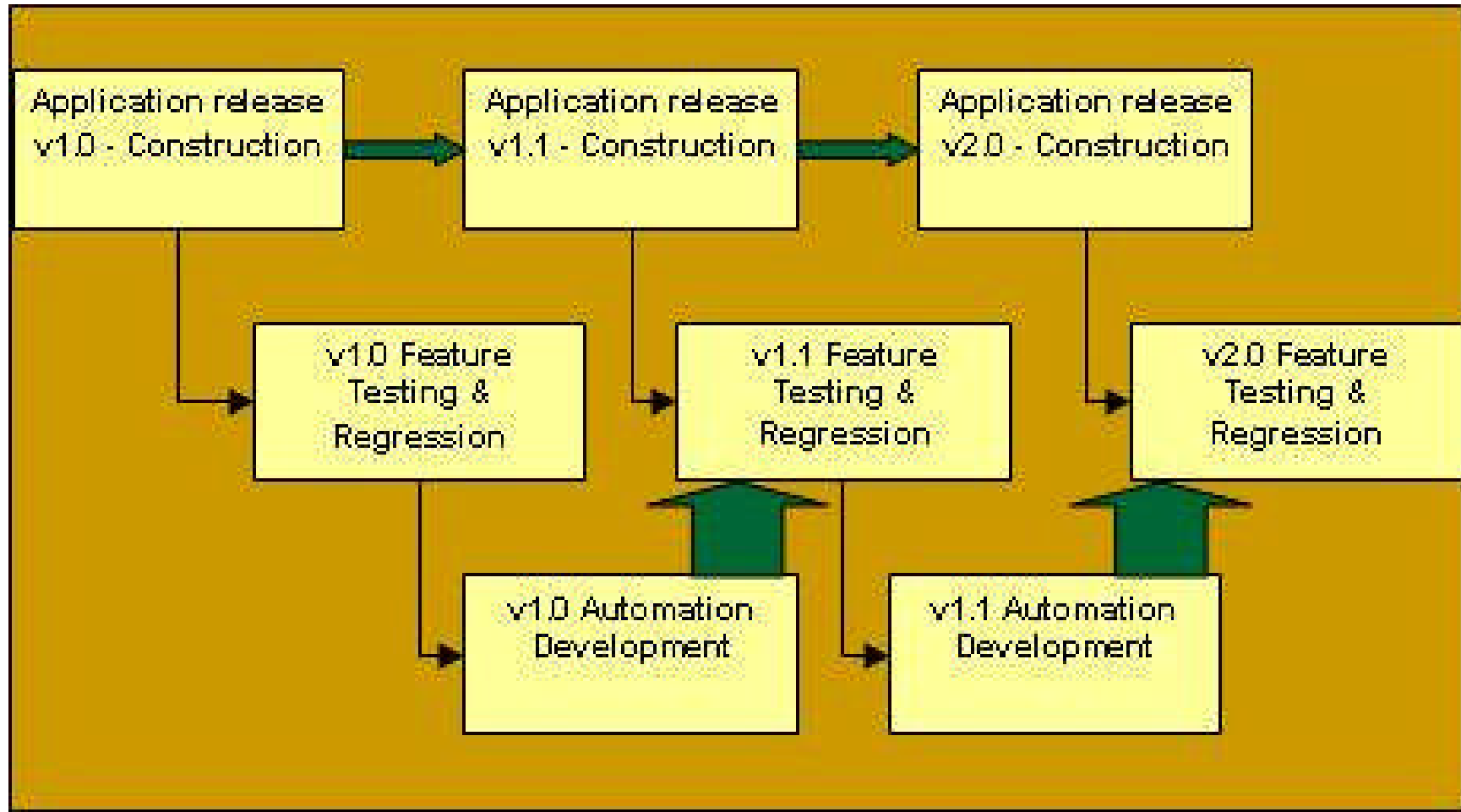
Infrastructure - (Tools & processes)

Automation - (Test case automation)

Execution - (Environment, scheduling, results analysis & reporting)

Automation-SDLC

Automation vs. Project Work Phasing





Test Case Selection

Where to Start?

- This section targeted towards group that have a large base of manual test cases that they want to automate
- Beyond architecture & tools, there's the simple question of – where do we begin?
- Establishing some notion of test case selection criteria thoughtfully addresses this question...

Test Case Selection

5 Selection Anti-Patterns

1. We Don't Need No Stinkin' Criteria
 - ❖ The “just do it” mentality
2. A Good Start – But What's Next
 - ❖ Prioritizing a small list, but then a free for all
3. By George, Let's Do It All
 - ❖ A very large list, pick your poison
4. In Tools We Trust
 - ❖ Not worrying about it at all – the tool will handle it
5. Make A List, Then Stick To It!
 - ❖ Picking your candidates, but never revisiting based on project changes

Test Case Selection

8 Selection Patterns

1. Low Hanging Fruit

- ❖ The early pilots, feasibility, learning curve. Making early, very visible progress.

2. Fleshing Out the Infrastructure

- ❖ Architecture needs to be tested and qualified. Checking feasibility and tools integration.

3. Minimizing Rework

- ❖ Waiting for features or functionality to stabilize. Considering the brittleness of the area your automating.

Test Case Selection

8 Selection Patterns

4. Driving with Value

- ❖ At the lowest level, time to automate vs. life expectancy, priority coverage and business need.

5. Planning – Consider Complexity & Packaging Tempo

- ❖ Think in terms of what you take on in complexity and release handling on the part of your execution team. Don't take on too much!

6. Project Impact – Consider Speed

- ❖ Speed of pulling the automation together – time-boxing; speed impact it will have on your execution.

Test Case Selection

8 Selection Patterns

7. Project Impact – Consider Risk

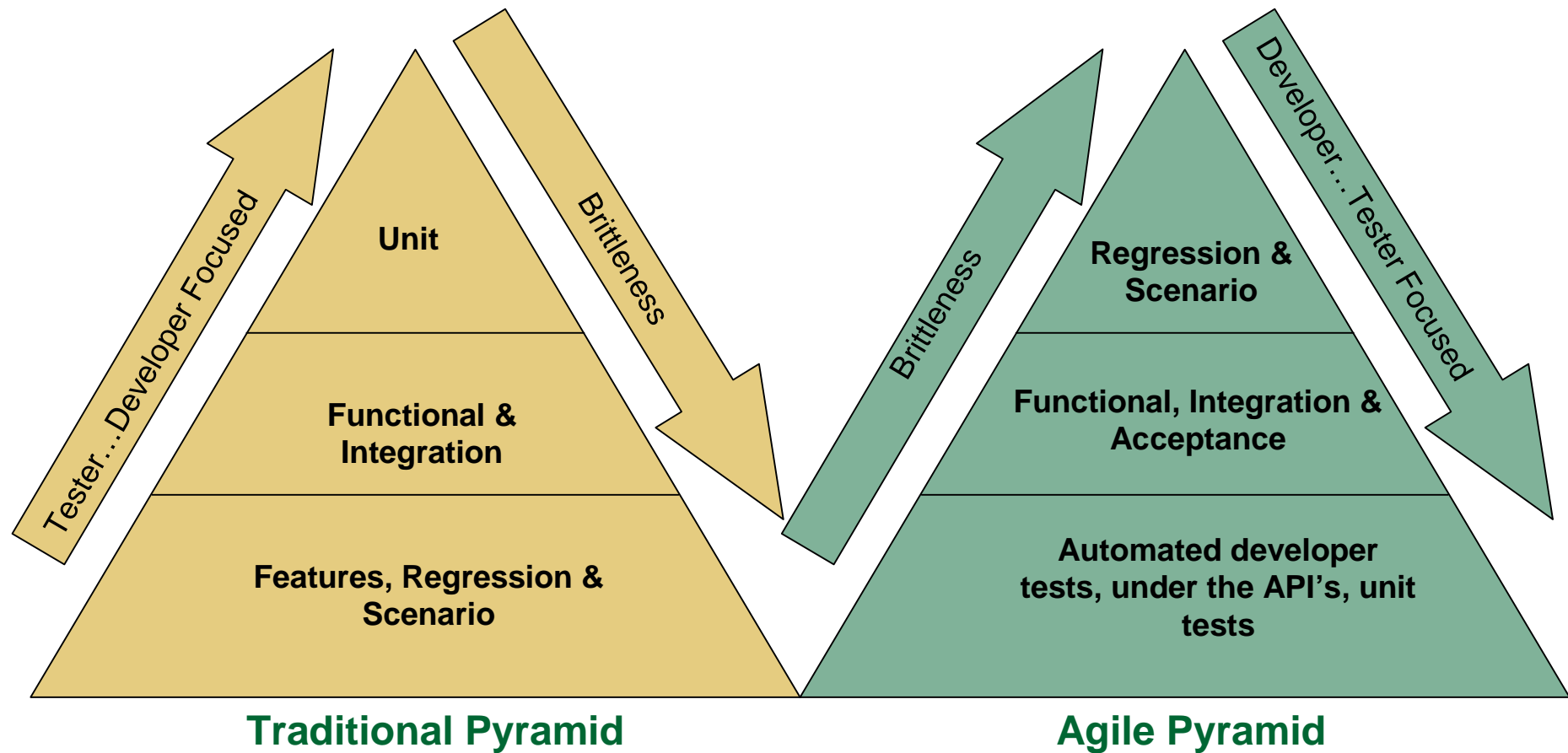
- ❖ Early on focused on automation setup risks. Later on, target application risk – volatility, change, interfaces, etc.

8. Project Impact – Consider Maintenance

- ❖ Before writing automation – consider it's life expectancy, application / business factor volatility that will influence change in the underlying feature.

Test Case Selection

Testing “Food” Pyramids



Test Case Selection

Changing Your Focus

1. Changes in Team Skill Set
2. Changes in Application Technology
3. Changes in Team Balance
4. Ongoing Setup Time
5. Evolution in Maintenance Costs
6. Evolution in Development Methods
7. Changing Business Conditions
8. Retirement Considerations

Test Case Selection

Weighting

Selection Factor	Considerations	Weight for New / Mature Apps
Project Impact	Will this automation directly help your projects speed, coverage, risk, and/or development nimbleness?	L / H
Complexity	Is the automation effort simple to implement, including data and other environmental challenges?	H / H
Level of Effort	How much time is required to implement the automation, including time spent on packaging with other work?	L / H
Early Requirement & Code Stability	Will the application requirements or early coding prove stable within the scope of the driving project?	L / M
Ongoing Maintenance	How volatile will this code be in the longer term, will the functionality change/evolve? How about the toolset?	L / M
Coverage	How broad will be the coverage impact in features and functionality exercised, of this automation? Will it cover critical features?	H / M
Resources	Does your team have the human, physical and data resource required to run the tests?	H / H
Hand-off Efforts	Do test execution resources have sufficient skill and time to run the automation?	L / H

Wrap-up

The theme is really to focus on the Start!

- Establish your business case and ensure your stakeholders realistically understand the challenges and benefits
- Spend time establishing a technology foundation – your architecture, A-SDLC, methods, skill-sets, etc.
- Finally, bite off small pieces and establish a release tempo that addresses different criteria & goals for each iteration

While always connecting your efforts to business value and making your progress visible

Questions?

Thank you!

References

Software Test & Performance - www.stpmag.com

■ Business Case

- Making the Case: Developing your Defense for Test Automation, January 2007, pp 24-29
- Link – www.stpmag.com/issues/stp-2007-01.pdf

■ Automation SDLC

- How to Build Tests and Apps in Tandem, November 2006, pp 14-19
- Link – www.stpmag.com/issues/stp-2006-11.pdf

■ Selection Criteria

- Sizing Up Automation Candidates, December 2006, pp 20-25
- Link - www.stpmag.com/issues/stp-2006-12.pdf

Elements of an Automation Architecture

<p>Team Elements</p>	<ul style="list-style-type: none"> ✓ Charter your team with clear architectural and leadership roles ✓ Assess & establish a base level of programming / development skills ✓ Develop training materials, on-line guidance, templates and checklists ✓ Establish mentoring or pair-testing relationships ✓ Obtain training for writing, running and framework automation development and maintenance activities
<p>Process Elements</p>	<ul style="list-style-type: none"> ✓ Create coding conventions, templates, and design patterns ✓ Adhere to naming conventions ✓ Use proper test case size guidance for decomposition, granularity & estimation ✓ Establish review, promotion, and removal models for test cases within the regression framework
<p>Tool Elements</p>	<ul style="list-style-type: none"> ✓ Map tool selection criteria to your applied automated testing model ✓ Establish standards for your Configuration Management and Defect Tracking tools ✓ Define <i>wrapping</i> conventions that target how you will insulate dependencies and workarounds
<p>Architectural Elements</p>	<ul style="list-style-type: none"> ✓ Decompose your AUT into meaningful layers or components for test case development and efficient regression testing ✓ Establish a versioning model for controlling “packages” of tests for new feature and maintenance updates ✓ Wherever possible, defining libraries of re-usable, well tested, automation components ✓ Develop standards for logging errors (files, formats, review practices) and interfaces to reporting systems (logging, web reporting, defects) ✓ Gather lab or H/W interface information for test environment interrogation, scheduling and control ✓ Use traditional test management – scheduling, results interrogation, defect reporting, metrics

Contact Info

Robert Galen
RGalen Consulting Group,
L.L.C.
PO Box 865, Cary, NC
27512
919-272-0719
www.rgalen.com
bob@rgalen.com

*Software Endgames: Eliminating Defects,
Controlling Change, and the Countdown to
On-Time Delivery* published by Dorset House
in Spring 2005. www.rgalen.com for order
info, misc. related presentations, and papers.

