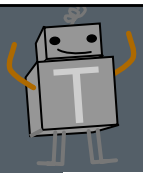


Software Test & Performance CONFERENCE

1202

LogiGear®

The 5% Challenges of Test Automation

STP Conference 2007

Hans Buwalda

LogiGear Corporation

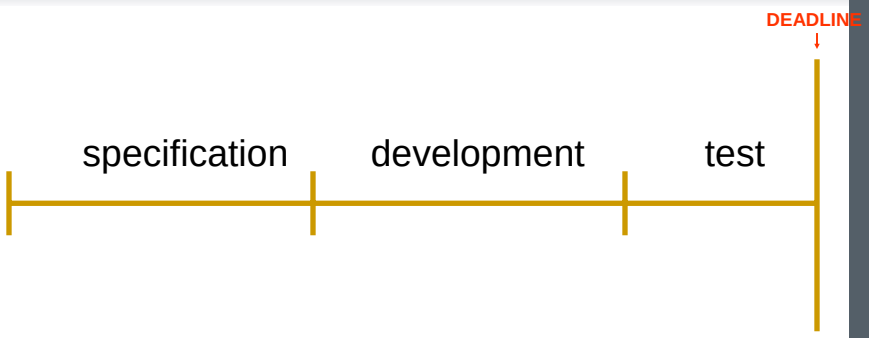
hans @ logigear.com

LogiGear®

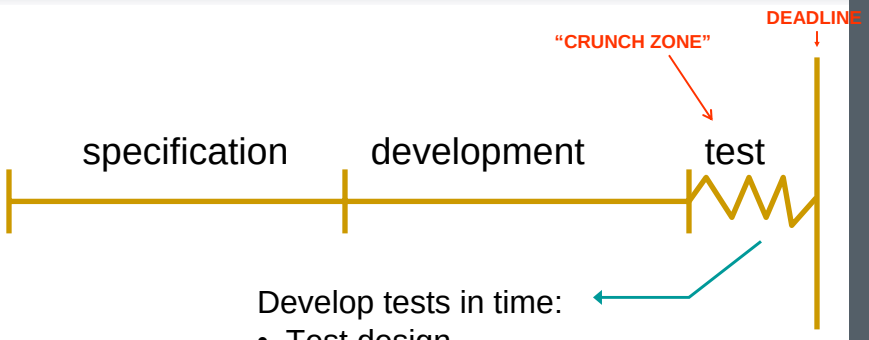
Scope of this Workshop

- Raising the bar for automation
  - the 5% challenges and their rationale
- Show techniques, concepts, ideas to achieve the higher bar
  - how could you achieve the challenges
- How feasible is all of this in practice . . .

Testing Under Pressure



Testing Under Pressure



Develop tests in time:

- Test design
- Auditing, acceptance
- Preparations
- Automation

The 5% Challenges

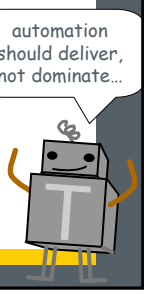
- No more than 5% of all tests should be executed manually
- No more than 5% of all efforts around testing should involve automating the tests

Why 95%+ Automation ??

- Credible pay-off for automation efforts
- Automation will find “bonus bugs”
  - on average 15% of fixes cause new bugs
  - many of these bugs are hard to find without integral testing
    - often a result of violating overall architectures
    - the bugs occur because data is left in an inconsistent state
- The automation should be the core of the testing life-cycle
  - otherwise it looses ground quickly over time
  - relieve the “crunch zone”
  - apart from “exploratory testing”

Why < 5% Efforts ??

- Automation should not dominate testing
  - it is not a goal in itself
  - may never be a bottleneck
- Testers should be able to focus on testing
  - better tests (higher ambition level)
  - communication with stake holders
- High level of effort will aggravate the "crunch zone"
  - "invitation to Murphy's law"



Approaches to Automation

1. Record and Playback (or "Capture and Replay")
2. Scripting / Programming
3. Action Based Testing (action words, keywords)

it is my belief that an approach based on keywords is a prerequisite to achieve a high level of stable automation

(simplified, see also: "Software Test Automation", Mark Fewster and Dorothy Graham)

1. Record and Playback

```
select window "Logon"
enter text "username", "administrator"
enter text "password", "testonly"
push button "Ok"
select window "Main"
push button "New Customer"
expect window "Customer Information"
select field "First Name"
type "Paul"
select field "Last Name"
type "Jones"
select field "Address"
type "54321 Space Drive"
   .
   .
   .
```


2. Scripting / Programming

```
. . .
Function EnterCustomer(FirstName, LastName, Address)
        Click("New Customer");
        ExpectWindow("New Customer");
        EnterField("First Name", FirstName);
        EnterField("Last Name", LastName);
        EnterField("Address", Address);
        . . .
        Click("OK");
End Function

. . .

Function Main()
        Logon();
        EnterCustomer("Mary", "Jones", "123 Palm Drive");
        EnterCustomer("Paul", "Franklin", "321 Regent Street");
        . . .
        LogOff();
End Function
. . .
```

3. Action Based Testing

- Provides a framework for effective automation
- Based on a notion that a test can be broken down in a number of consecutive actions (keywords)
- Both actions and their data are in products, called "Test Modules"
  - Excel spreadsheets for easy development and communication
  - test data is explicit or with a place holder
  - explicit checks with specified expected result values
  - most actions are "high level", omitting as many unneeded details as possible
- Instead of implementing test cases, the automation concentrates on the programming individual actions

Separation of Tests and Automation

	number	name	quantity
new product	P-9009	Sledge Hammer	5
	number	quantity	
add quantity	P-9009	20	
add quantity	P-9009	3	
add quantity	P-9009	6	
	number	quantity	
check quantity	P-9009	34	

Test Module
(Excel Spreadsheet)

Actions

…
add quantity
check quantity
new product
…

Automation →

tools, scripts, etc

Example of "Old and New" Formats

Most values are implicit. The tester has to figure them out during execution….

Same instruction is repeated over and over again…

Current format

Enter a user id that is greater than 10 characters, enter proper information for all other fields, and click on the "Continue" button	An error message should be displayed stating that "User Id must be less than 10 characters".
Enter a User Id with special character's), enter proper information for all other fields and click on the "Continue" button	An error message should be displayed indicating that "User Id cannot contain some special characters".
Enter the information, with a password of 4 characters and click on the "Continue" button	An error message should be displayed with "Password must contain at least 5 characters".

ABT format

check registration dialog	user id aaaaabbbbbc	message User Id must be less than 10 characters
check registration dialog	user id résoudre	message User Id cannot contain some special characters
check registration dialog	password test	message Password must contain at least 5 characters

Achieving the Challenges...

Three major success factors:

1. Test design
2. Automation architecture
3. Organization

LogiGear

Success Factor 1: Test Design

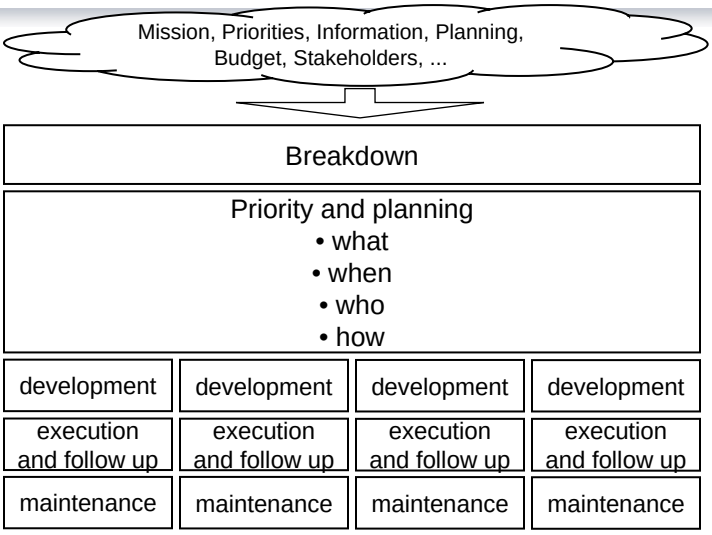
LogiGear

Test Design

- Effective test breakdown (into test modules)
  - make sure every test has a clear focus
  - keep different kinds and levels of tests separate
- Right level of actions
  - as “high level” if possible, hiding as many details as much as possible
  - but not if the details are relevant for the test

It is my believe that test design, not automation or “the tool”, is the deciding factor in achieving the 5% challenges

Breakdown: Divide and Conquer



Breakdown "Cheat Sheet"

- Straightforward Criteria
  - Architecture of the system under test (which part are we testing)
  - Functionality and other requirements
  - Kind of test (navigation flow, quality attributes, ...)
  - Ambition level (smoke test, regression, requirement based, aggressive, …)
- Additional Criteria
  - Stakeholders
  - Complexity of the test
  - Technical aspects of execution
  - Planning and control
  - Risks involved

Splitting into Test Modules

- UI oriented tests
  - does function key F4 work
  - does listbox xyz the right values
  - is the tab order correct
- Do individual functions work
  - like transactions in a financial system
- Alternate paths in use cases
  - like cancel a transaction
- End-to-end tests
- Simulating business processes
- Tests with specific automation needs
  - like multi station tests
- Tests of non-UI functions
- High ambition level tests (aggressive tests)

Levels of Actions

- High level actions to hide details
  - to hide unneeded details
    - always hunt down details
  - implemented in terms of lower level actions
- Low level actions to access the automation technology
  - either built-in in a tool like TestArchitect
  - or create yourself in a “harness”
    - this is the only real programming you would do

Example of “Going High Level”

	window	tree	tree item path
click tree item	global	main tree	/my projects/Main
	window	max	
wait for window	view	10	
	window	list	item
check item exists	view	item list	Master Account

	project	item
check project item	Main	Master Account

Example of “Going High Level”

section	**Get cost values**			
	key			
key navigate	F7			
key navigate	5			
	window			
wait for controls loaded	view account info			
	text			
check breadcrumb	# view account info breadcrumb			
	window	field		
check focus	view account info	cancel		
	start cost	cost in	cost out	net cost
capture cost values	>> starting cost	>> total cost in	>> total cost out	>> total net cost

Example of “Going High Level”

section	Get status			
	window	menu	item	
click select menu	general look	menu bar	system	
	keys			
type	{S}			
	window			
check window exists	status information			
	cost no	logon	dep level	cost min limit
capture status information	>> dep num	>> logon_id	>> account level	>> cost min
	cost max limit	shop	region	branch
>>>	>> cost max	>> shop	>> region	>> branch
	source	control		
click	status information	ok		

Example of “Going High Level”

section	Capture initial number		
	key		
key navigate	F7		
key navigate	3		
	page tab		
locate page tab	Scan Criteria		
	window		
wait for controls loaded	search		
	text		
check breadcrumb	# search breadcrumb		
	window	control	value
select	search	scan direction	Backward
	window	control	value
enter value	search	business date match	# bus date
	source	control	
click	search	go	
	window		
wait for controls loaded	search results		
	store as		
capture sequence num	>> seq num		

Example of “Going High Level”

	start cost	cost in	cost out	net cost
get cost values	>> starting cost	>> total cost in	>> total cost out	>> total net cost
	cost no	logon	dep level	
get status info	>> dep num	>> logon_id	>> account level	
	shop	region	branch	
>>>	>> shop	>> region	>> branch	
	cost min limit	cost max limit		
>>>	>> cost min	>> cost max		
	number			
get seq number	>> seq num			

Success Factor 2: Automation Architecture

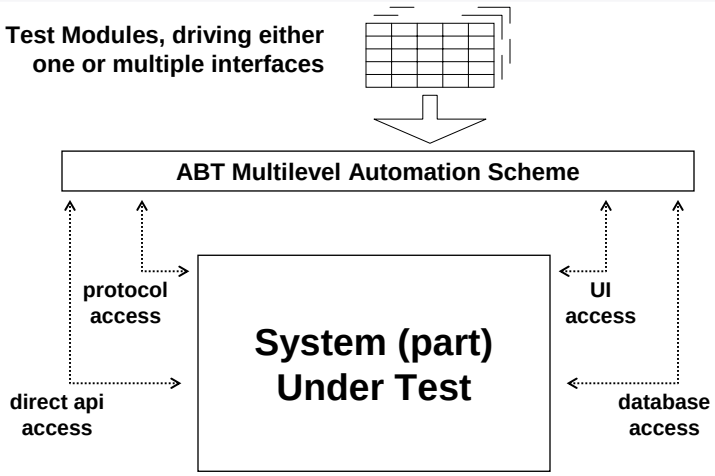
Automation Architecture

- Pay attention to where and how to implement actions
  - using the “action definitions” (or something similar) for straightforward actions
  - use programming for
    - complex actions (typically involving loops)
    - wrap specific technical code to access the interface
- Select the right tools and technologies
  - some of the considerations are subtle
  - technical choices strongly influence the ease of the automation
  - make sure the technology is organized as durable/re-usable as possible
  - Go for a “mix” of technologies (UI, API’s, SQL, TCP/IP, ...)
- Manage the handling of the interfaces (UI, API, ...)
  - make sure all dialogs, pages, controls etc get meaningful “logical” names (mapped to the actual items in the UI etc)

Select Tools and Technologies

- Technical choices strongly influence the ease of the automation
  - some of the considerations are subtle, invest some thinking
  - often the technology is not the deciding factor, but the way you package it is
- Make sure the technology is organized as durable/re-usable as possible
- Go for a “mix” of technologies (UI, API’s, SQL, TCP/IP, ...)

Multiple System Access



Some Tips to Get Stable Automation

- Make the system under test automation friendly
  - developers are not always motivated to do that, but it pays off
  - in particular ask to add specific property values to the GUI interface controls for automated identification
    - like “accessible name” in .Net and Java, or “id” in Web controls
- Pay attention to timing matters
  - in particular use “active timing”, based on the system under test, not fixed amounts of “sleep”
- Test your automation
  - develop a separate test set to verify that the actions work
  - make separate people responsible for the automation
- Use automation to identify differences between versions of the system under test

LogiGear

Success Factor 3: Organization

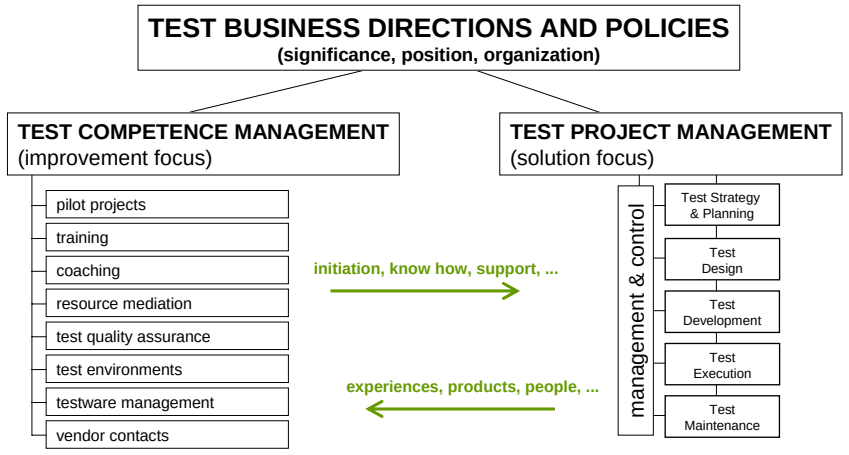
LogiGear

Organization

- Much of the 5% success is also gained or lost in how you organize the process
- Write a plan of approach:
  - scope, assumptions, risks, planning
  - methods, best practices
  - tools, technologies, architecture
  - stake holders, including roles and processes for input and approvals
  - team
  - . . .
- Assemble the right team
  - testers, lead testers
  - automation engineer(s)
  - managers, ambassadors, ...

see also:
"QA All-Stars: Building Your Dream Team", Hans Buwalda, "Better Software" September 2006

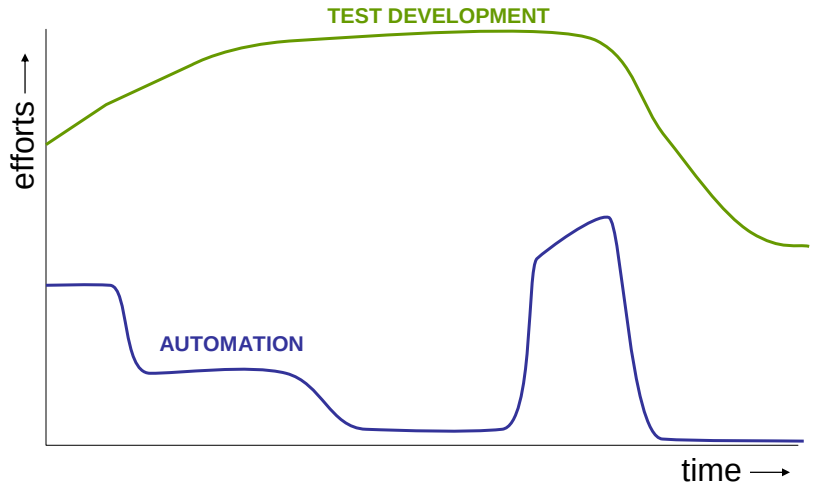
Test Governance



Are the 5% rules Achievable?

- To achieve to high automation coverage tends to be fairly straightforward
  - once the actions work all tests are automated
  - don't settle for too little
- The automation efforts need to be well managed, but then can be kept low as well
  - need of automation specialization and intelligence in the team
  - a method like ABT can provide a proper framework, but is only a start
- The key to success is: think it through
  - don't hesitate to revisit your design and automation
  - make sure the tests are easy to run

Typical Time Allocation



Case Study

- Large IT provider
- New version of one of their major web-sites
- Test scope was user acceptance test (functional acceptance)
  - the users were the “business owners”
- Development was off-shore

Case Study

- Test development was done separate from automation
  - time-line for test development: May – Oct
  - time-line for automation (roughly): Jan – Feb
- All tests were reviewed and approved by the business owners
  - acceptance was finished by the end of the test development cycle

Case Study, Results

- All tests were developed and reviewed on schedule
  - many notes and questions during test development phase
- The automation was 100% of the tests
  - all actions were automated, thus automating all test modules
- The test development took an estimated 18 person months
  - one on-shore resource, two off-shore resources
- The automation took between one and two months
  - focused on actions
  - most time was spent in handling changes in the interface (layout of pages etc)

Summary

- Good to set aggressive targets for the automation:
  - it costs money and efforts
  - it should not get in the way of the testers making good tests
- Elements for success:
  - test design
  - automation architecture
  - organization
- Focus on the method first
  - the “tool” is only a minor element
  - use some form of keyword driven method like ABT
- The 5% targets can be achieved:
  - automation coverage can even reach 100%
  - to limit efforts (to 5% of all efforts) requires serious planning and thinking ahead