
Test Automation

Past, present, future

Ayal Zylberman



0-1970

Pre-Testing phase

- Dedicated testers were almost unknown
- Programmers tested software.
- Testing was often not distinguished from debugging. (James Bach)

70s

Formal Testing begins

- In the 1970's the dedicated testers did appear.
- In 1972, at Chapel Hill, the first conference on testing software was held, and the proceedings of that conference show that testing was beginning to be thought of as a discipline worthy of study apart from programming. (James Bach)

80s

Static Capture/Replay Tools (without scripting Language)

- In the mid-1980's automated testing tools emerged to automate the manual testing effort to improve the efficiency and quality of the target application
- These tools were initially fairly primitive and did not have advanced scripting language facilities
- With these early tools, tests were performed manually and the inputs and outputs were captured in the background
- The costs associated with maintaining such scripts were astronomical, and unacceptable
- These scripts were not reliable, even if the application had not changed, and often failed on replay (pop-up windows, messages, and other things can happen that did not happen when the test was recorded)
- More time was spent on debugging the scripts

(from: Software testing and continuous quality improvement - William E. Lewis, Gunasekaran Veerapillai)

90s

Variable Script-Based Testing

- The next generation of automated testing tools introduced added variable test data to be used in conjunction with the capture/replay features
- The role of a test automation expert and tester were distinguished
- Variable capture/replay and extended methodologies reduce the risk of not performing regression testing on existing features and improving the productivity of the testing process
- However, the problem with variable capture/replay tools is that they still require a scripting language that needs to be programmed
- More functional decomposition approach was used

(from: Software testing and continuous quality improvement - William E. Lewis, Gunasekaran Veerapillai)

OOs

Keyword Driven Testing

- This data-driven approach (since the late 1990's) uses the test case document developed by the tester (a spreadsheet containing table-driven "keywords") and functional decomposition script development (like structured software development) to reduce all test cases to their most fundamental tasks
- The following keyword controlled types of scripts are written: Item, business function scripts (Sequences) and Pre-defined scripts (utility functions) that perform these tasks independently of each other.
- Some examples of what these scripts do include:
 - Navigation (e.g., "Access Loan Screen from Main Menu")
 - Specific (Business) Function (e.g., "Post a Loan")
 - Data Verification (e.g., "Verify Loan Payment Updates Current Balance")
 - Return navigation (e.g., "Return to main Menu")
- The activity of test automation was reunited as a mutual tasks of the manual tester and the test automation expert

Automated Software Testing: Introduction, Management, and Performance - Elfriede Dustin Jeff Rashka, John Paul

Test Language

Test Language is a comprehensive test approach that hands over the responsibility of automation plan, design and execution to the functional testers by using KDT based solution

KDT Architecture

Keyword

- **Item** - an action that performs a specific operation on a given GUI component
- **Sequence** - a set of keywords that produces a business process
- **Utility Functions** - a script that executes a certain functional operation that is hard\ non-effective to implement as a Sequence

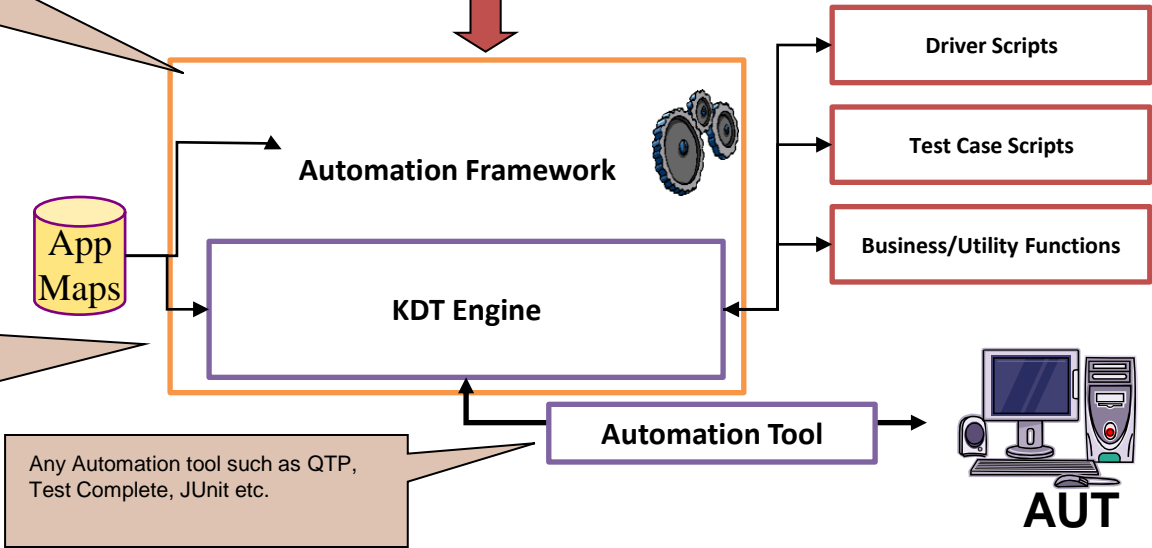
Automation Framework

Automation Framework that allows testers to design, maintain and execute automation scripts. Can be Tailored Excel sheets or comprehensive solutions such as Funtasy

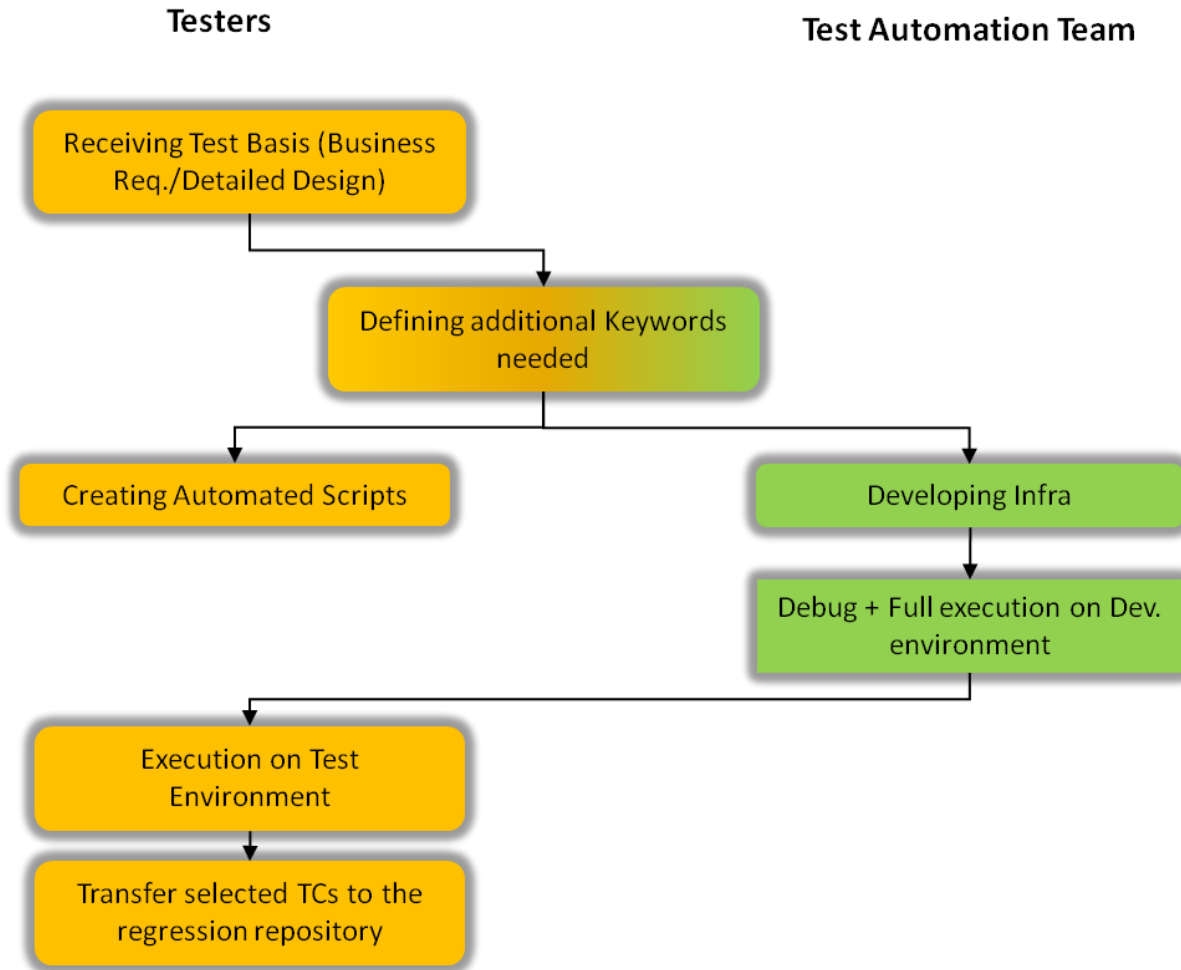
Type	Description	Action	Value	Data
Sequence	Log in to TAS	tas_login		
Sequence	Create new residential customer	tas_create_residential		
Sequence	Check account info	tas_check_residential	Success	
Function	Authorise the new account	tas_autorisatie		
Item	Show all res customers	Button	ShowReport	Click
Item	Make sure the new account is in the report	search_report	CustomersList	qualitest1
Item	Check status message	status_bar	state	Success
Item	Click OK and go back to main window	Button	OK	Click

KDT Engine

- Read the Keywords line by line
- Perform the required operation on the Application Under Test (AUT) using the needed information from the functions and Application Maps



Test Automation Development Lifecycle



10s

Model Based Testing

- Since approximately 2002, Model-Based Testing became available.
- Model-Based Testing is the application of Model based design for designing and executing the necessary artifacts to perform software testing.
- This is achieved by having a model that describes all aspects of the testing data, mainly the test cases and the test execution environment.
- Usually, the testing model is derived in whole or in part from a model that describes some (usually functional) aspects of the system under test (SUT). (Wikipedia)

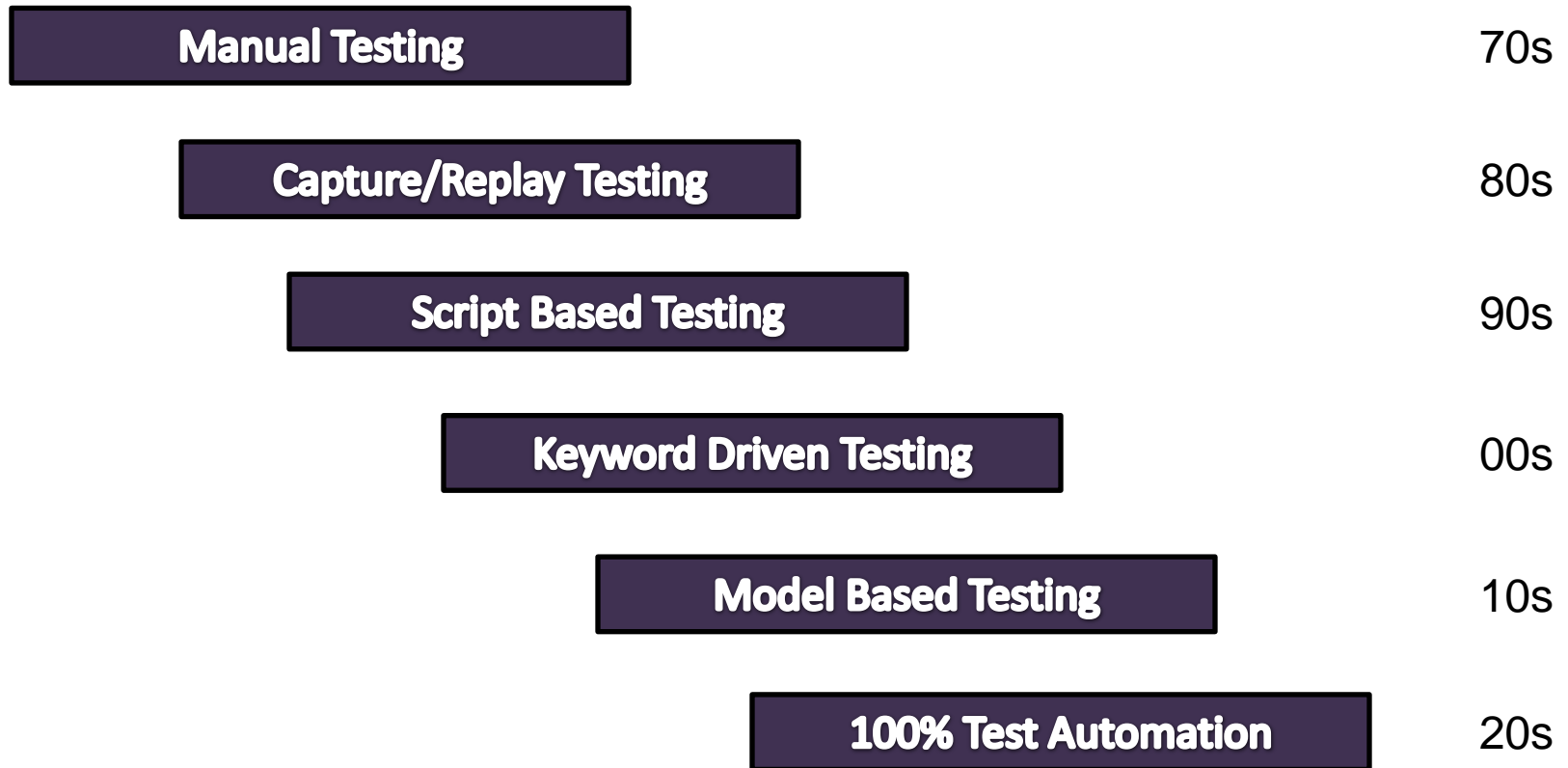
(from: Software testing and continuous quality improvement - William E. Lewis, Gunasekaran Veerapillai)

Test Automation Model 2020

100% Test Automation

- Business constraints requires organizations to shorten their software development lifecycle
- Shorter SDLC means less time for testing. Less time for testing means more test automation.
- Advanced techniques and approaches for test automation will spread and more and more organizations will end up having 100% of the test execution to be done automatically.
- Testers will focus on planning more test and investigating issues found by test automation.

Summary



Introducing FUNTASY

- FUNTASY is a functional test automation system that is based on GUI and non-GUI Test Automation and using Keyword Driven Testing (KDT) approach
- It enables cross-organization test design by employing a keyword driven testing rather than any formal language
- It allows testers to plan, design and execute tests with no coding knowledge required
- Test Case are written way before the system is ready for testing
- Full integration with all leading test tools such as HP's QTP, TestComplete, JUnit and more

FUNTASY[©]

Fantasy - Functional Test Automation System [Execution only mode]

File Tools Plugins Automation Manager Help

Log Parser MSSQL SNMP Telnet Test Complete WebGUI CMD

Automation Manager [ayalz, Online] Test Complete KDT

Test Complete KDT Add-in Manager Automation

Details

Name: Log in Test Case

Author:

Purpose:

Tested Subjects:

Last Update: Never

Objects Sequences Functions

- Sequences
 - Ayal
 - avishai
 - Eldar
 - MyFolder
 - Mark Special

New

Local Params

Input Params

Load Objects

Has Output

	InUse	Step ...	Description	Expected Result	Step Type	Item ID	Operation	Output
▶	<input checked="" type="checkbox"/>	1	Sets the value of an edit field		Item	(Edit)	Set	"Adm
	<input checked="" type="checkbox"/>	2	Sets the value of an edit field		Item	(Edit)	Set	"1234
	<input checked="" type="checkbox"/>	3			Item	OK (Button)	Click	
	<input checked="" type="checkbox"/>	4			Sequence		Login2	<User

Save as Test Save as Sequence

Fantasy version: 2.1.0.19794 Enterprise

Non-GUI Test Automation

What is it?

- Testing non-GUI components by invoking parts of the applications under test and catch the return values of the invocation

Where can I use it?

- Any application that support open architecture (SOA, XML, API, Network Elements, Web etc.)

Benefits

- Less Development
- Less Maintenance
- Fast Execution
- Robust

Conclusion

- Prepare test automation before the system is ready for testing >> allow focus on new features automation and not just regression
- No double documentation
- Anyone can plan, design and execute test automation
- Creating test cases is easier
- Improve communication
- Reach 100% automation



ayalz@qualitest.co.il

Phone: 972.3.737.1300

www.QualiTEst.co.il

Or visit QualiTest at:



Thank You!

www.QualiTestGroup.com

