Challenges in Software Testing

Prepared and presented for JaSST by

Dorothy Graham

email: info@dorothygraham.co.uk

Twitter: @DorothyGraham

www.DorothyGraham.co.uk

DG

Contents

- software testing in the past
  - software testing in Europe
  - qualifications and ISTQB
- practical advice for the present
  - measuring the value of testing
  - intelligent mistakes in test automation
- challenges for the future
  - for testing, automation and testers

DG

70s and before

- mainframes, few turnarounds/day
- or real time – assembly language
- developers test their own code, no testing careers
- tools are “utilities” developed for purpose
- first books on testing (USA)
  - 1st mention, Jerry Weinberg - programming book, 1961
  - Bill Hetzel, Program Test Methods, 1973.
  - Glenford Myers, The Art of Software Testing, 1979
- Hot topics
  - optimising for hardware space
  - high level languages

DG See www.testingreferences.com for a testing timeline – and Keizo Tatsumi 3

80s

- IEEE829 and other test standards
- US testing conference USPDI Washington DC ‘83
- testers - seen as 2nd class, developers paid more
- commercial test tools
  - Linda Hayes’ Autotester, 1985
- UK SIGIST begins 1989
- Hot topics
  - structured programming
  - PCs, GUIs, client/server, OO, connectivity
  - new development methods to do away with expensive people (CASE tools) – and testing?

DG 4

info@dorothygraham.co.uk

Technology

- My new computer (1989)
  - Apple Macintosh SE/30 HD 2/40
    - Motorola 68030 processor - 16 Mhz
    - 2 MB memory, 40 MB internal disc
    - black & white screen
- Cost: £2223 (with 30% discount) (¥ 300,000)
  - equivalent to £5000 today (¥660,000)
  - Powerpoint and Word £110 each (¥15,000)
- No connection to outside world
  - no internet, no email, no mobile phones!

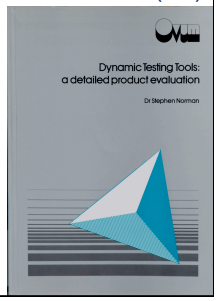
DG

90s

- tool reports (CAST Report, Ovum)
- 1st EuroSTAR 1993, London
- email and the internet
- BS7925 published by BSI (98)
- qualifications, career testers
- more books, magazines
- Hot topics:
  - tools
  - tester/develop ratios
  - testing GUIs, RAD, OO
  - process, Quality Management, ISO9000

82 tools described (from '91)

17 evaluated ('93)



DG

info@dorothygraham.co.uk

European testing groups

- SAST (Swedish Association for Software Testing) founded 1995
- TestNet (Netherlands) founded 1997
  - Tmap book published 1995 (in Dutch)
- first SQC conference 1996 (Germany)
- FAST (Finnish Association of Software Testing) founded 2001
- Soft Test Ireland founded 2002
- SJSI – Polish IS Quality group founded 2003
- Norwegian computer society

DG

00s to now

- tool suites, open source tools
- developers re-discover testing (TDD)
- context-driven school, exploratory testing
- professionalism, specializations in testing
- many SIGs, publications, conferences
- Hot topics
  - outsourcing / off-shoring
  - certification / anti-certification
  - agile, exploratory testing
  - test automation
  - social networks, cloud, virtualisation

DG

info@dorothygraham.co.uk

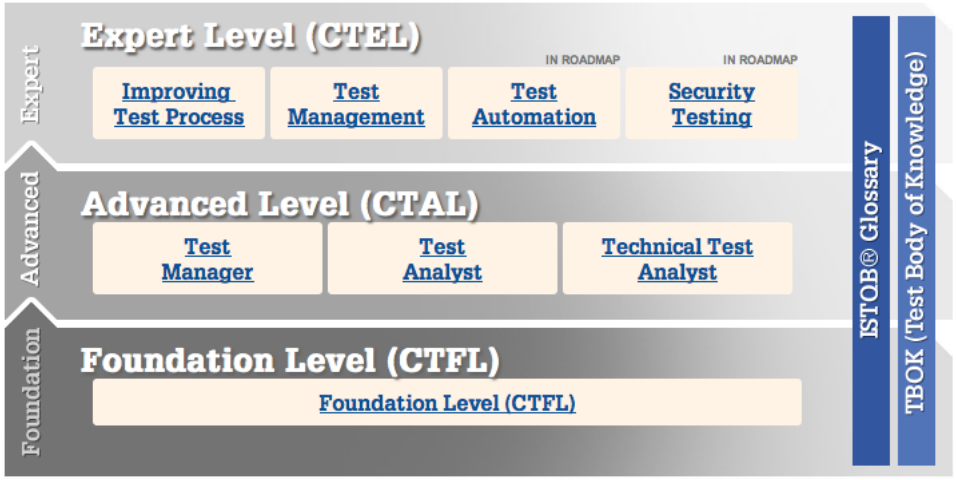
ISTQB

- ISTQB = International Software Testing Qualifications Board
  - precursor: UK's ISEB Software Testing Board
    - first software testing foundation certificate issued 1998
  - ISTQB formed in 2002 (group meetings in 2001)
  - common syllabus, accreditation of trainers, shared exam questions, international governance
  - 46 member boards, covering 70 countries
  - 250,000 ST certificates world-wide

DG

Source: ISTQB.org

Qualifications



DG

Source: ISTQB.org

info@dorothygraham.co.uk

Before ISTQB

- some training, all different
- varied terms in testing
- testers not respected, not a real career
- pockets of interest / expertise
- “anyone can test”

After ISTQB

- common syllabus for training
- common vocabulary
- recognized as a profession, more respect
- coordinated sharing of expertise & knowledge
- something to be learned

DG 11

What has changed over the years?

- a growing profession
  - qualifications and respect
  - from one book to hundreds
  - from “tester” to specialisms within testing
- technical change
  - from mainframes to the cloud, SoLoMo
  - from homegrown utilities to commercial & free tools
  - from KB to GB to TB
  - from print (books, journals), to internet, blogs, twitter..
  - from “turnarounds” to continuous integration

DG 12

info@dorothygraham.co.uk

What hasn't changed?

- managers don't understand testing
- testing seldom taught at university (or to CEOs)
- tools are seen as a panacea/silver bullet syndrome
- people new to testing don't know much
- lots of people are new to testing
- no/little desire to learn from the past
- new technology, constant change
  - testing follows the technology
- testing is still testing; people are still people

DG

Contents

DG

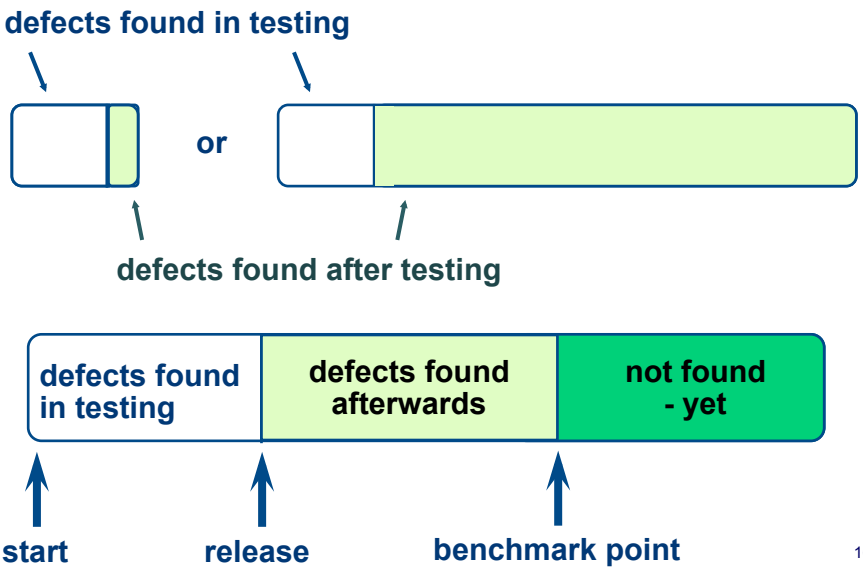
info@dorothygraham.co.uk

What value does testing provide?

- Potential value
  - finds defects
  - gives confidence
  - gives information, e.g. about risk
- How can you tell if value is provided?
  - how can you measure this value? (examples)
    - for finding defects – DDP
    - for assessing confidence
    - for showing risk

DG 15

How effective are we at finding defects?



DG 16

Defect Detection Percentage (DDP)

$$\frac{\text{defects found by this testing}}{\text{total defects including those found afterwards}}$$

- "this" testing could be
  - testing of a sprint, increment or story
  - a test stage, e.g. component, integration, acceptance, regression, etc.
  - testing for a function, subsystem or defect type
  - all testing for a system

DG

DDP example

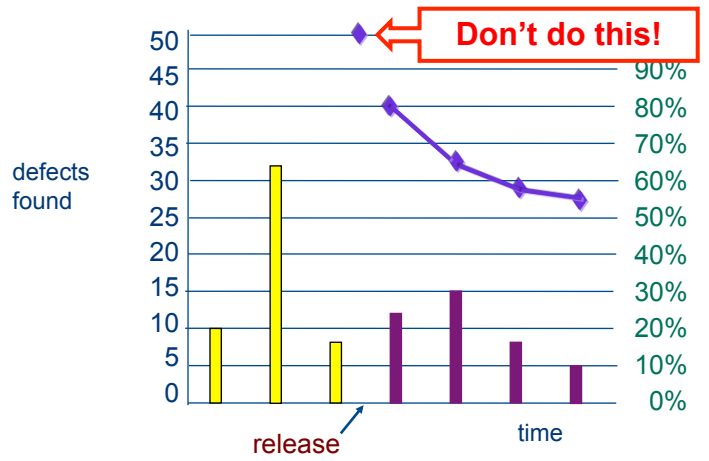
testing	live running	DDP after live
150	50	75%

$$DDP = \frac{150}{150 + 50} = \frac{150}{200} = 75\%$$

DG

info@dorothygraham.co.uk

Effectiveness at finding defects

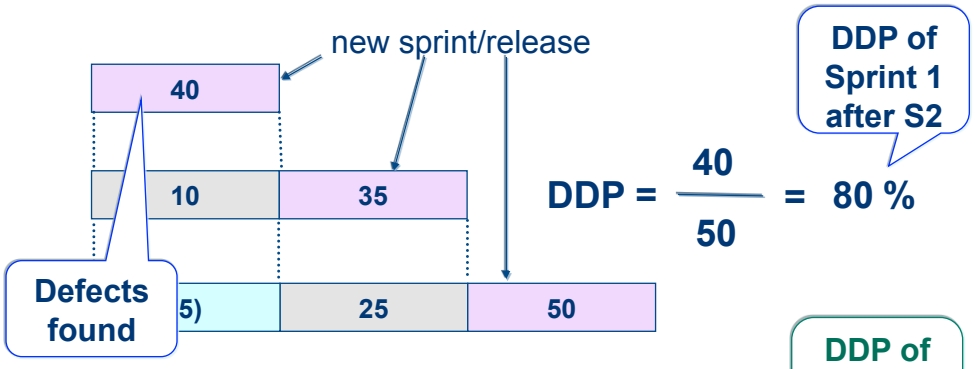


defects found in testing: 50 | defects found after testing: 40 | total defects found: 90

$$DDP = \frac{50}{90} = 55\ \%$$

19

DDP in agile/iterative development



DDP of Sprint 1 after S2

$$DDP = \frac{40}{50} = 80\ \%$$

DDP of Sprint 2 after S3

$$DDP = \frac{35}{60} = 58\ \%$$

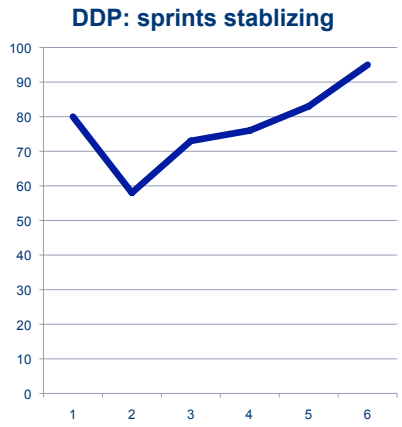
Options: by sprint, all previous sprints, all defects for each sprint, etc!

DG

20

info@dorothygraham.co.uk

DDP examples over Sprints



DG

What is your DDP?

- How many bugs found in testing for the system or area that is now live/released?
- How many bugs found since it was released?
- Your DDP will be:
  - guaranteed to be
    - between 0% and 100%
  - your actual number doesn't matter a lot
    - it's how it changes over time

DG

info@dorothygraham.co.uk

Case studies from clients

	1 mo	10 mo
year 1	70%	50% est
year 2	92%	

Finance (insurance)

Operating system

System Test Group DDP = 38%
(before performance testing)

Priority 1 & 2 only: DDP = 31%

23% to 87% by application

Defects: 1 / 4 160 / 200

Scientific software
(chemical analysis)

Not useful for low numbers of defects

DG

DDP Summary for AP Europe

Project or App.	Months	DDP	DDP Status	Comments
Before New Testing Process				
S4		50%	ESTIMATED	
After New Testing Process				
R1	3	81%	FINAL	*Major re-engineering*
LBS	4	91%	FINAL	
CP	7	100%	FINAL	*Reporting System*
DS	3	95%	FINAL	
APC	4	93%	FINAL	
ELCS	4	95%	FINAL	*Eur impl. of US system*
SMS	3	96%	FINAL	*Enhancement Release*
C	4	96%	FINAL	
E7 (US)	5	83%	FINAL	*Global Enhancements*
E7 (Eur)	1	97%		*Global Enhancements*

Source: Stuart Compton, Air Products plc

DG

info@dorothygraham.co.uk

What does it mean?

- DDP is very high (> 95%)
  - testing is very good?
  - system not been used much yet?
  - next stage of testing was very poor?
    - e.g. ST looks good but UAT was poor, ST after UAT is high – but live running will find many defects!
- DDP is low (< 60%)
  - testing is poor?
  - poor quality software (too many to find in the time)?
  - deadline pressure – testing was squeezed?

DG 25

DDP benefits

- DDP can highlight
  - testing improvements
  - the effect of severe deadline pressure
- can raise the profile of testing
- is applicable over different projects
  - reflects testing process in general
- can give on-going monitoring of testing

DG 26

info@dorothygraham.co.uk

Options for measuring DDP

- what to measure
  - simplest: all test defects / all defects so far
  - by severity level
- how "deep" to go?
  - deeper levels give more detailed information
  - deeper levels more complex to measure
- advice: start simple
  - simple information is much better than none
  - trend is important, detail & accuracy are not
  - build DDP calculation into your bug system

e.g. many sprints, eliminate duplicate defects

DG

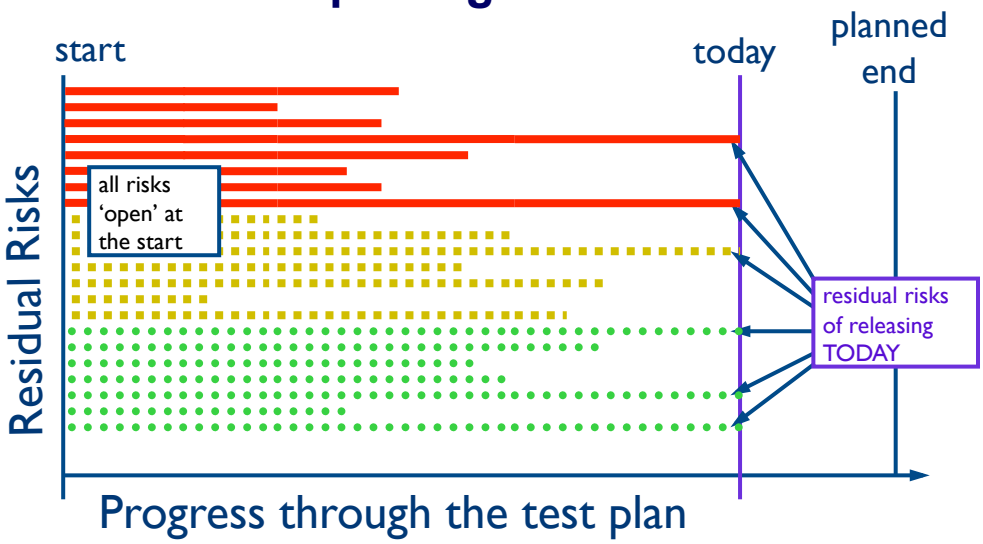
Confident about what?

- objective measures
  - the system does the right things, usable, reliable
    - if tests pass [and a good set of tests]
  - is the testing good [enough]
    - coverage (of code, functions, menus etc)
    - review against standards
- subjective measures – confidence is a feeling
  - ask “how confident are you” (on a scale of 0 to 5)
  - assess against targets
    - e.g. target = 4.5 (out of 5.0)
    - average confidence = 3.5 ➔ not there yet

DG

info@dorothygraham.co.uk

Risk-based reporting

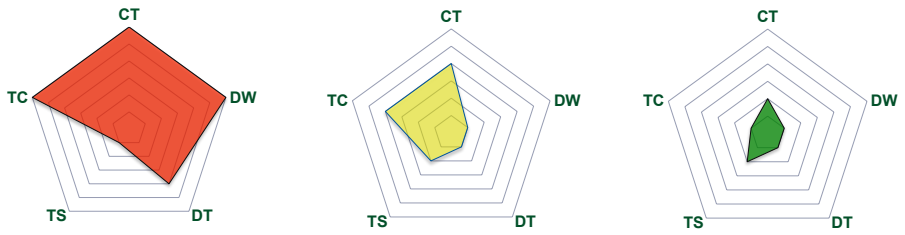


DG Source: Paul Gerrard & Neil Thompson, Risk-based e-business testing, Artech House, 2002

"Risk spider"

Weekly or daily updates on the top risk factors, e.g.

- **CT:** **Code Turmoil**
- **DW:** **Defects found this Week**
- **DT:** **Defects open in Total**
- **TS:** **Test Success Rate (% that passed)**
- **TC:** **Test Completion Rate (% of planned tests run)**



DG Source: Mike Ennis, Managing the End Game of a Software Project, EuroStar 09 30

info@dorothygraham.co.uk

Contents

- software testing in the past
  - software testing in Europe
  - qualifications and ISTQB
- practical advice for the present
  - measuring the value of testing
  - intelligent mistakes in test automation
- challenges for the future
  - for testing, automation and testers

DG

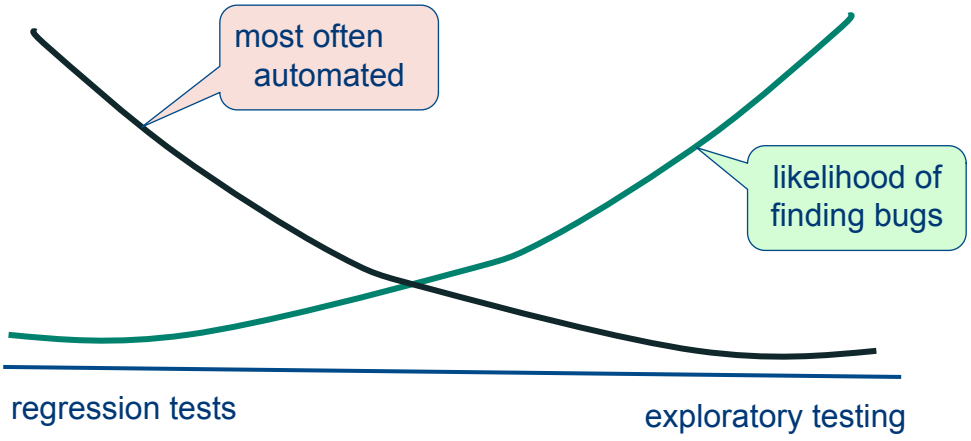
Intelligent mistakes in automation

- automated regression tests should find [lots of] bugs
- tools can and will replace testers
- the tool will provide the architecture for the tests
- automation is just test execution
- automate all manual tests
- ROI is essential

DG

info@dorothygraham.co.uk

Bug-finding and automation



DG

Automation should find bugs?

- automation doesn't find bugs: tests find bugs
- the bug-finding ability of a test is not affected by the manner in which it is executed
- automation is a mechanism for running tests
  - a test-running activity, not a bug-finding activity
- mistake: confuse objectives for testing with objectives for automation

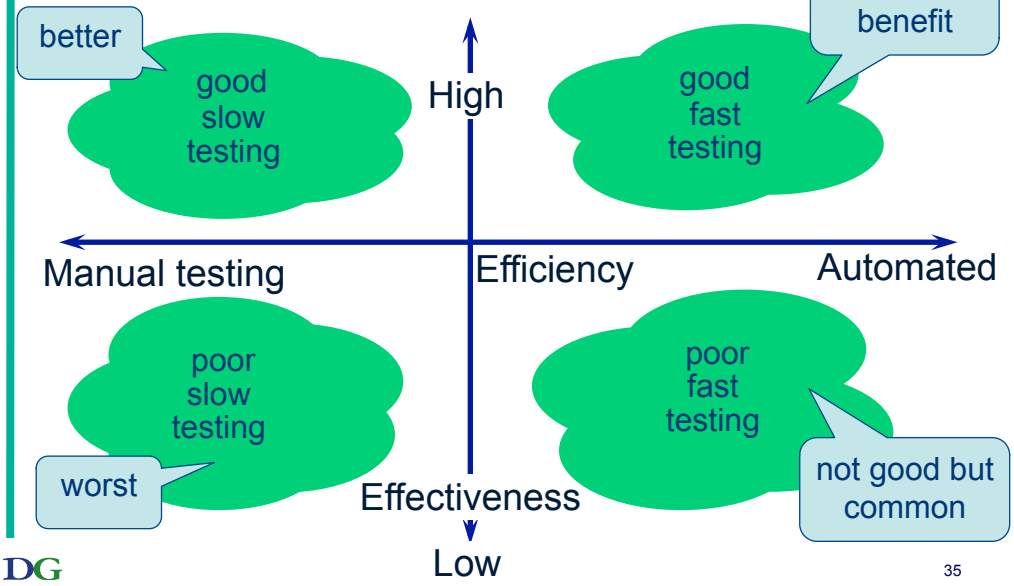
Automated tests	Manual Scripted	Exploratory	Fix Verification
9.3%	**24.0%**	**58.2%**	**8.4%**

DG

Experiences of Test Automation, Ch 27, p 503, Ed Allen & Brian Newman

info@dorothygraham.co.uk

Efficiency and effectiveness



DG

When is "find more bugs" a good objective for automation?

- when the first run of a given test is automated
  - Model-Based Testing (MBT), exploratory test automation, automated test design, monkey testing
  - keyword-driven (e.g. users populate spreadsheet)
- find bugs in parts we wouldn't have tested?
  - indirect result of automation
  - direct result of running more tests

DG

info@dorothygraham.co.uk

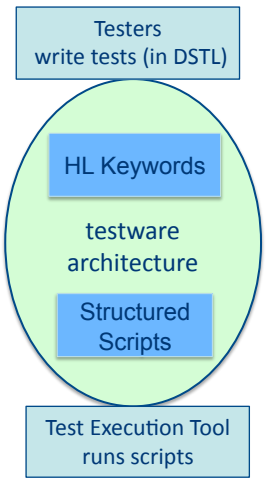
Tools will replace testers?

- "we can reduce the number of testers once we have the tool"
  - what are your testers like?
    - mindless mechanical machines, or
    - intelligent investigators?
  - need more skills, not fewer
    - development skills to work directly with tools
  - automation can free testers to do more test design, exploratory testing
    - and find more bugs
  - tools don't replace testers, they support them

DG

Testware architecture

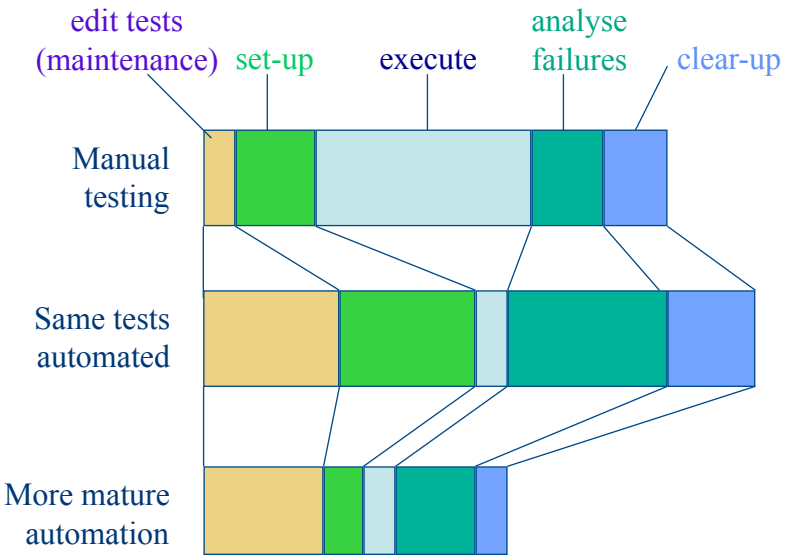
- use your tool's scheme?
  - locked in, great for vendor!
  - define what's best for you
- a poor architecture gives high maintenance cost
  - often leads to shelfware
- two layers of abstraction are critical for success
  - technical: for long life
  - human: for wide use



DG

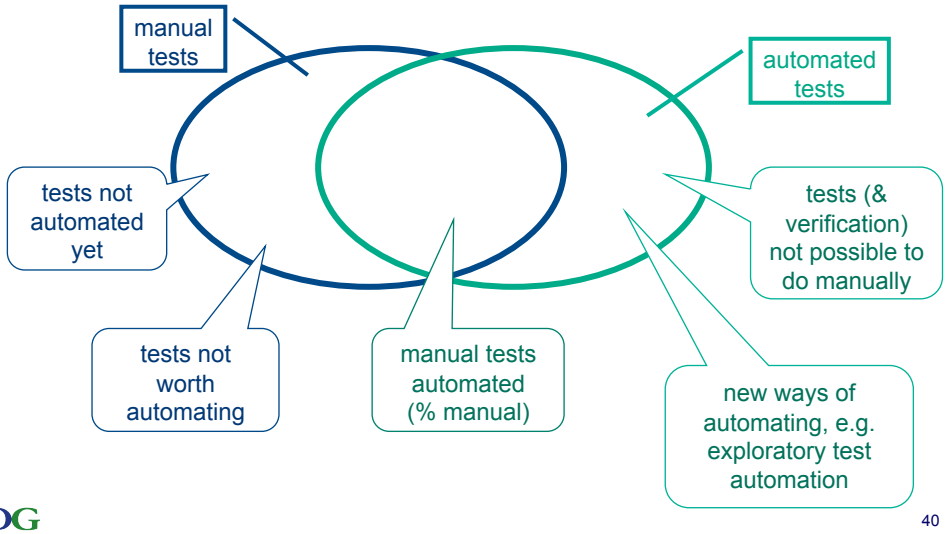
info@dorothygraham.co.uk

Automation is more than execution



DG

Automate manual tests?



DG

info@dorothygraham.co.uk

Is this Return on Investment (ROI)?

- tests are run more often
- tests take less time to run
- it takes less human effort to run tests
- we can test (cover) more of the system
- we can run the equivalent of days / weeks of manual testing in a few minutes / hours
- faster time to market

ROI = (benefit – cost) / cost

these are (good) benefits but are not ROI

DG

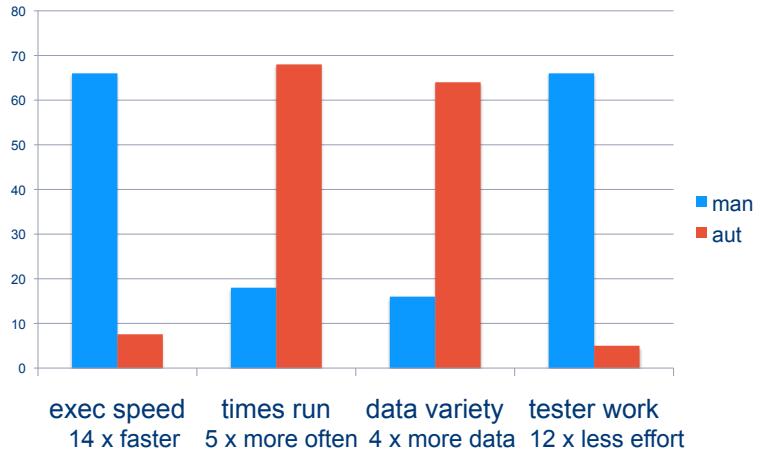
How important is ROI?

- ROI can be dangerous
  - easiest way to measure: tester time
  - may give impression that tools replace people
- "automation is an enabler for success, not a cost reduction tool"
  - Yoram Mizrachi, "Planning a mobile test automation strategy that works, ATI magazine, July 2012
- many achieve lasting success without measuring ROI (depends on your context)
  - need to be aware of benefits (and publicize them)

DG

info@dorothygraham.co.uk

An example comparative benefits chart



ROI spreadsheet – email me for a copy

DG

Contents

- software testing in the past
  - software testing in Europe
  - qualifications and ISTQB
- practical advice for the present
  - measuring the value of testing
  - intelligent mistakes in test automation
- challenges for the future
  - for testing, automation and testers

DG

info@dorothygraham.co.uk

Challenges for the future: testing

- technical challenges
  - cloud, virtualization, SoLoMo
  - Agile, Continuous Integration, DevOps
  - Testing in Production (TiP)
- different focus for testing
  - consumer-driven focus: usability, quality?
    - Service-Driven Test Management (Martin Pol)
  - testing as a Service
  - tests to be a commodity? (James Whittaker)
    - download from iTest?

DG

Challenges for the future: automation

- more automation
  - unit tests with continuous integration
  - routine for regression tests to be automated
- better automation - levels of abstraction
  - structured and efficient for any tester to use
    - MBT, DSTL, Exploratory Test Automation, “scriptless”
  - good design principles for minimal maintenance
- wider scope
  - “outside the box”, not just traditional automation
  - manual tests supported/surrounded by automation

DG

info@dorothygraham.co.uk

Challenges for the future: testers

- the tester's role may change
  - out of a job?
    - because developers now test so well…?
    - all testers have now become developers??
  - crowd-sourcing, social media, user testers
    - communication, collaboration
  - technical advisors to developers/co-working?
    - design for testability / automatability
  - root cause analysis & defect prevention
  - customer/business facing at the highest level?

DG

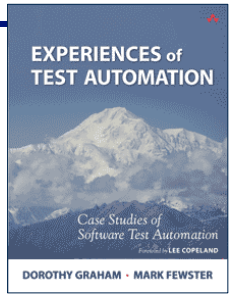
Summary: key points

- software testing has come a long way
  - from unnoticed and unappreciated
  - to respected and qualified
- measure the value of testing (e.g. DDP)
- beware of "intelligent mistakes" in test automation
- the future looks very interesting!
  - for testing, testers, and automation

DG

info@dorothygraham.co.uk

More information



- downloads www.DorothyGraham.co.uk
  - articles and papers
- email info@DorothyGraham.co.uk for
  - Framework and test execution tool list
  - ROI calculator
  - my random newsletter
- blog http://dorothygraham.blogspot.com
  - including automation, DDP, certification
- twitter
  - @DorothyGraham

DG

info@dorothygraham.co.uk