The world of the testing has changed during the years and has developed hugely. But what kind of developments have happened and what things have changed? What does the present look like? What great things have been created? Can we see a trend to the future?

This presentation was originally crafted in 2010 and updated in 2014 and in 2019. Still, it is an ongoing draft...

Matti Vuori [www.mattivuori.net](http://www.mattivuori.net) matti.vuori@mattivuori.net
Contents 1/7

Testing develops without us noticing 9
Disclaimer 10
Testing develops with the software development 12
Caricatyric stages of software development 13
... Behind this are big things 18
...and smaller things 19
Trajectories of the features of the testing 20
OVERALL VIEW 21
OPERATING ENVIRONMENT AND ROLE OF TESTING 22
  Role in the software production 23
  Role of testing from viewpoint of acquirer of an information system 24
  Development of the quality thinking by layers 25
  Position of testing in the world of quality assurance 26
  New "critical systems" 27
  Megatrend: Pervasive computing 28
## Contents 3/7

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tester's stereotypes</td>
<td>44</td>
</tr>
<tr>
<td>If you’d hire another kind of professional to do testing, what profession is it?</td>
<td>45</td>
</tr>
<tr>
<td>Organising</td>
<td>46</td>
</tr>
<tr>
<td>Actors in “ecosystem” of testing</td>
<td>47</td>
</tr>
<tr>
<td>Supplier of testing, locations and culture</td>
<td>48</td>
</tr>
<tr>
<td>Megatrend: Communality</td>
<td>49</td>
</tr>
<tr>
<td>Phenomenon: Postmodern organisation</td>
<td>50</td>
</tr>
<tr>
<td><strong>ACTORS' ROLES</strong></td>
<td>51</td>
</tr>
<tr>
<td>Tester's role and status</td>
<td>52</td>
</tr>
<tr>
<td>Tester's central information needs</td>
<td>53</td>
</tr>
<tr>
<td>Tester's fears &amp; turn-offs</td>
<td>54</td>
</tr>
<tr>
<td>Tester's pressures</td>
<td>55</td>
</tr>
<tr>
<td>Tester's communities</td>
<td>56</td>
</tr>
<tr>
<td>Megatrend: Fragmenting of communication, communities and forums</td>
<td>57</td>
</tr>
<tr>
<td>Programmer's relation to her products</td>
<td>58</td>
</tr>
</tbody>
</table>
## Contents 5/7

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concept testing artefacts</td>
<td>74</td>
</tr>
<tr>
<td>Timing of functional testing</td>
<td>75</td>
</tr>
<tr>
<td><strong>TEST TYPES</strong></td>
<td></td>
</tr>
<tr>
<td>Unit testing</td>
<td>77</td>
</tr>
<tr>
<td>Separation and unification in unit testing</td>
<td>78</td>
</tr>
<tr>
<td>Low level integration testing</td>
<td>79</td>
</tr>
<tr>
<td>System integration testing</td>
<td>80</td>
</tr>
<tr>
<td>Acceptance testing of information system</td>
<td>81</td>
</tr>
<tr>
<td>The whole testing of information systems</td>
<td>82</td>
</tr>
<tr>
<td>Stress/load testing of information systems</td>
<td>83</td>
</tr>
<tr>
<td>Motivation for stress/load testing</td>
<td>84</td>
</tr>
<tr>
<td>Usability testing</td>
<td>85</td>
</tr>
<tr>
<td>Localisation testing</td>
<td>86</td>
</tr>
<tr>
<td>Information security testing</td>
<td>87</td>
</tr>
<tr>
<td>Regression testing</td>
<td>88</td>
</tr>
</tbody>
</table>
## DEVELOPMENT OF TESTING

- Development of agile testing
- Drivers of developing testing
- Standards, maturity models and framework models
- Tester certification systems
- Megatrend: Communality in standards
- Megatrend: Globalisation

## TESTING TOOLS

- Testing tools
- View on good test environment
- Acquisition and management of test environments
- Test automation technologies
- SUT actuation
- Megatrend: Open systems

## Conclusions
The world of the testing has changed along the years and has developed hugely.

But and what kind of developments have happened and what things have changed? What does the present look like?

What great things have been created? Have new things been created that might cause new problems?

The present is always very trivial. Even the innovations cannot be seen at the stage when they really hit it, because by then they are part of our everyday life.

Therefore it healthy to examine the trajectories of things from far away from the history. It helps to appreciate the present and to see the possibility of change – and to prepare for the following steps perhaps earlier than others.
Disclaimer 1/2

• Note that the pages are really “slides” and traditionally any slide should take at least 3 minutes to explain. So there is always more stuff than what one can see.

• But many of the slides describe complex things that would take much longer to explain and many pages of text if written out.

• So, in ideal case the reader should have good understanding of the issues to make proper sense of the slides.
Disclaimer 2/2

- A side notice...
- The world of testing and software development is so diverse and different in different contexts that none of the things presented in this slide set is comprehensive and in many cases the issues are simplified in sometimes almost violent or silly ways.
- Likewise, the whole set is necessarily lacking several themes that are important to many readers.
Testing develops with the software development

- It is essential in the development of testing that it has taken place with the change in the software development
  - Meeting the challenges of the time: processes, scope of the projects, organising methods, need for the controllability etc.
  - Software crisis – and trying to win it
- At the same testing people have of course learned about what testing is and what is possible for it
- And all this only during a few decades
  - Humans have learned to build bridges for thousands of years and it still does not always succeed
- In the following slides there are a few caricature type phases in s/w development through which we will look at the changes
1. "Small" programs made by one programmer
   - <= 1980s
   - 1 maker
   - A couple of modules, a source code file
   - Languages such as RPG, COBOL, Fortran
   - No process in development or testing
   - Programmer does everything directly to his customers
   - Work of hero coder, skill based action
2. Developing of a larger program
   - Information systems and similar
   - 1980's – >
   - A couple of programmers
   - Several components
   - A controlled, led process
   - Organising: a manager, division of labour
   - Testing is debugging – that is, reactive, later proactive
3. Distributed software development

- Projects done by several parties
- Systems of system or wide and complex applications
- 1990's –>
- Many teams, foreign to each other
- Client controls
- Delivery orientation
- Various systematic processes & lifecycle models are applied
- Testing is systematic, planned, competent
- Separate testing teams (in 2000's)
4. Distributed software development – agile era
   - As # 3, but agile project management
   - 2000's – >
   - Testers brought to development teams
   - Still separate testing teams exist
   - In addition to pre-planned testing, agile testing is used
   - Testers usually are skilled for their tasks
Caricatyric stages of software development 5/5

5. Cloud time
- Software that offers services is produced to a chaotic networked whole
- Huge number of other suppliers, not known or controlled
- Management of things is based on interfaces
- Systems updated often
- Automation in big role, CI systems and DevOps pipelines
- 2010's >
... Behind this are big things

- The software development does not revive by itself either
- In the background there are influencing things (to which this small listing makes no justice), which affect testing too
  - Learning about the nature of information technology systems
  - Growth of software development competence
  - Spreading of information technology to everywhere
  - Growth of the size of projects
  - Growth and networking of systems
  - Globalisation
  - Etc...
...and smaller things

- For example it is thought that the project models of agile developing put many things anew, changing rules, preconditions, possibilities.
- Some of that is true, but some of this are just ripples in the big trends.
Trajectories of the features of the testing

- On the following pages there are trajectories of the different characteristics of testing
- The time periods that were described above are only as a background orientation, they are not perfectly followed because different things develop at different pace
- To point out some especially good or bad / dangerous things, some visual symbols are sometimes used
OVERALL VIEW

Reactive debugging

User experiment of functions

The developers' debugging

Systematic end testing of the software

Acceptance testing

Several testing levels

Constant testing

Skilled testers Systematic system testing

Special testing of quality characteristics

Exploratory testing

Testing integrated in workflows

Bug bounties in infosec

Maturing diverse culture

Points

Testing has become infinitely more diverse and rich!
OPERATING ENVIRONMENT AND ROLE OF TESTING
Role in the software production

Testing is a mandatory parasite

Important phase

Item of expenditure to be minimised

Item of expenditure to be optimised

Test policy

Critical stage and process characteristic

The quality system obliges to test

Customers require testing

Customers require good (!) testing

Test policy

Testing precondition for market entry (for example safety critical).

Platform requires certification testing

Points

The role begins to be already generally understood. The testing does not need to be hidden from customers!
Role of testing from viewpoint of acquirer of an information system

Need for testing is a sign of badness of the supplier ("Can't they code!")

Mandatory item of expenditure

The quality management system obliges to test

One must invest self a lot

Developing of acceptance testing culture

Sign of goodness of the supplier

Missing acquisition process

Projectisation of acquisition

Systematic acquisition models

Points

The acquirers understand the importance of testing too
Development of the quality thinking by layers

- Risk management
- Prevention of defects, not repair
- Analysis is as important as testing
- Robustness thinking in the complex world
- User-centred quality, user & customer experience
- Customer-oriented quality
- Quality management, leadership
- Stages and gates in processes, reviews
- Quality of the processes produces the quality of the product
- Technical quality
- Security & privacy

Points

During every era people will focus on the new things just noticed to be lacking, but old things remain critical!
Position of testing in the world of quality assurance

- Code inspection
- Static analysis of code programmatically
- Dynamic analysis of code
- Architecture assessment
- Programmatic inspection and architecture in integration testing
- Project phase reviews
- Inspection of project phase products
- Review of project plans
- Analysis of state model
- Model-based testing
- Testing of value
- Evaluation of the ergonomics of the user interface
- Usability testing
- Usability analysis
- User experience testing
- Risk analysis
- Risk based testing
- Programmatic inspection of technical usability properties

Points: Testing is a part of a rich whole
New "critical systems"

Points
What is "critical" varies in contexts

Safety critical systems

Business critical systems

Experience critical systems
Megatrend: Pervasive computing

• How is it seen:
  – Software demystified
  – Understanding of features of applications and information systems
  – People can be realists: software is acquired in a different way than office furniture
  – It is understood that testing is always necessary

• Think:
  – How will it have effect in future?
  – How could we utilise it?
  – What drawbacks does it have?
National challenges

- How to get companies to test at all
- How to make companies understand testing
- How to get companies to use capable testers
- How to build capable testers
- How to make companies test more efficient
- How to support business better with testing
- How to do testing in continuous deployment
- How to do testing in continuous deployment
- How to do testing in agile processes
- How to build the tester's occupation with identity and respect
- How to help startups do testing
- How to help transforming industries in testing (machine builders)
- Continuous learning and renewal of the competence, approaches, craft

Points
Challenges change all the time – what will be the next...
NATURE OF TESTING
Schools of testing

There are always "schools" due to environments, business and politics and they too change with time. Note that this applies mostly on functional testing.
World views

How the world works

- Mechanical, causal
- Systemic
- Process
- Activity

Testability of things

- Things are provable
- Things can be questioned

Systems used by humans

- Technical systems
- Socio-technical systems, human-computer systems
- Rational tools
- Products of meaning, representation, desire

How people act

- Rationally
- In just about any way
Researchers’ biggest challenge

**Computing view**

- Development of techniques and methods

**Hope of “best practices”**

- Development of tools
- Development of practices

**Planning for diverse world and future**

- Development of new approaches
- Organisational issues and change
- Understanding of what really happens in organisations
Evolution of ethics in testing

Worker’s ethics (for employer) → Quality ethics → Tester’s ethic → Social responsibility (e.g. security, privacy)

Worker’s ethics (for employer) → Consultant’s ethics → Ethics in service provision
Each era brings with it a new set of terms that express new things and a change of culture.
Great things in testing

Getting programs to work well

Customer's satisfaction is assured

Low risk level to customer is assured

Professionalism, management of things

Tester's identity, special know-how

Risks of society managed – good society enabled

Behaviour of software is understood

Software all time working and stable

Good ethical business

Risk to customers managed

Cyber risks managed

Points

The world changes and the great things change with it
Aesthetics of work

- Beauty of well-working practice
- Control of things
- Complexity, clock-mechanisms of processes
- Beauty of thinking, intellect
- Dream of machines
- Flow, rhythm
The worst hype of the period

Testing is not needed...

Faultless code

Manual testing is not needed... 100% automation

Complete testing

Systematic test planning

Done lightly, as needed

Validation of code

Generation of faultless programs

No need to test, just fix when users find bugs! (As Google does)

Model based testing a silver bullet

A/B testing best way to develop usability

Test automation

Unit testing and integration testing are enough

Only exploratory testing

Points
Each era has its own hype, which will be only later be understood us such
Marketing terms for testing methods

**Points**

Marketing terms tell a lot of the current thinking

**Tools**

- "Automated"
- "Complete coverage"
- "Fast", "Easy"

**Manual**

- "Systematic"
- "Based on standard"
- "Agile"
- "Supports agile development"

**AI, ML**
Silliest things of praise

- Complexity of test process ("just look at this chart!")
- Automation level & number of test cases
- All testing is automated (everything can be automated)
- Doing only exploratory testing
- Whole process is documented and fully instructed (no matter how good it is)
- No documentation & we let teams do as they wish (cause none of us has no clue)

Points: It is good to be proud, but preferably for the right reason
Defects and view to quality

- Code and its defects
- Application and its defects
- Development process and its quality
- Correct working
- Working of functions
- All quality factors
- Success of usage scenarios
- Success of deliveries
- Applications' life cycle
- Lack of deterministic systems
- Realities of chaotic systems

Points: Here shows a learning process and hierarchy of needs
Publicity of defects and criticism of testing

Media era... Internet era

- Customer complains
- Service did not work
- Product has faults
- Closed systems with secret bugs
- Due to software bugs, service did not work
- Constant complaining of product in net
- Active 3rd party security research
- Open source with public bug databases
- New security vulnerabilities are news
- Due to bad testing, service did not work
- Constant complaining of testing in net
- Making headlines

Points: In the world of Internet, anyone can get in the news!
PEOPLE IN TESTING
Tester’s stereotypes

QA tester in engineering domain

Information system / business tester in ICT department of insurance company

Hard core testing tool specialist in telecom

Generic skilled tester

Generic unskilled tester

Automation specialist

Exploratory testing expert

Automation specialist in CD context

Security testing specialist

Usability testing / assessment specialist
If you’d hire another kind of professional to do testing, what profession is it?

Yes, this is about mapping stereotypes to analogies

Mathematician

Electrical technician

Typist

Accountant

Fire safety inspector

Anthropologist

Guerrilla warrior

Chess player
Organising

"Customer tests"

Coder

Separate testing team

Coder colleagues

1 tester in team

Customer defining tests (A)

Testers in development teams

Separate testing teams

Role of developers re-emphasised

Special program testers of quality characteristics (for example load, security, usability)

Points

In organising of testing, enrichment of testing is seen
Actors in "ecosystem" of testing

- Publishers
- Certifiers
- Testing companies
- Trainers
- Consultants
- Makers of testing tools
- Client
- Own organisation
- Customer

Points: Number of actors increases all time
Supplier of testing, locations and culture

Points: Outsourcing, off-shoring and decentralisation came here to stay
Megatrend: Communality

• How is it shown:
  – Open source code
  – Increase in public beta testing
  – Social web techniques in test management systems

• Think:
  – How will it have effect in future?
  – How could we utilise it?
  – What drawbacks does it have?
Phenomenon: Postmodern organisation

• How is it shows:
  – Organisation is not controlled by one big story and mission but in everyone's work own meanings must be found
  – Organisation lives from these meanings and management still has an important role in their creation
  – Everyone creates her own relationship with quality, bugs and testing
  – That relation is built with a dialogue! It is not created for example with reports in information systems

• Think:
  – How will this have effect in future?
  – How could we utilise it?
  – What drawbacks does it have?
ACTORS' ROLES
Tester's role and status

- Occasional assistant
- Assisting staff in project
- Colleague
- Low-waged assisting staff
- Professional of testing
- Specialist of testing technology
- External abstraction
- In team responsible for testing

Points: Status of testers improves but work must still be done for it
Tester's central information needs

- How can this be tested?
- How is this supposed to work? Where are up-to-date specifications?
- For whom has this been done and to what purpose?
- Personas. How do people act?
- What things are here most important?
- How is the information produced by me used?
- What is the business like?
- What are the risks?

Points
Information needs are linked to the information needs of others! Tester is a supporter of decision-making
Tester's fears & turn-offs

Can I test? What should a tester do?

Why don't we get the information? Where are the docs?

Why does working pace vary? Why must we test the same things again and again?

How will I find my role in an agile development team?

Will there be work for me when all is automated & developers do unit testing? Do I need to become a programmer?

How can I grasp new tech (AI, CD, containers) & security and privacy stud?

Need training

Points
The tester's pressures change but why do not the decrease?

Need proper competence & identity to give value
Tester's pressures

- Non-inspiring routine, narrow work profiles
- Managing documentation
- Time and priorities
- Complex technology
  Tools & scripting
Tester's communities

- Coders + customer + work mates
- Project organisation
- ICT clubs
- E-mail lists and web sites
- Customer's project team
- Project team
- Organisation's network
- Testing club
- Web communities
- Communities, organisations
- All other testing peers

Points: Challenge: how are communities used and how people act in them
Megatrend: Fragmenting of communication, communities and forums

- How is it shown:
  - There are not only few Internet-forums for testers, such as was during the Usenet news era – WWW forums will come and go
  - A plus: for each topic a suitable size forum with a suitable profile can be found
  - Anyone can found a "community" but few know how to grow them
  - Google will find all essential information for free when you need it
  - Net world: a new citizen's basic skill for testers! Searching, communication, asking of things

- Think:
  - How will it have an effect in future?
  - How could we utilise it?
  - What drawbacks could it have?
Programmer's relation to her products

- **Own program, own customers, own users**
- **Team's shared customer** Users are unknown, no direct contacts
- **Software is a piece of a big whole, but its role is not understood** Users are an abstraction
- **Software is a piece of a big whole, but understood** Users and usage are made somewhat familiar
- **Software is an independent actor in world which is not controlled or influence and the other actors of which are not known**

**Points**: A coder's view to her doings influences the possibilities for the testers to act
Programmer's relation to testing

What it is?

When bugs are found, will test in debugger

Others' job. Not mine. I just code

Co-operation. Each has her own role

Has a big role
- Unit testing
- Integration
- Testability
- Develops automation

Points
It is great that coders include testing to their tasks
COMPETENCES
View on characteristics of good tester

- Analytic
- Careful
- Diligent
- Representing users
- Technology expert (test systems)
- Knows product technology
- Being familiar with testing
- Active
- Adaptable
- Well communicating
- Team player
- Takes her own role
- Makes initiatives
- Proactive helper
- Fast
- Has different working styles
- Can program or at least script
- Smart
- Experienced
- Certified

Points
The view about the key competences changes all the time – we must not develop yesterday's skills
Testing wisdom

One must think very carefully who to listen to...
There are many kinds of training and every year some more.
SYSTEMS UNDER TEST
Technologies and testability

- Manufacturer-specific technologies
- Generic standardised interfaces (for example HTTP)
- Accessibility thinking in user interfaces
- Layer architectures and MVC, microservices
- Understandability and testability
- Neglect of testability in platforms
- Neglect of testability in application development
- Testability requirements
- Testability review
- Smoke tests in processes

Points: Possibilities of testability are good even if systems are complex.
New intelligence

Simple logic in simple systems → Complex and changing logic in complex systems and systems of systems → Machine learning (AI) → Need to understand uncertainty, data, deviations...

Points: AI era brings new challenges…

Big data → Image and signal processing
Testing software robotics in offices

Manual data processing between systems

Test automation tools or manual testing

API level glue tools

GUI testing tools Macros

Software robotics for UI

Robot as test tool for itself – just feed it test data. Robots work like UI level test automation tools. Note: tests only integration

Points: New interactivity needs testing too
Needs for better testing

- Software everywhere
- Use over Internet or in local network
- Part system of systems
- Demands raise radically for robustness, security

- Constant growth of size of software
- Number of bugs increases at same bug density
- Relative amount and quality of testing must be increased!

- Complexity of systems and subsystems
- Difficult to understand. Security critical
- More focus on security, robustness

Points: If testing does not improve strongly every year, quality will get worse!
Open technologies

- **Open source code**
- **Undocumented proprietary interfaces**
- **Closed testing tools**
- **Closed operating systems in all kinds of environments and products**

**Myth:** Openness as such improves code

- **Openness makes all kinds of testing possible for anyone**
- **Use of open standards**
- **Generic testing tools**
- **Open testing tools**
- **Tools available for all**
- **Open operating systems (= Linux) & their tools becoming common**
- **Integration of testing tools with systems is easier**

**Points** Openness is an advantage to testers for many reasons
Delivery for managed testing

Production server

Stared test server

Levels of test servers – different test stages

Instantiation in cloud

File based transfer

Configuration based system deployment

Configuration based system deployment in container

Observability & debuggability
TESTING STRATEGIES
Approach

Points
Approaches have become diverse – horses for courses
Considering the software requirements

- Requirements as criteria of acceptance testing and system testing
- Monitoring of requirements coverage
- Traceability of requirements
- Special treatment of safety critical requirements
- Considering of risk related to requirements (to utilization of s/w)
- Focus on stories and scenarios
- Risk-based testing
- Linking of test management with requirements management
- Reviewing of testability of requirements

Points: The relation with requirements has matured all the time
Concept testing artefacts

- Assessment of specifications & static pictures
- Rapid prototyping in non-implementation tool
- Rapid first working version in Agile
- MVP ("customer development")
Timing of functional testing

- Correcting bugs after delivery
- After development, at the end of project
- During project at several stages
- Continuously
- Test driven implementation
- Rapid testing of new things
- Many level tests in continuous integration & deployment systems
- Running of tests in compilation
- Rhythmic processes

Points: All time earlier and more continuously at all testing levels
TEST TYPES
**Unit testing**

- Ad-hoc test drivers and debugger testing
- Compulsory systematic testing
- Coverage measurement
- Test-driven implementation
- Continuous unit testing
- Use of general frameworks (xUnit)
- Support tools for all IDEs
- Outside tester
- No integration with compiling

**Points**
Unit testing progresses steadily but still does not always take place
Separation and unification in unit testing

Programmer tests with lean, integrated test functions

Separate tester seen optimal

Code and tests separated, different tools

Unit testing developer's key task

Code and tests closer in IDEs

TDD

Tool-aided test creation (unit tests and mocks)
Low level integration testing

Ad-hoc integration, no testing

Rhythmic, systematic integration

Rhythmic, systematic integration and testing

Version control

Pre-integration testing in developer’s workstation in unit testing

Continuous integration, testing and versatile analysis of code

Points
Pre-integration in workstations was really the biggest quantum leap

Much more integration than integration testing is still done
System integration testing

Not needed – a monolithic system

Big bang – when systems are completed
Doesn't work!

Continuous systems integration testing

Integration testing as indicator of progress

Vendors hide information

Integration & testing hard

Points: Integration thinking has grown after bitter experiences have been obtained
Acceptance testing of information system

Points
Acceptance testing has developed strongly during the years

- Vendors leads & manipulates
- Somebody tries it
- Systematic trying of functions
- Systematic multi-phased testing in test environments of different levels
- Non-functional criteria included
- Having of acceptance testing done by a testing house (lead, some testers)

The "acceptance testing" in agile has very little to do with serious acceptance testing of information systems

No real user, customer-led testing
The whole testing of information systems

Points: Testing of information systems has gotten rich. Automation and agile testing supplement each other.
Stress/load testing of information systems

- Large number of users doing testing
- Use of load testing programs
- Load testing a routine before going to production
- Free load testing programs in use widely
- Load testing early and then continuously or often

Points: JMeter and others have brought load testing tools accessible to everybody
Motivation for stress/load testing

- Handling large usage volume
- Managing launches and campaigns
- Handling denial or service attacks

Points: JMeter and others have brought load testing tools accessible to everybody.
Usability testing

What is it?

1 outsourced testing round done too late

Review of visual design

Screen design checklists

Design analysis

User studies produce information basis

Usability analyses and testing at a few stages

Low-design, blind A/B testing

UX testing

Assuring of usability is still usually very lacking

Points

Assuring of usability is still often lacking
Localisation testing

Inspection of translation (terms, sentences) → Cultural analysis of proposed translation (text, concepts, visuals) → Smoke test of compiled program and visual inspection → Functional testing of compiled program and visual inspection → Usability testing with localisation / cultural viewpoint

Why does ISTQB vocabulary still not know the whole thing?

Points
Localisation testing is essential to all software products
Information security testing

**What is it?**
- Mechanistic testing of simple vulnerability (by a consultant)
- Testing tools

**Reviews**
- Code analysis for potential problems
- Assuring of safe programming

**Testing tools**
- Rich systemic understanding of mechanism of vulnerabilities
- Information risk analysis
- System audit and review

**Points**
Information security is critical in the era of net but still taken care of weakly.

**Bug bounties**

Assuring of information security is still usually lacking.
Regression testing

Testing of change and an "trying" of other parts of program

Comprehensive user interface testing

Analysis of effects of change

Risk analysis of changes

Automatic regression testing in unit testing and integration testing

Automated regression testing at UI level

Manual regression testing

Continuous integration, testing and versatile analysis of code

Exploratory regression testing

Critical in agile development, but the safety net is not always there

Points

Regression caused by changes remains a big problem
PROCESSES AND METRICS
View on good testing

Testing techniques

- Code coverage
- Standards
- As early as possible test planning
- Once at the end
- V-model
- Requirements coverage
- Test planning at right time
- As early as possible and continuously
- W-model
- Best practices
- Test management
- Process
- Activity system of testing
- Management of quality risks
- Sprint model
- Context thinking
- Kanban

Points: When techniques are managed, we can work at a higher activity level.
Test plans

Project plan in which testing was not mentioned

Project plan in which testing is described

Project plan and lists of things to be tested

Quality assurance plan
Plans for testing of quality properties

Separate testing plans

Testing manuals (instructions)

Little planning in agile projects

In weak organisations regression of culture

Points
Test planning has generally gotten richer
Typical picture in a slide set
Reporting has changed to offering views.
Testing rhythm during development

- One testing round
- A couple of rounds
- Continuous testing
  - For a new version
  - For a new build
  - For a new feature
  - For a change
Management of test assets

Ad-hoc specifications

Mystery excel

Unstructured Word plan

Structured Word plan

Files in random network places

Files in version / configuration control

Low level test designs

High level test designs with adaptation

Task management / backlog

Version control

Points: Handling of specs and other material becoming like handling of requirements and code
Nature of test data

Only little data, well behaving so that tests pass

Realistic production data – a lot

Dangerous data (attacks)

Large-scale testing with poorly behaving, random, broken data

Fuzzing

Privacy considerations (GDPR)

Points: The goal of breaking the program is emphasised in test data
Test design styles

- Design techniques
  - Common sense
  - Risk based
  - Scenario based
  - Model based
  - Random (monkey, fuzzing)

Experience based & tacit
Testing of embedded software

- Systematic testing in simulator
  - Unit testing
  - Definition of test cases

- Versatile software, no OS
  - Limitation is communications

- Simple programmable logic

- Emulator testing
  - All testing levels

- Operating system

- IP communications

- Embedded program part of wider system – end to end testing

- Software can be significantly tested on a PC

- Generic operating system
  - Generic software platforms

**Points**

Testing of embedded programs begins to be mainstream
Most important formal languages in testing

- Flow chart
- ER
- UML
- BDD
- Testing language
- Language of the automatisation environment
- Implementation language
- Efficient scripting languages
- Excel's macro language
- JSON
- SQL
- XML
- HTML
- Gantt chart
- Burndown chart

Points: Every tester needs skills in some formal languages
Data stores of communication

Points: Movement from email data management to shared systems
Risk management of testing

- Project risk analysis
  
  Schedule, infrastructure

- Continuous project risk management

- Own risk analysis for testing
  
  Schedule, infrastructure, deliveries, testability, security and privacy

Security & privacy management

Points

Risk analysis culture spreads and finds own applications in testing
Testing services for functional testing

- Hired testers
- Outsourcing of testing phase / task
- Projectised testing service
- Renting of test managers
- Improvement consultation of testing
- Result-defined "black box" testing service
- Services with various cost profiles
  - In Finnish: in which country is the testing done
- Holistic assessment of testing activity
- Auditing of testing activity (QMS)
- Maturity models assessments

Points:
Testing services evolve all the time. Testing dev. services are important.
Testing metrics

Product metrics
- Open bugs
- Bug situation Maturing
- Situation by seriousness class
- Situation by risk class, residual risk

Process metrics
- Amount of work in testing
- Progress of testing
- Effectiveness of testing
- Quality and effectiveness of testing process
- Number of testers
- Organisational metrics
- Development of testing process
- Test coverage
- How much automated

Points: Measuring develops because it is cheaper than before
DEVELOPMENT OF TESTING
Development of agile testing

- Trying out software
- Testing of functions with positive test cases
- Testing of functions with negative test cases
- Ad-hoc testing
- Exploratory testing
- Systematic exploratory testing
  - Points of view
  - Objectives
  - In different ways at separate stages of project

Points: There is no "one size fits all" agile testing
Drivers of developing testing

- Need to find defects
- Dealing with a complex system (amount of testing)
- Reduction of costs
- Reacting to changes
- Quick development cycles (agile development)
- Keeping software stable
- Decentralisation of development
- Speed of deployment pipeline
- Meeting new requirements for security, UX, CX
- Changing goals, requirements
- Points: There is always a lack of money, but also needs regarding processes
Standards, maturity models and framework models

- Quality systems and maturity models
  - ISO 9001
  - ISO 9000-3
  - CMMI (DEV and ACQ)
  - TMap

- Process development
  - Internal testing training
  - Certification (ISEB, ISTQB)

- ISO, IEEE, IEEE 829 – documenting
  - (TPI) TMMi
  - Introducing of agile development models (developer-centric)

- Showing of competence
  - Test management view
  - ISO/IEC 29119 Software testing
  - Safety standards (IEC 61508)

Points: There are many models to support development – some already at second generation.

Testing activity is in need of balanced metrics.
Tester certification systems

ISEB (Great Britain) Foundation, Intermediate, Practitioner

ISTQB Foundation

ISTQB Advanced

Proprietary cert systems for e.g. agile

Extensions for e.g. agile, MBT

ASQF (Germany)

CAST, CSTP (USA)

ISTQB Expert (many modules, under development)

Points

In certification systems, ISTQB is global and displaced ISEB. Why is the latter still trained...
Megatrend: Communality in standards

• How could it be shown:
  – Situation of standards is catastrophic: they are expensive secrets made by secret societies (fortunately there are plans to open – at least in terms of being free, so that people would get to know about their contents!)
  – Even many known frameworks (with almost de facto standard status) are closed commercial systems
  – ISTQB is still the most globally communal system of building internationally shared views for testing – in other words, standards, though flawed in many ways
Megatrend: Globalisation

- How is it shown:
  - Outsourcing around globe
  - Work in projects in separate time zones
  - Global standards, global testing know-how, global certificates (ISTQB)

- Think:
  - How will it have effect in future?
  - How could we utilise it?
  - What drawbacks does it have?
Hunt Down the Bugs!
SCORE 1200
Testing tools

Execution environment
Debugger
User interface

Terminals
Test harnesses, drivers and stubs
Debugger
Test environments

Ready-made harnesses and tools
Defect management systems
Test management systems

Test automation programs
Model-based testing tools
Expensive, commercial

"All" testing tool types available as open source versions, on user's language. For most purposes many high-quality alternatives.

Points
We live in a good era...
View on good test environment

- Clean, "drawn from pack"
- Corresponds to users' environment
- Different at different test levels and in different tests (change is always an advantage)
- Computers for various operating systems
- Disk images for various operating systems
- Virtualisation – many environments in use at the same time. Rapid creation of defined environments
- Replace environment with stubs
- Replace environment with copy
- Replace external environment with simulating mocks
- Containers

Points: Test environments are getting more and more understood
Acquisition and management of test environments

- Computers for various operating systems
- Disk images for various operating systems
- Virtualisation – many environments in use at the same time
- Global test computer cloud – immediate deployment
- Test equipment from manufacturer
- Test equipment from testing company
- Use of emulators
- Public beta testing
- Replace environment with stubs
- Replace environment with copy
- Replace environment with simulating mocks

Points: Test environments are getting virtualised
Test automation technologies

- Tailored testing languages
- Capture & replay tools
- Keyphrase-driven testing
- Era of easy test specification (e.g. Robot Framework)
- Generic and standardised testing languages
- Data driven testing
- Synthesis of UI testing and distributed multi-language code-based testing (Selenium)
- Model based protocol testing
- Model based UI testing
- Ad-hoc unit testing
- Unit testing frameworks
- xUnit
- Building programs
- Programmatic code inspection
- Tools for continuous integration testing

Points: Test automation will become more ordinary all the time to everybody.
SUT actuation

- User interface API
  - Manual
  - Internal API
- Machine vision and use of robots
  - Next? Control by thought?
Megatrend: Open systems

• How is it shown:
  – Excellent free source applications (and dual licensed a little worse ones)
  – Open culture has promoted openness of closed software too
  – Open interfaces and standards – great thing to testing
  – Opens licences of documents facilitate reuse of information
  – Public and private sector will require and promote openness
• Think:
  – How will this have effect in future?
  – How could we utilise it?
  – What drawbacks will it have?
Conclusions

- Testing has developed to a positive direction even though times will come necessarily be times when things regress somewhat
- Testing has diversified, tools and methods have evolved
- Things are understood from many viewpoints and on many abstraction levels
- It is extremely rare that in some area of operation reforms are mainly positive – but testing is such
- Doesn’t the future look bright?