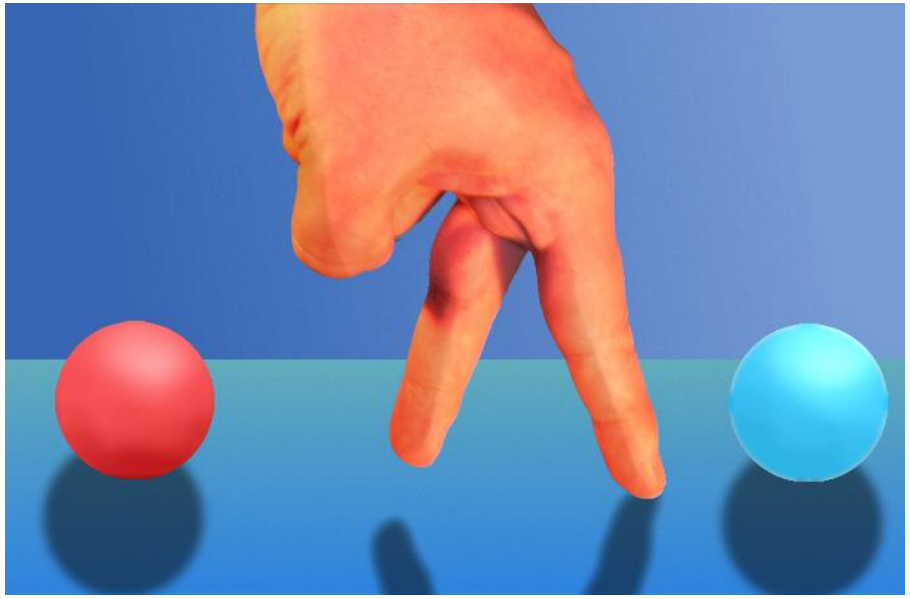
Trajectories of testing – checkpoint 2019



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 matti.vuori@mattivuori.net

Contents 1/7

Contents 2/7

Contents 3/7

Contents 4/7

Contents 5/7

Contents 6/7

Contents 7/7

Testing develops without us noticing

- 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 a testing and software development is so diverse and different in 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

Caricatyric stages of software development 1/5

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

Caricatyric stages of software development 2/5

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

Caricatyric stages of software development 3/5

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)

Caricatyric stages of software development 4/5

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

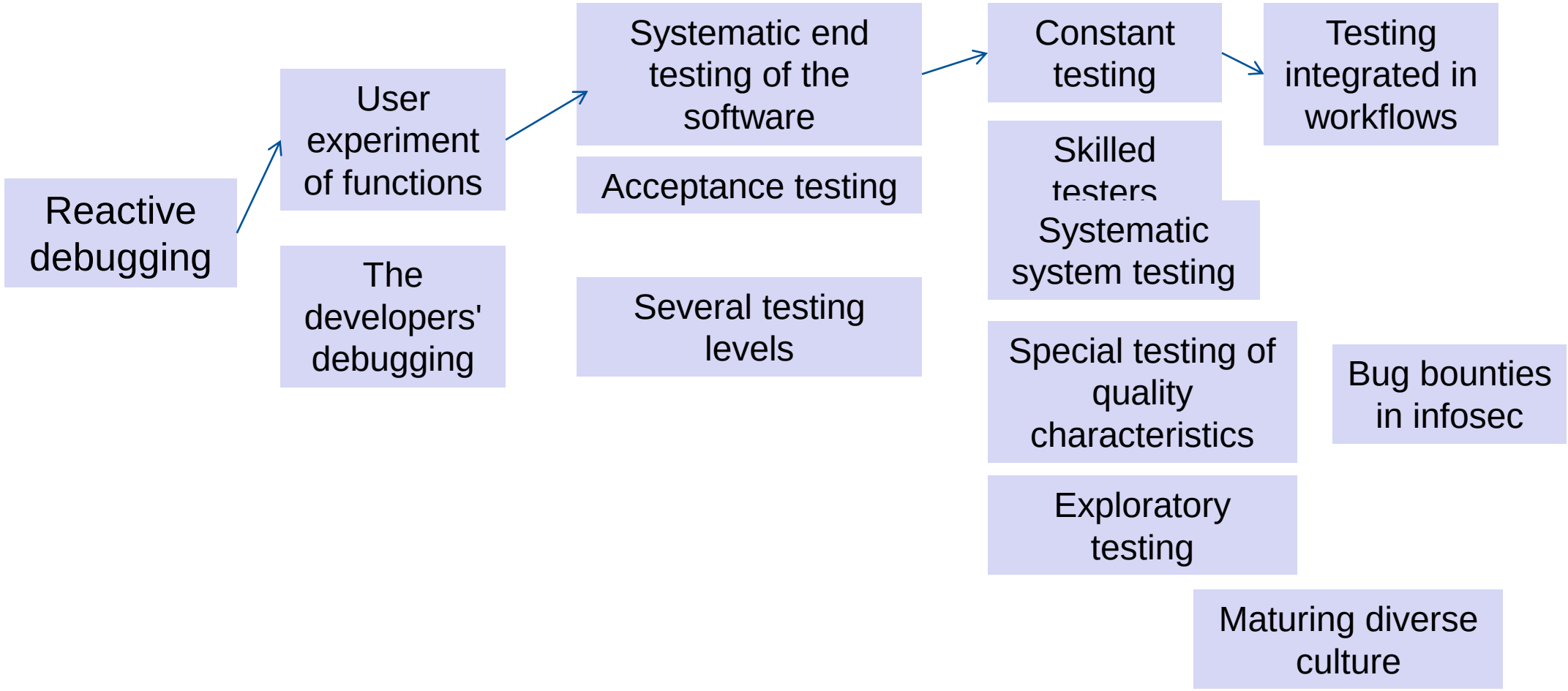
- For example it is thought that the project models of agile developing put many things anew, changing rules, preconditions, possibilities.
- Some of that is true, but some of this are just ripples in the big trends.

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

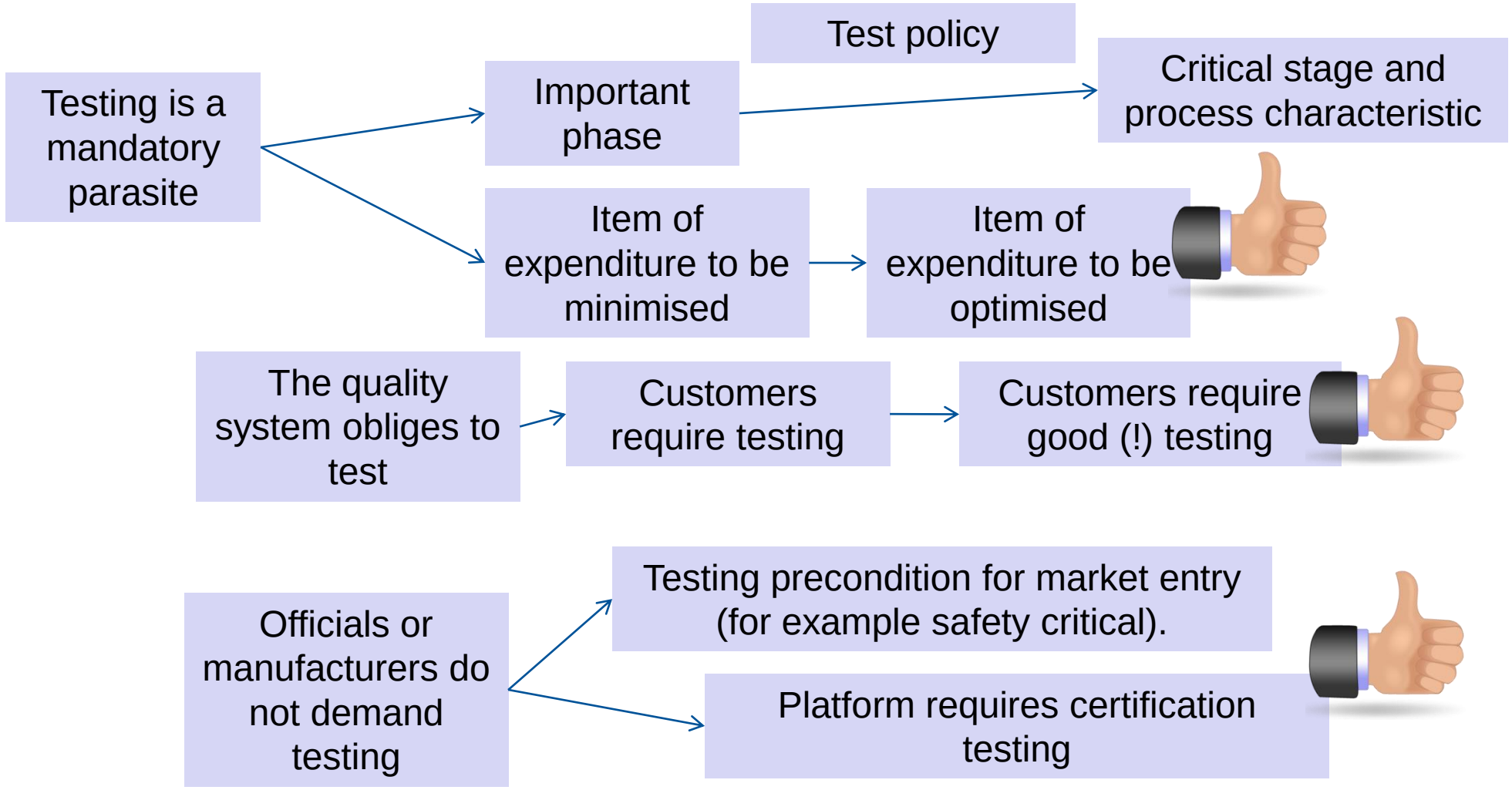


Points

Testing has become infinitely more diverse and rich!

OPERATING ENVIRONMENT AND ROLE OF TESTING

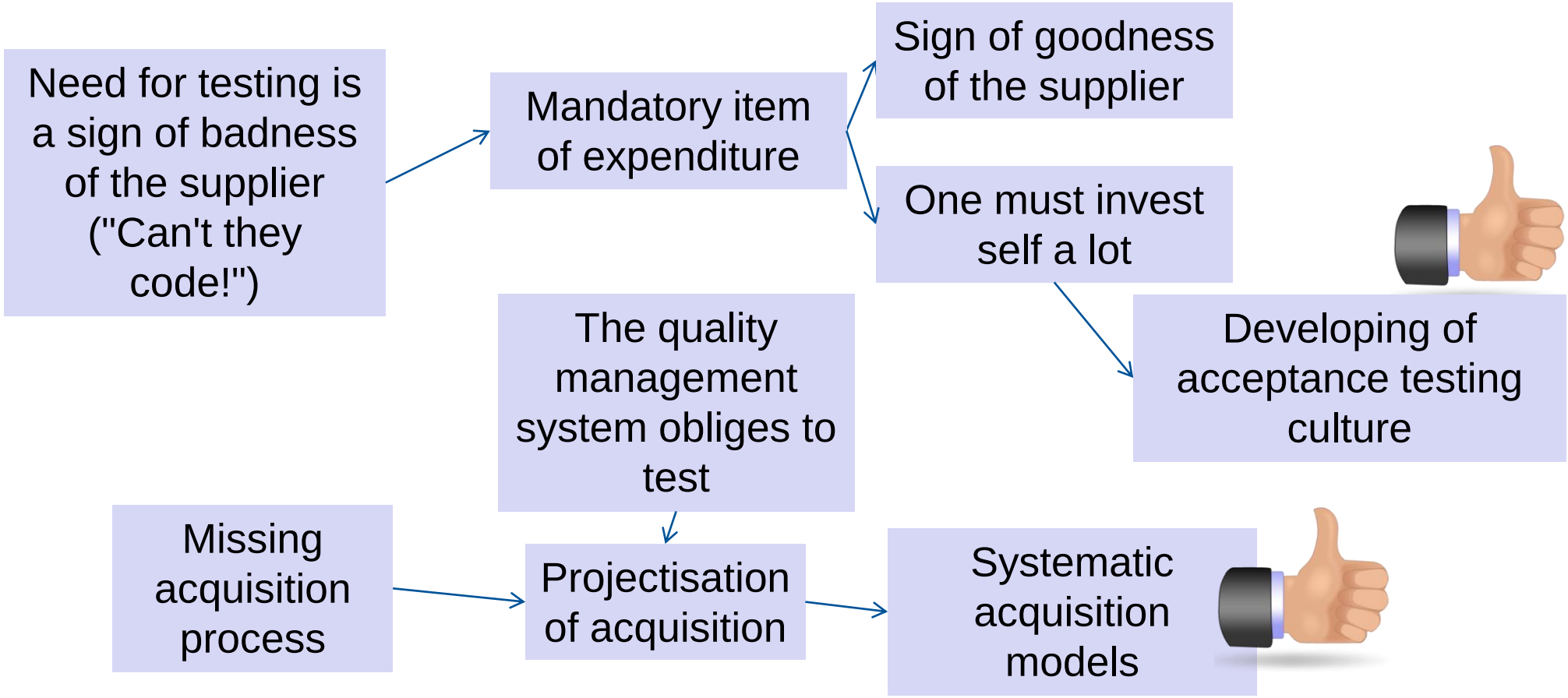
Role in the software production



Points

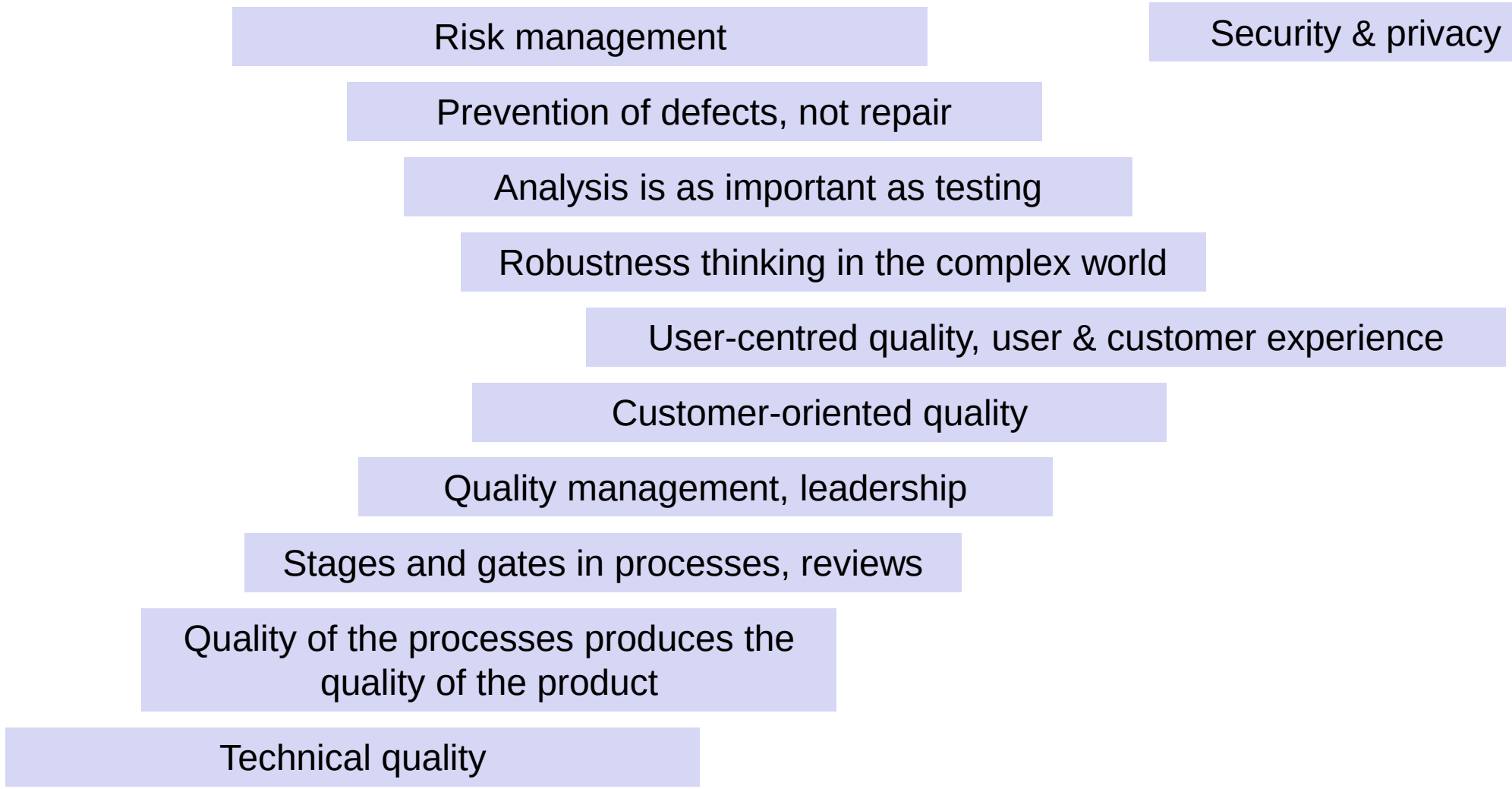
The role begins to be already generally understood. The testing does not need to be hidden from customers!

Role of testing from viewpoint of acquirer of an information system



Points
The acquirers understand the importance of testing too

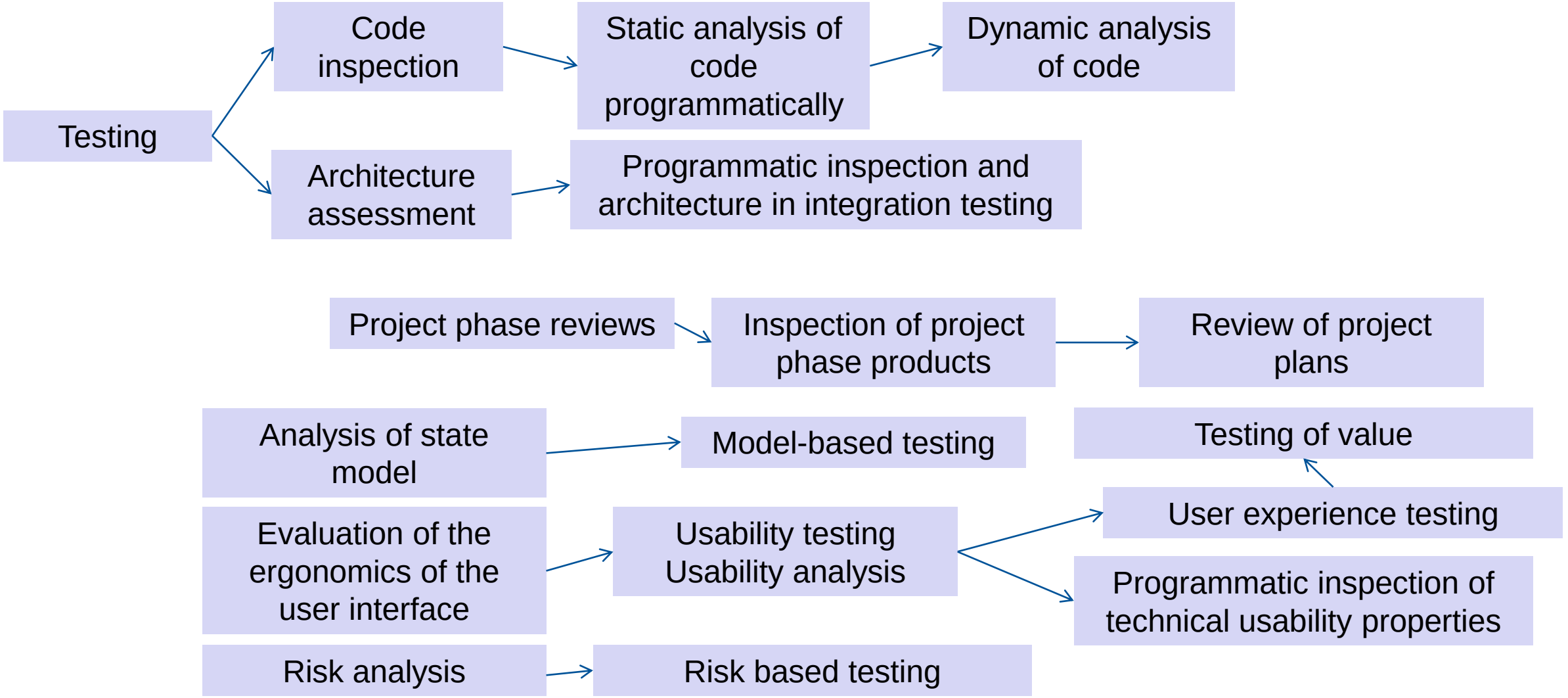
Development of the quality thinking by layers



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



Points Testing is a part of a rich whole

New "critical systems"

Safety critical systems

Business critical systems

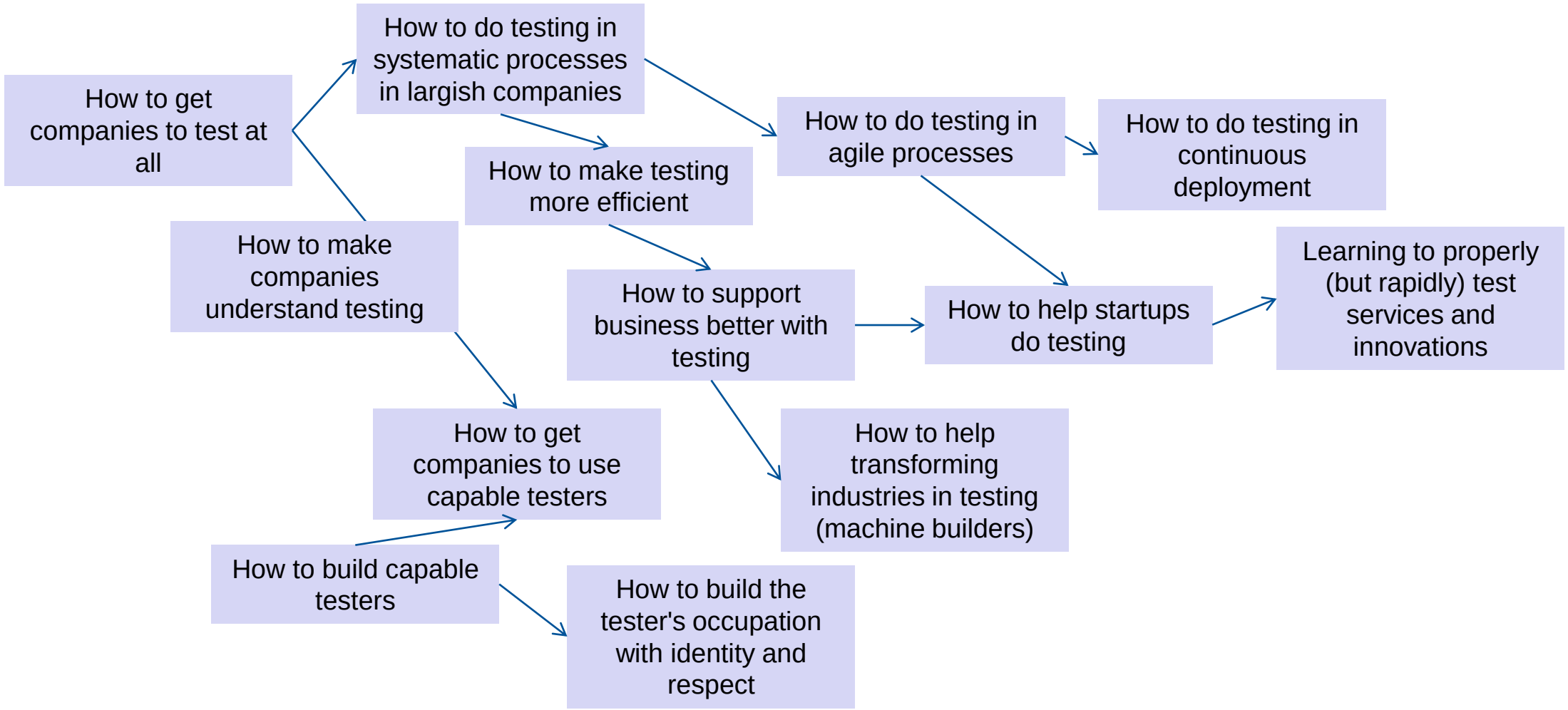
Experience critical systems

Points What is "critical" varies in contexts

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?

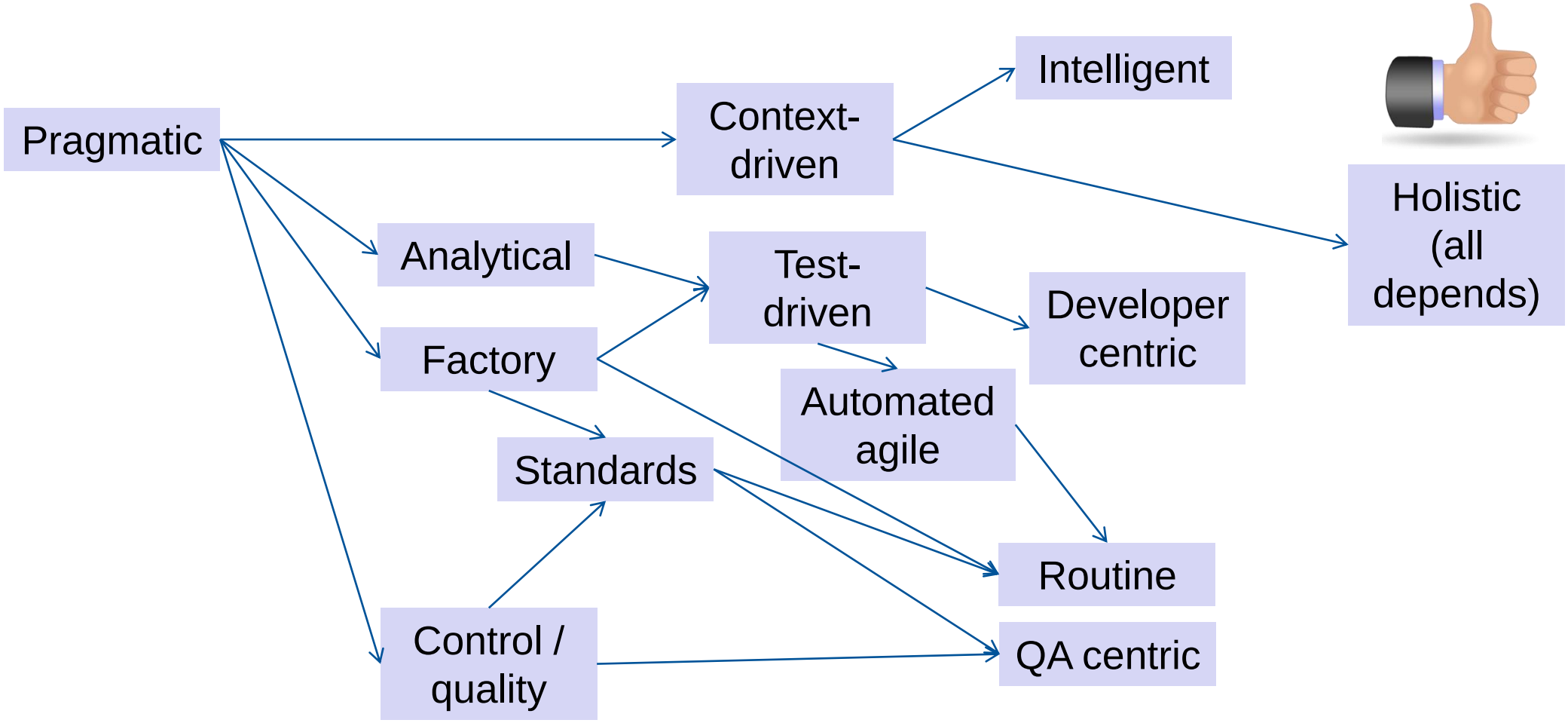
National challenges



Points Challenges change all the time – what will be the next…

NATURE OF TESTING

Schools of testing



Points

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

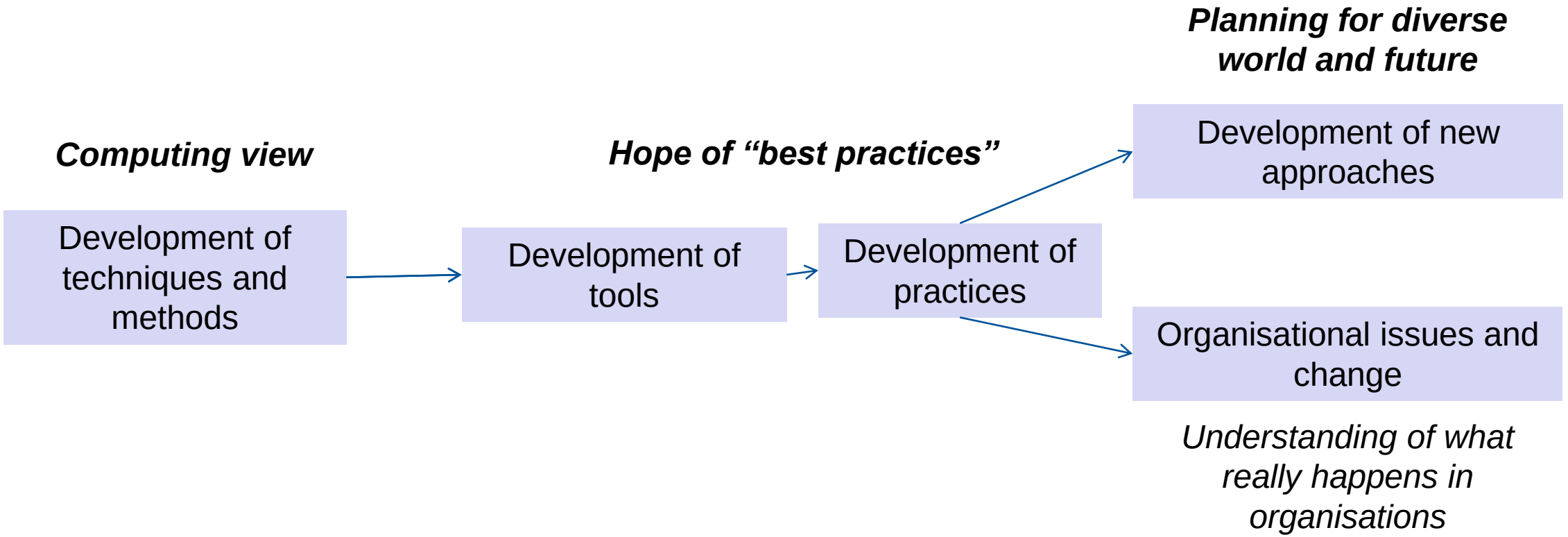
Systems used by humans

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

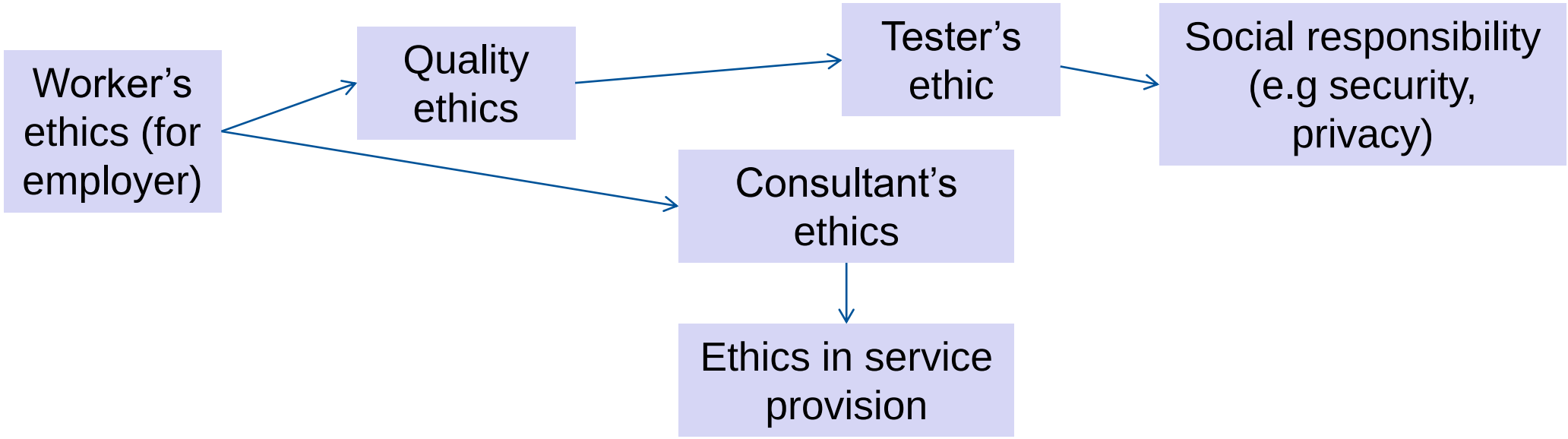
How people act

Rationally	In just about any way

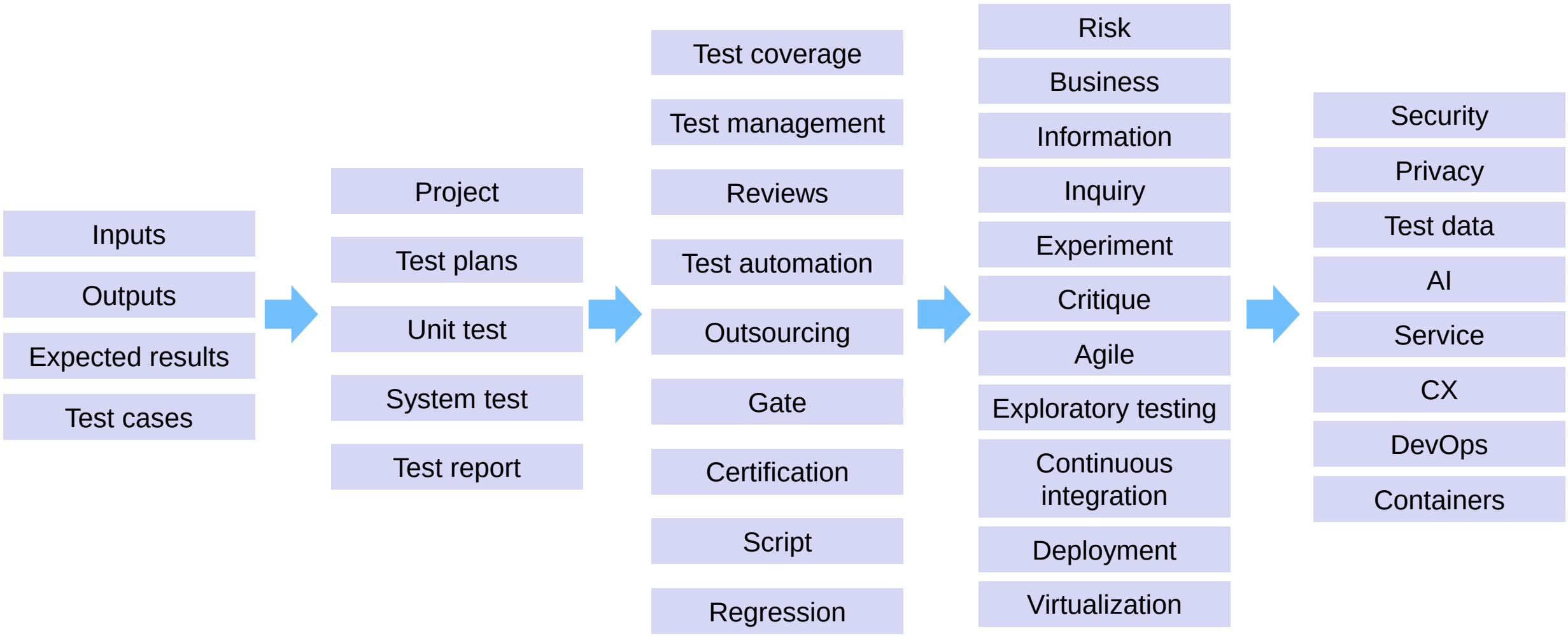
Researchers' biggest challenge



Evolution of ethics in testing



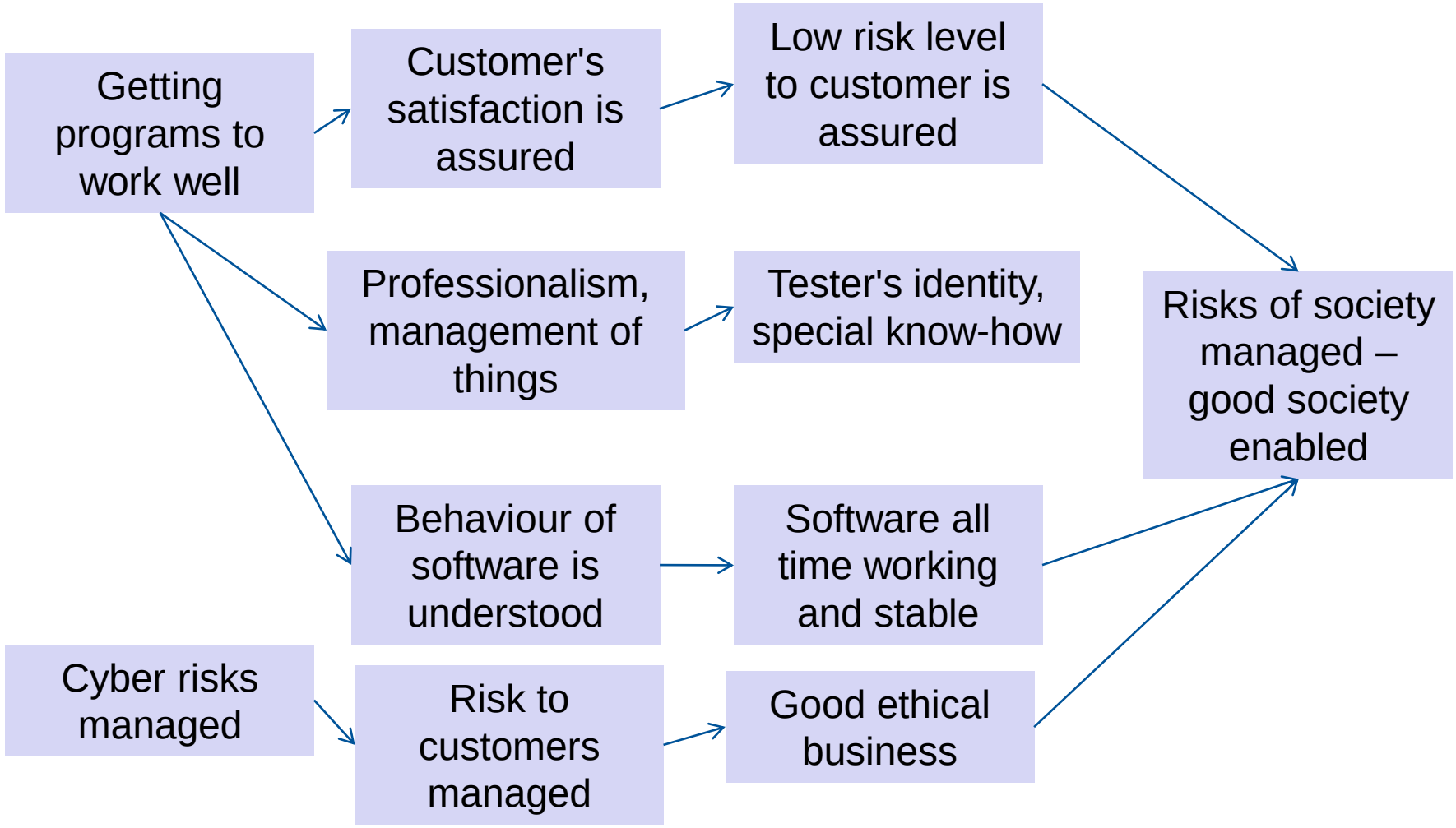
Biggest (new) terms of the trade



Points

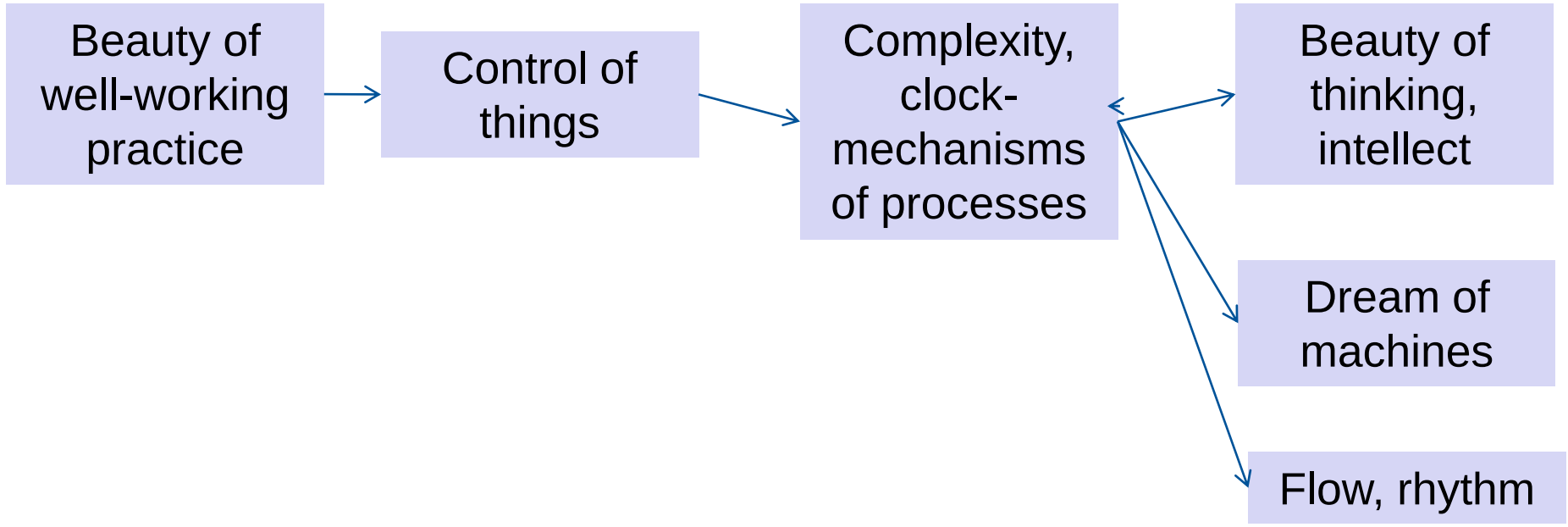
Each era brings with it a new set of terms that express new things and a change of culture

Great things in testing



Points The world changes and the great things change with it

Aesthetics of work



The worst hype of the period

Testing is not needed...

Faultless code → Validation of code → Generation of faultless programs

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

Manual testing is not needed... 100% automation

Model based testing a silver bullet

A/B testing best way to develop usability

Complete testing

Test automation

Systematic test planning

Unit testing and integration testing are enough

AI will soon do all testing

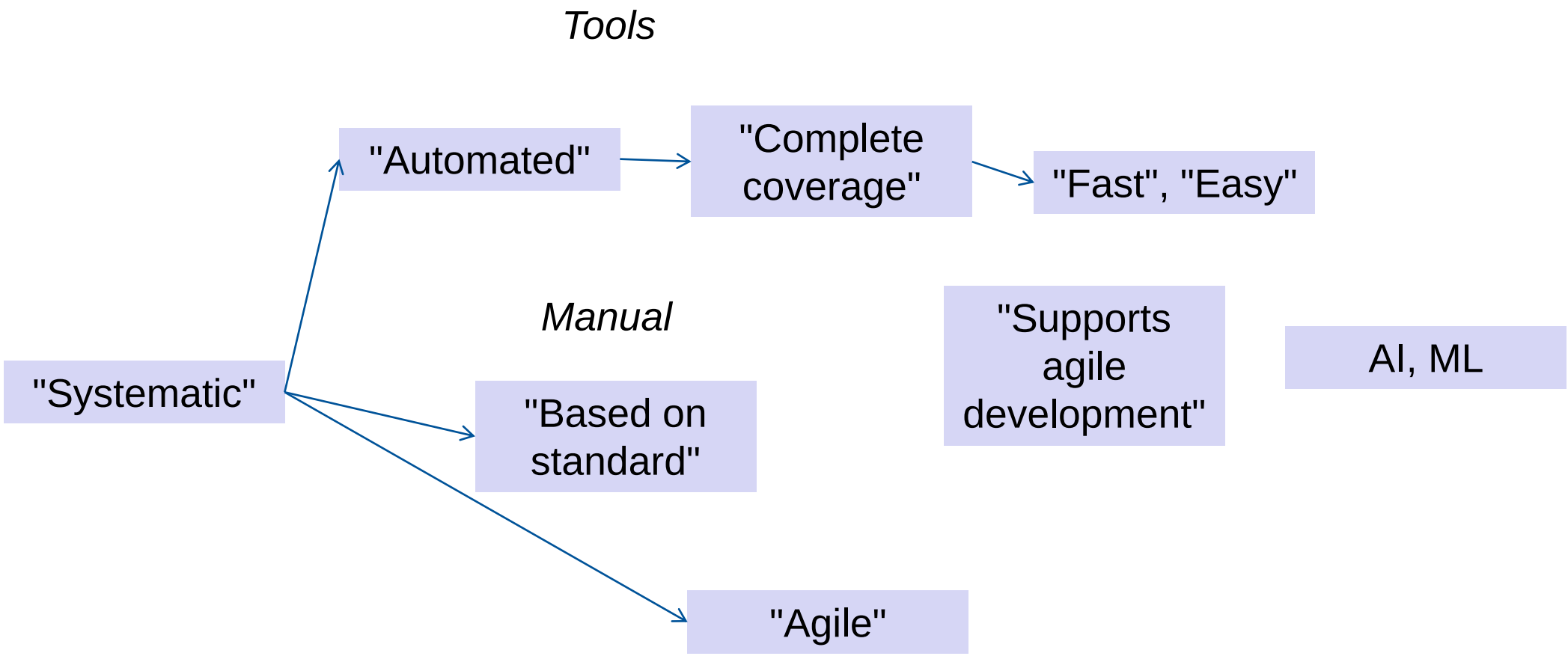
Done lightly, as needed

Only exploratory testing

Points

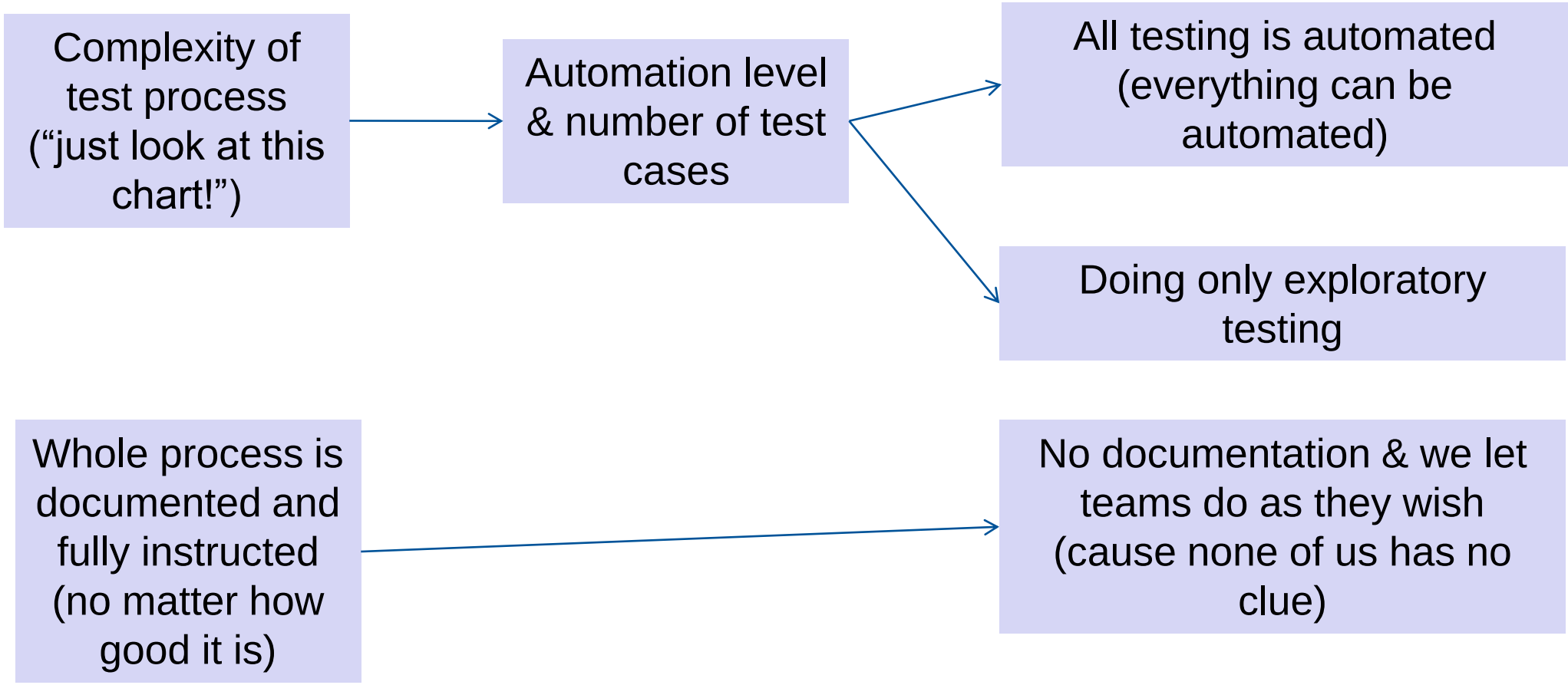
Each era has its own hype, which will be only later be understood us such

Marketing terms for testing methods



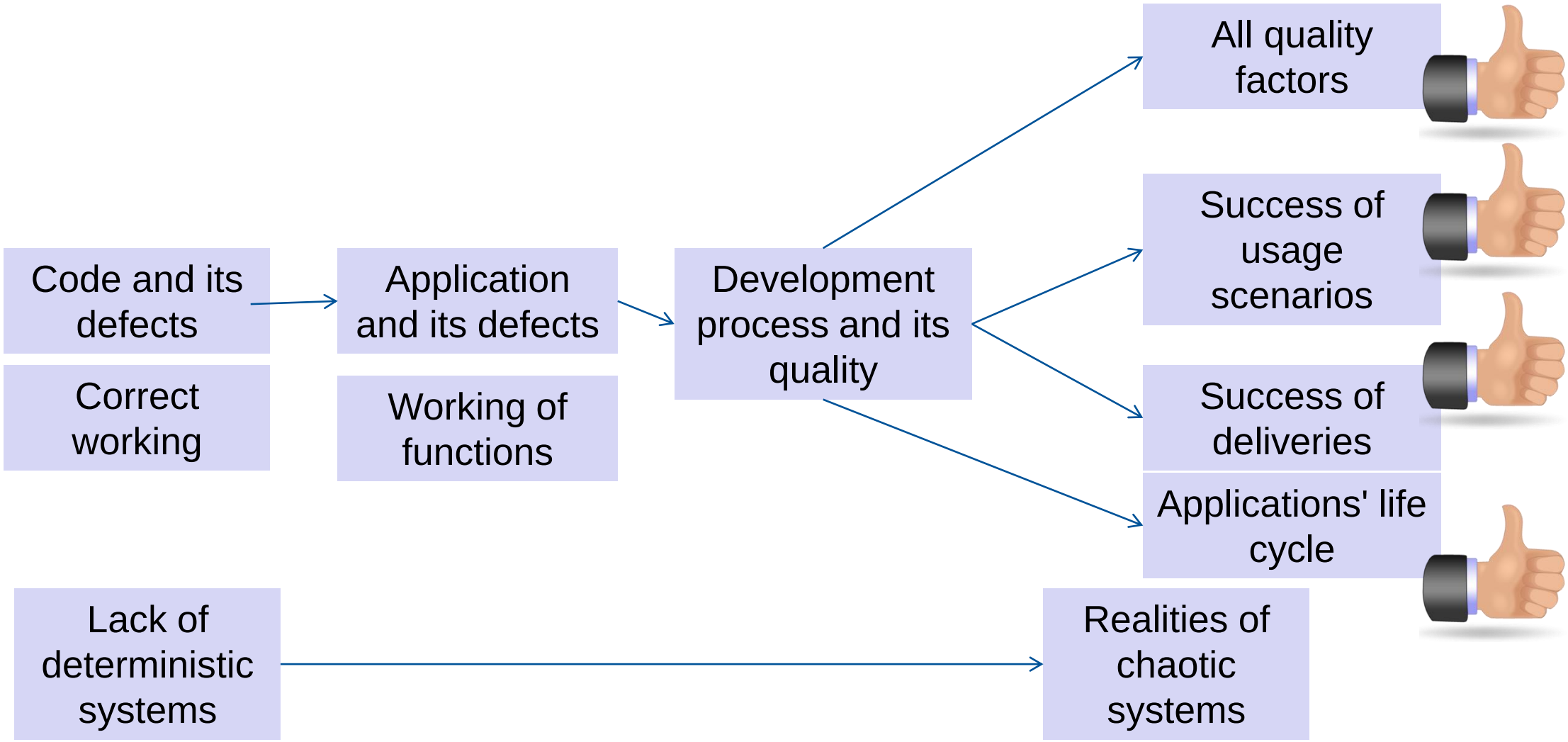
Points Marketing terms tell a lot of the current thinking

Silliest things of praise



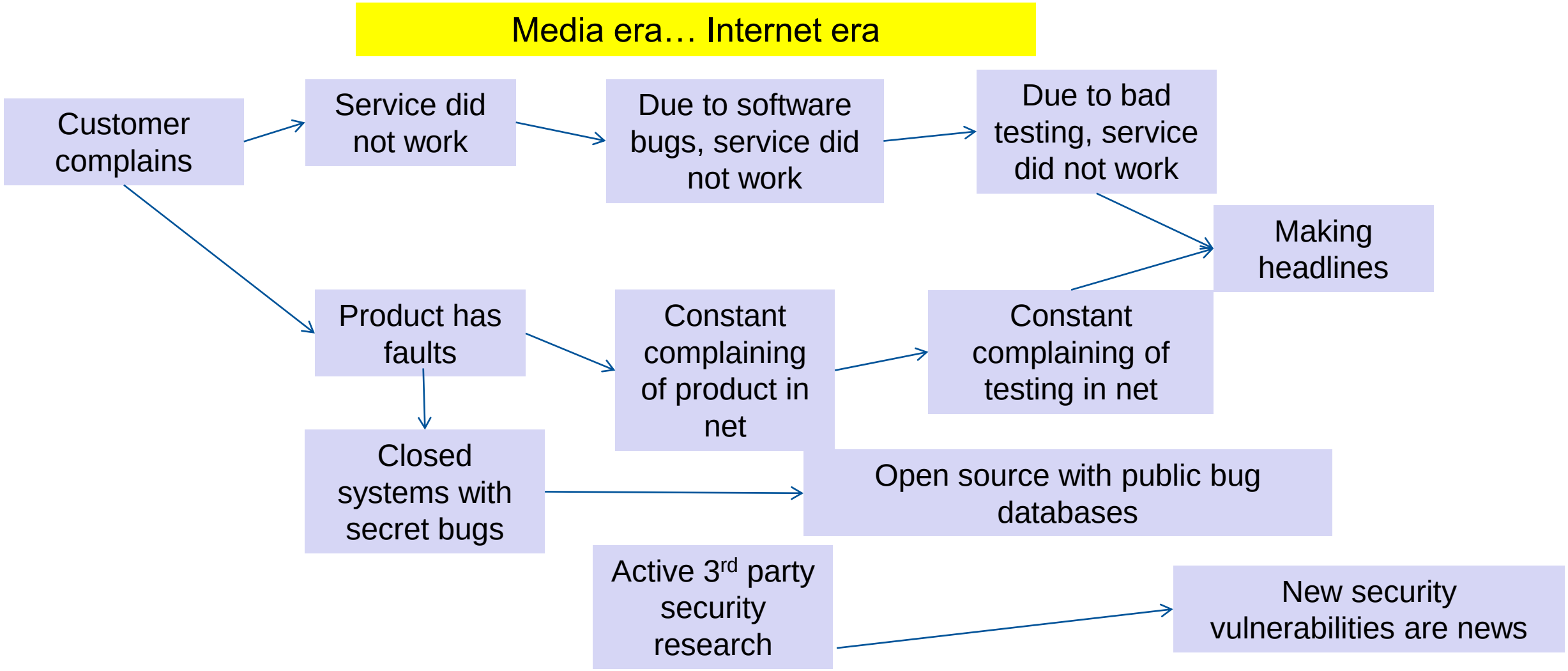
Points It is good to be proud, but preferably for the right reason

Defects and view to quality



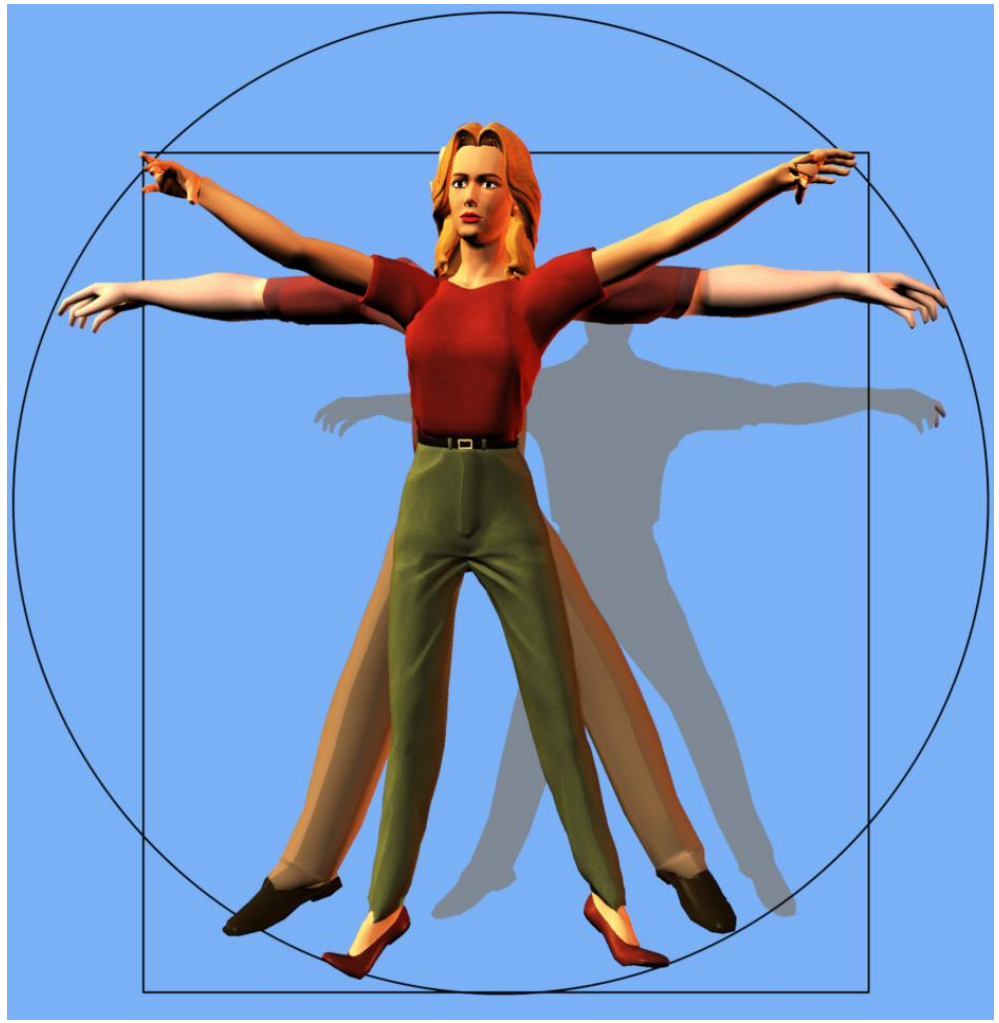
Points Here shows a learning process and hierarchy of needs

Publicity of defects and criticism of testing

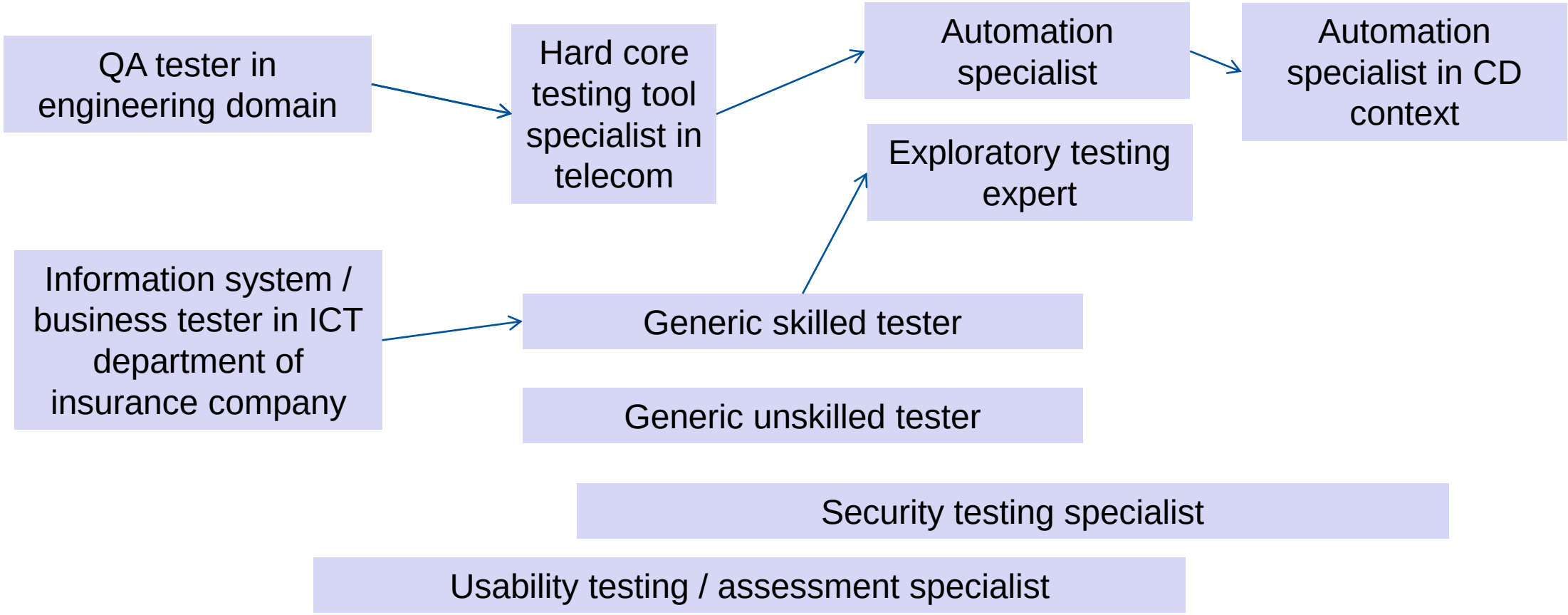


Points In the world of Internet, anyone can get in the news!

PEOPLE IN TESTING

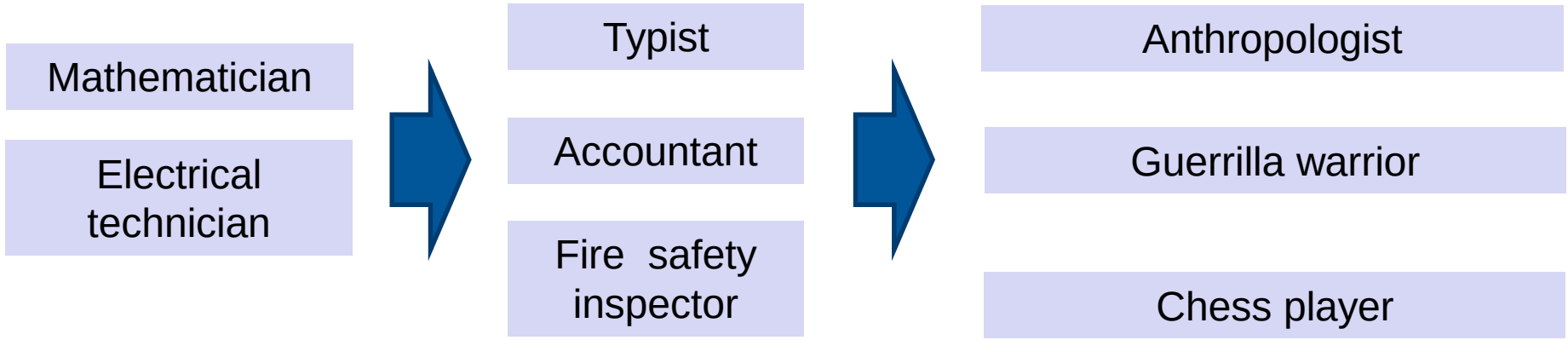


Tester’s stereotypes



If you'd hire another kind of professional to do testing, what profession is it?

Yes, this is about mapping stereotypes to analogies



Organising

”Customer tests”

Coder → Separate testing team

Coder colleagues

1 tester in team

Customer defining tests (A)

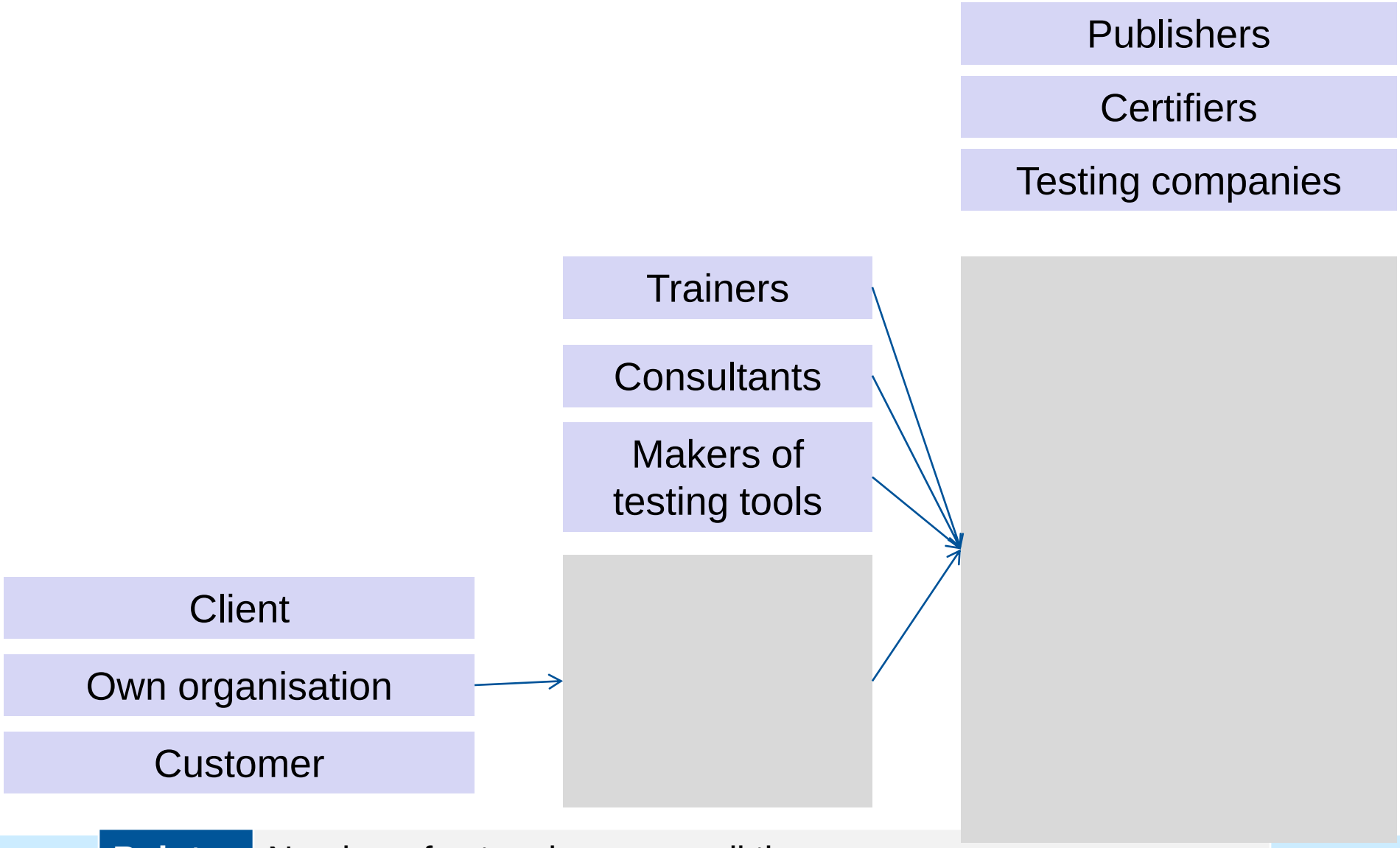
Testers in development teams → Role of developers re-emphasised

Separate testing teams

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

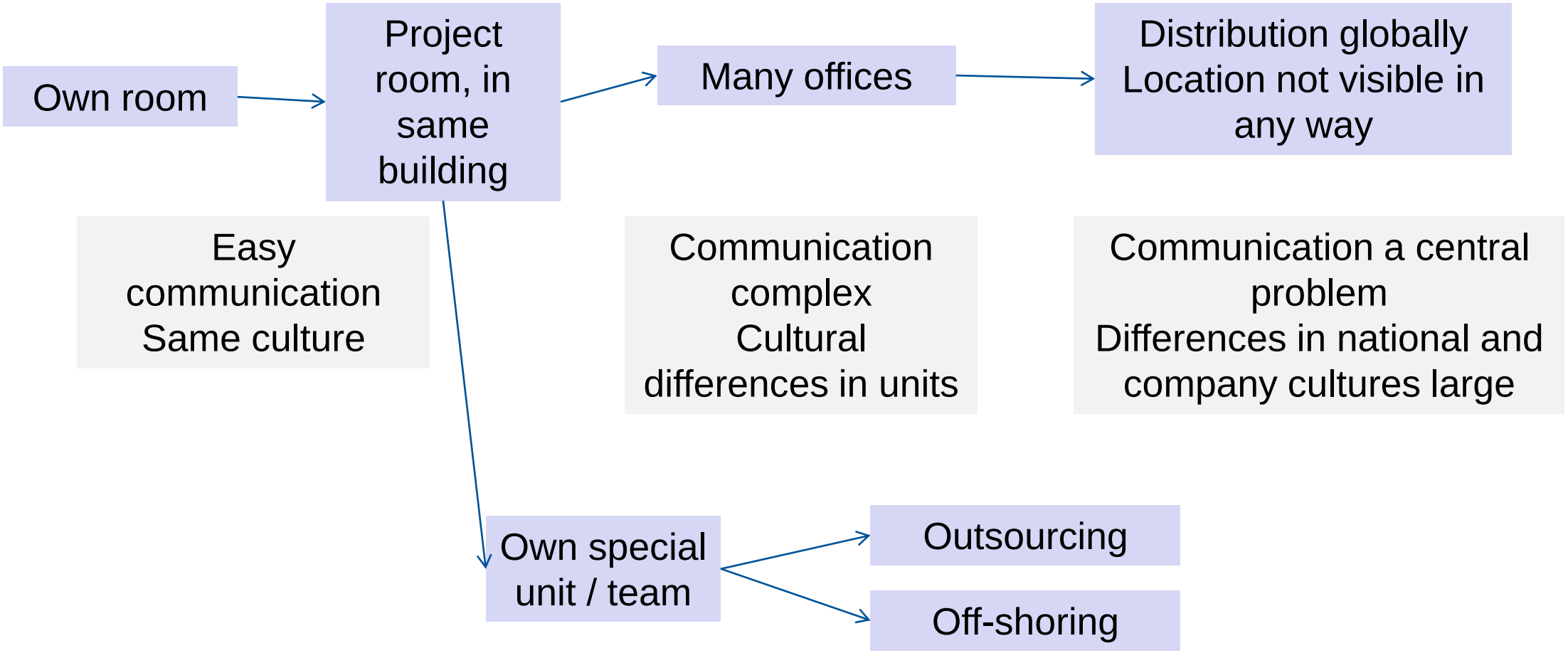
Points In organising of testing, enrichment of testing is seen

Actors in "ecosystem" of testing



Points Number of actors increases all time

Supplier of testing, locations and culture



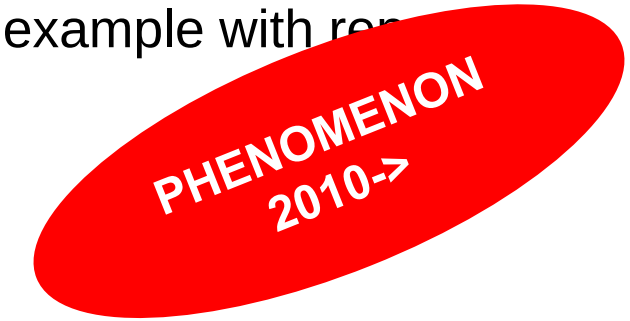
Points Outsourcing, off-shoring and decentralisation came here to stay

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?

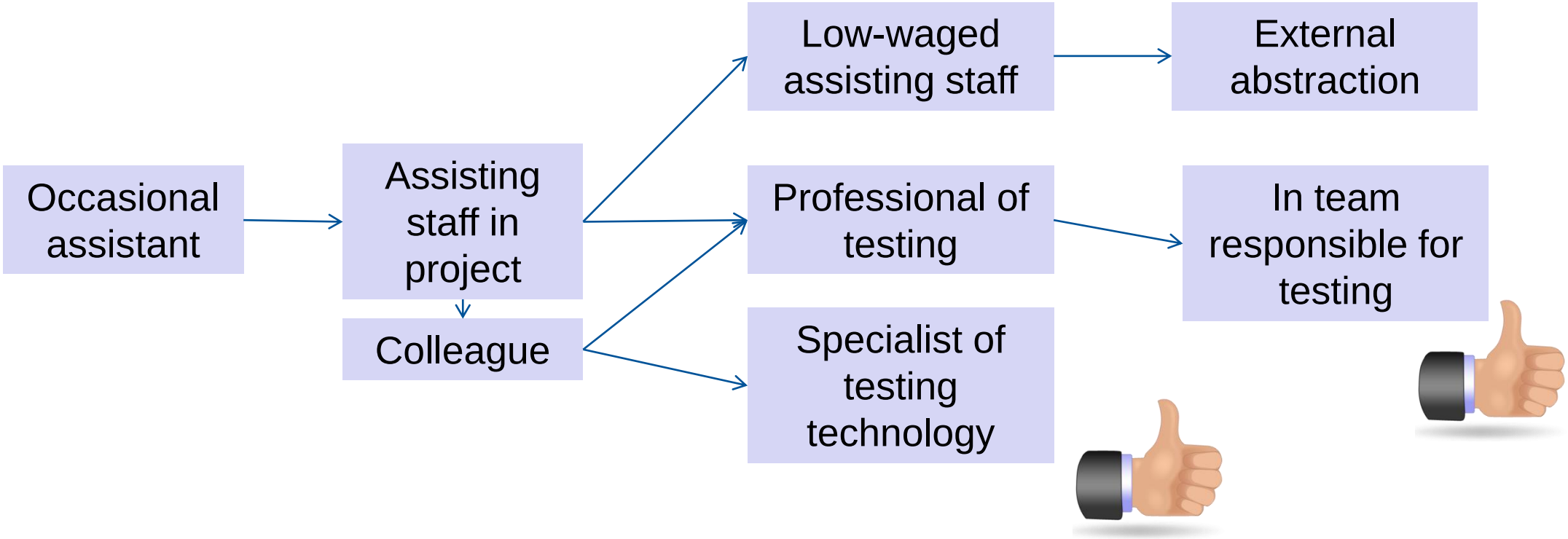
Phenomenon: Postmodern organisation

- 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 re… information systems
- Think:
  - How will this have effect in future?
  - How could we utilise it?
  - What drawbacks does it have?



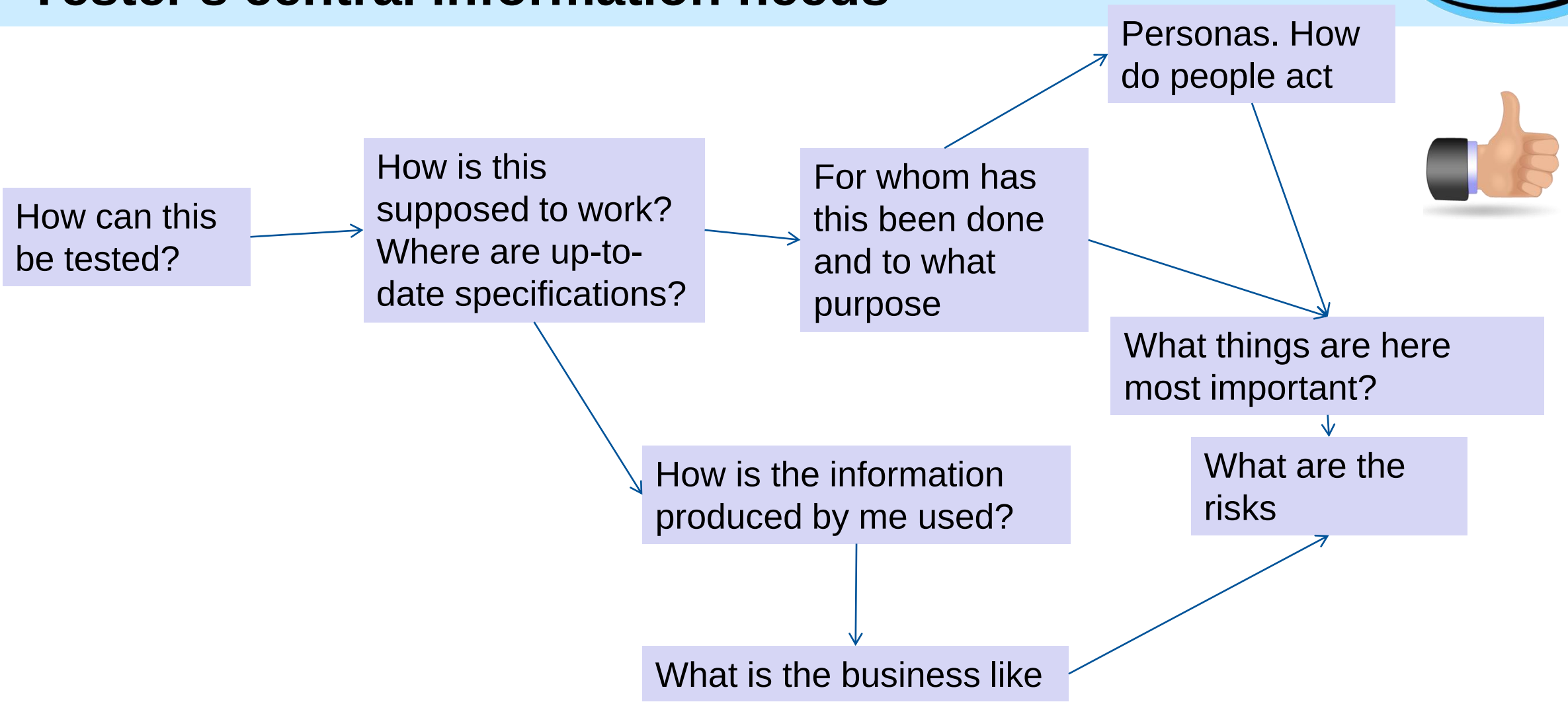
ACTORS' ROLES

Tester's role and status



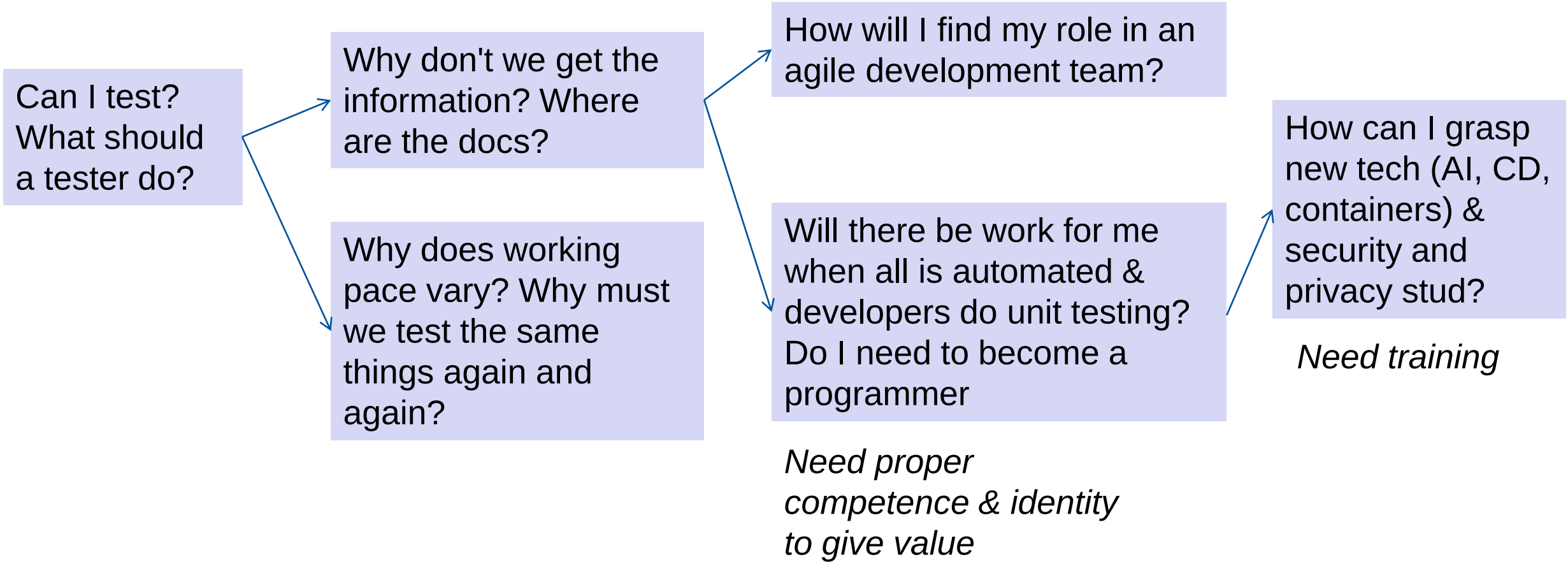
Points Status of testers improves but work must still be done for it

Tester's central information needs



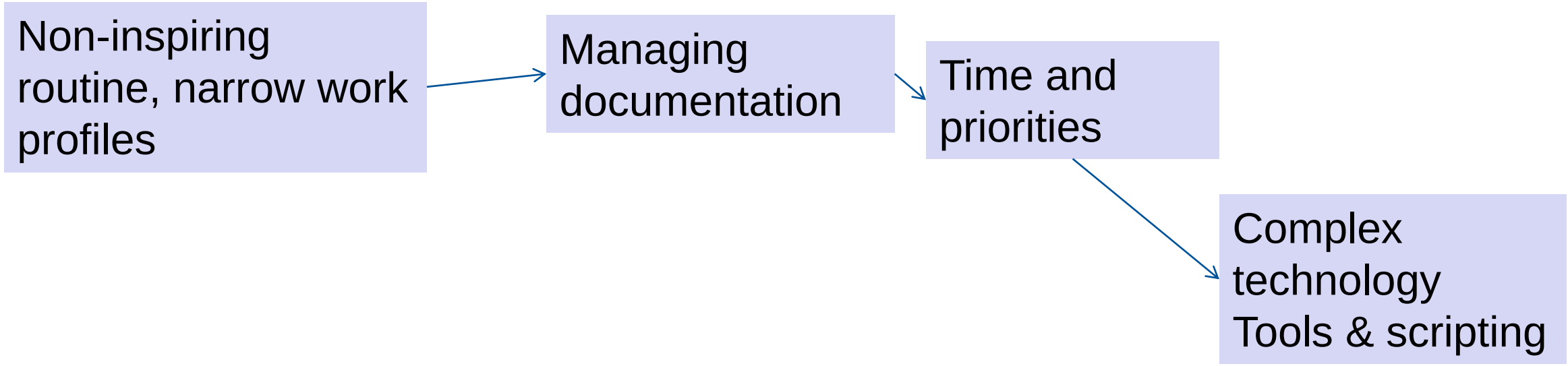
Points Information needs are linked to the information needs of others! Tester is a supporter of decision-making

Tester's fears & turn-offs

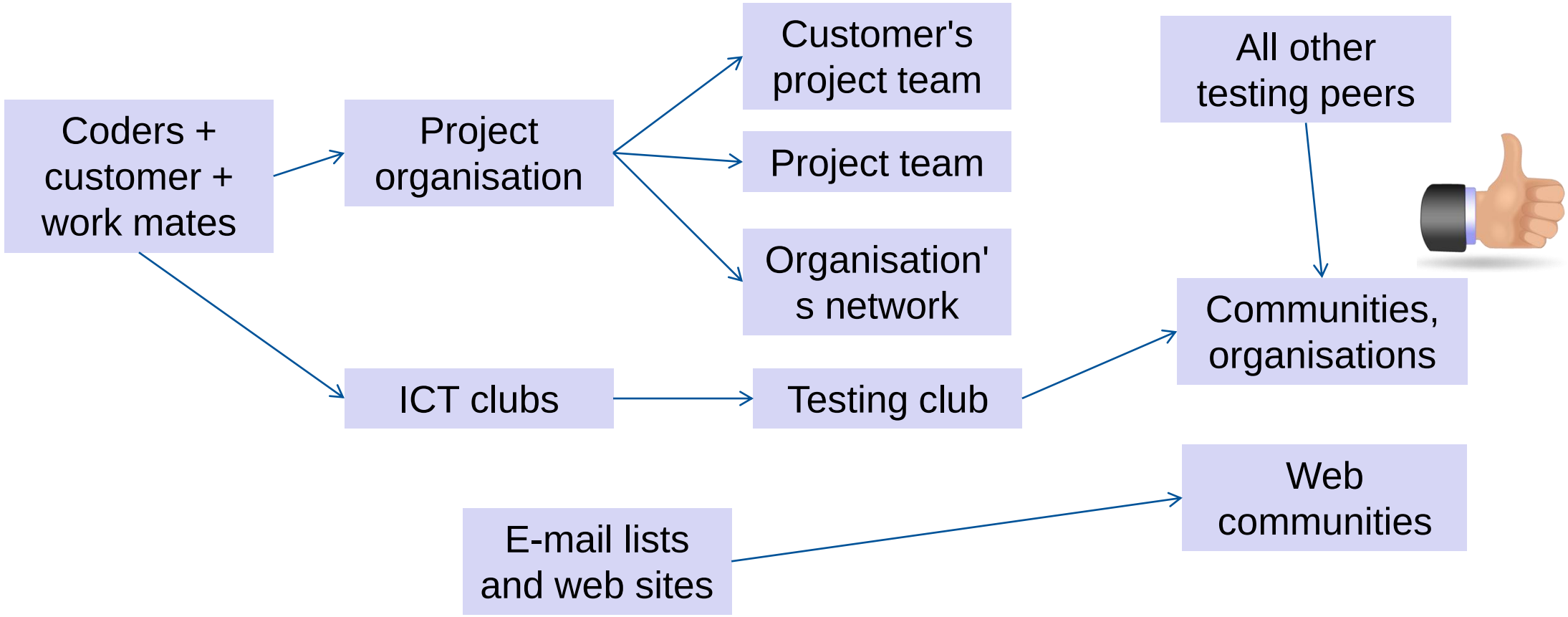


Points The tester's pressures change but why do not the decrease?

Tester's pressures



Tester's communities

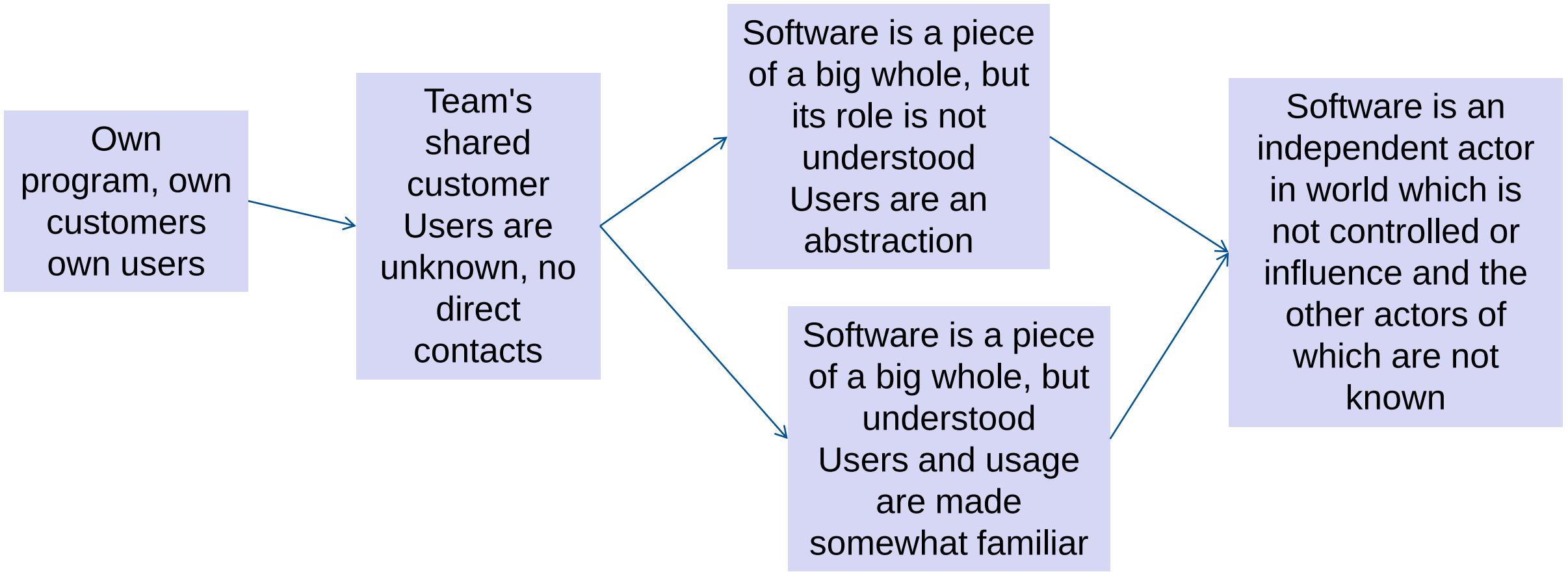


Points Challenge: how are communities used and how people act in them

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



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
- mation

Points It is great that coders include testing to their tasks

COMPETENCES

View on characteristics of good tester

Analytic
Careful

Diligent
Representing users

Being familiar with testing
Active
Adaptable
Well communicating

Technology expert (test systems)
Knows product technology

Team player
Takes her own role
Makes iniatives
Proactive helper
Fast
Has different working styles

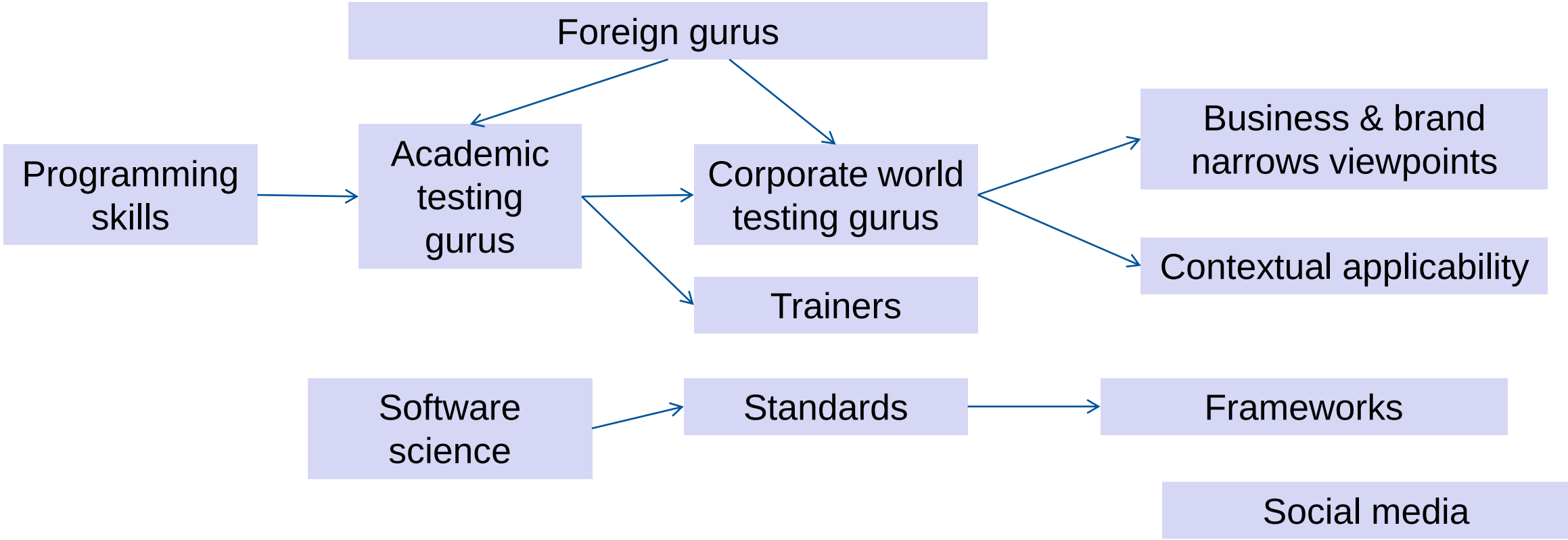
Can program or at least script

Smart → Experienced → Certified

Points

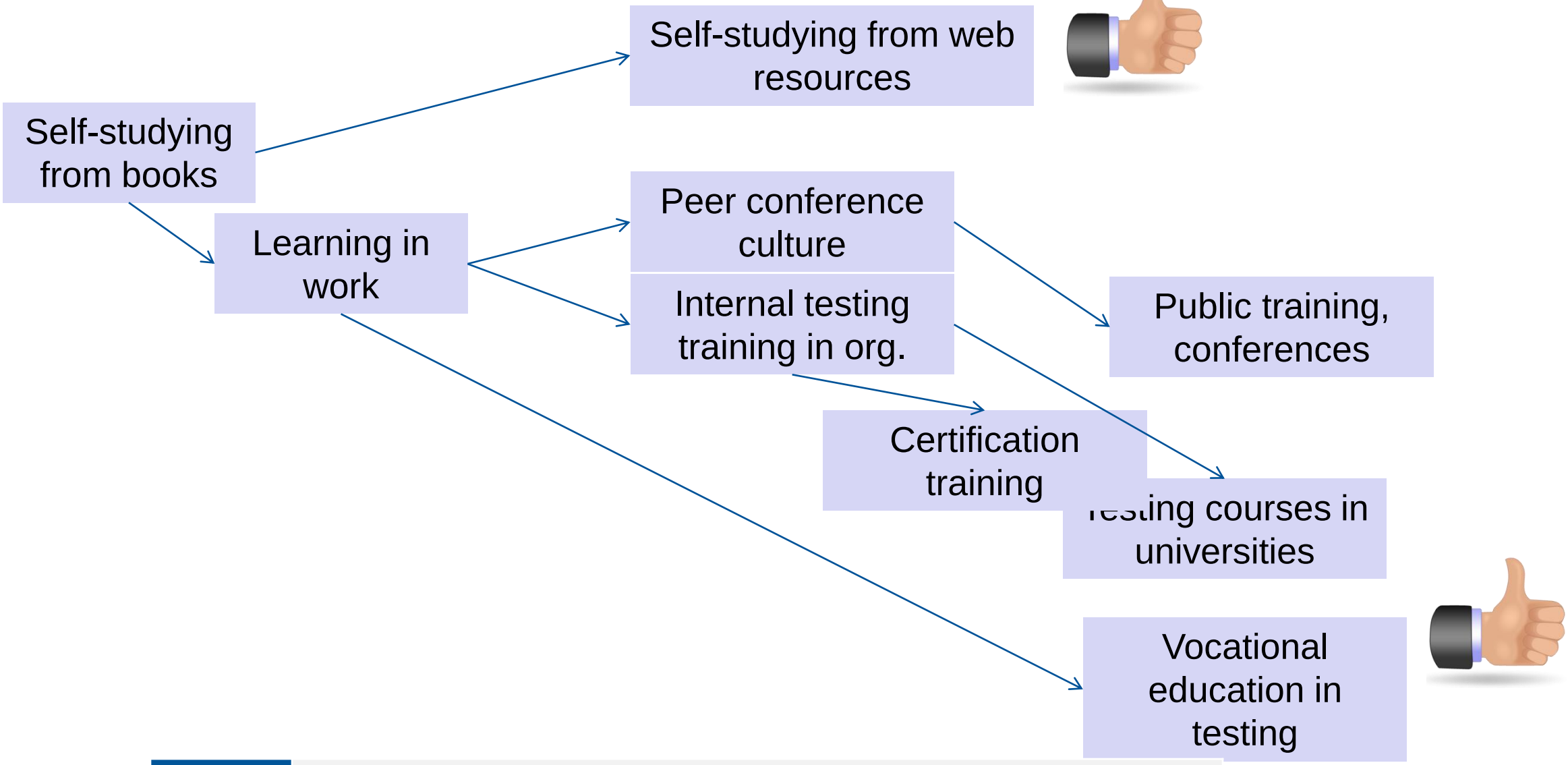
The view about the key competences changes all the time – we must not develop yesterday's skills

Testing wisdom



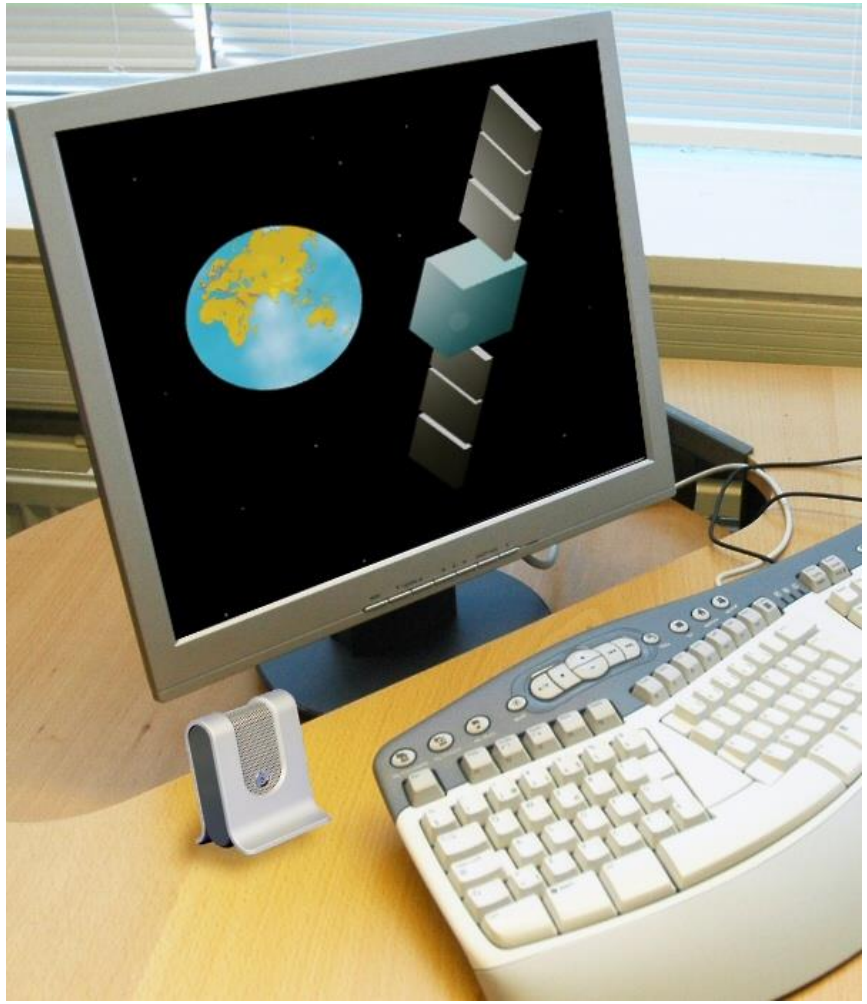
Points One must think very carefully who to listen to...

Testing training

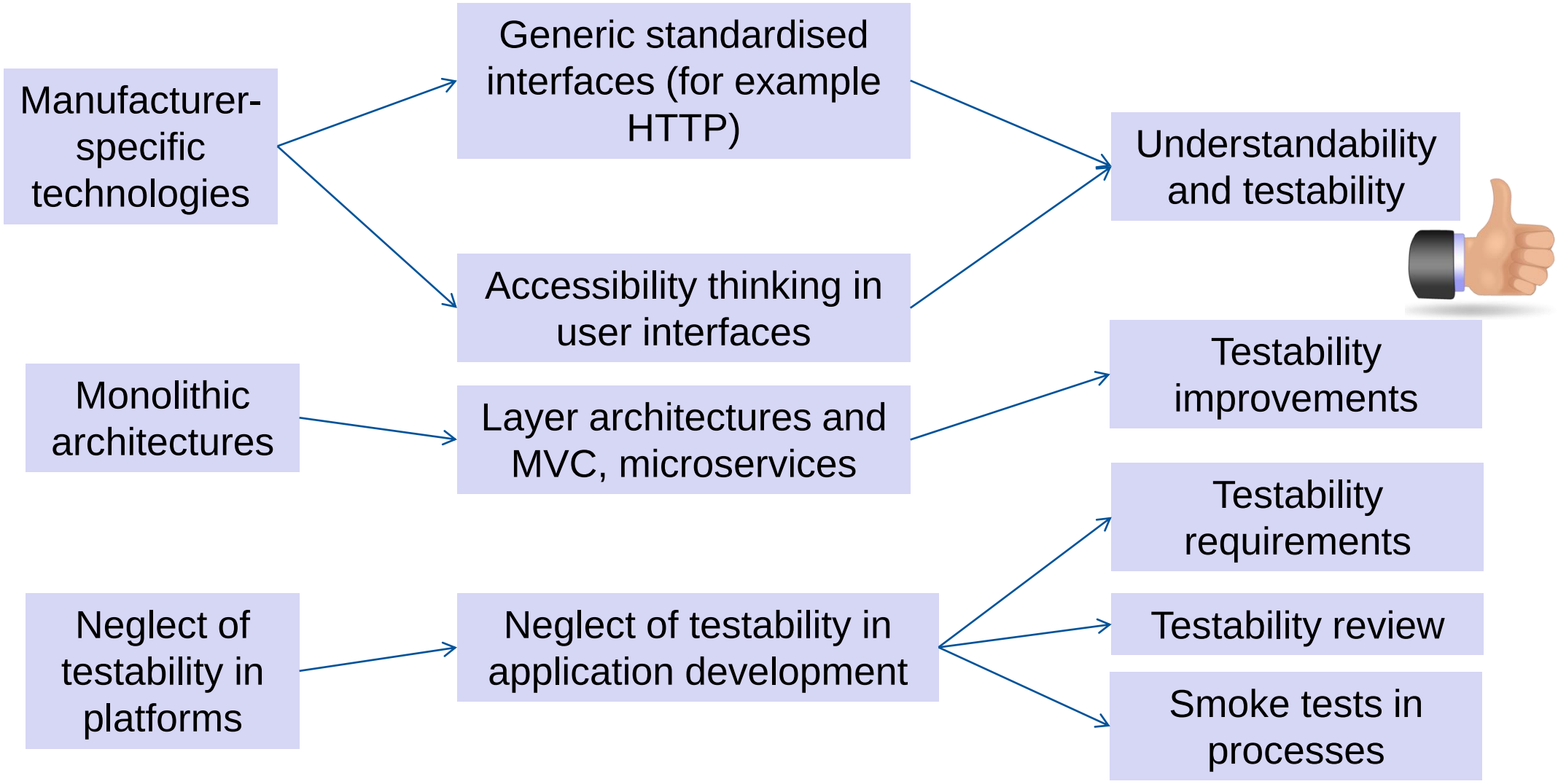


Points There are many kinds of training and the every year some more

SYSTEMS UNDER TEST



Technologies and testability



Points Possibilities of testability are good even if systems are complex

New intelligence

Big data

Image and signal processing

Simple logic in simple systems

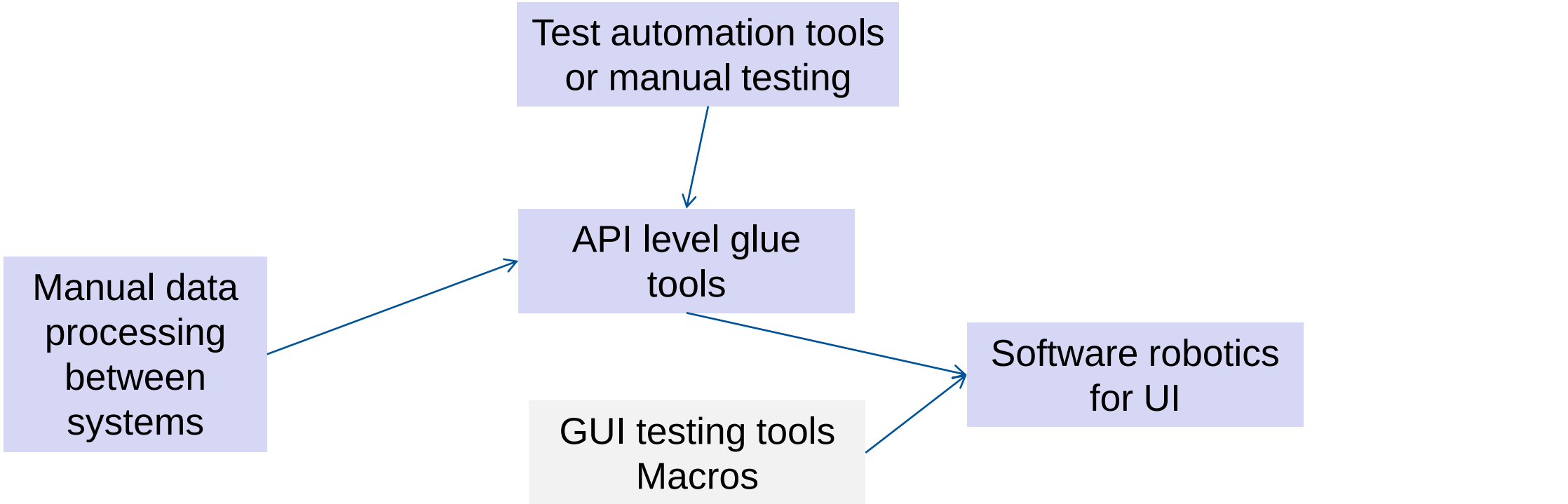
Complex and changing logic in complex systems and systems of systems

Machine learning (AI)

Need to understand uncertainty, data, deviations…

Points AI era brings new challenges…

Testing software robotics in offices



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

Demands raise radically for robustness, security

Part system of systems

End to end testing of processes challenging

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

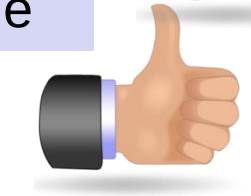
Points If testing does not improve strongly every year, quality will get worse!

Open technologies

Myth: Openness as such improves code

Open source code → Openness makes all kinds of testing possible for anyone

Undocumented proprietary interfaces → Use of open standards



Generic testing tools

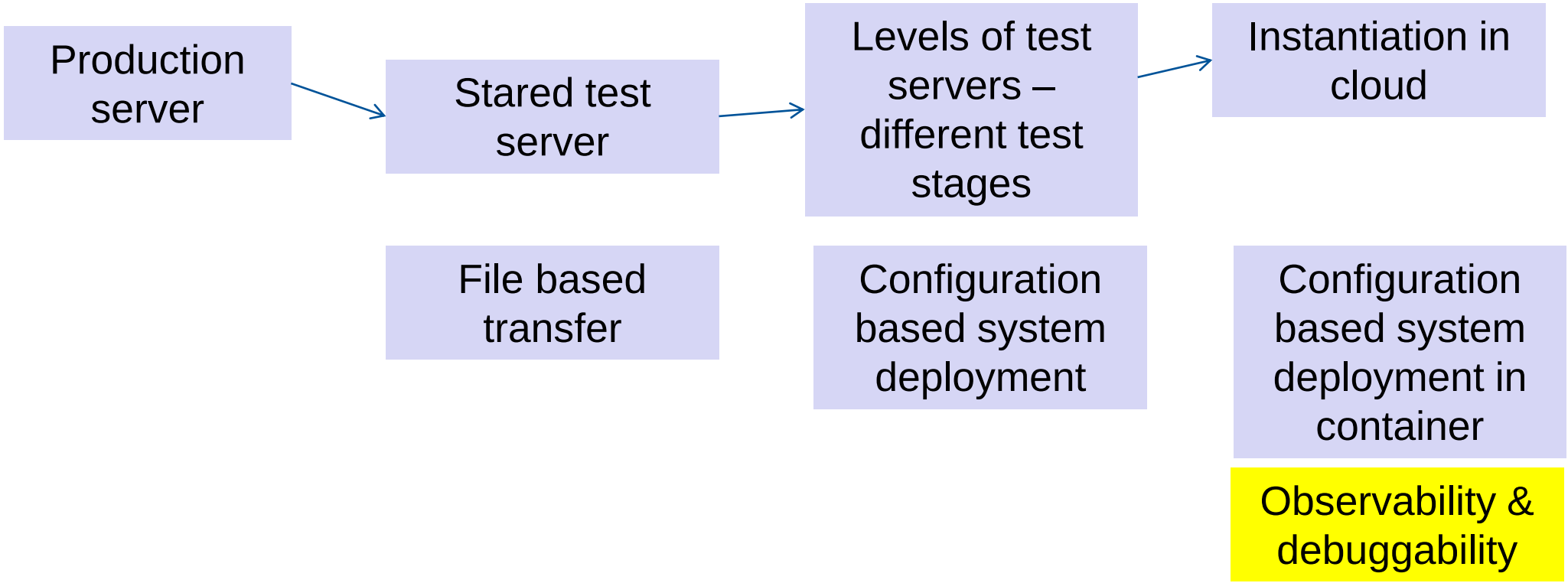
Closed testing tools → Open testing tools → Tools available for all

Closed operating systems in all kinds of environments and products → Open operating systems (= Linux) & their tools becoming common

Integration of testing tools with systems is easier

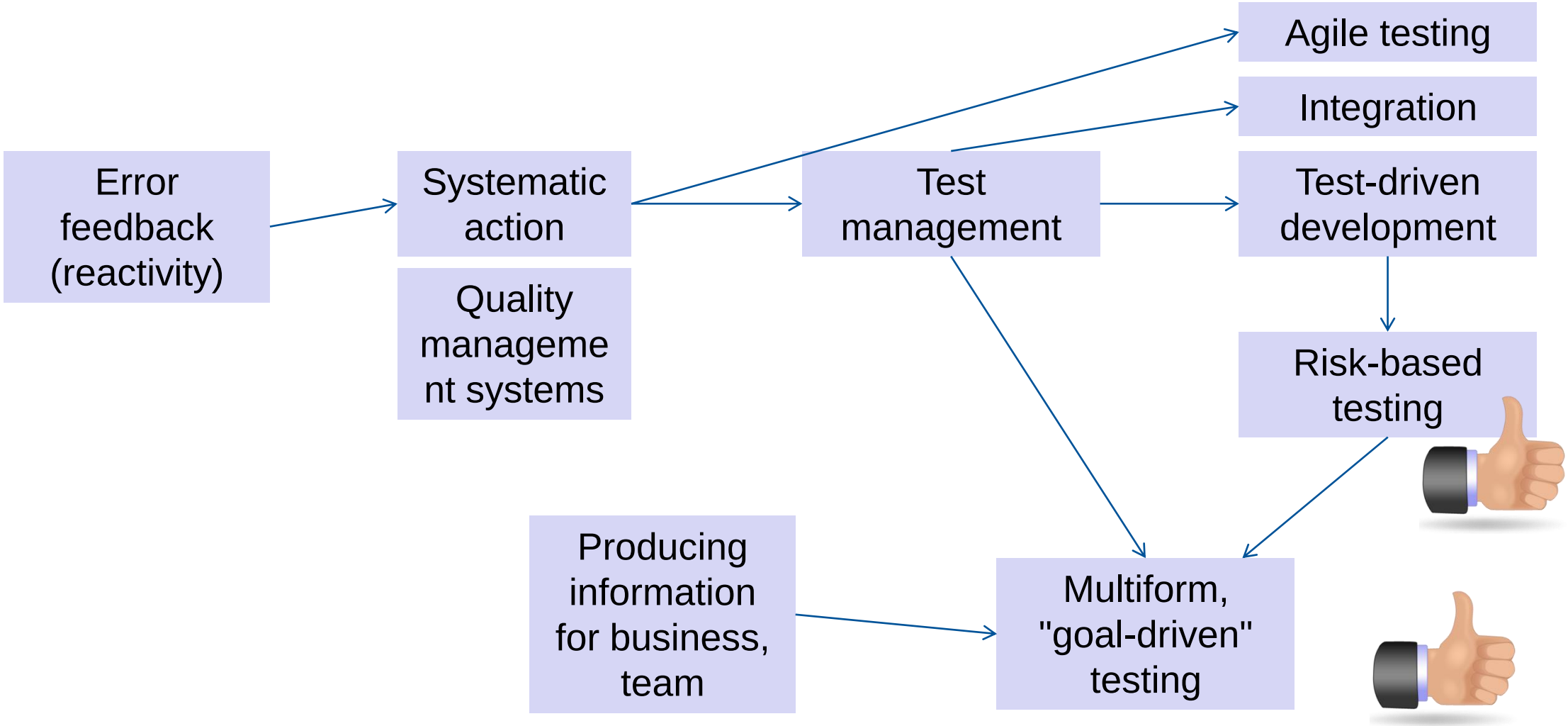
Points Openness is an advantage to testers for many reasons

Delivery for managed testing



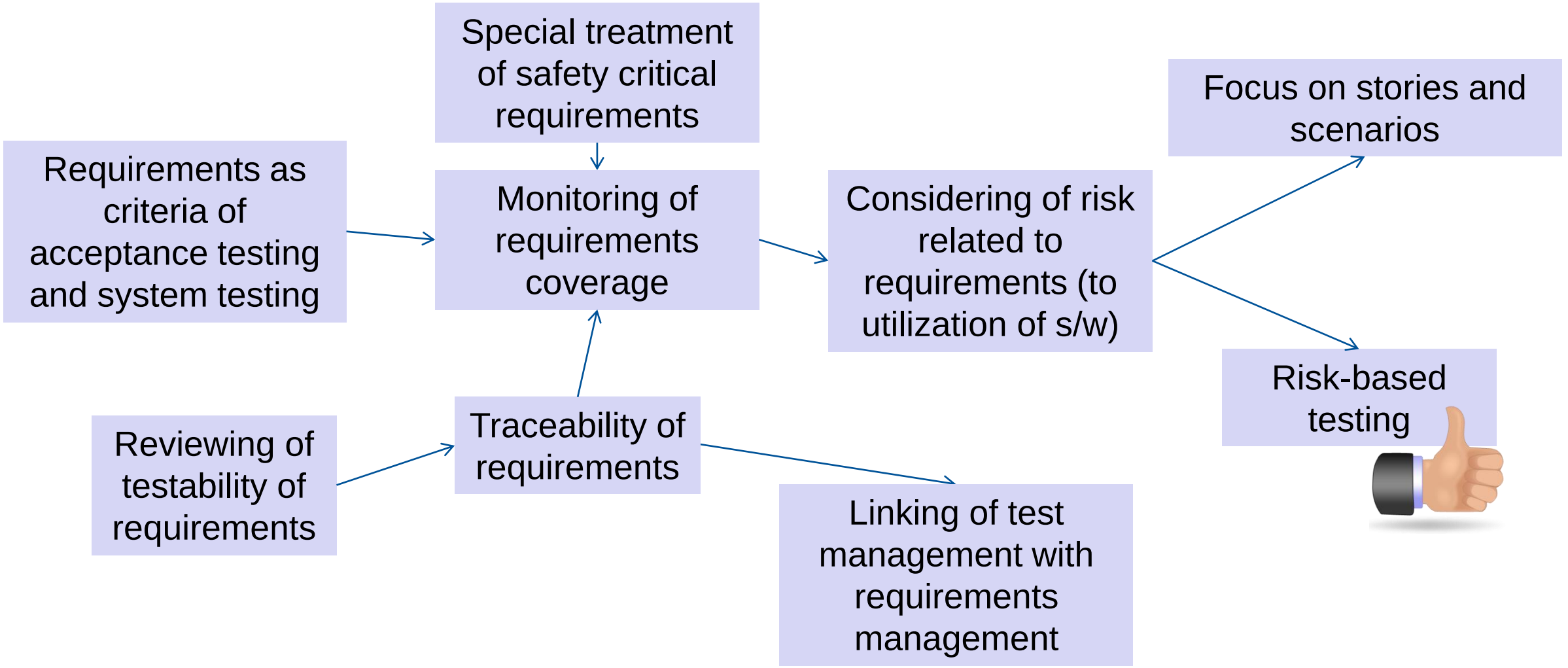
TESTING STRATEGIES

Approach



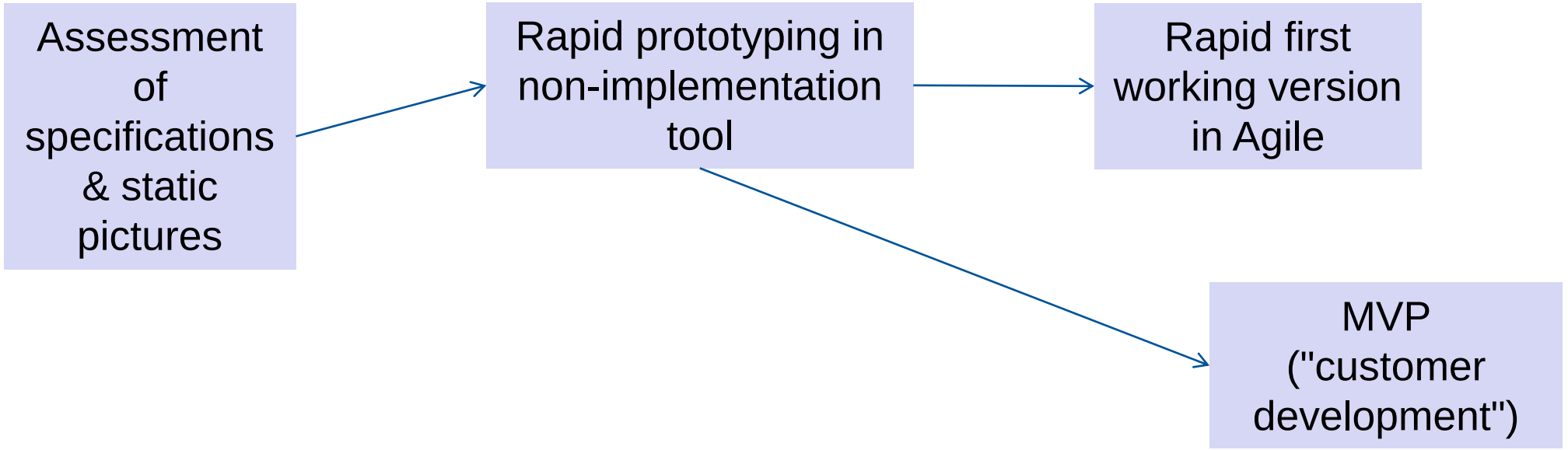
Points Approaches have become diverse – horses for courses

Considering the software requirements

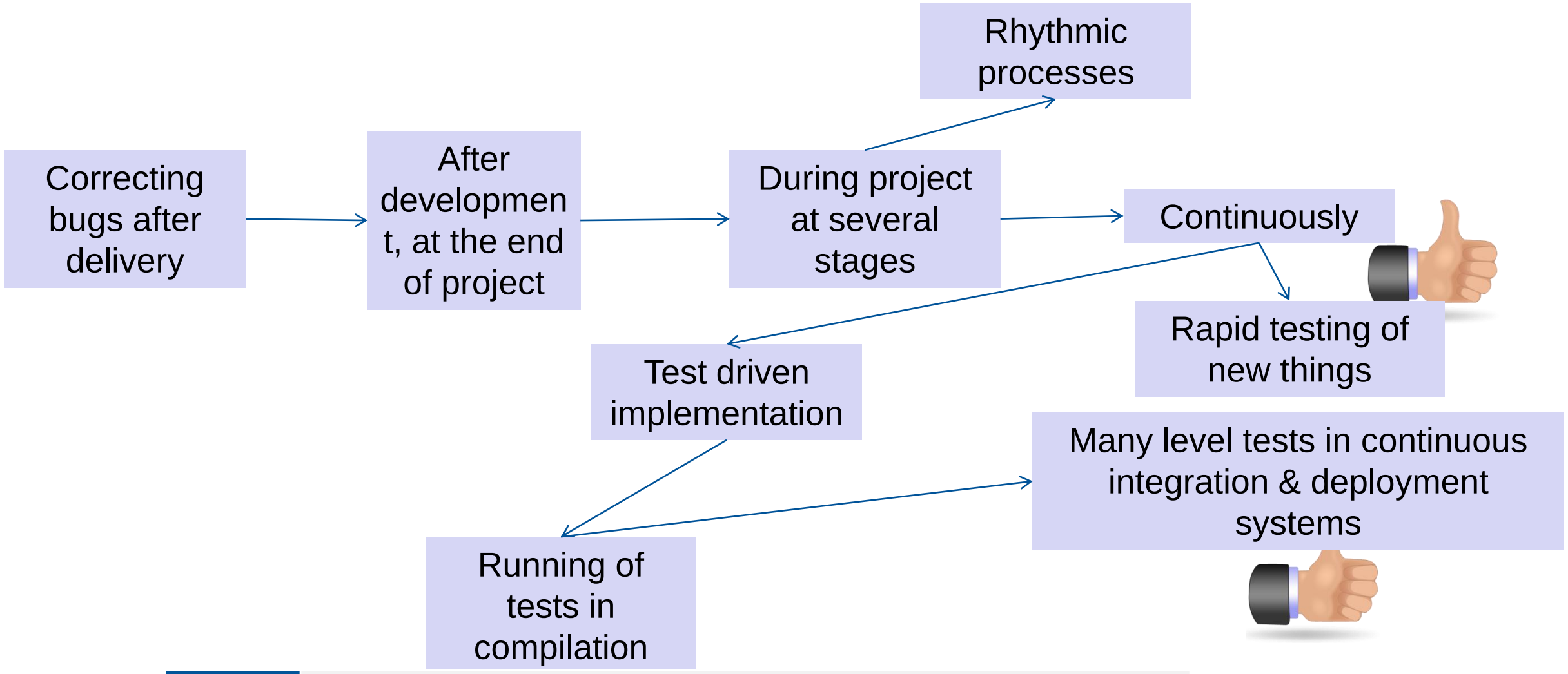


Points The relation with requirements has matured all the time

Concept testing artefacts

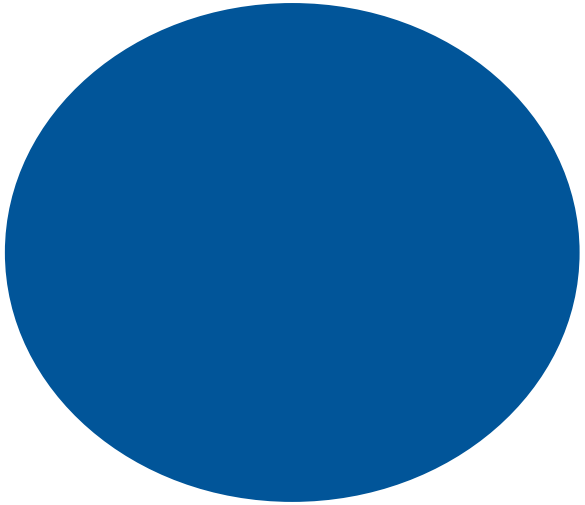


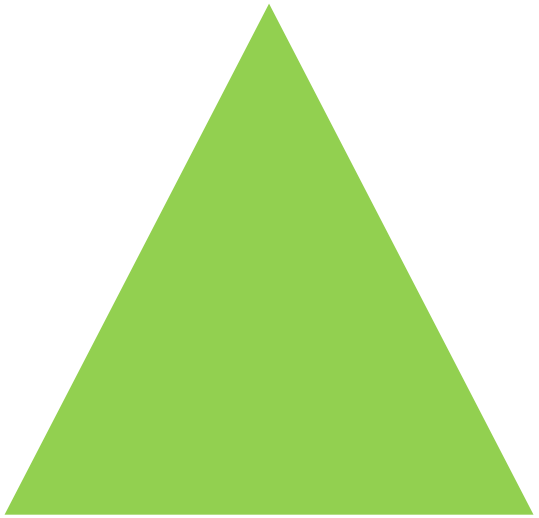
Timing of functional testing



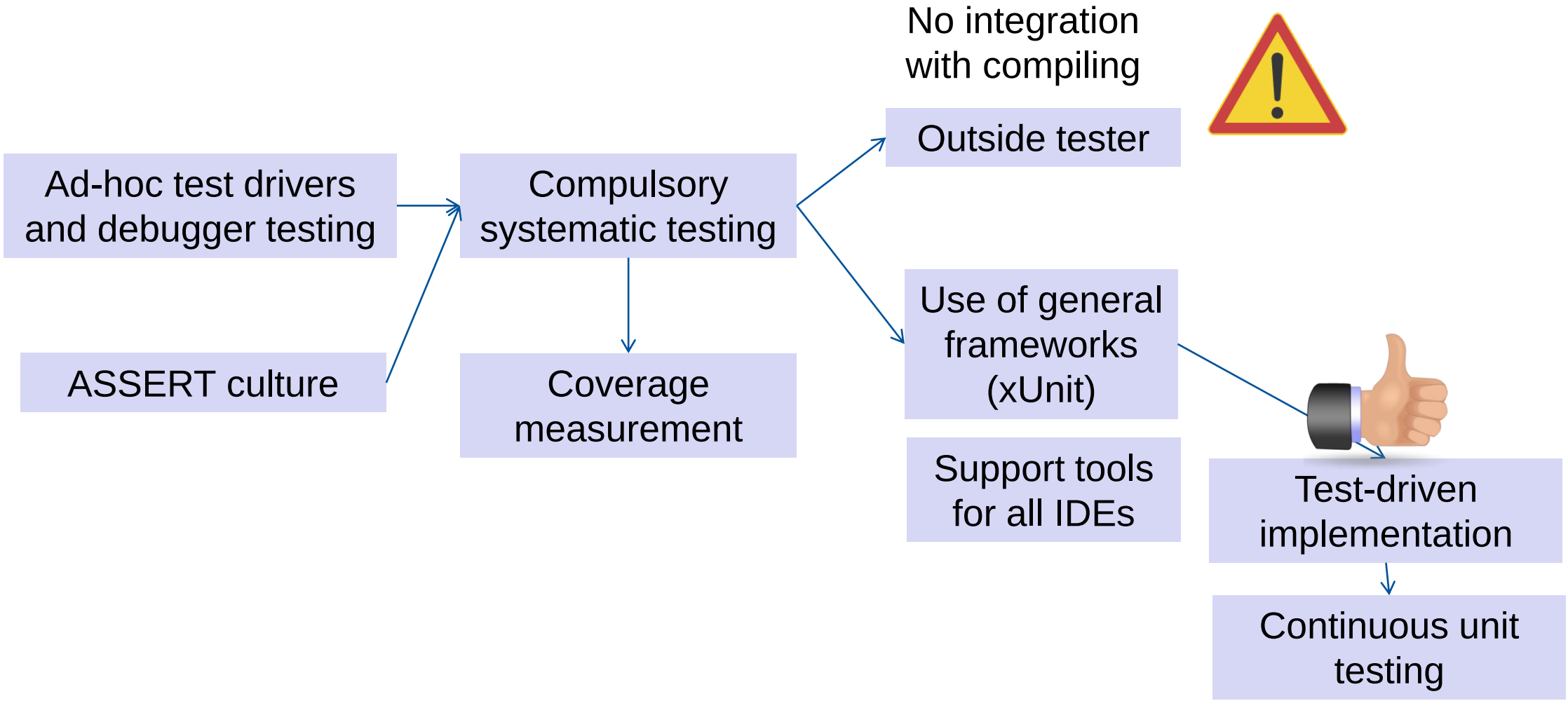
Points All time earlier and more continuously at all testing levels

TEST TYPES



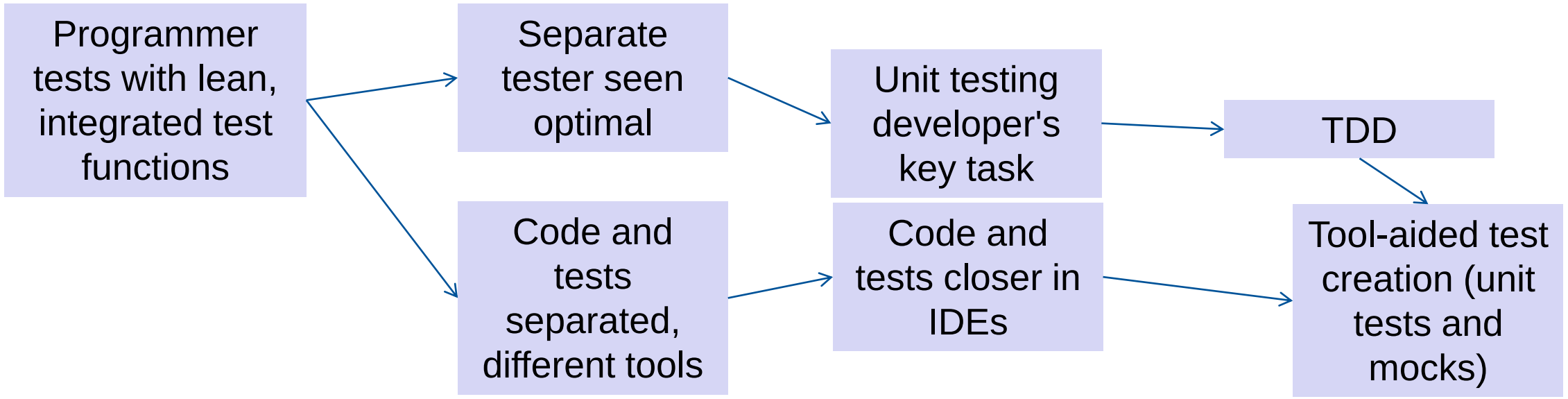


Unit testing

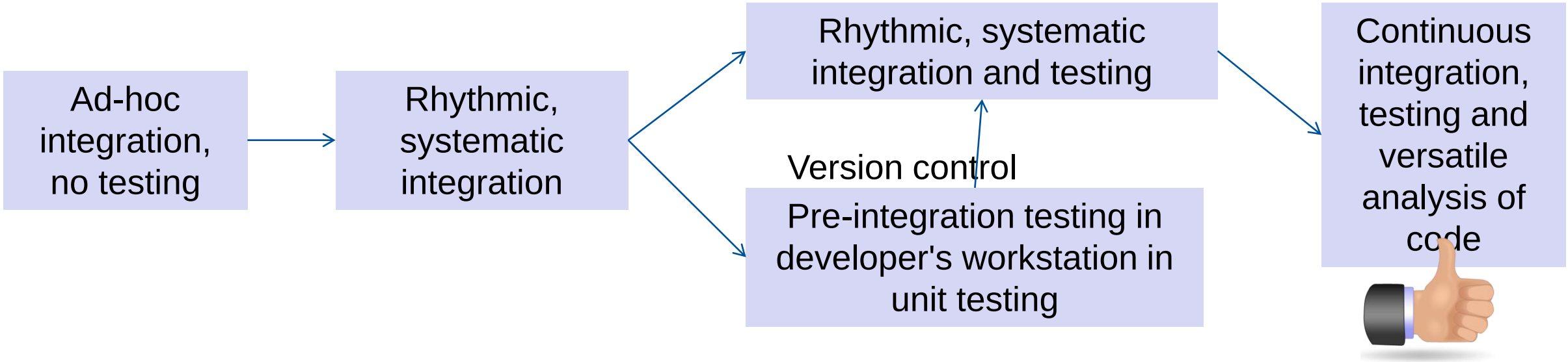


Points Unit testing progresses steadily but still does not always take place

Separation and unification in unit testing



Low level integration testing



Much more integration than integration testing is still done

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

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

Vendors leads & manipulates → No real user, customer-led testing

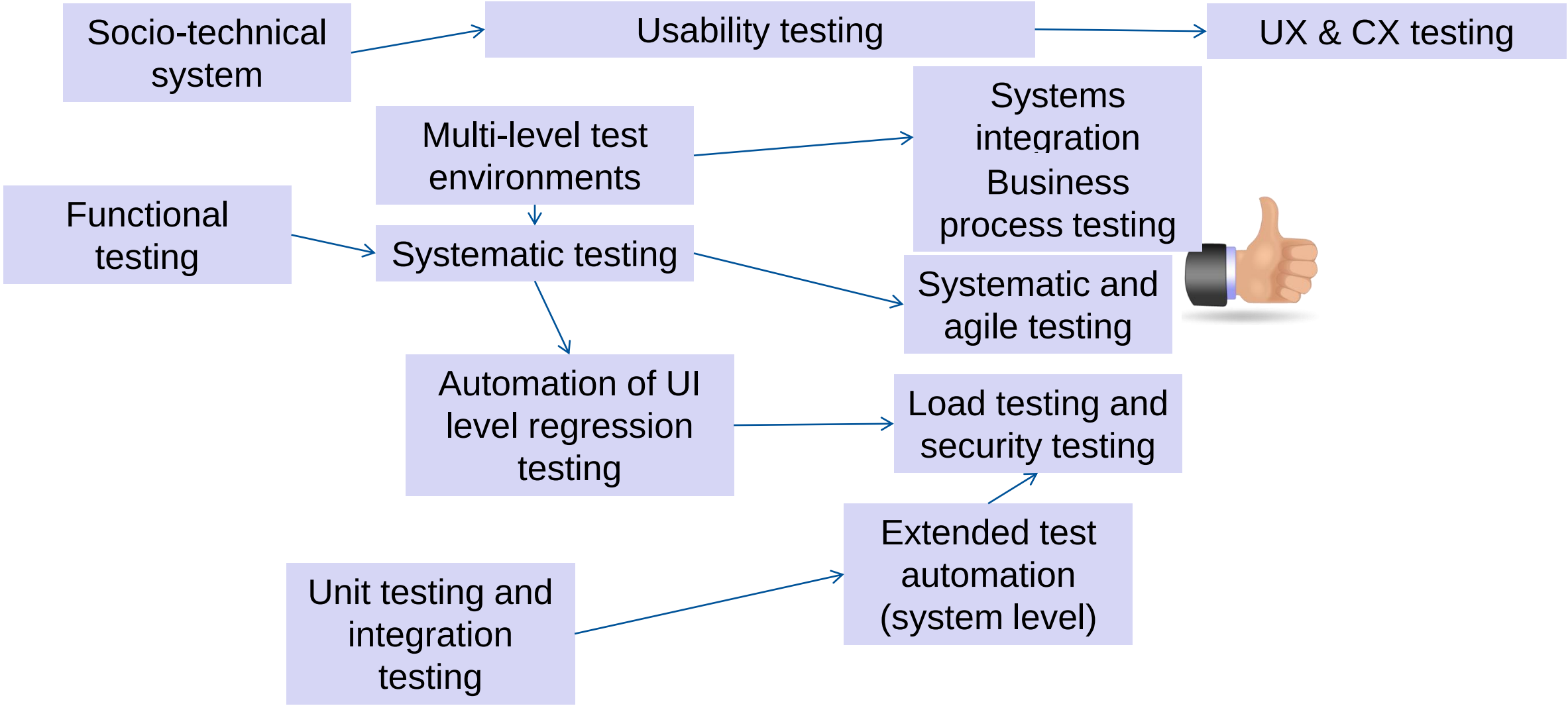
Somebody tries it → Systematic trying of functions → Systematic multi-phased testing in test environments of different levels → Non-functional criteria included

Systematic trying of functions → 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

Points Acceptance testing has developed strongly during the years

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

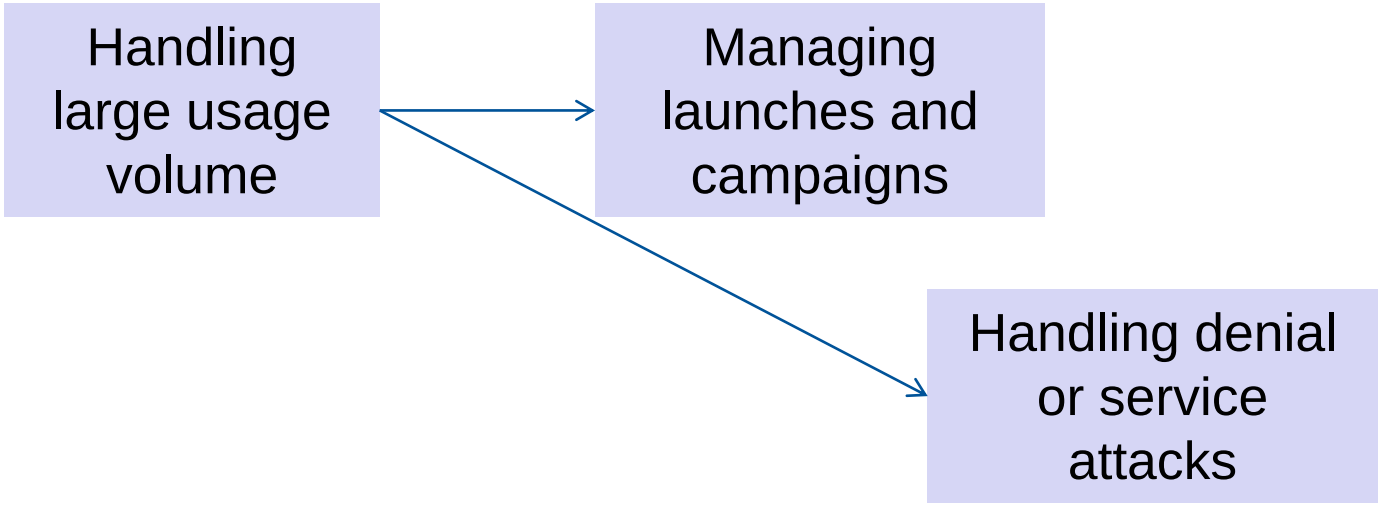
Load testing a routine before going to production

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



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

Usability testing

What is it?

1 outsourced testing round done too late

User studies produce information basis

Low-design, blind A/B testing

Usability analyses and testing at a few stages

UX testing

Review of visual design

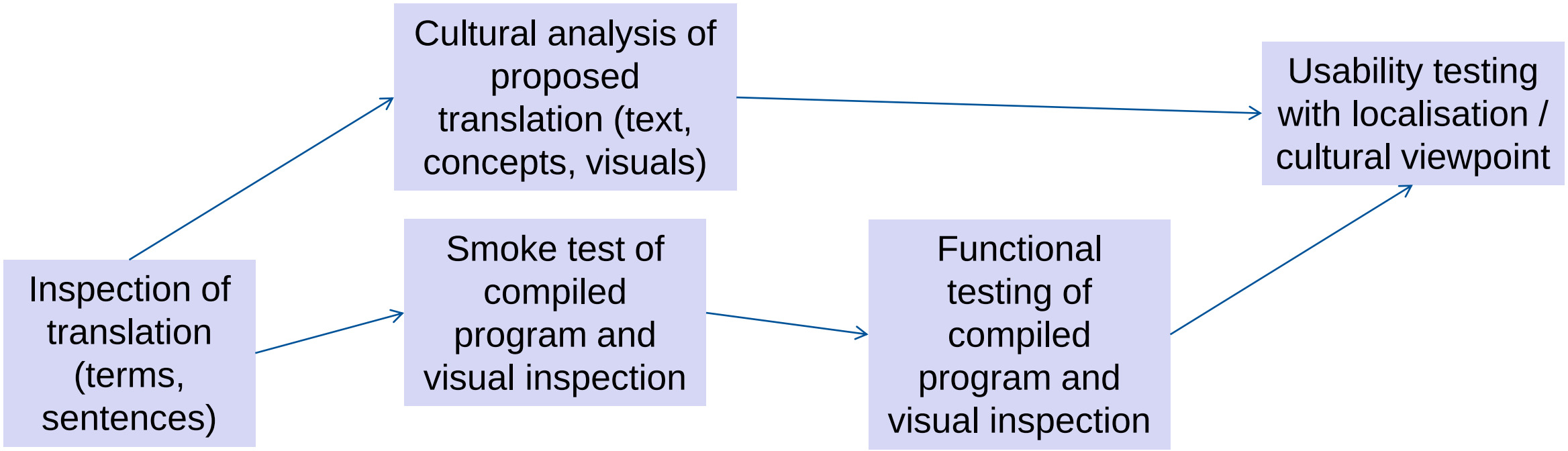
Screen design checklists

Design analysis

Assuring of usability is still usually very lacking

Points Assuring of usability is still often lacking

Localisation testing



Why does ISTQB vocabulary still not know the whole thing?

Points Localisation testing is essential to all software products

Information security testing

Reviews

Code analysis for potential problems

Assuring of safe programming

Bug bounties

What is it?

Mechanistic testing of simple vulnerability (by a consultant)

Testing tools

Rich systemic understanding of mechanism of vulnerabilities

Information risk analysis

System audit and review

Assuring of information security is still usually lacking

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

Regression testing

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

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

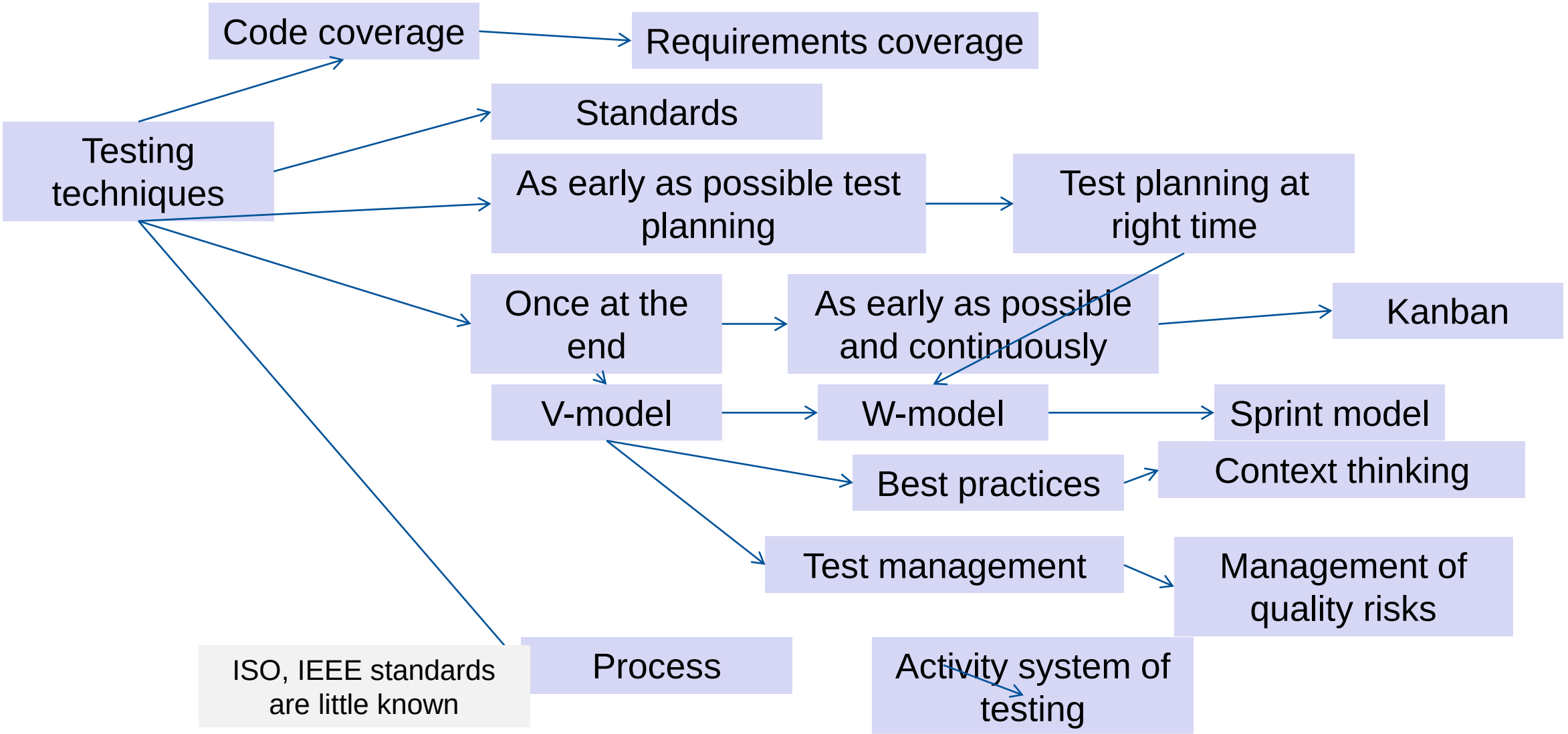
Continuous integration, testing and versatile analysis of code

Exploratory regression testing

Points Regression caused by changes remains a big problem

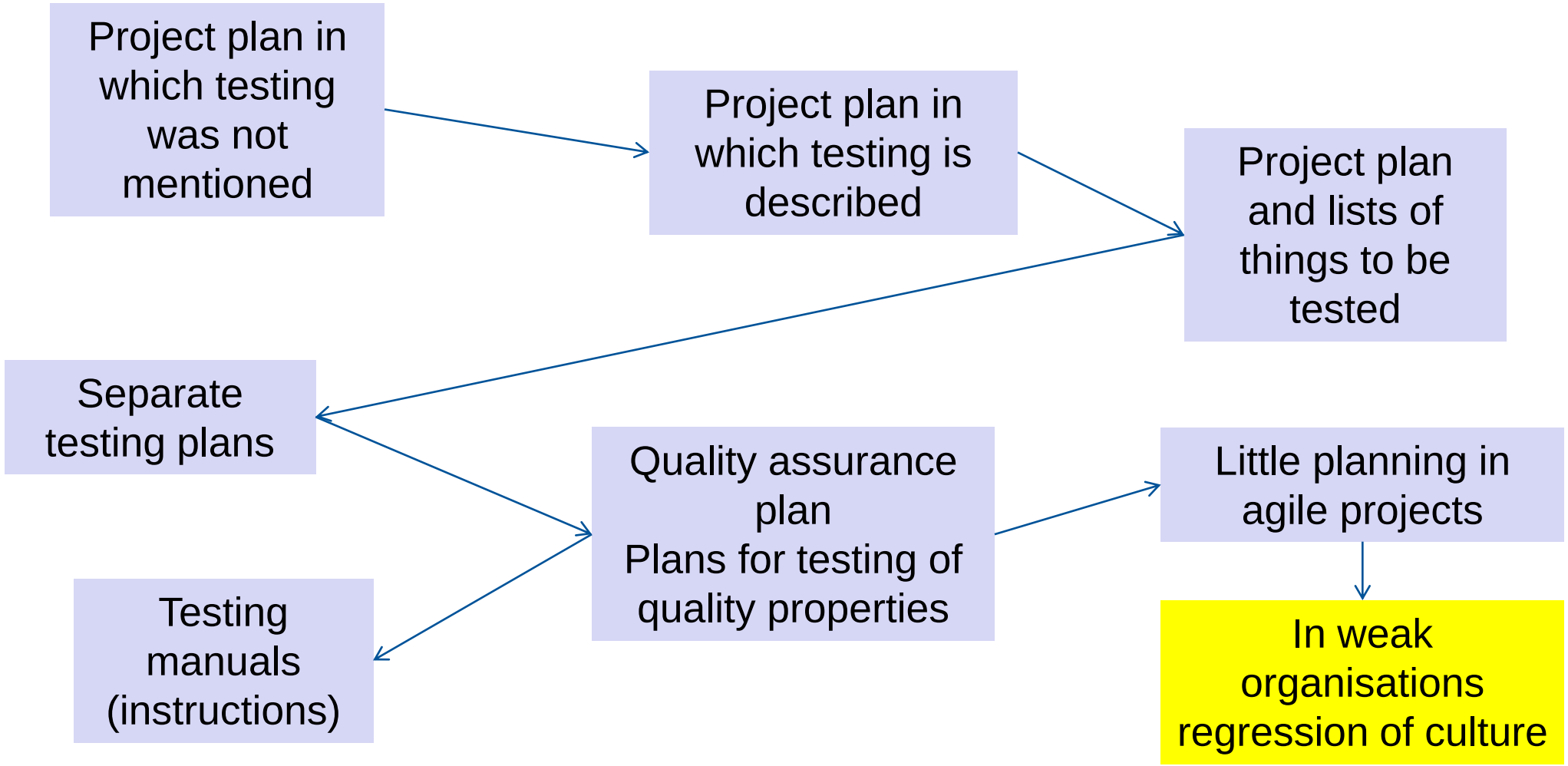
PROCESSES AND METRICS

View on good testing



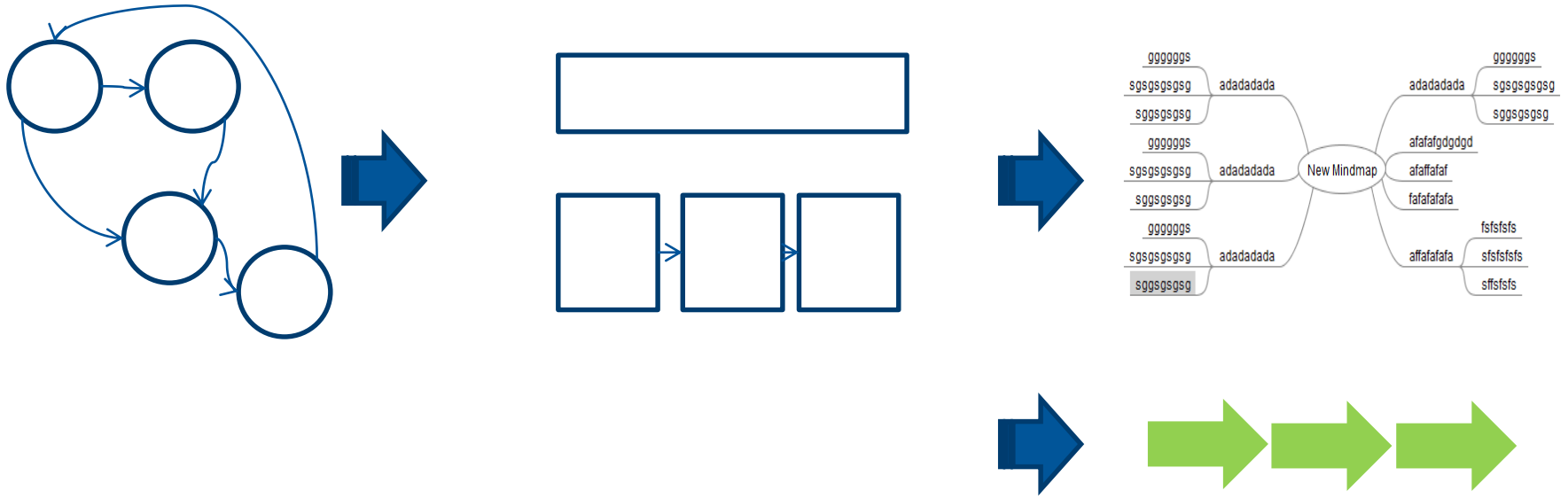
Points When techniques are managed, we can work at a higher activity level

Test plans

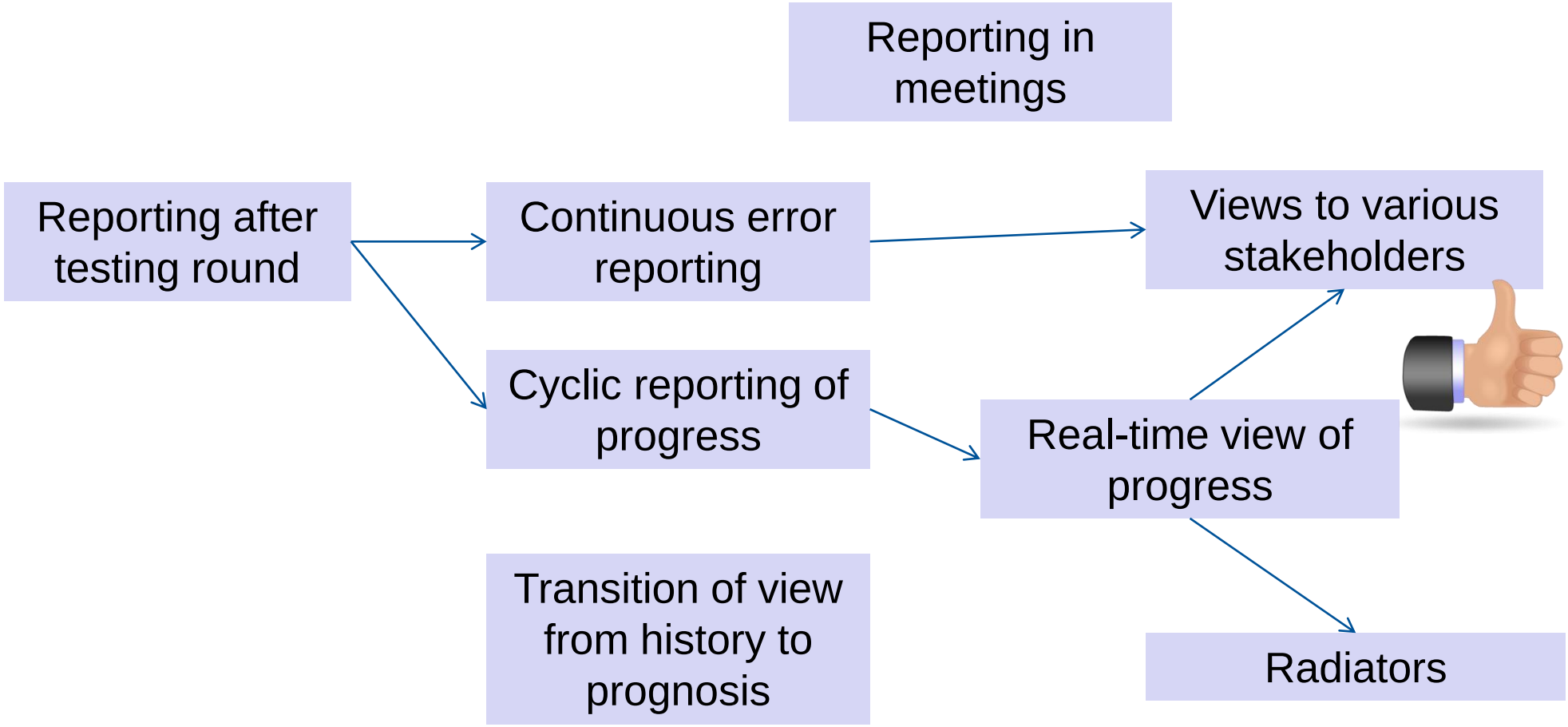


Points Test planning has generally gotten richer

Typical picture in a slide set

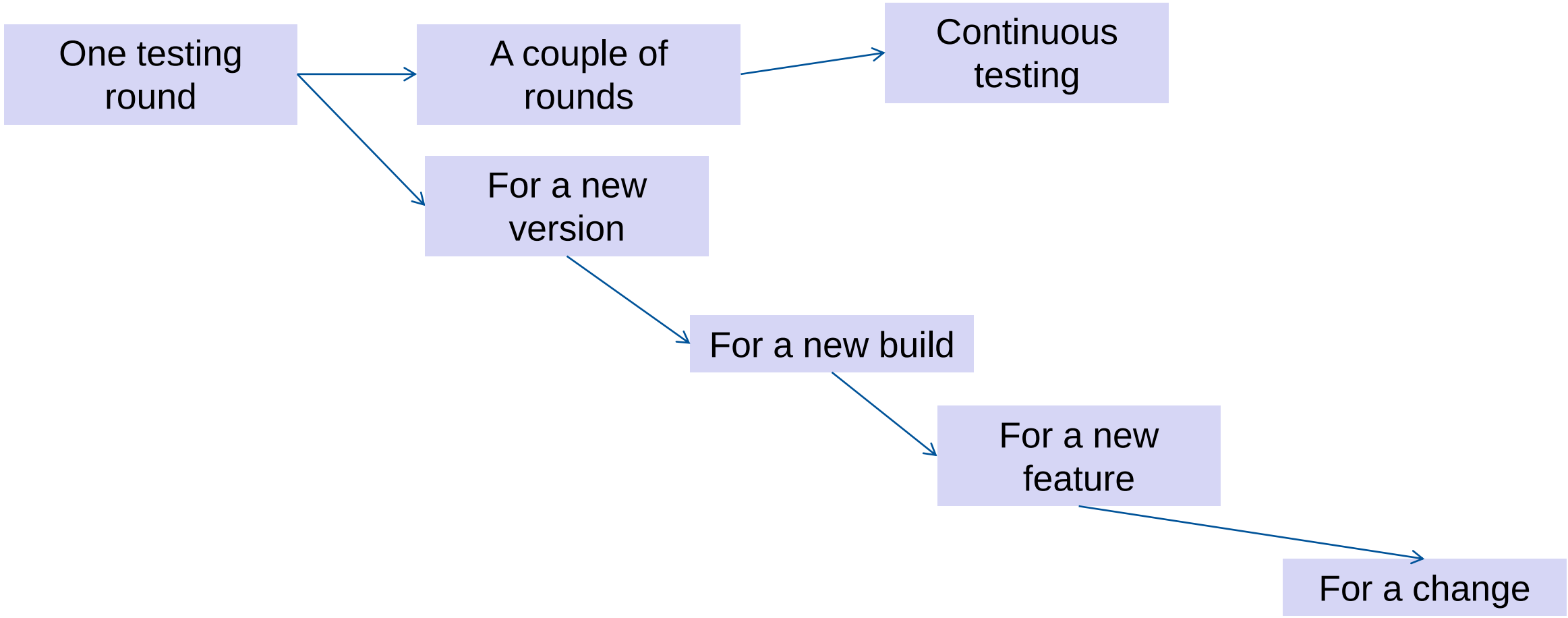


Reporting

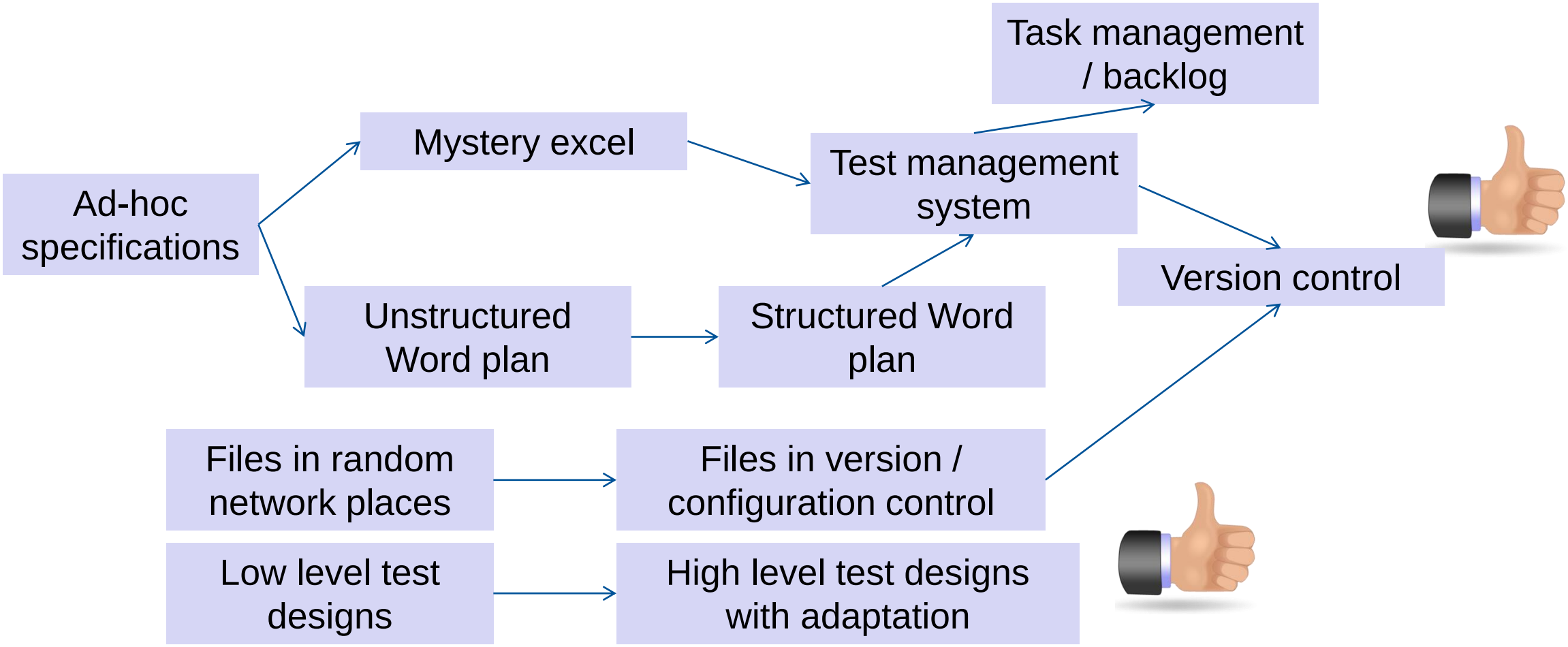


Points Reporting has changed to offering views

Testing rhythm during development



Management of test assets



Points	Handling of specs and other material becoming like handling of requirements and code

Nature of test data

Privacy considerations (GDPR)

Only little data, well behaving so that tests pass

Realistic production data – a lot

Dangerous data (attacks)

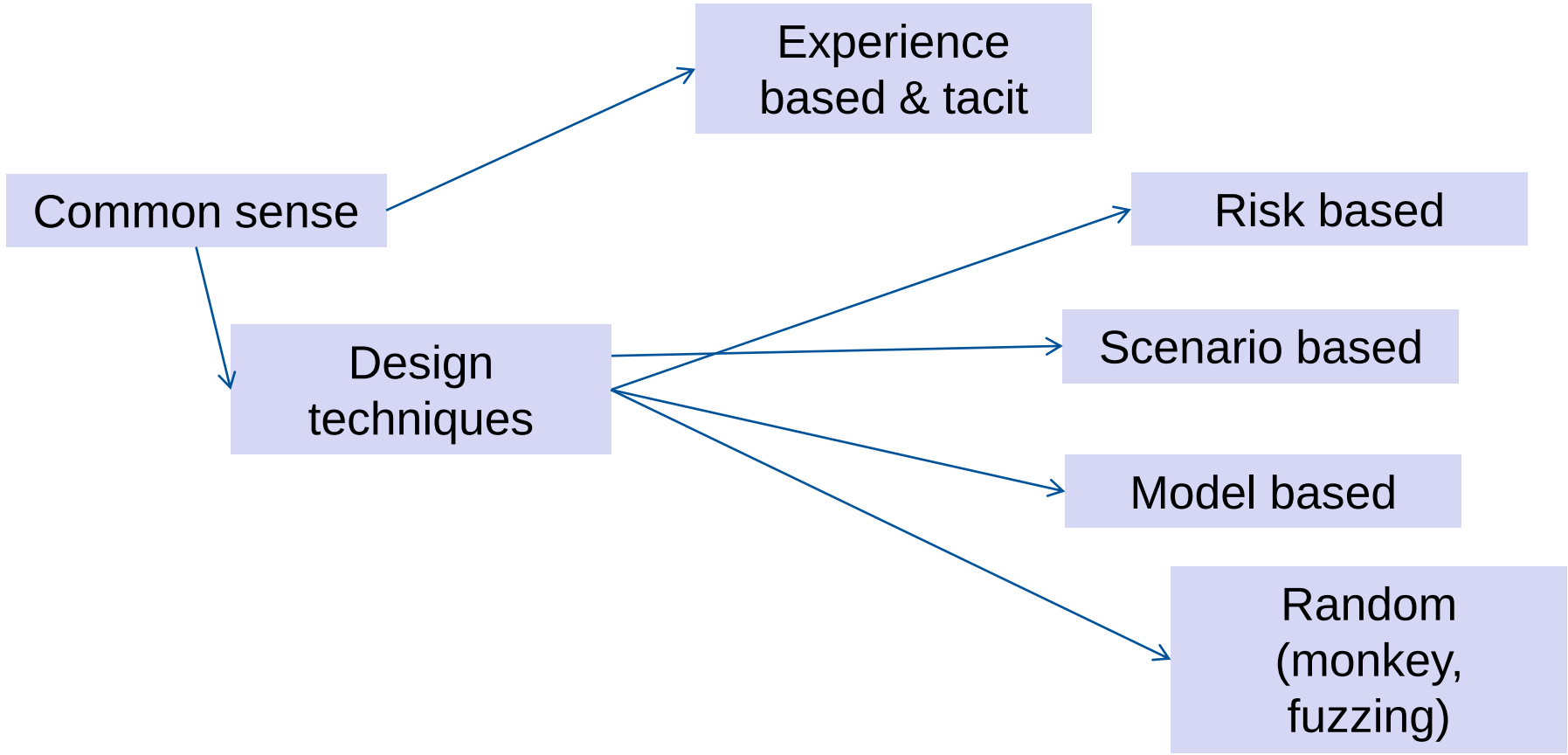
Data generation

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

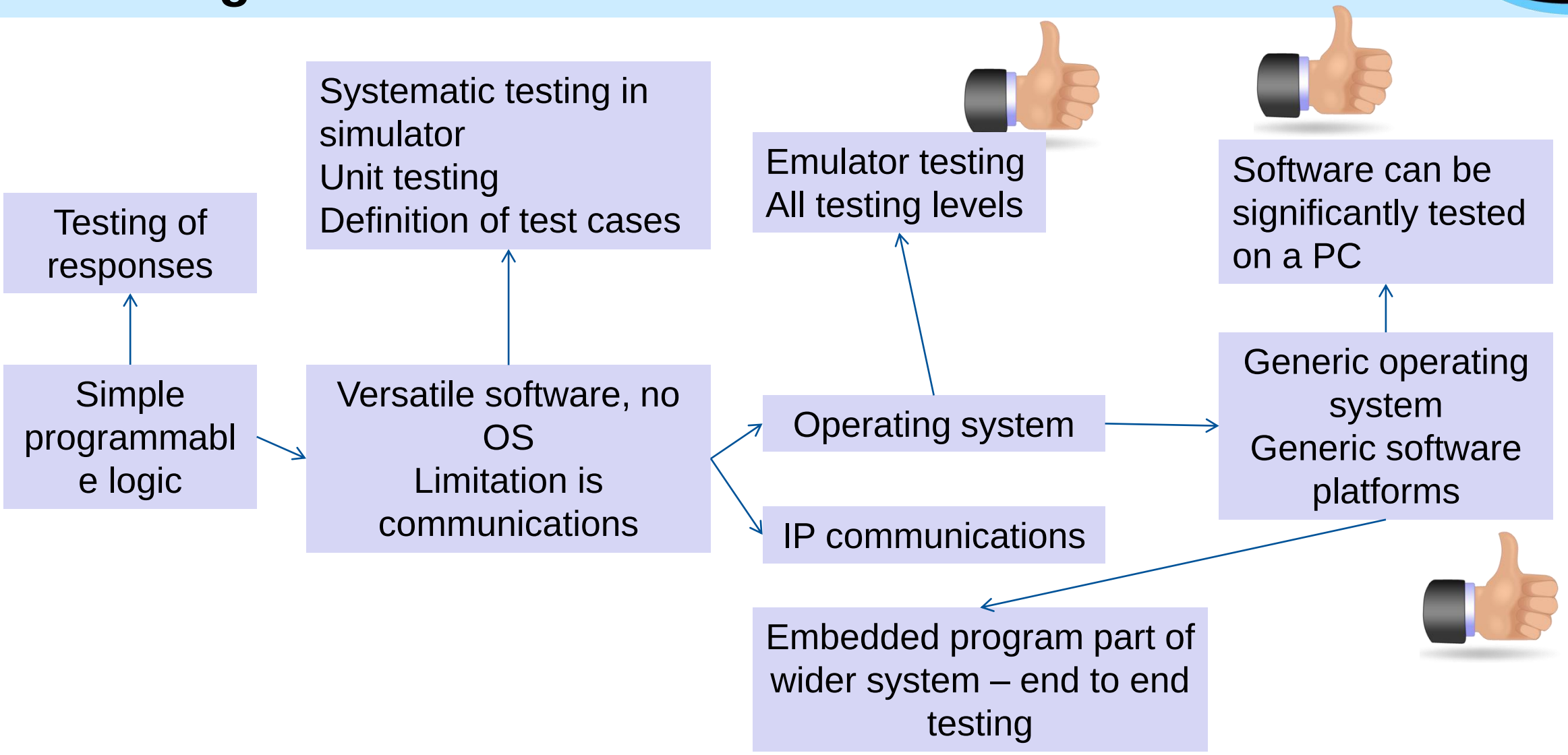
Fuzzing

Points The goal of breaking the program is emphasised in test data

Test design styles

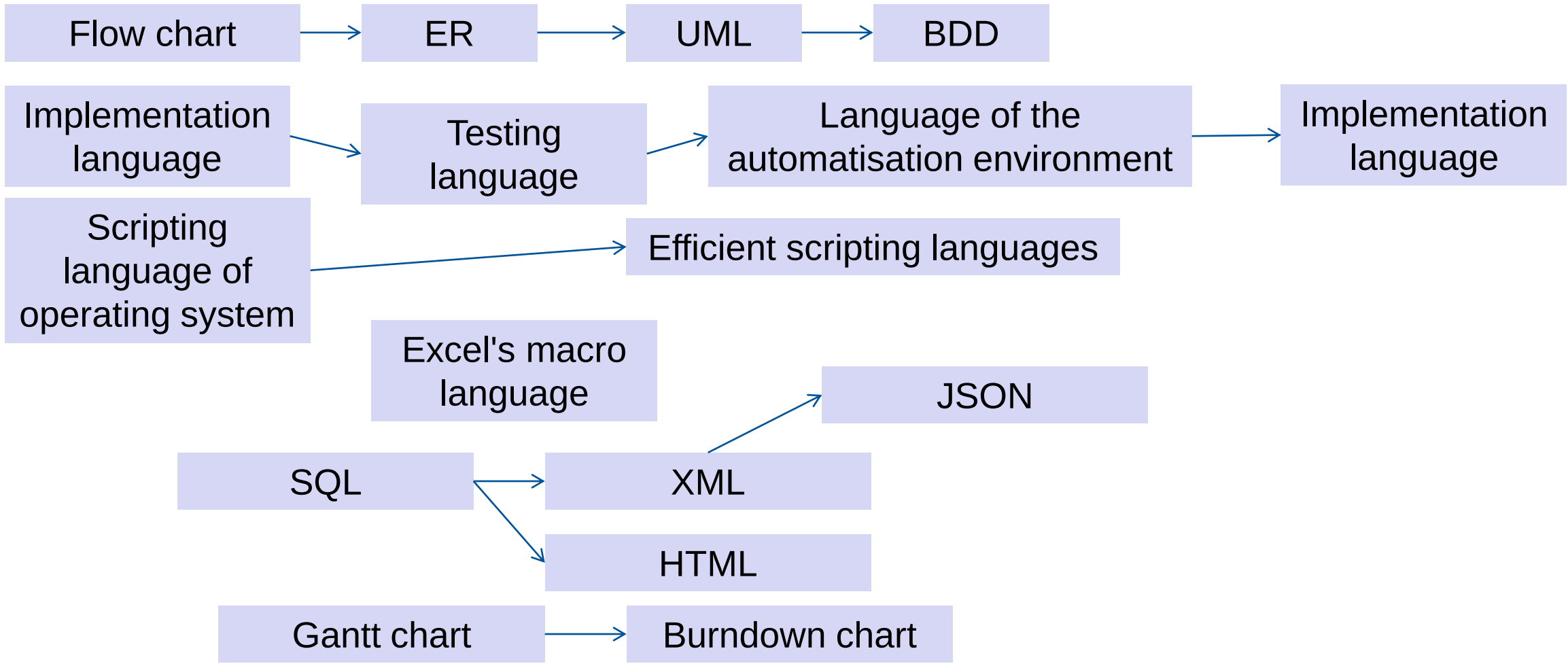


Testing of embedded software



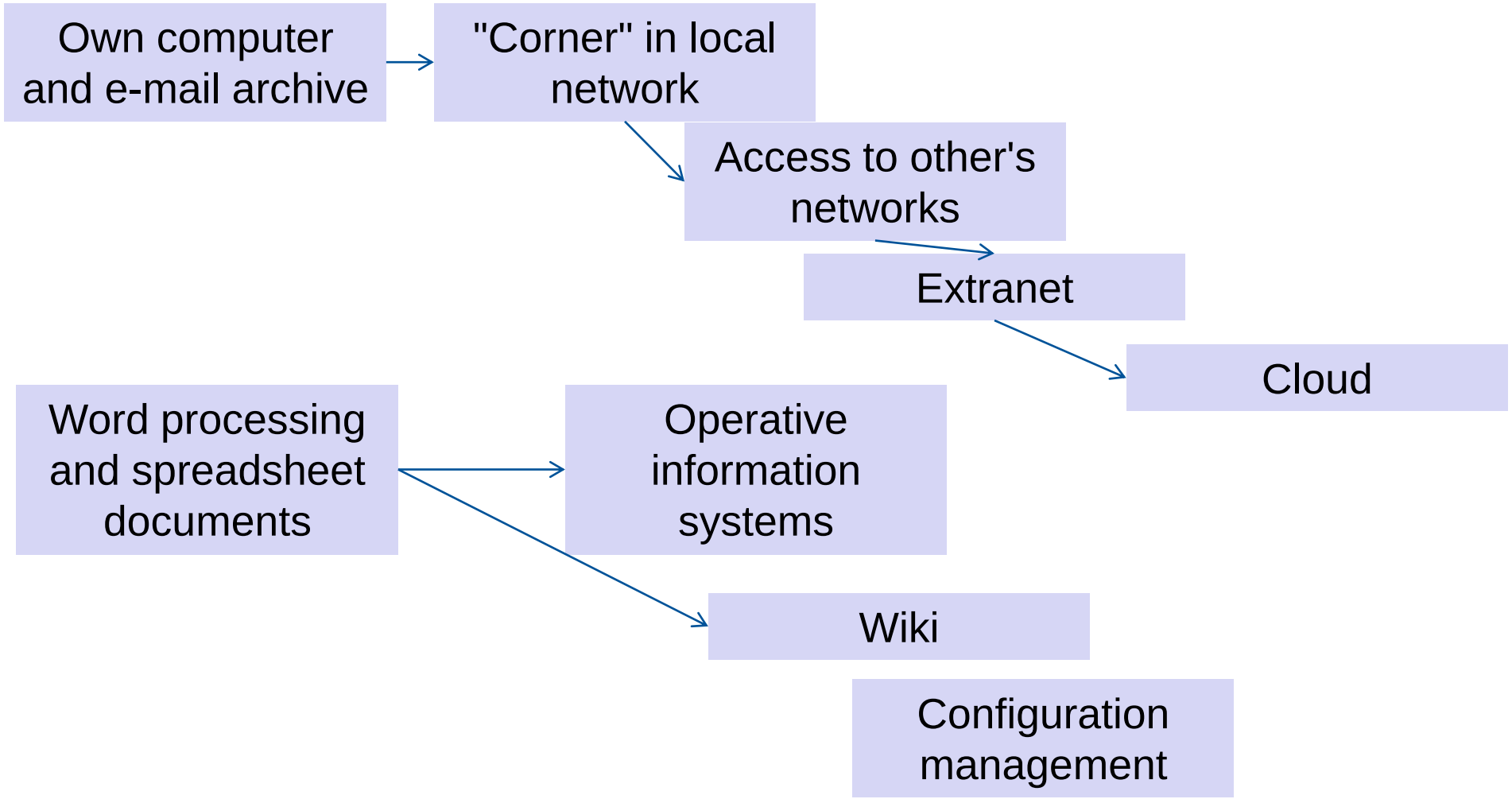
Points Testing of embedded programs begins to be mainstream

Most important formal languages in testing



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

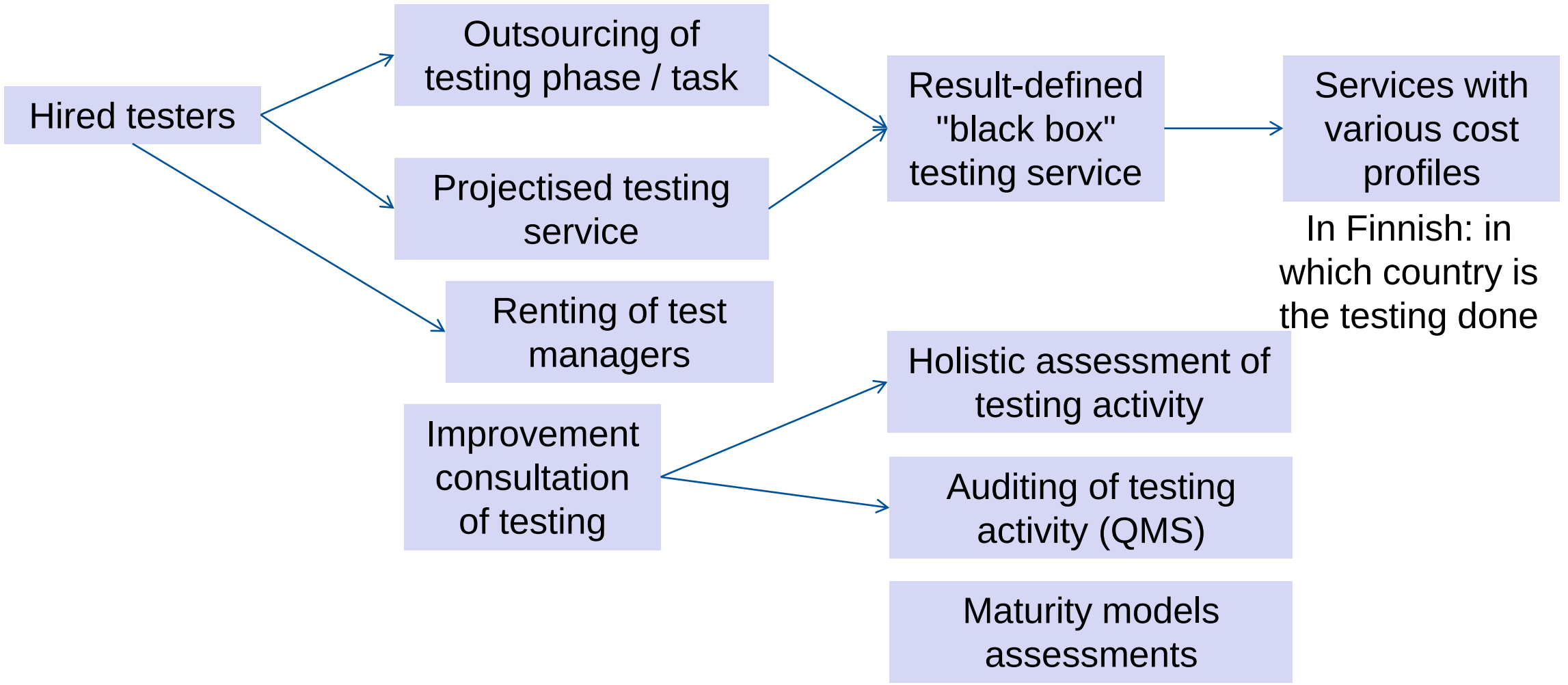
Own risk analysis for testing

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



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

Effectiveness of testing → Development of testing process

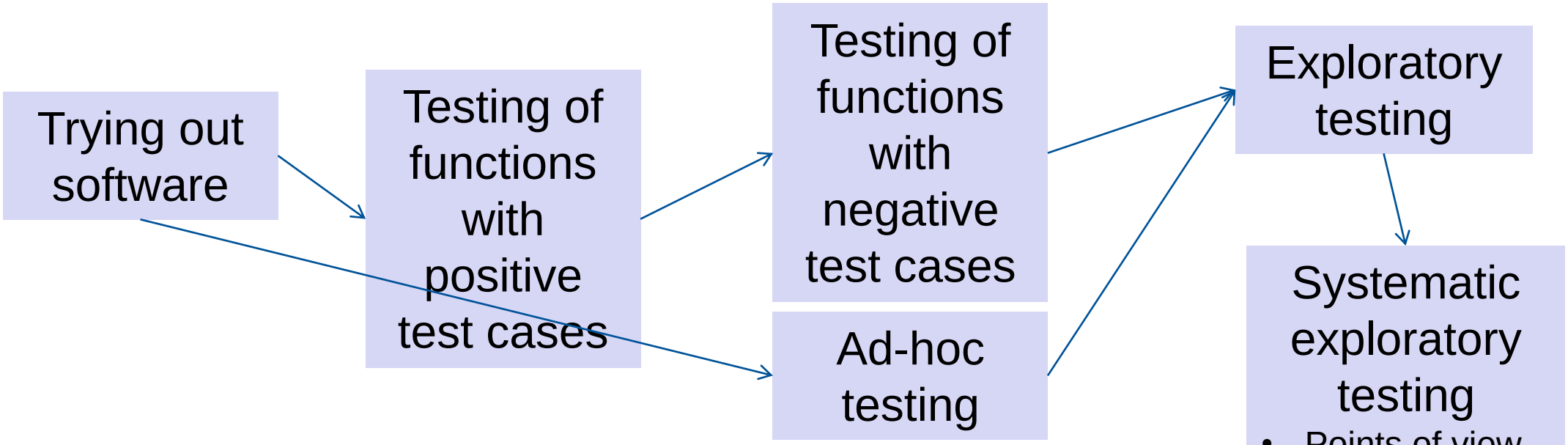
Number of testers → Organisational metrics

Test coverage → How much automated

Points Measuring develops because it is cheaper than before

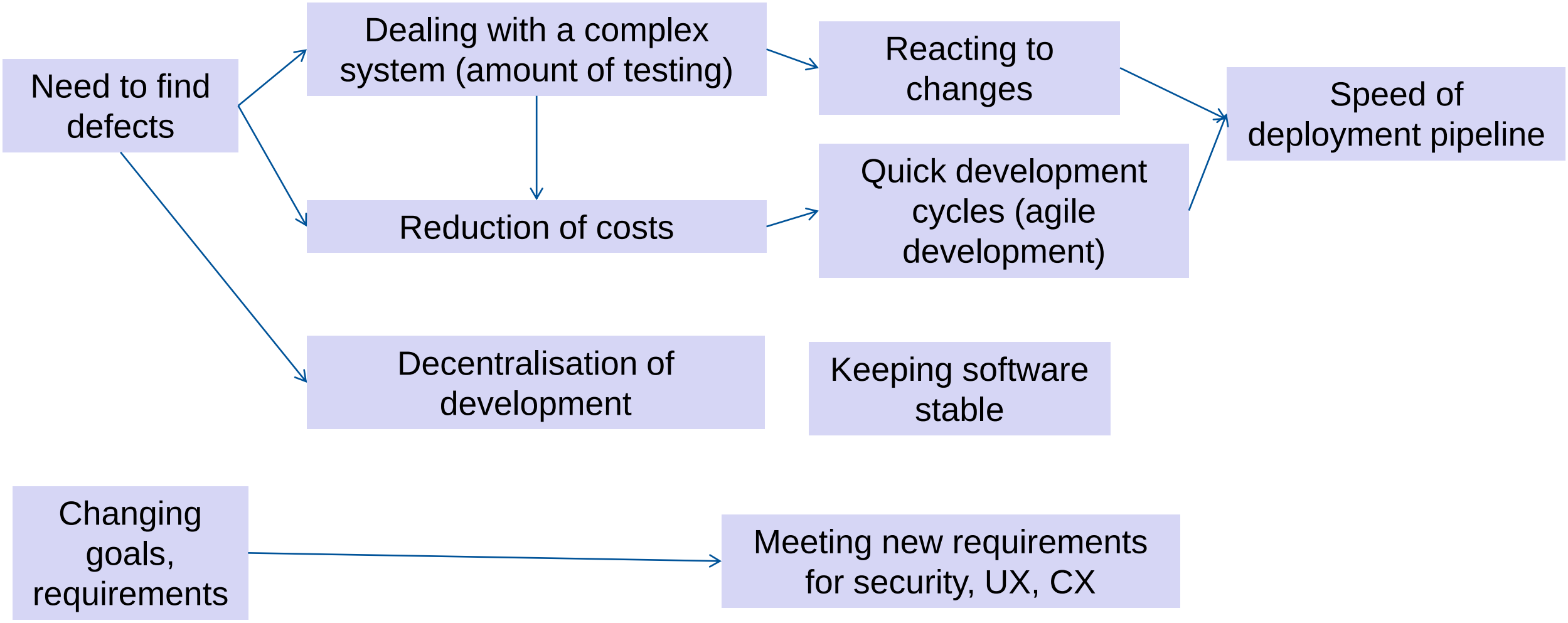
DEVELOPMENT OF TESTING

Development of agile testing



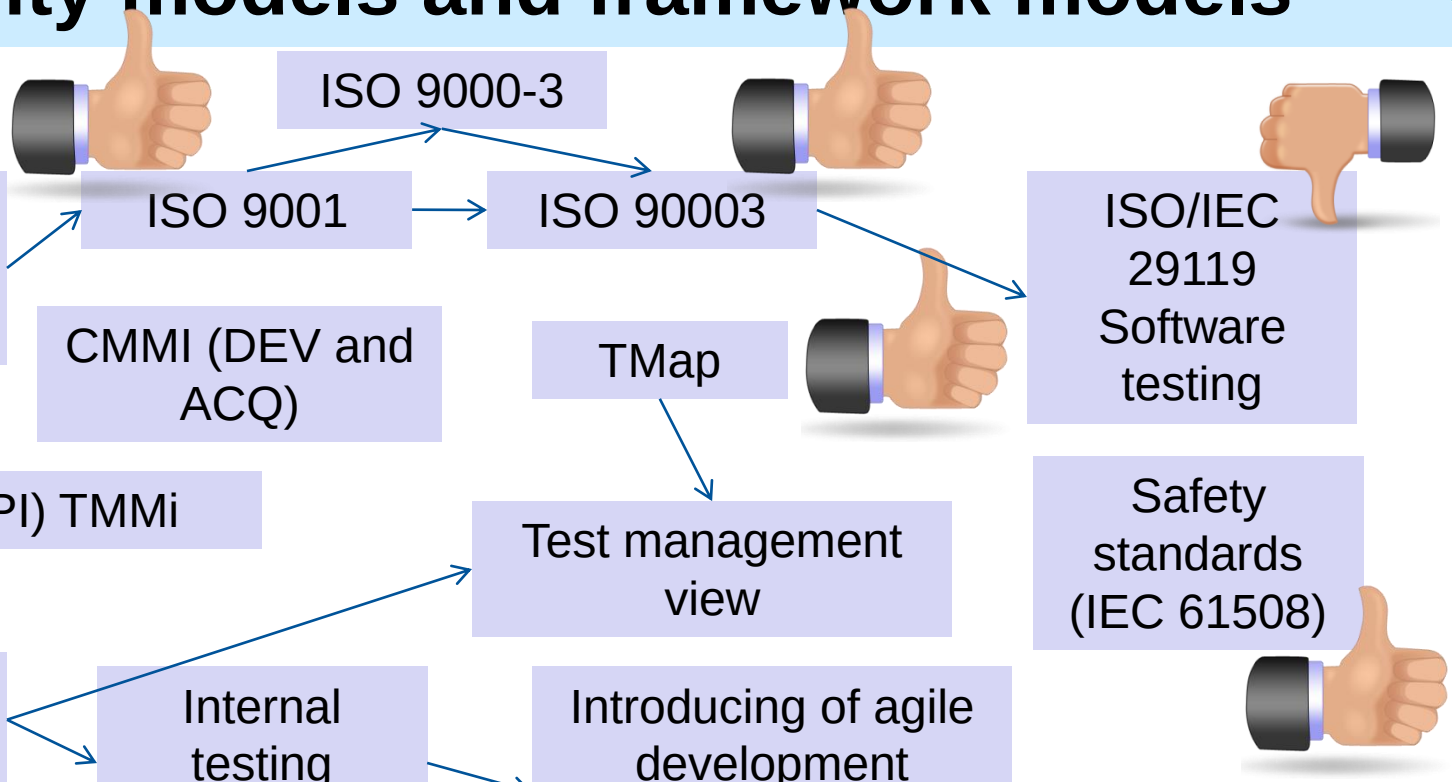
Points There is no "one size fits all" agile testing

Drivers of developing testing



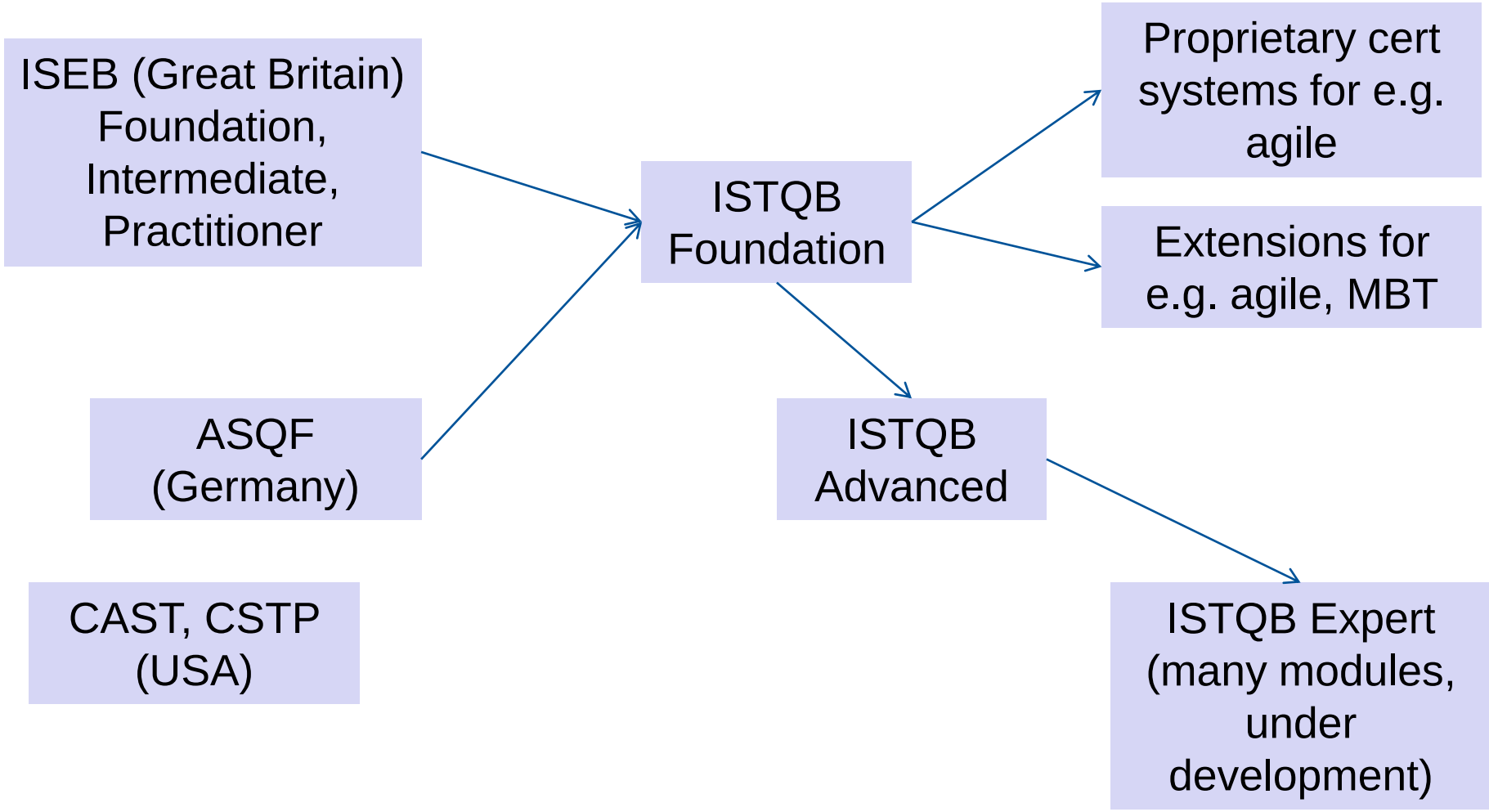
Points There is always a lack of money, but also needs regarding processes

Standards, maturity models and framework models



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

Tester certification systems



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?

TESTING TOOLS

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

Same as developers have so that bugs can be repeated

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

Containers

Replace environment with stubs

Replace environment with copy

Replace external environment with simulating mocks

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

Public beta testing

Test equipment from manufacturer

Test equipment from testing company

Use of emulators

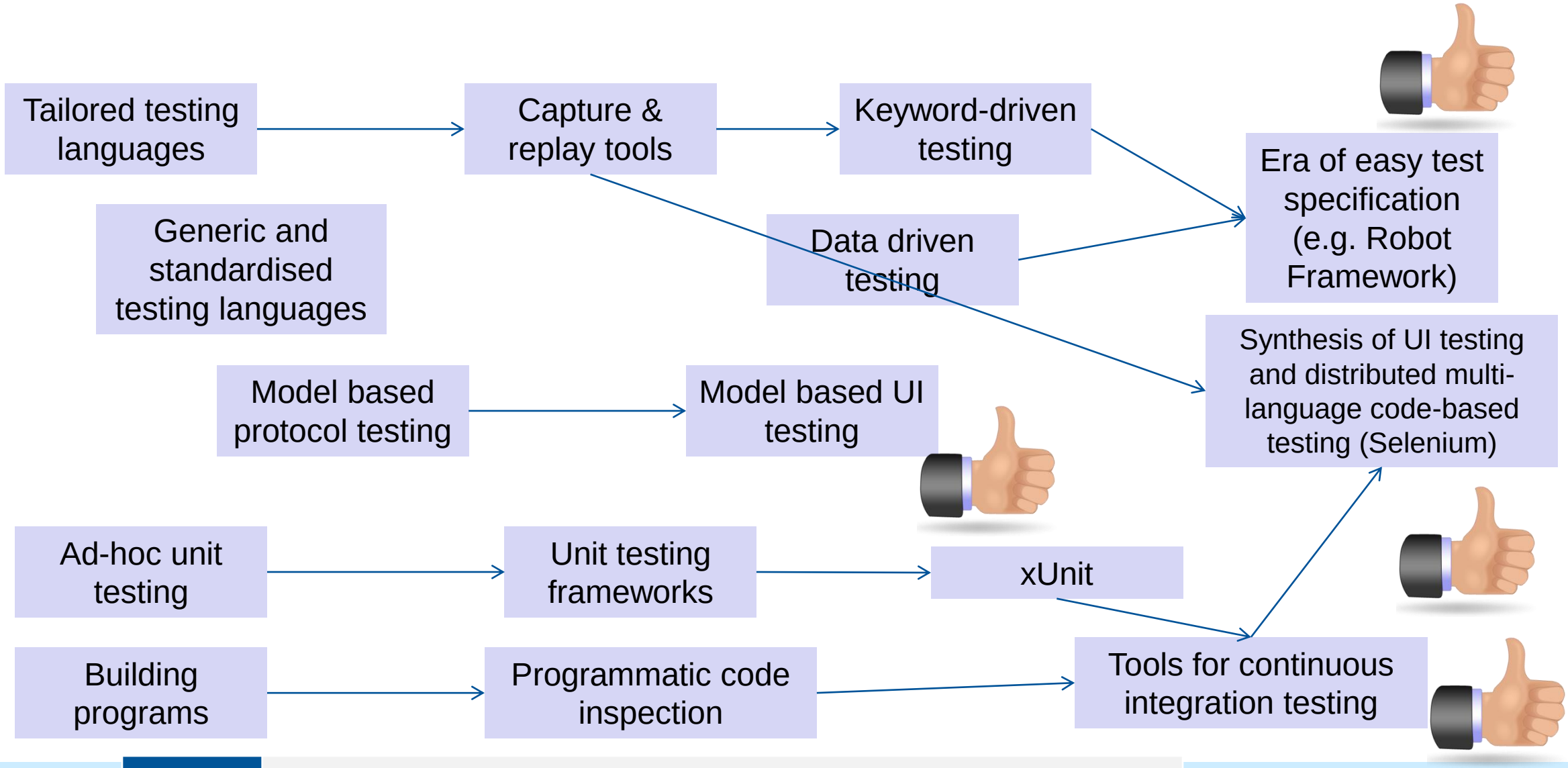
Replace environment with stubs

Replace environment with copy

Replace environment with simulating mocks

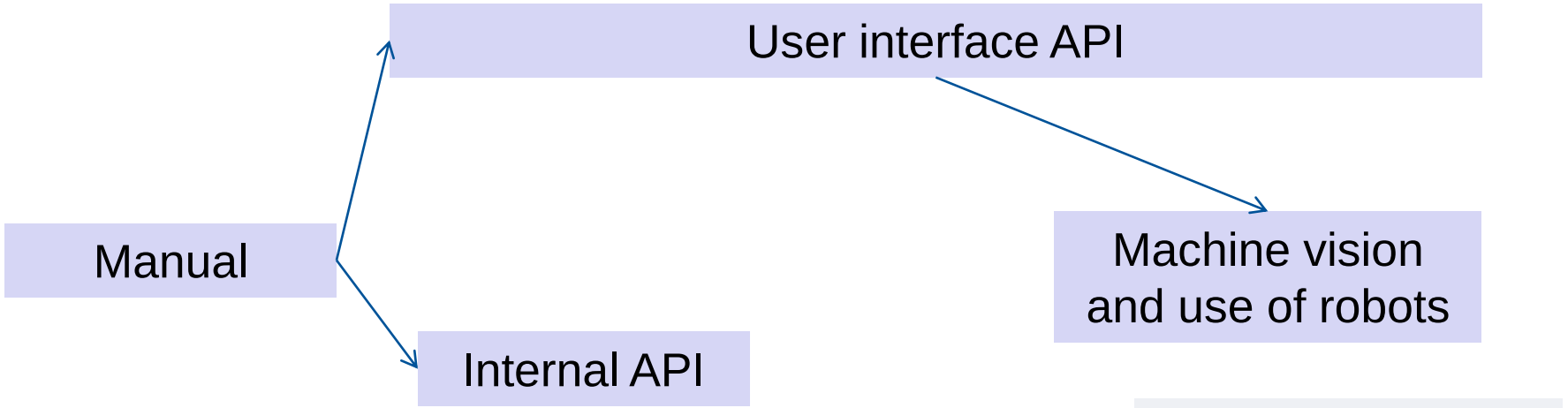
Points Test environments are getting virtualised

Test automation technologies



Points Test automation will become more ordinary all the time to everybody

SUT actuation



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?