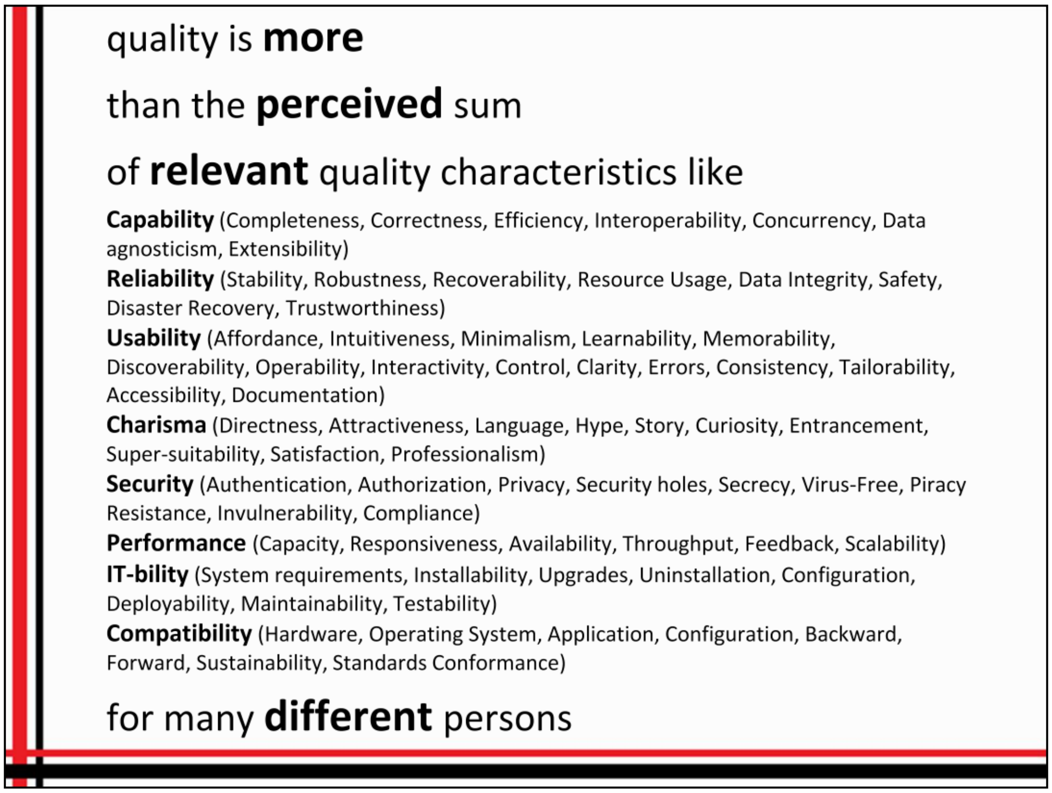
77 Test Idea Triggers

Rikard Edgren

TIBCO Spotfire

SAST Öresund 2011-05-04

V1.0.1



quality is **more**

than the **perceived** sum

of **relevant** quality characteristics like

Capability (Completeness, Correctness, Efficiency, Interoperability, Concurrency, Data agnosticism, Extensibility)
Reliability (Stability, Robustness, Recoverability, Resource Usage, Data Integrity, Safety, Disaster Recovery, Trustworthiness)
Usability (Affordance, Intuitiveness, Minimalism, Learnability, Memorability, Discoverability, Operability, Interactivity, Control, Clarity, Errors, Consistency, Tailorability, Accessibility, Documentation)
Charisma (Directness, Attractiveness, Language, Hype, Story, Curiosity, Entrancement, Super-suitability, Satisfaction, Professionalism)
Security (Authentication, Authorization, Privacy, Security holes, Secrecy, Virus-Free, Piracy Resistance, Invulnerability, Compliance)
Performance (Capacity, Responsiveness, Availability, Throughput, Feedback, Scalability)
IT-bility (System requirements, Installability, Upgrades, Uninstallation, Configuration, Deployability, Maintainability, Testability)
Compatibility (Hardware, Operating System, Application, Configuration, Backward, Forward, Sustainability, Standards Conformance)

for many **different** persons

There are many definitions of quality:

”guess it’s that people like it”

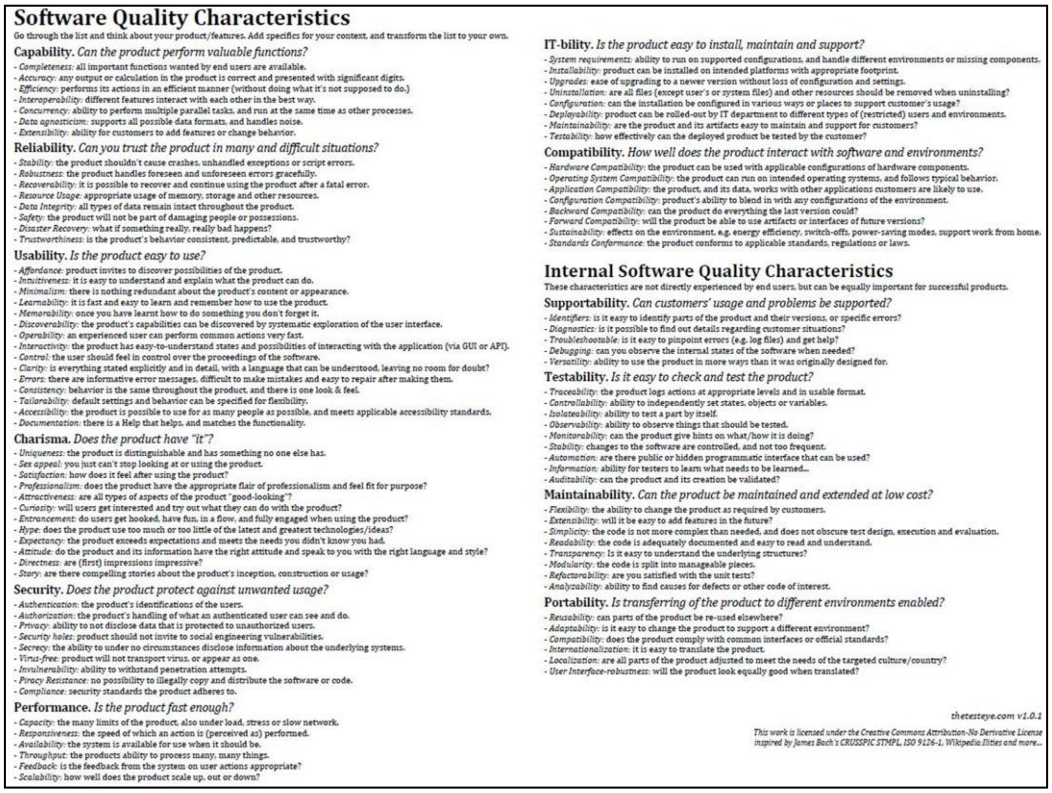
”quality is value to some person”

doesn’t help me so much if these persons can’t tell me more

quality is more than the subjective sum of relevant quality attributes like capability, reliability, usability, charisma, security, performance, IT-bility, compatibility”

e.g. a stable product is better than an unstable

A long definition like this is of course practically useless.



Software Quality Characteristics

Go through the list and think about your product/features. Add specifics for your context, and transform the list to your own.

Capability. *Can the product perform valuable functions?*
- *Completeness:* all important functions wanted by end users are available.
- *Accuracy:* any output or calculation in the product is correct and presented with significant digits.
- *Efficiency:* performs its actions in an efficient manner (without doing what it's not supposed to do.)
- *Interoperability:* different features interact with each other in the best way.
- *Concurrency:* ability to perform multiple parallel tasks, and run at the same time as other processes.
- *Data agnosticism:* supports all possible data formats, and handles noise.
- *Extensibility:* ability for customers to add features or change behavior.

Reliability. *Can you trust the product in many and difficult situations?*
- *Stability:* the product shouldn't cause crashes, unhandled exceptions or script errors.
- *Robustness:* the product handles foreseen and unforeseen errors gracefully.
- *Recoverability:* it is possible to recover and continue using the product after a fatal error.
- *Resource Usage:* appropriate usage of memory, storage and other resources.
- *Data Integrity:* all types of data remain intact throughout the product.
- *Safety:* the product will not be part of damaging people or possessions.
- *Disaster Recovery:* what if something really, really bad happens?
- *Trustworthiness:* is the product's behavior consistent, predictable, and trustworthy?

Usability. *Is the product easy to use?*
- *Affordance:* product invites to discover possibilities of the product.
- *Intuitiveness:* it is easy to understand and explain what the product can do.
- *Minimalism:* there is nothing redundant about the product's content or appearance.
- *Learnability:* it is fast and easy to learn and remember how to use the product.
- *Memorability:* once you have learnt how to do something you don't forget it.
- *Discoverability:* the product's capabilities can be discovered by systematic exploration of the user interface.
- *Operability:* an experienced user can perform common actions very fast.
- *Interactivity:* the product has easy-to-understand states and possibilities of interacting with the application (via GUI or API).
- *Control:* the user should feel in control over the proceedings of the software.
- *Clarity:* is everything stated explicitly and in detail, with a language that can be understood, leaving no room for doubt?
- *Errors:* there are informative error messages, difficult to make mistakes and easy to repair after making them.
- *Consistency:* behavior is the same throughout the product, and there is one look & feel.
- *Tailorability:* default settings and behavior can be specified for flexibility.
- *Accessibility:* the product is possible to use for as many people as possible, and meets applicable accessibility standards.
- *Documentation:* there is a Help that helps, and matches the functionality.

Charisma. *Does the product have "it"?*
- *Uniqueness:* the product is distinguishable and has something no one else has.
- *Sex appeal:* you just can't stop looking at or using the product.
- *Satisfaction:* how does it feel after using the product?
- *Professionalism:* does the product have the appropriate flair of professionalism and feel fit for purpose?
- *Attractiveness:* are all types of aspects of the product "good-looking"?
- *Curiosity:* will users get interested and try out what they can do with the product?
- *Entrancement:* do users get hooked, have fun, in a flow, and fully engaged when using the product?
- *Hype:* does the product use too much or too little of the latest and greatest technologies/ideas?
- *Expectancy:* the product exceeds expectations and meets the needs you didn't know you had.
- *Attitude:* do the product and its information have the right attitude and speak to you with the right language and style?
- *Directness:* are (first) impressions impressive?
- *Story:* are there compelling stories about the product's inception, construction or usage?

Security. *Does the product protect against unwanted usage?*
- *Authentication:* the product's identifications of the users.
- *Authorization:* the product's handling of what an authenticated user can see and do.
- *Privacy:* ability to not disclose data that is protected to unauthorized users.
- *Security holes:* product should not invite to social engineering vulnerabilities.
- *Secrecy:* the ability to under no circumstances disclose information about the underlying systems.
- *Virus-free:* product will not transport virus, or appear as one.
- *Invulnerability:* ability to withstand penetration attempts.
- *Piracy Resistance:* no possibility to illegally copy and distribute the software or code.
- *Compliance:* security standards the product adheres to.

Performance. *Is the product fast enough?*
- *Capacity:* the many limits of the product, also under load, stress or slow network.
- *Responsiveness:* the speed of which an action is (perceived as) performed.
- *Availability:* the system is available for use when it should be.
- *Throughput:* the products ability to process many, many things.
- *Feedback:* is the feedback from the system on user actions appropriate?
- *Scalability:* how well does the product scale up, out or down?

IT-bility. *Is the product easy to install, maintain and support?*
- *System requirements:* ability to run on supported configurations, and handle different environments or missing components.
- *Installability:* product can be installed on intended platforms with appropriate footprint.
- *Upgrades:* ease of upgrading to a newer version without loss of configuration and settings.
- *Uninstallation:* are all files (except user's or system files) and other resources should be removed when uninstalling?
- *Configuration:* can the installation be configured in various ways or places to support customer's usage?
- *Deployability:* product can be rolled-out by IT department to different types of (restricted) users and environments.
- *Maintainability:* are the product and its artifacts easy to maintain and support for customers?
- *Testability:* how effectively can the deployed product be tested by the customer?

Compatibility. *How well does the product interact with software and environments?*
- *Hardware Compatibility:* the product can be used with applicable configurations of hardware components.
- *Operating System Compatibility:* the product can run on intended operating systems, and follows typical behavior.
- *Application Compatibility:* the product, and its data, works with other applications customers are likely to use.
- *Configuration Compatibility:* product's ability to blend in with any configurations of the environment.
- *Backward Compatibility:* can the product do everything the last version could?
- *Forward Compatibility:* will the product be able to use artifacts or interfaces of future versions?
- *Sustainability:* effects on the environment, e.g. energy efficiency, switch-offs, power-saving modes, support work from home.
- *Standards Conformance:* the product conforms to applicable standards, regulations or laws.

Internal Software Quality Characteristics

These characteristics are not directly experienced by end users, but can be equally important for successful products.

Supportability. *Can customers' usage and problems be supported?*
- *Identifiers:* is it easy to identify parts of the product and their versions, or specific errors?
- *Diagnostics:* is it possible to find out details regarding customer situations?
- *Troubleshootable:* is it easy to pinpoint errors (e.g. log files) and get help?
- *Debugging:* can you observe the internal states of the software when needed?
- *Versatility:* ability to use the product in more ways than it was originally designed for.

Testability. *Is it easy to check and test the product?*
- *Traceability:* the product logs actions at appropriate levels and in usable format.
- *Controllability:* ability to independently set states, objects or variables.
- *Isolateability:* ability to test a part by itself.
- *Observability:* ability to observe things that should be tested.
- *Monitorability:* can the product give hints on what/how it is doing?
- *Stability:* changes to the software are controlled, and not too frequent.
- *Automation:* are there public or hidden programmatic interface that can be used?
- *Information:* ability for testers to learn what needs to be learned...
- *Auditability:* can the product and its creation be validated?

Maintainability. *Can the product be maintained and extended at low cost?*
- *Flexibility:* the ability to change the product as required by customers.
- *Extensibility:* will it be easy to add features in the future?
- *Simplicity:* the code is not more complex than needed, and does not obscure test design, execution and evaluation.
- *Readability:* the code is adequately documented and easy to read and understand.
- *Transparency:* Is it easy to understand the underlying structures?
- *Modularity:* the code is split into manageable pieces.
- *Refactorability:* are you satisfied with the unit tests?
- *Analyzability:* ability to find causes for defects or other code of interest.

Portability. *Is transferring of the product to different environments enabled?*
- *Reusability:* can parts of the product be re-used elsewhere?
- *Adaptability:* is it easy to change the product to support a different environment?
- *Compatibility:* does the product comply with common interfaces or official standards?
- *Internationalization:* it is easy to translate the product.
- *Localization:* are all parts of the product adjusted to meet the needs of the targeted culture/country?
- *User Interface-robustness:* will the product look equally good when translated?

thetesteye.com v1.0.1

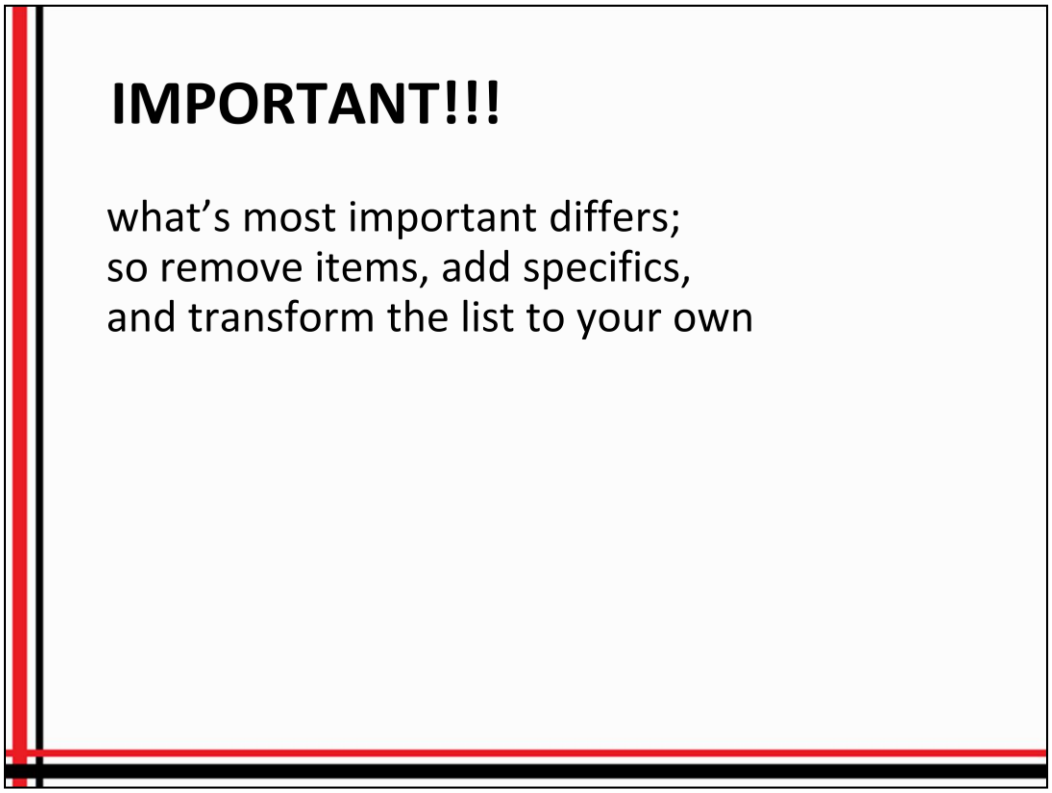
Quality characteristics describe attributes that most software benefit from. They can be used on the whole product or for details.

The whole is made by the details. The quality of a detail is defined by the whole.

This is a thorough extension in the same spirit as Bach's CRUSSPIC STMPL.

Version 1.0 available at http://thetesteye.com/posters/TheTestEye_SoftwareQualityCharacteristics.pdf

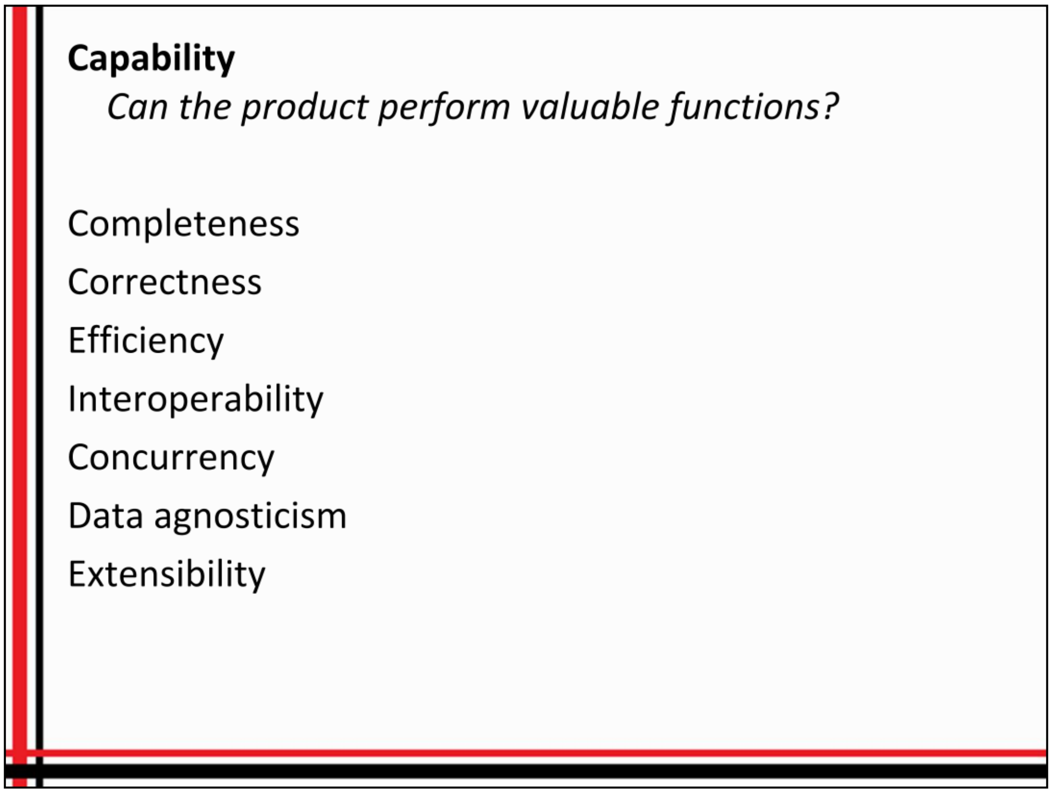
Does anyone have an example application we can use today?



IMPORTANT!!!

what’s most important differs;
so remove items, add specifics,
and transform the list to your own

At my company there are a dozen that always are fruitful.



Capability

Can the product perform valuable functions?

Completeness

Correctness

Efficiency

Interoperability

Concurrency

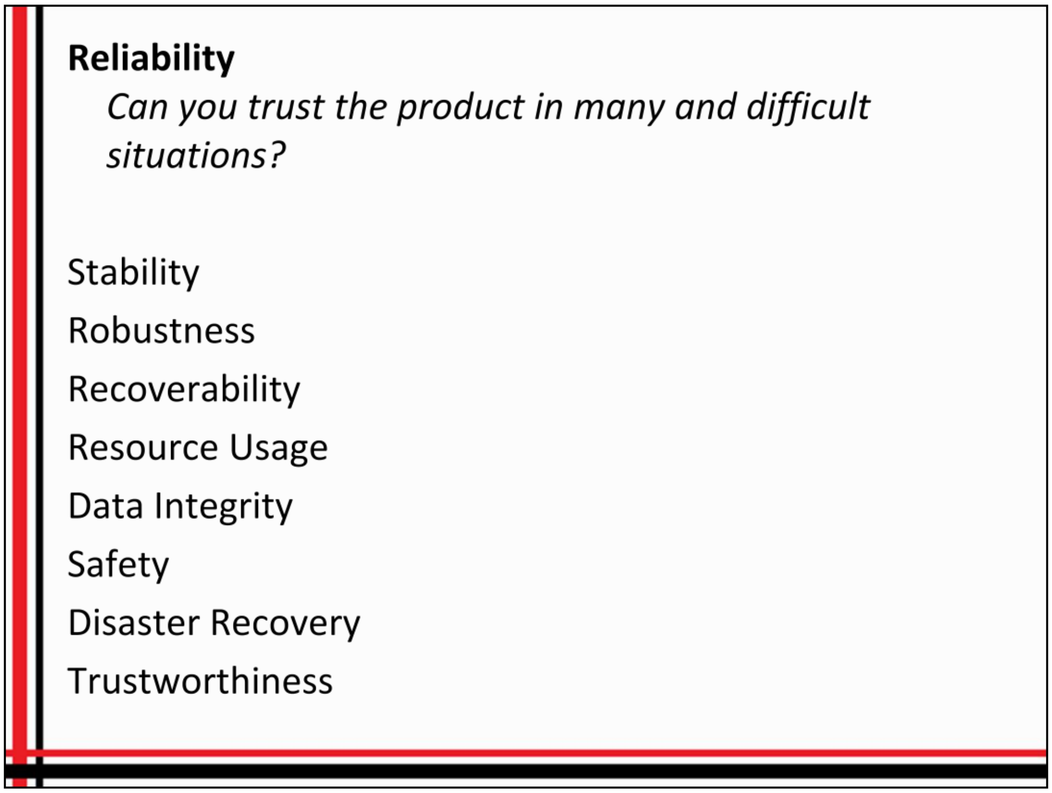
Data agnosticism

Extensibility

Many test efforts stop with capability.

Capability. *Can the product perform valuable functions?*

- *Completeness:* all important functions wanted by end users are available.
- *Correctness:* any output from the product is correct.
- *Efficiency:* performs its actions in an efficient manner (without doing what it's not supposed to do.)
- *Interoperability:* different features interact with each other in the best way.
- *Concurrency:* ability to perform multiple parallel tasks, and run at the same time as other processes.
- *Data agnosticism:* supports all possible data formats, and handles noise.
- *Extensibility:* ability for customers or 3rd parties to add features or change behavior.


Reliability

Can you trust the product in many and difficult situations?

Stability

Robustness

Recoverability

Resource Usage

Data Integrity

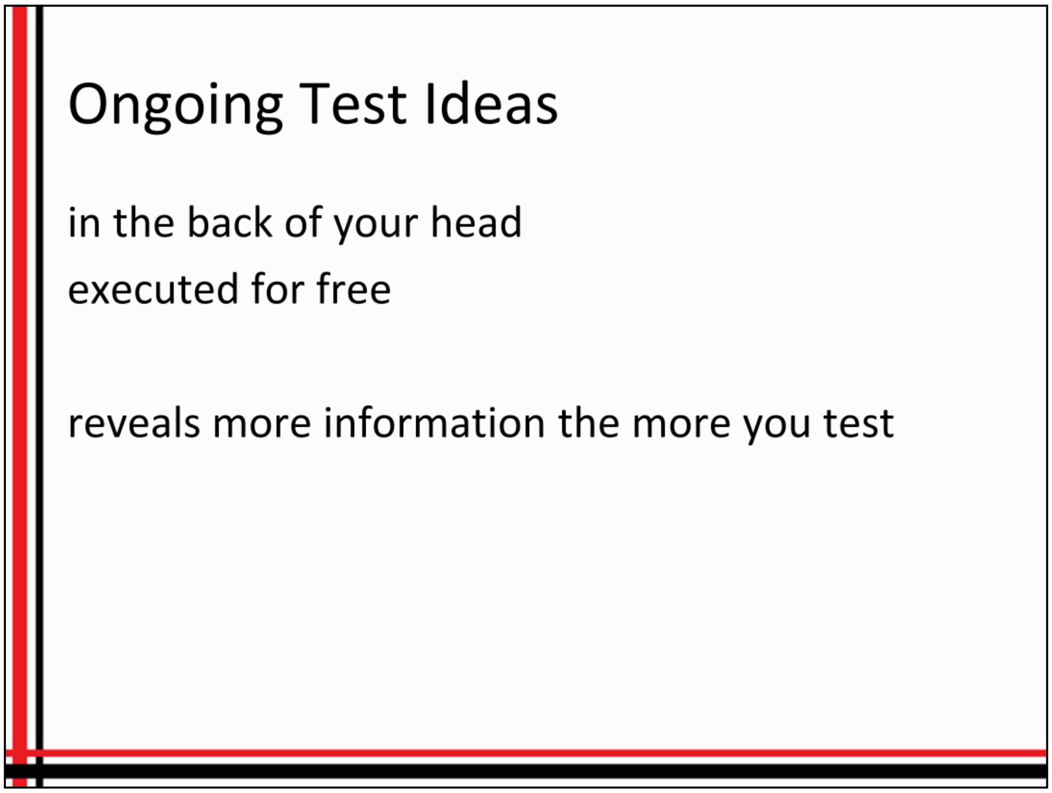
Safety

Disaster Recovery

Trustworthiness

Reliability. *Can you trust the product in many and difficult situations?*

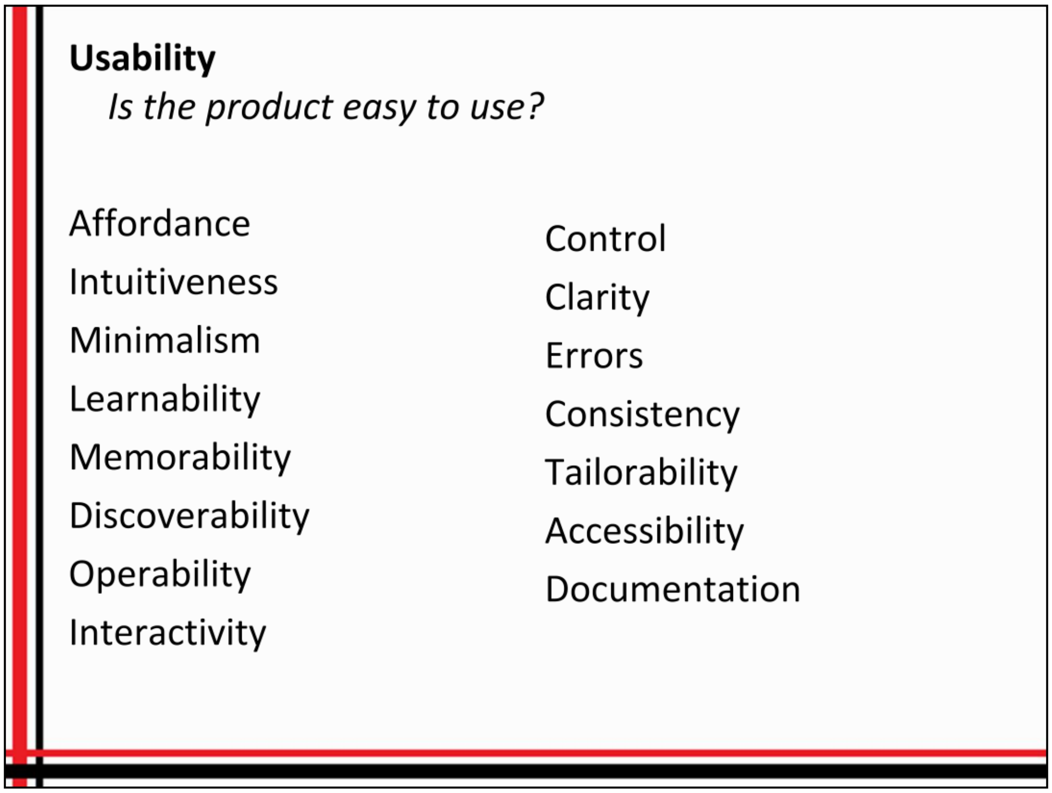
- *Stability:* the product shouldn't cause crashes, unhandled exceptions or script errors.
- *Robustness:* the product handles foreseen and unforeseen errors gracefully.
- *Recoverability:* it is possible to recover and continue using the product after a fatal error.
- *Resource Usage:* appropriate usage of memory, storage and other resources.
- *Data Integrity:* all types of data remain intact throughout the product.
- *Safety:* the product will not be part of damaging people or possessions.
- *Disaster Recovery:* what if something really, really bad happens?
- *Trustworthiness:* is the product's behavior consistent, predictable, and trustworthy?


Ongoing Test Ideas

in the back of your head
executed for free

reveals more information the more you test

things you have in the back of your head all the time
Characteristics (e.g. Stability, Performance) that are executed for free all the time



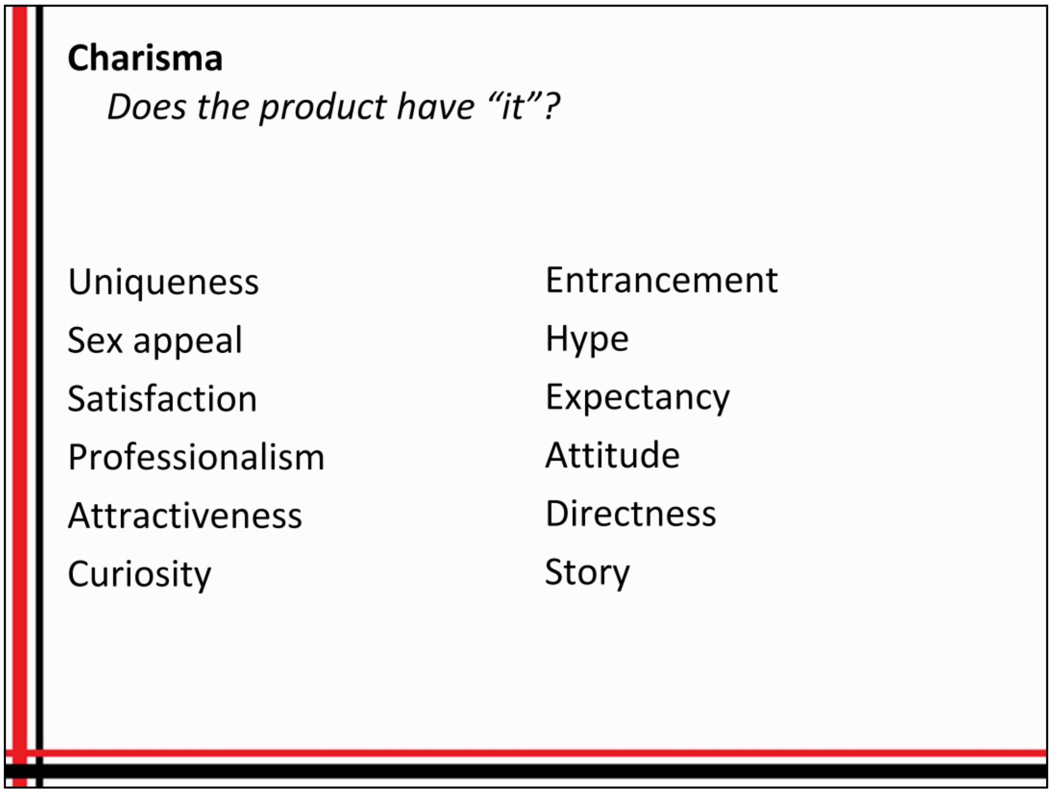
Usability

Is the product easy to use?

Affordance
Intuitiveness
Minimalism
Learnability
Memorability
Discoverability
Operability
Interactivity
Control
Clarity
Errors
Consistency
Tailorability
Accessibility
Documentation

Usability. *Is the product easy to use?*

- *Affordance:* product invites to discover possibilities of the product.
- *Intuitiveness:* it is easy to understand and explain what the product can do.
- *Minimalism:* there is nothing redundant about the product's content and appearance.
- *Learnability:* it is fast and easy to learn and remember how to use the product.
- *Memorability:* once you have learnt to do something you don't forget it.
- *Discoverability:* the product's capabilities can be discovered by systematic exploration of the user interface.
- *Operability:* an experienced user can perform common actions very fast.
- *Interactivity:* the product has easy-to-understand states and possibilities of interacting with the application (via GUI or API).
- *Control:* the user should feel in control over the proceedings of the software.
- *Clarity:* is everything stated explicitly and in detail, with a language that can be understood, leaving no room for doubt?
- *Errors:* there are informative error messages, difficult to make mistakes and easy to repair after making them.
- *Consistency:* behavior is the same throughout the product, and there is one look & feel.
- *Tailorability:* default settings and behavior can be specified for flexibility.
- *Accessibility:* the product is possible to use for as many people as possible, and meets applicable accessibility standards.
- *Documentation:* there is a Help that helps, and matches the functionality.



Charisma

Does the product have "it"?

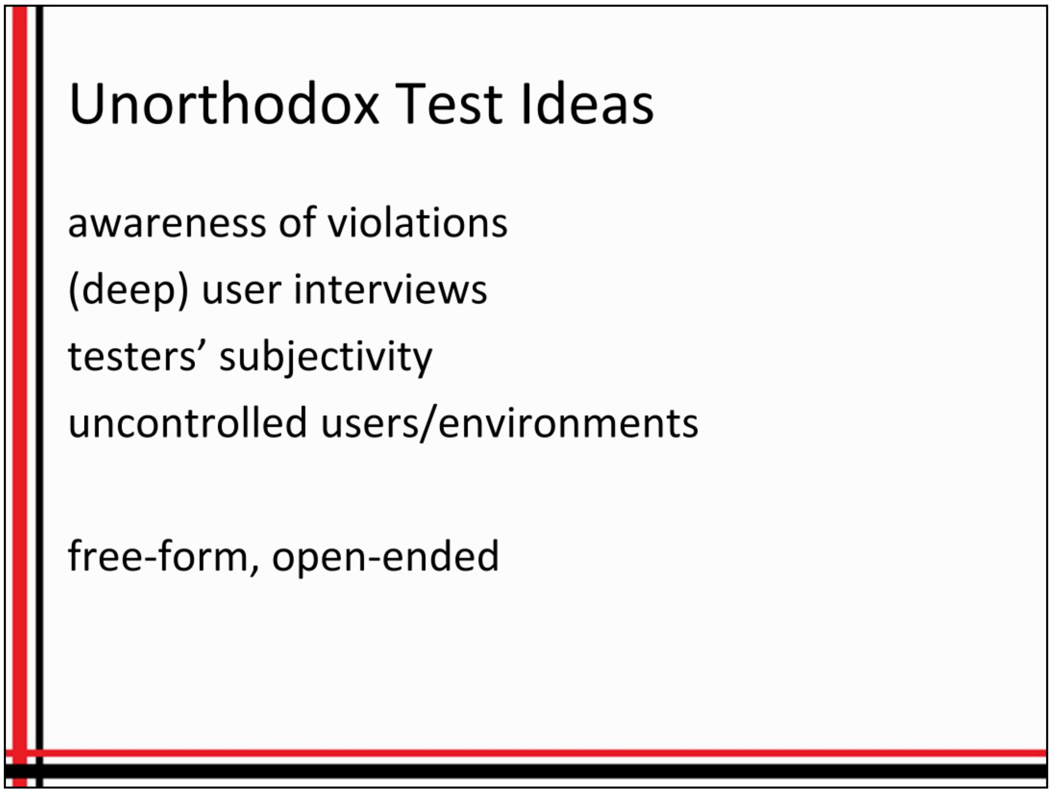
Uniqueness	Entrancement
Sex appeal	Hype
Satisfaction	Expectancy
Professionalism	Attitude
Attractiveness	Directness
Curiosity	Story

Charisma. *Does the product have "it"?*

- *Uniqueness:* the product is distinguishable and has something no one else has.
- *Sex appeal:* you just can't stop looking at or using the product.
- *Satisfaction:* how does it feel after using the product?
- *Professionalism:* does the product have the appropriate flair of professionalism and feel fit for purpose?
- *Attractiveness:* are all types of aspects of the product "good-looking"?
- *Curiosity:* will users get interested and try out what they can do with the product?
- *Entrancement:* do users get hooked, have fun, in a flow, and fully engaged when using the product?
- *Hype:* does the product use too much or too little of the latest and greatest technologies/ideas?
- *Expectancy:* the product exceeds expectations and meets the needs you didn't know you had.
- *Attitude:* do the product and its information have the right attitude and speak to you with the right language and style?
- *Directness:* are (first) impressions impressive?
- *Story:* are there compelling stories about the product's inception, construction or usage?

I'm confident you agree that many of these are important aspects for the users' perception of your product.

Then how come you don't test charisma??



Unorthodox Test Ideas

awareness of violations

(deep) user interviews

testers' subjectivity

uncontrolled users/environments

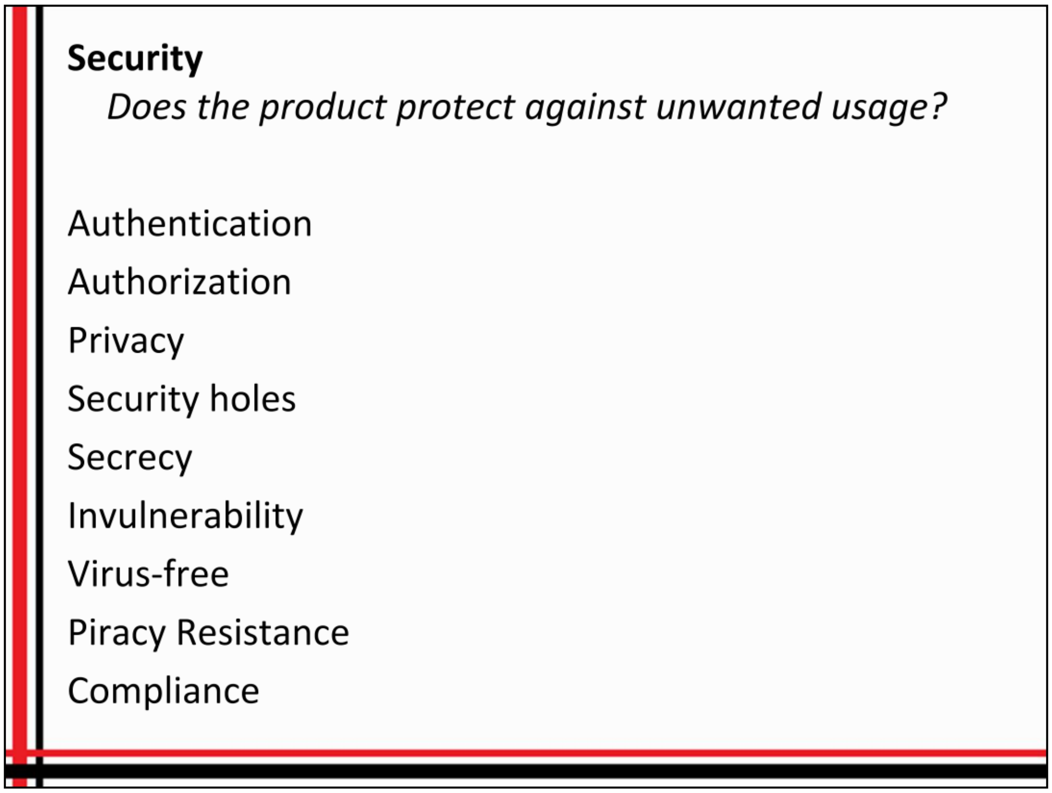
free-form, open-ended

Things like Charisma aren't easily captured in tests.

You have to take alternative actions.

free-form: E.g. think about Charisma for a feature, and brainstorm around what can be done testing-wise.

Get rid of Pass/Fail-thinking, we can communicate noteworthy information.



Security

Does the product protect against unwanted usage?

Authentication

Authorization

Privacy

Security holes

Secrecy

Invulnerability

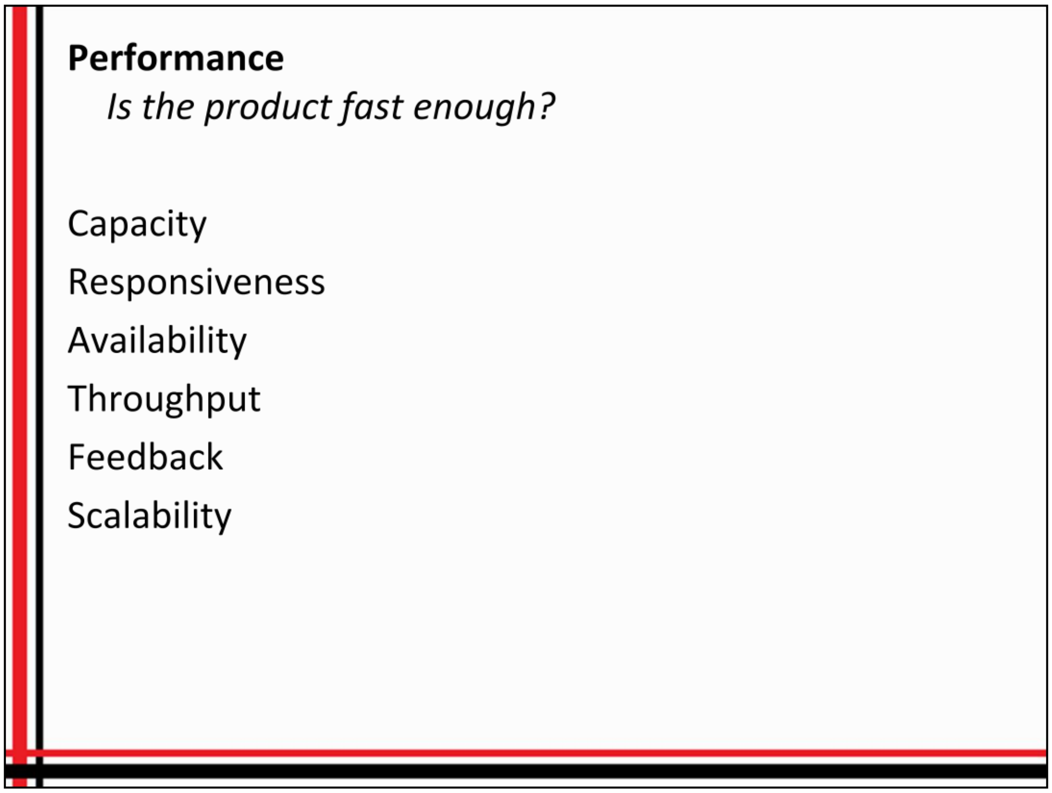
Virus-free

Piracy Resistance

Compliance

Security. *Does the product protect against unwanted usage?*

- *Authentication*: the product's identifications of the users.
- *Authorization*: the product's handling of what an authenticated user can see and do.
- *Privacy*: ability to not disclose data that is protected to unauthorized users.
- *Security holes*: product should not invite to social engineering vulnerabilities.
- *Secrecy:* the ability to under no circumstances disclose information about the underlying systems.
- *Invulnerability:* ability to withstand penetration attempts.
- *Virus-free:* product will not transport virus, or appear as one.
- *Piracy Resistance:* no possibility to illegally copy and distribute the software or code.
- *Compliance:* security standards the product adheres to.



Performance

Is the product fast enough?

Capacity

Responsiveness

Availability

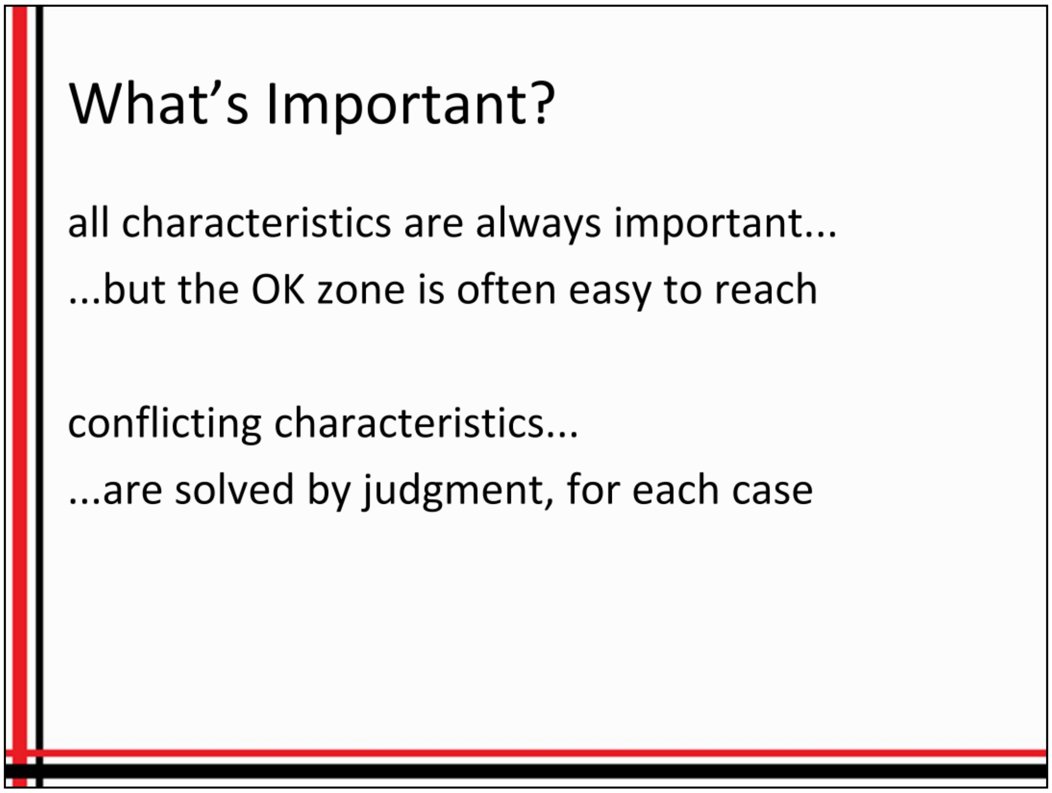
Throughput

Feedback

Scalability

Performance. *Is the product fast enough?*

- *Capacity:* the many limits of the product, also under load, stress or slow network.
- *Responsiveness:* the speed of which an action is (perceived as) performed.
- *Availability:* the system is available for use when it should be.
- *Throughput:* the products ability to process many, many things.
- *Feedback:* is the feedback from the system on user actions appropriate?
- *Scalability*: how well does the product scale up, out or down?

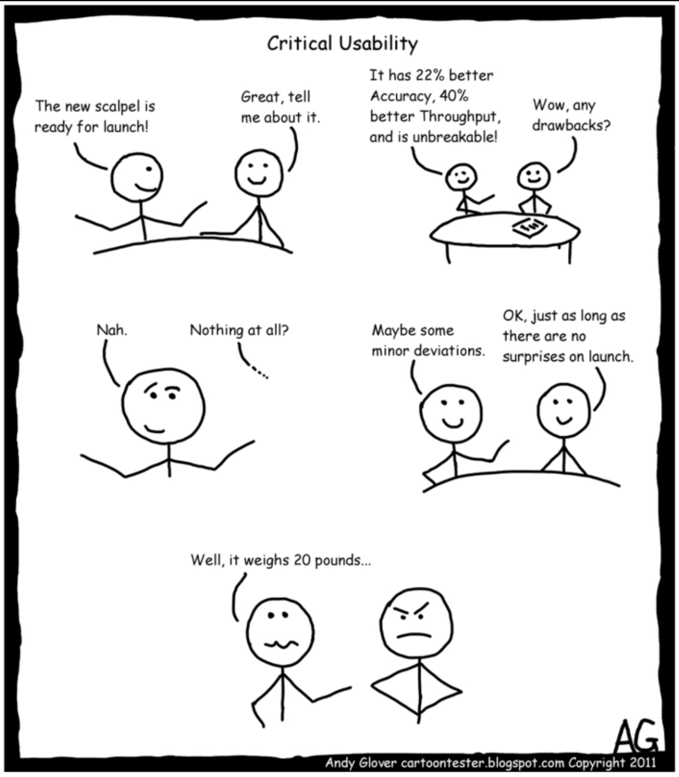


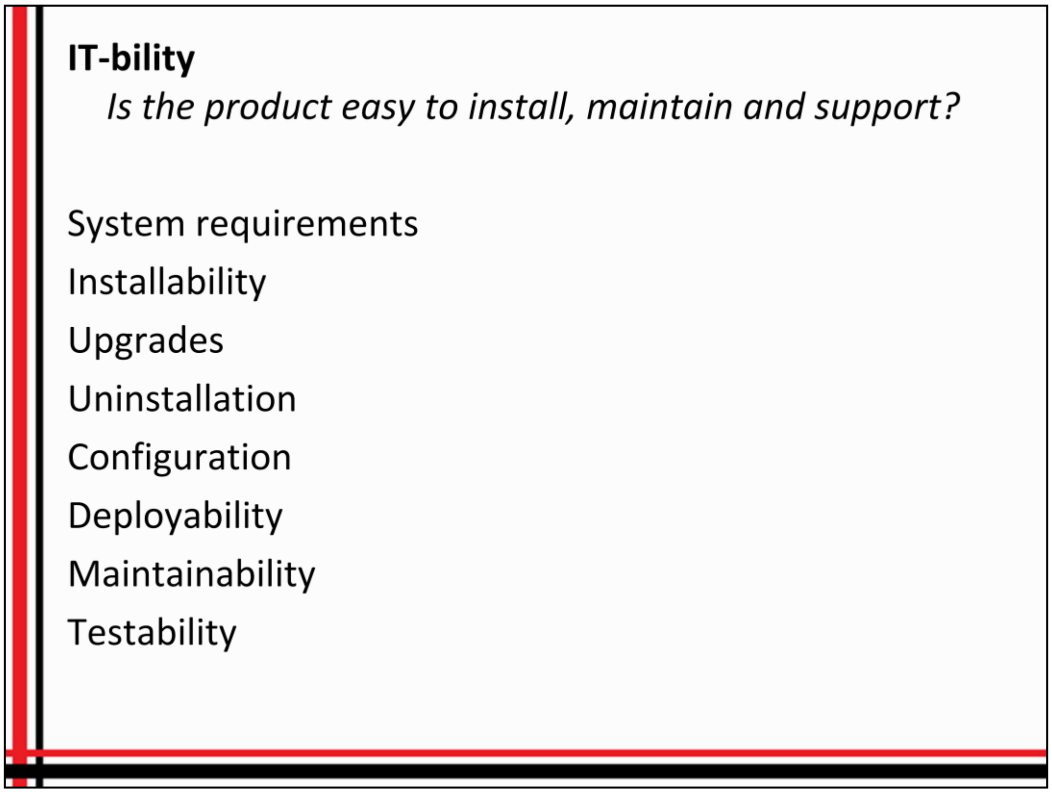
What’s Important?

all characteristics are always important...
...but the OK zone is often easy to reach

conflicting characteristics...
...are solved by judgment, for each case

Judgment needs a lot of knowledge, collaboration and feelings.


Critical Usability
The new scalpel is ready for launch!
Great, tell me about it.
It has 22% better Accuracy, 40% better Throughput, and is unbreakable!
Wow, any drawbacks?
Nah.
Nothing at all?
Maybe some minor deviations.
OK, just as long as there are no surprises on launch.
Well, it weighs 20 pounds...
AG
Andy Glover cartoontester.blogspot.com Copyright 2011


IT-bility

Is the product easy to install, maintain and support?

System requirements

Installability

Upgrades

Uninstallation

Configuration

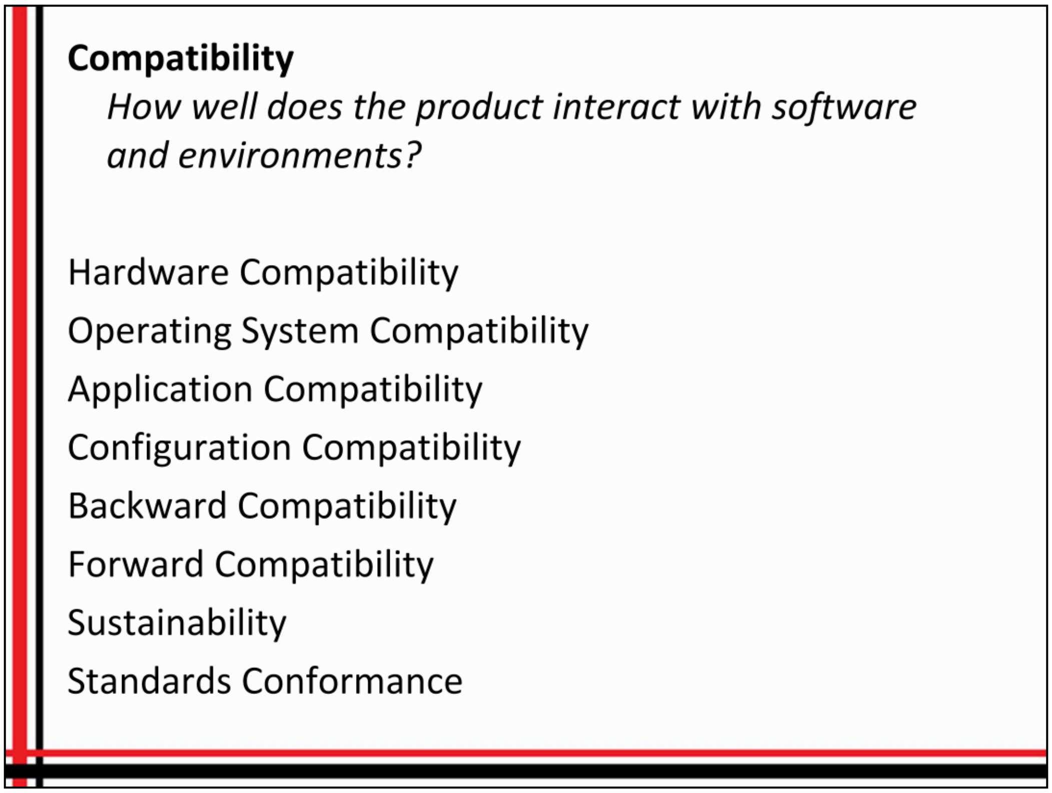
Deployability

Maintainability

Testability

IT-bility. *Is the product easy to install, maintain and support?*

- *System requirements:* ability to run on supported configurations, and handle different environments or missing components.
- *Installability:* product can be installed on intended platforms with appropriate footprint.
- *Upgrades:* ease of upgrading to a newer version without loss of configuration and settings.
- *Uninstallation:* are all files (except user's or system files) and other resources should be removed when uninstalling?
- *Configuration:* can the installation be configured in various ways or places to support customer's usage?
- *Deployability:* product can be rolled-out by IT department to different types of (restricted) users and environments.
- *Maintainability:* are the product and its artifacts easy to maintain and support for customers?
- *Testability:* how effectively can the deployed product be tested by the customer?



Compatibility

How well does the product interact with software and environments?

Hardware Compatibility

Operating System Compatibility

Application Compatibility

Configuration Compatibility

Backward Compatibility

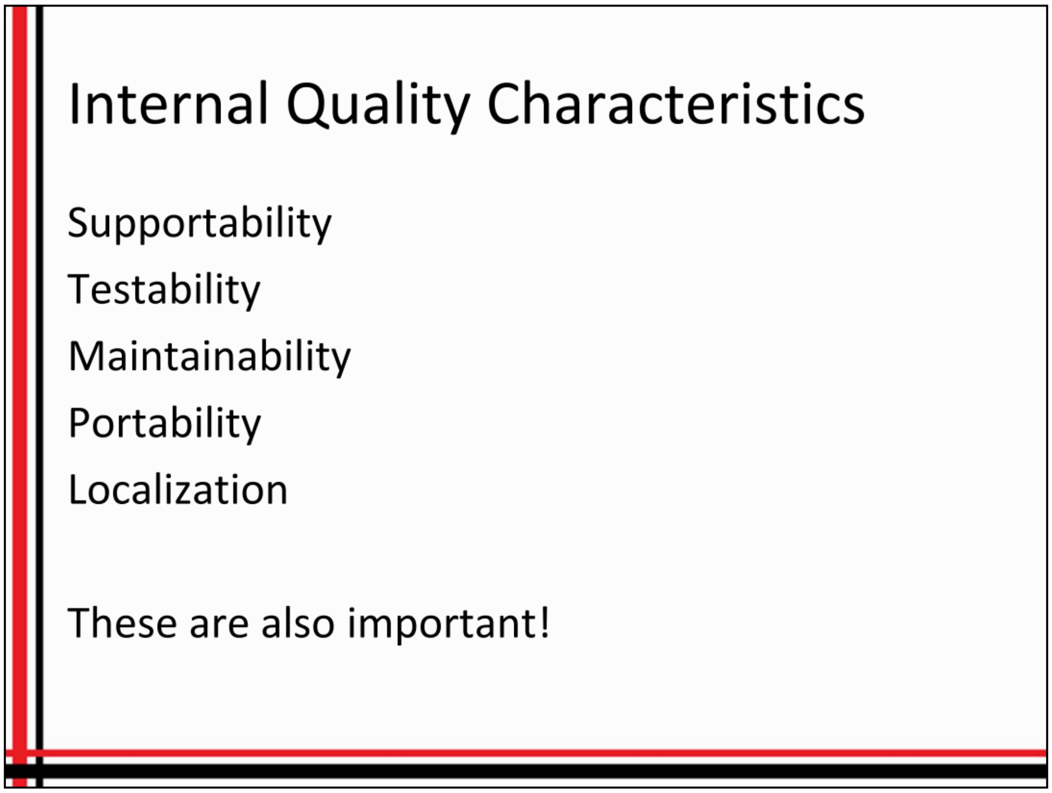
Forward Compatibility

Sustainability

Standards Conformance

Compatibility. *How well does the product interact with software and environments?*

- *Hardware Compatibility*: the product can be used with applicable configurations of hardware components.
- *Operating System Compatibility*: the product can run on intended operating systems, and follows typical behavior.
- *Application Compatibility:* the product, and its data, works with other applications customers are likely to use.
- *Configuration Compatibility*: product’s ability to blend in with configurations of the environment.
- *Backward Compatibility:* can the product do everything the last version could?
- *Forward Compatibility:* will the product be able to use artifacts or interfaces of future versions?
- *Sustainability:* effects on the environment, e.g. energy efficiency, switch-offs, power-saving modes, support work from home.
- *Standards Conformance:* the product conforms to applicable standards, regulations, laws or ethics.



Internal Quality Characteristics

Supportability

Testability

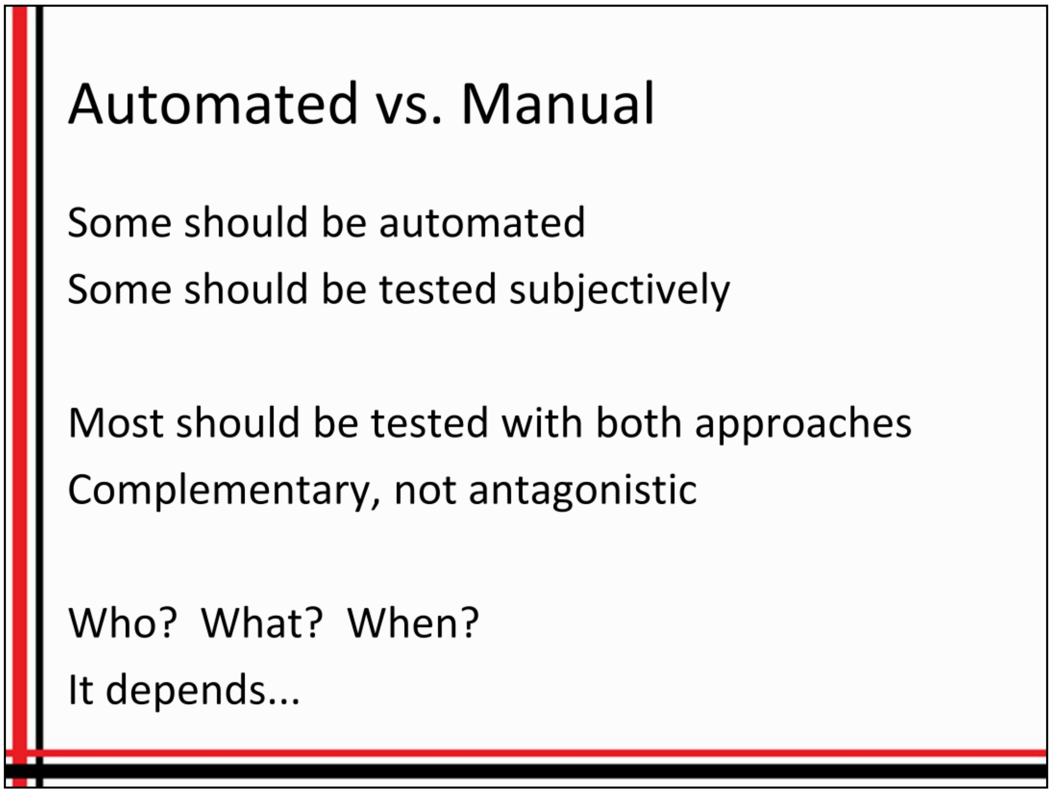
Maintainability

Portability

Localization

These are also important!

...but we will not go into details today.



Automated vs. Manual

Some should be automated

Some should be tested subjectively

Most should be tested with both approaches

Complementary, not antagonistic

Who? What? When?

It depends...

Some, but not all, of these characteristics can be quantified and made regressionable as automated tests.

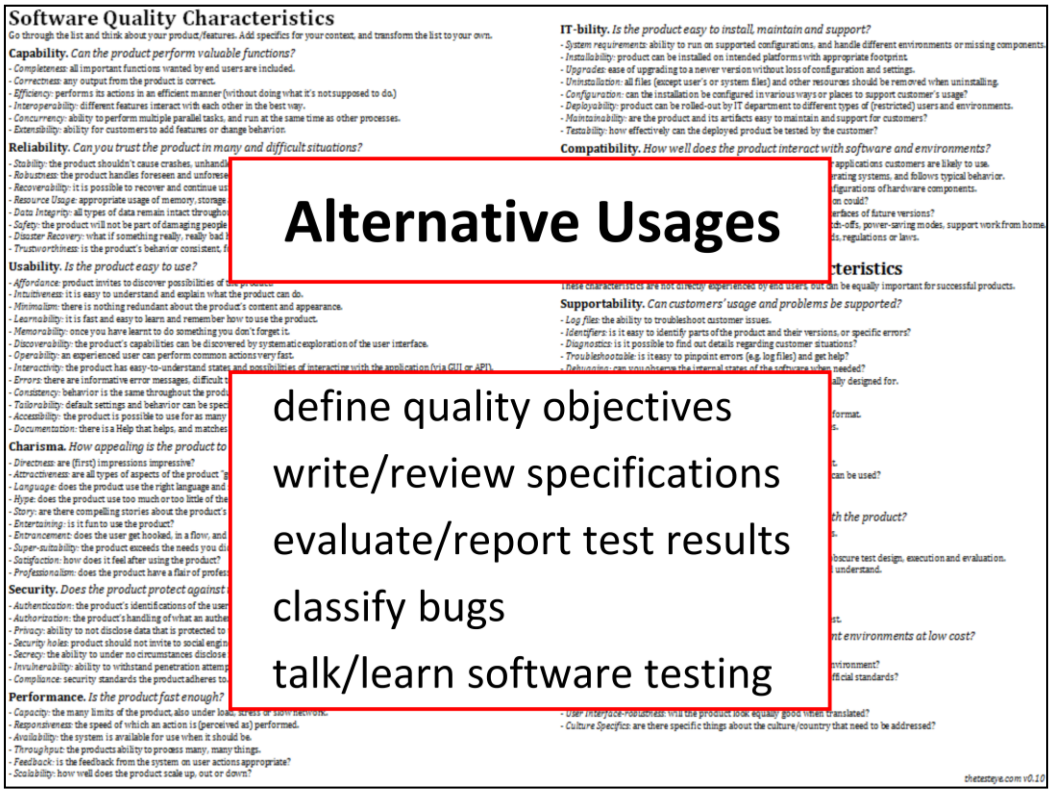
Some are better tested manually and subjectively.

For most, both automated and manual approaches are recommended, because repetitive automated tests and changing manual tests are complementary, not antagonistic. If they feel in conflict, re-check your thinking.

Computers are faster, humans are better judges of importance.

A perfect scenario could be where developers create automated tests for the ”checking”, and system testers can use diverse approaches to do the necessary testing (without being bothered by ”easy” bugs.)

If you’re doing regression testing, automation is closer. When testing something new, manual is a better fit. Tools can always be used!



Software Quality Characteristics

Go through the list and think about your product/features. Add specifics for your context, and transform the list to your own.

Capability. *Can the product perform valuable functions?*
- *Completeness:* all important functions wanted by end users are included.
- *Correctness:* any output from the product is correct.
- *Efficiency:* performs its actions in an efficient manner (without doing what it's not supposed to do.)
- *Interoperability:* different features interact with each other in the best way.
- *Concurrency:* ability to perform multiple parallel tasks, and run at the same time as other processes.
- *Extensibility:* ability for customers to add features or change behavior.

Reliability. *Can you trust the product in many and difficult situations?*
- *Stability:* the product shouldn't cause crashes, unhandl
- *Robustness:* the product handles foreseen and unforese
- *Recoverability:* it is possible to recover and continue us
- *Resource Usage:* appropriate usage of memory, storage
- *Data Integrity:* all types of data remain intact througho
- *Safety:* the product will not be part of damaging people
- *Disaster Recovery:* what if something really, really bad h
- *Trustworthiness:* is the product's behavior consistent, f

Usability. *Is the product easy to use?*
- *Affordance:* product invites to discover possibilities of
- *Intuitiveness:* it is easy to understand and explain what the product can do.
- *Minimalism:* there is nothing redundant about the product's content and appearance.
- *Learnability:* it is fast and easy to learn and remember how to use the product.
- *Memorability:* once you have learnt to do something you don't forget it.
- *Discoverability:* the product's capabilities can be discovered by systematic exploration of the user interface.
- *Operability:* an experienced user can perform common actions very fast.
- *Interactivity:* the product has easy-to-understand states and possibilities of interacting with the application (via GUI or API).
- *Errors:* there are informative error messages, difficult t
- *Consistency:* behavior is the same throughout the produ
- *Tailorability:* default settings and behavior can be speci
- *Accessibility:* the product is possible to use for as many
- *Documentation:* there is a Help that helps, and matches

Charisma. *How appealing is the product to*
- *Directness:* are (first) impressions impressive?
- *Attractiveness:* are all types of aspects of the product "g
- *Language:* does the product use the right language and
- *Hype:* does the product use too much or too little of the
- *Story:* are there compelling stories about the product's
- *Entertaining:* is it fun to use the product?
- *Entrancement:* does the user get hooked, in a flow, and
- *Super-suitability:* the product exceeds the needs you di
- *Satisfaction:* how does it feel after using the product?
- *Professionalism:* does the product have a flair of profess

Security. *Does the product protect against*
- *Authentication:* the product's identifications of the user
- *Authorization:* the product's handling of what an authe
- *Privacy:* ability to not disclose data that is protected to
- *Security holes:* product should not invite to social engin
- *Secrecy:* the ability to under no circumstances disclose
- *Invulnerability:* ability to withstand penetration attemp
- *Compliance:* security standards the product adheres to.

Performance. *Is the product fast enough?*
- *Capacity:* the many limits of the product, also under load, stress or slow network.
- *Responsiveness:* the speed of which an action is (perceived as) performed.
- *Availability:* the system is available for use when it should be.
- *Throughput:* the products ability to process many, many things.
- *Feedback:* is the feedback from the system on user actions appropriate?
- *Scalability:* how well does the product scale up, out or down?

IT-bility. *Is the product easy to install, maintain and support?*
- *System requirements:* ability to run on supported configurations, and handle different environments or missing components.
- *Installability:* product can be installed on intended platforms with appropriate footprint.
- *Upgrades:* ease of upgrading to a newer version without loss of configuration and settings.
- *Uninstallation:* all files (except user's or system files) and other resources should be removed when uninstalling.
- *Configuration:* can the installation be configured in various ways or places to support customer's usage?
- *Deployability:* product can be rolled-out by IT department to different types of (restricted) users and environments.
- *Maintainability:* are the product and its artifacts easy to maintain and support for customers?
- *Testability:* how effectively can the deployed product be tested by the customer?

Compatibility. *How well does the product interact with software and environments?*

r applications customers are likely to use.
erating systems, and follows typical behavior.
figurations of hardware components.
on could?
erfaces of future versions?
ch-offs, power-saving modes, support work from home.
ds, regulations or laws.

cteristics

These characteristics are not directly experienced by end users, but can be equally important for successful products.

Supportability. *Can customers' usage and problems be supported?*
- *Log files:* the ability to troubleshoot customer issues.
- *Identifiers:* is it easy to identify parts of the product and their versions, or specific errors?
- *Diagnostics:* is it possible to find out details regarding customer situations?
- *Troubleshootable:* is it easy to pinpoint errors (e.g. log files) and get help?
- *Debugging:* can you observe the internal states of the software when needed?

ally designed for.

format.
s.

t.
can be used?

th the product?
s.

bscure test design, execution and evaluation.
l understand.

st.
nt environments at low cost?

nvironment?
fficial standards?

- *User Interface-robustness:* will the product look equally good when translated?
- *Culture Specifics:* are there specific things about the culture/country that need to be addressed?

thetesteye.com v0.10

Alternative Usages

define quality objectives

write/review specifications

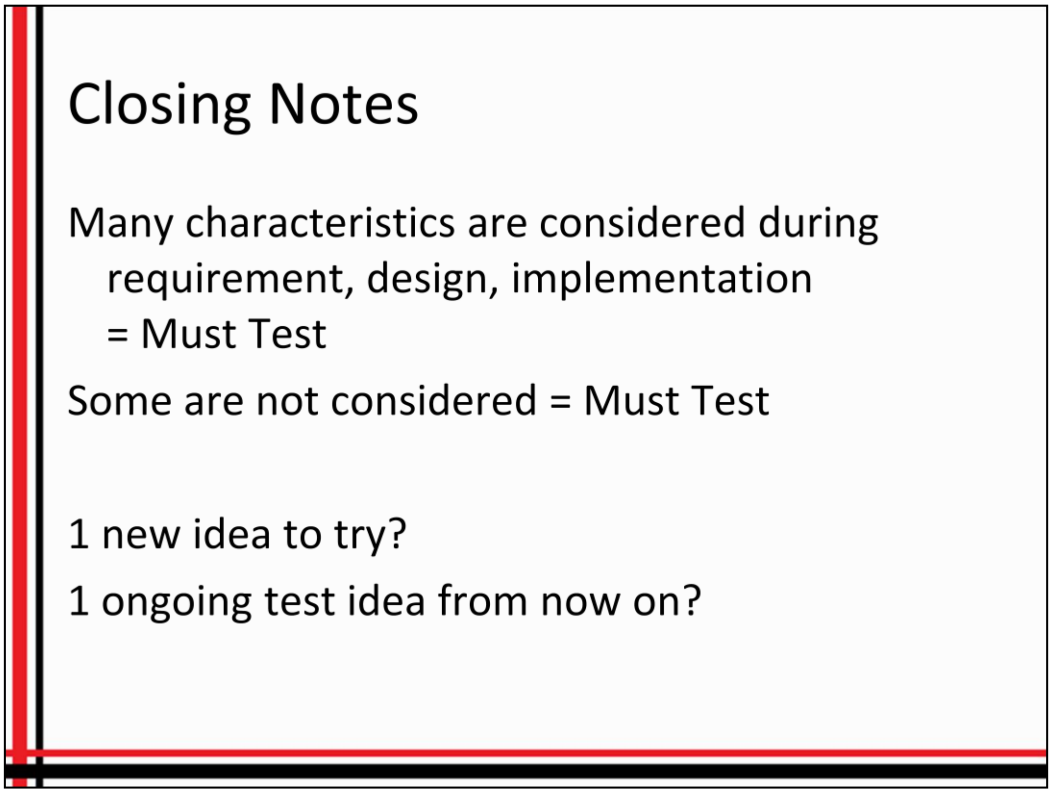
evaluate/report test results

classify bugs

talk/learn software testing

It is fast to write twenty guiding statements about the most important characteristics for your system.

You can use the list to create your own, shorter, better definition of quality characteristics.



Closing Notes

Many characteristics are considered during requirement, design, implementation
= Must Test

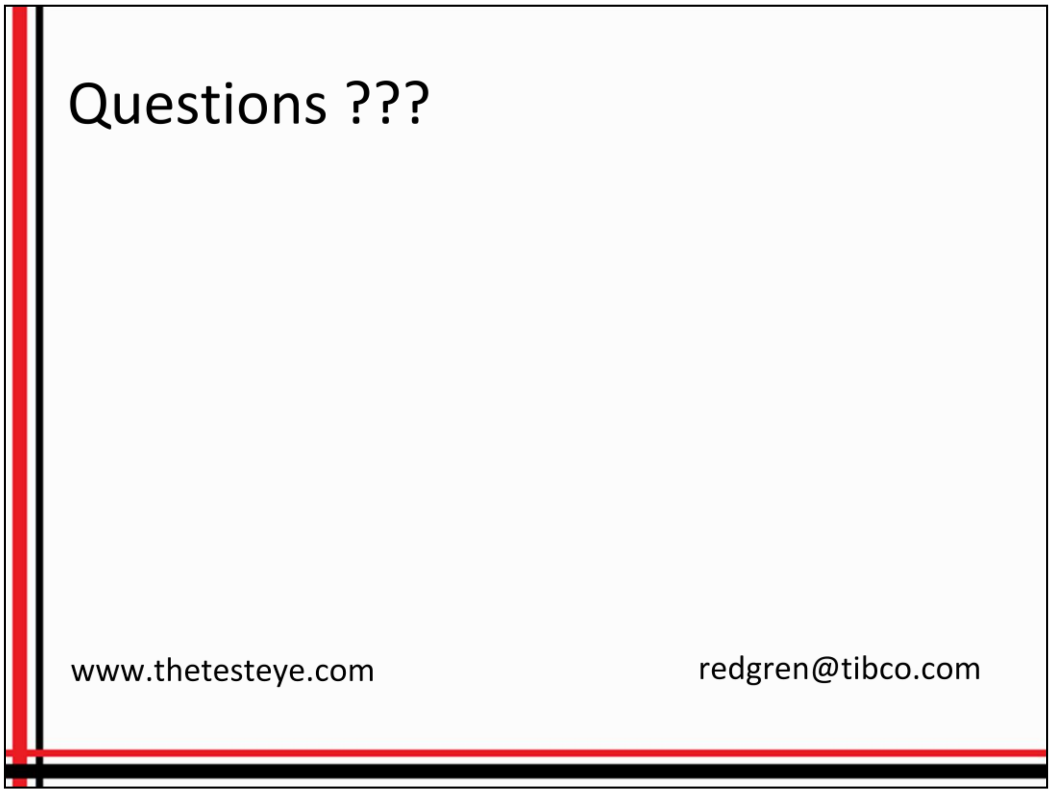
Some are not considered = Must Test

1 new idea to try?

1 ongoing test idea from now on?

A lot of these have been thought about during requirements, design and implementation; so we must test them.

Those that haven't been considered, yet still important, we really must test.



Questions ???

www.thetesteye.com

redgren@tibco.com

Isn’t it better to use the list in the beginning of the project?

- One way doesn’t rule out the other. But I’m a tester, and testers are my main audience.

I counted to 8 categories and 73 sub-characteristics; does that add up to 77?

- I have no comments on that remark ;)