Contents

Risk Mitigation Using Exploratory and Technical Testing

This document has slides and notes for a webinar presented on 28th July 2016 in partnership between QASymphony and Alan Richardson of Compendium Developments Ltd

This document is not a transcript, it is a set of notes to expand on the slides. Where the slide is text and self explanatory we don't add a lot of notes.

Books:

- "Dear Evil Tester",
- "Java For Testers"
- and "Selenium Simplified";

Online training courses on:

- Technical web testing,
- Java and Selenium WebDriver
- Java and Technical testing

Websites and blogs:


OUR PRESENTER
Guest Speaker: Alan Richardson
• Alan has worked in Software Development for over 20 years; as a programmer, tester, test manager. As an independent test consultant he helps organisations improve their agility, technical skills and testing processes. Alan wrote the books "Dear Evil Tester", "Java For Testers" and "Selenium Simplified"; he also created online training courses on technical web testing, Java and Selenium WebDriver.
• Alan blogs at EvilTester.com, SeleniumSimplified.com, and JavaForTesters.com; you can find information on his consultancy, training and conference talks at CompendiumDev.co.uk. Follow him on twitter as @EvilTester.
Dear Evil Tester
Alan Richardson
Java For Testers
Alan Richardson
CompendiumDev.co.uk
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 1:

- EvilTester.com,
- seleniumsimplified.com,
- and JavaForTesters.com

You can find information on his consultancy, training and conference talks at:

- Compendium Developments Ltd compendiumdev.co.uk.

Follow him on twitter as [@EvilTester](http://twitter.com/eviltester)

A Commonsense Guide to Risk

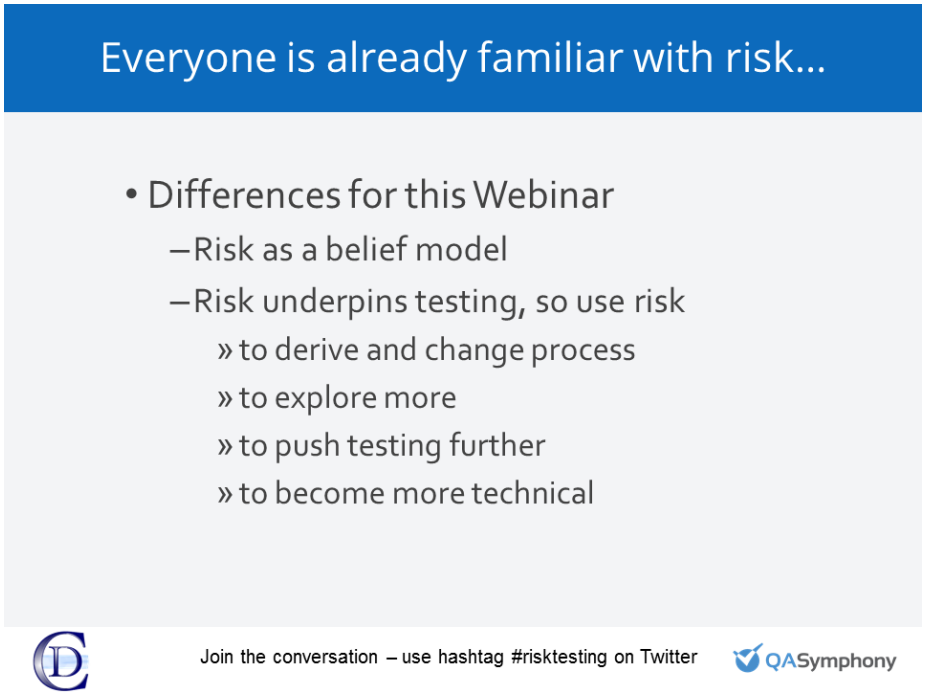


Figure 2:

I want to get across the model that I have for risks, which is that risks are "beliefs" and a result of our beliefs. We believe some things will go wrong more than others. And because our beliefs are limited but the range of risks is not, we need to somehow go beyond our beliefs and look at tools and processes for doing that.

Also we know that risk is important for testing. What I want to do in this talk is present risk as the underpinning and driving force behind everything we do in testing.

You can use risk to justify why you test, what you test and how you test.

Figure 3:

A commonsense risk process is:

- Identify
- Mitigate
- Detect
- Accept

Essentially, we scare ourselves silly to 'identify' risk.

Then decide what we are going to do something about 'mitigate'.

What we are going to keep an eye out for (just in case) - 'detect'.

And what we are going to shrug our shoulders at 'accept'. But at least we know what we are prepared to live with, and can tell people if they ask.

So with the Webinar...

- Identify
- Mitigate


Commonsense Webinar Risks
What might go wrong
With me
What if I'm ill? What if I still have a cold and can't talk?
What if I forget what I'm talking about?
With my broadband
What if the connection drops? What if the speed is poor?
With my computer
What if it crashes?
With the webinar system
What if it stops?
What might go wrong
With the phone?
What if it cuts out?
With the locale
What if there is a power cut?
With the content
What if I bore people?
What if they drop out?
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 4:

  - Illness – sleep more, pre-record webinar just in case
  - Forget – presenter notes, practice
  - Broadband – give slides to host
  - Computer crash – multiple computers
  - Power cut – battery, UPS
  - Phone – landline, mobile
- Accept
  - Boredom, Drop out (partially mitigate with presentation style and some images, roughly 1 slide every 33.8 seconds to keep interest up)
- Detect
  - Have computer watching webinar – risk: impacts Mbps

If I try to detect by having another computer 'watch' the webinar then there is a risk that that increases the bandwidth used and slows down the webinar. (So that 'detction' process creates a RISK).

Figure 5:

A simple risk process is to 'fear everything'. If there is something in your system that you do not have anything to fear about it, then you haven't identified enough risk.

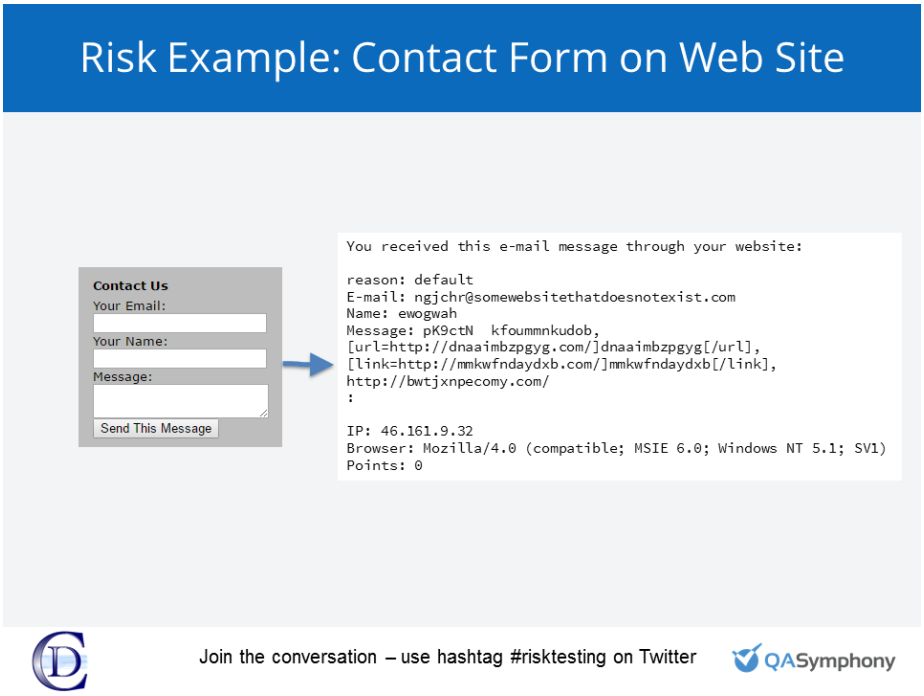

Risk Example: Contact Form on Web Site
Contact Us
Your Email:
Your Name:
Message:
Send This Message
You received this e-mail message through your website:
reason: default
E-mail: ngjchr@somewebsitethatdoesnotexist.com
Name: ewogwah
Message: pK9ctN kfoummnkudob,
[url=http://dnaaimbzpgyg.com/]dnaaimbzpgyg[/url],
[link=http://mmkwfndaydxb.com/]mmkwfndaydxb[/link],
http://bwtjxnpecomy.com/
:
IP: 46.161.9.32
Browser: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
Points: 0
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 6:

Risk Management Process Simplified.

Risk that form becomes a spam magnet by bots.

Impact – I have to deal with a lot of spam in my inbox.

Mitigation - Add Captcha to contact form.

Risk – no-one contacts me because of the captcha.

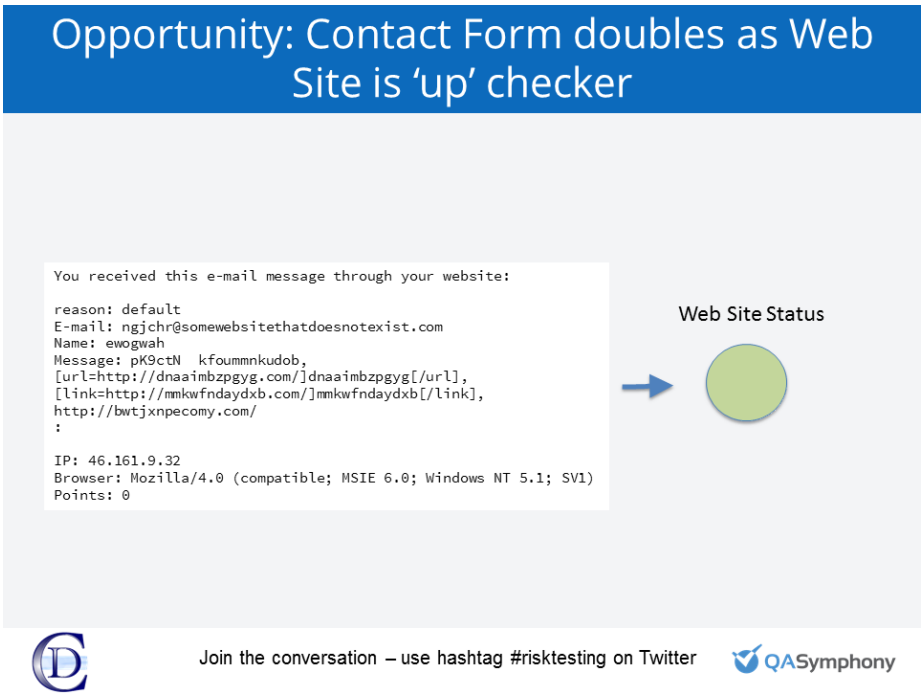


Figure 7:

When contact form becomes a spam magnet by bots I have to deal with a lot of spam in my inbox.

And I know that the website is still up and running and don't have to use an uptime checker and visit the site to check reports and see if my site is still running.

Spam saves me a fortune in maintaining automated scripts to monitor my website.

We view risk as a 'bad' thing

If we can model it differently we can use it as an opportunity to expand the scope of what we test and how we conduct process improvement.


Risk & Opportunity
What is risk?
Something that might go wrong
Probability
Impact
Opportunity for testing
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 8:

Risk and Testing

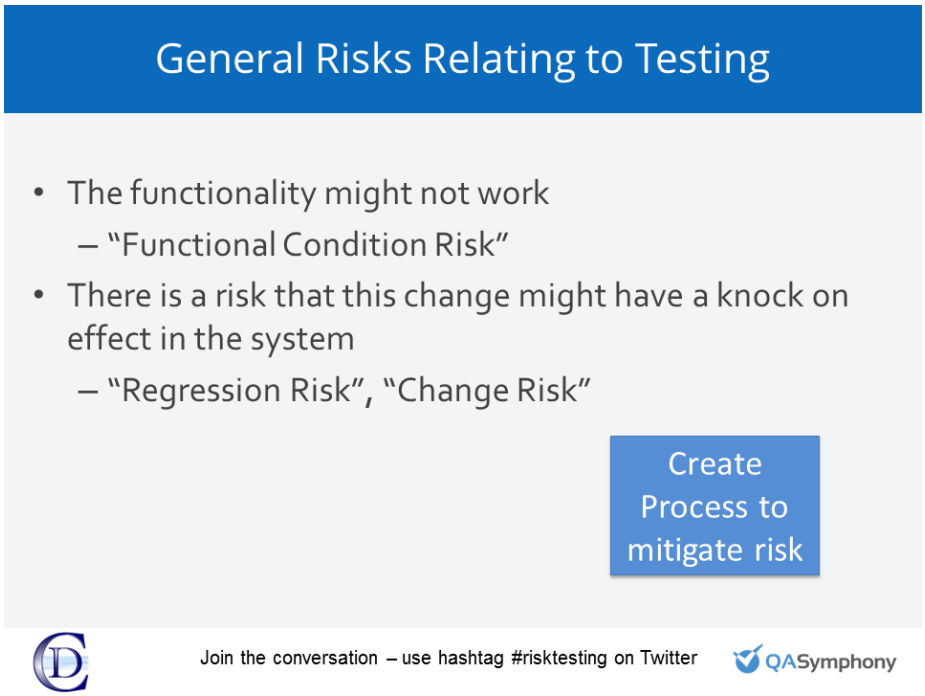


Figure 9:

Regression testing is not a risk mitigation. It is often a process added to a test approach without even having a 'risk' profile. It is most often something we do, because we are testing. Rather than something we do because we are trying to mitigate risk.

If we seriously wanted to mitigate regression risk we would use a QA process, not a testing process.

You probably already use risk in your testing.

- Business Risk
- Project Risk
- Functional Risk

Typical Risk Modeling: Business Risks

e.g.

- We might run out of funding

- Our requirements might be wrong
- We might be hit by a regulatory requirement

Less likely to be used for testing

We usually mean project and functional risk:

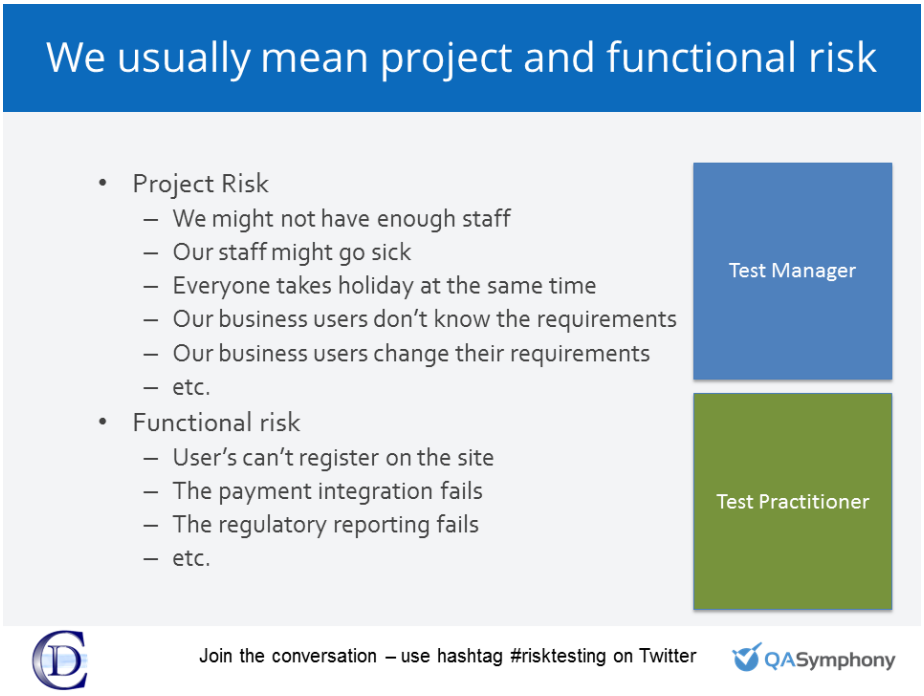


Figure 10:

Very easy to spend a lot of time trying to:

- Quantify probability
- Quantify Impact
- Use a 'formula' for deciding' priority

To avoid (and because of fear of. . .)

- Deciding what to focus on in case we
- Choosing the wrong thing
- Picking the wrong risk
- Blame


Risk
What is risk?
Something that might go wrong
Probability
Impact
People get hung up here
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 11:

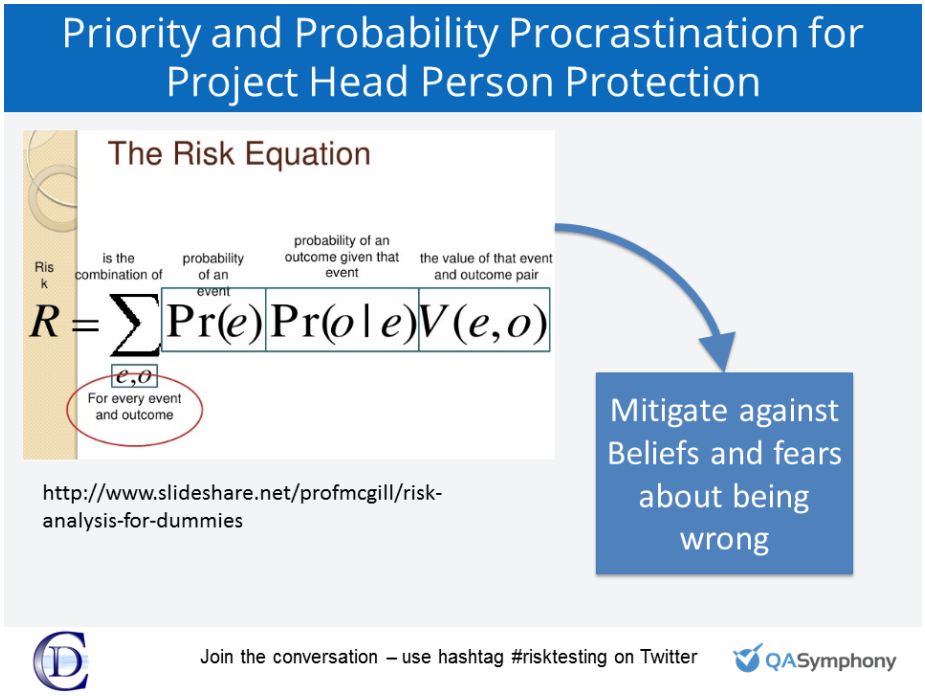

Priority and Probability Procrastination for Project Head Person Protection
The Risk Equation
Risk
is the combination of
probability of an event
probability of an outcome given that event
the value of that event and outcome pair
$R = \sum_{e,o} \Pr(e) \Pr(o \mid e) V(e,o)$
For every event and outcome
http://www.slideshare.net/profmcgill/risk-analysis-for-dummies
Mitigate against Beliefs and fears about being wrong
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 12:

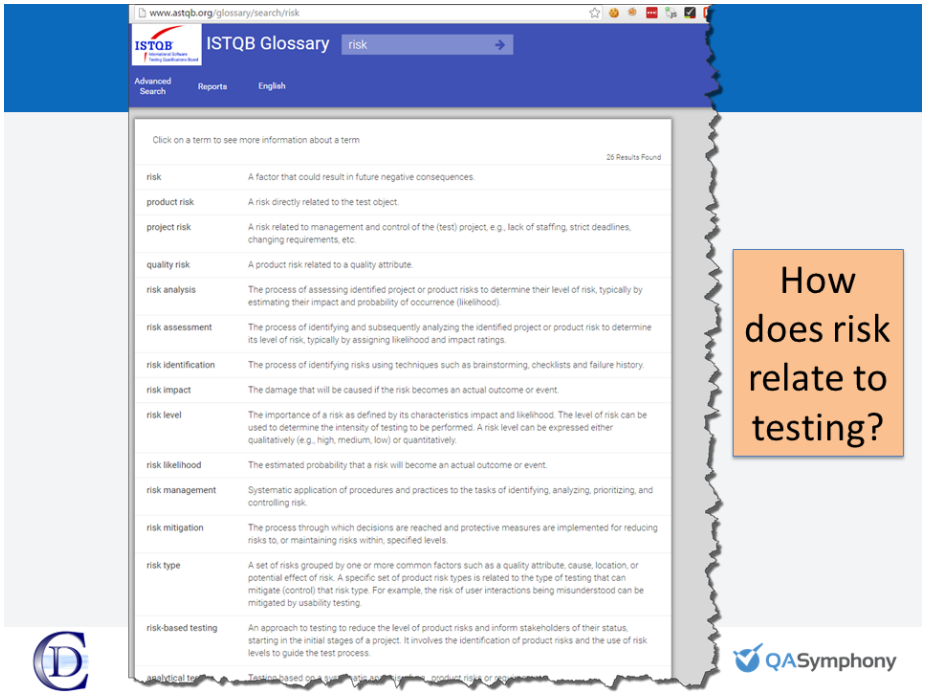

www.astqb.org/glossary/search/risk
ISTQB Glossary
risk
Advanced Search
Reports
English
Click on a term to see more information about a term
26 Results Found
risk
A factor that could result in future negative consequences.
product risk
A risk directly related to the test object.
project risk
A risk related to management and control of the (test) project, e.g., lack of staffing, strict deadlines, changing requirements, etc.
quality risk
A product risk related to a quality attribute.
risk analysis
The process of assessing identified project or product risks to determine their level of risk, typically by estimating their impact and probability of occurrence (likelihood).
risk assessment
The process of identifying and subsequently analyzing the identified project or product risk to determine its level of risk, typically by assigning likelihood and impact ratings.
risk identification
The process of identifying risks using techniques such as brainstorming, checklists and failure history.
risk impact
The damage that will be caused if the risk becomes an actual outcome or event.
risk level
The importance of a risk as defined by its characteristics impact and likelihood. The level of risk can be used to determine the intensity of testing to be performed. A risk level can be expressed either qualitatively (e.g., high, medium, low) or quantitatively.
risk likelihood
The estimated probability that a risk will become an actual outcome or event.
risk management
Systematic application of procedures and practices to the tasks of identifying, analyzing, prioritizing, and controlling risk.
risk mitigation
The process through which decisions are reached and protective measures are implemented for reducing risks to, or maintaining risks within, specified levels.
risk type
A set of risks grouped by one or more common factors such as a quality attribute, cause, location, or potential effect of risk. A specific set of product risk types is related to the type of testing that can mitigate (control) that risk type. For example, the risk of user interactions being misunderstood can be mitigated by usability testing.
risk-based testing
An approach to testing to reduce the level of product risks and inform stakeholders of their status, starting in the initial stages of a project. It involves the identification of product risks and the use of risk levels to guide the test process.
How does risk relate to testing?
QASymphony

Figure 13:

Technical knowledge might allow you to prioritise differently.

Risk in using a formal risk process that you paralyse your process.

ISTQB "Risk-Based Testing"

"An approach to testing to reduce the level of product risks and inform stakeholders of their status, starting in the initial stages of a project. It involves the identification of product risks and the use of risk levels to guide the test process."

http://www.astqb.org/glossary/search/risk-based%20testing

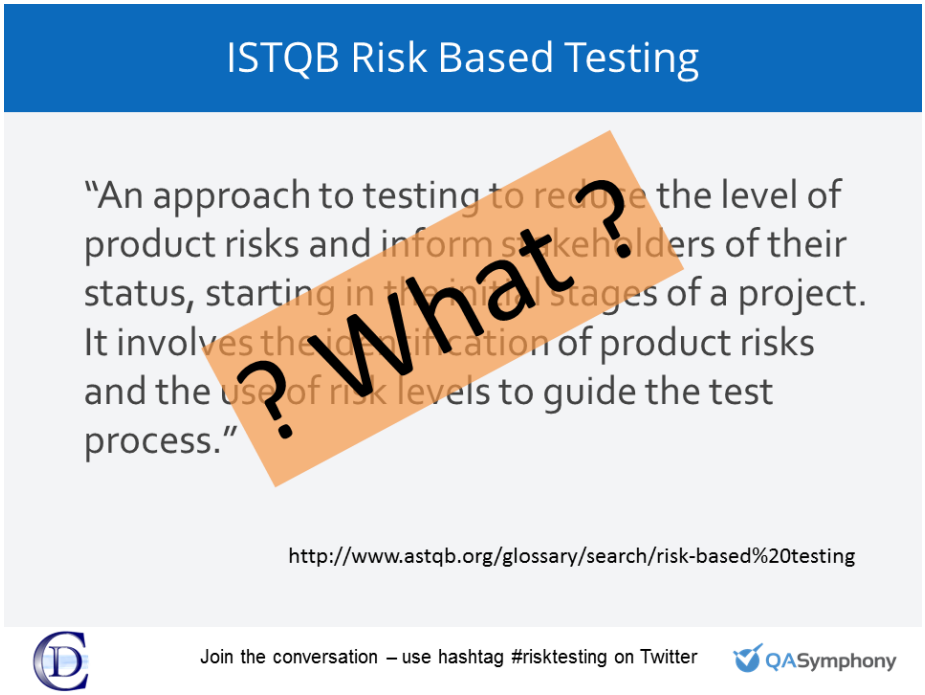


Figure 14:

"...It involves the identification of product risks..."

What is a Product Risk?

"A risk directly related to the test object."

What is a Test Object?

"The component or system to be tested."

Oh! They mean:

"A risk directly related to ... the component or system to be tested"

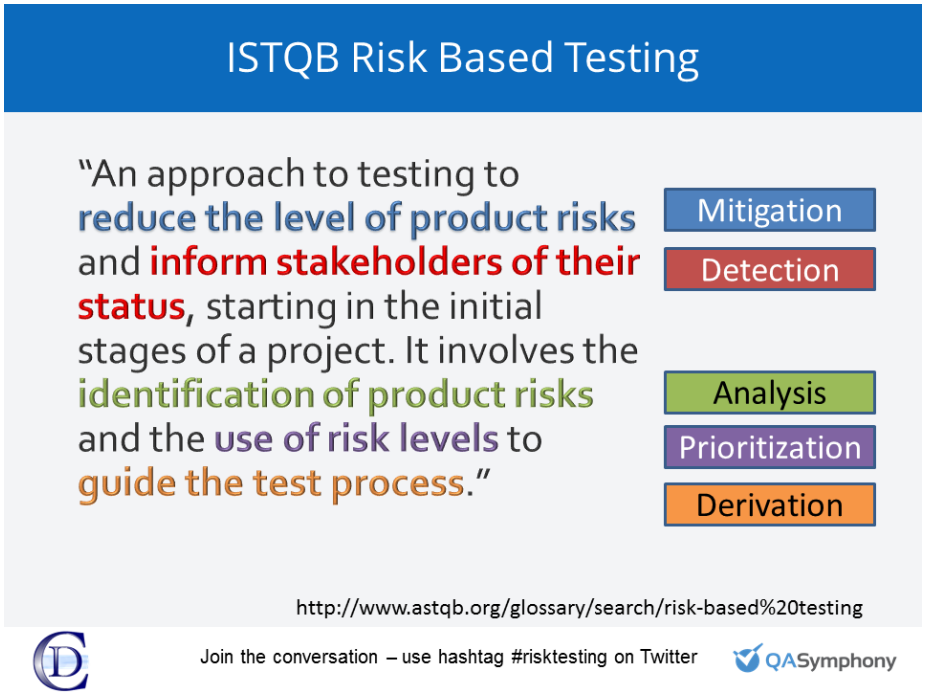

ISTQB Risk Based Testing
"An approach to testing to reduce the level of product risks and inform stakeholders of their status, starting in the initial stages of a project. It involves the identification of product risks and the use of risk levels to guide the test process."
Mitigation
Detection
Analysis
Prioritization
Derivation
http://www.astqb.org/glossary/search/risk-based%20testing
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 15:

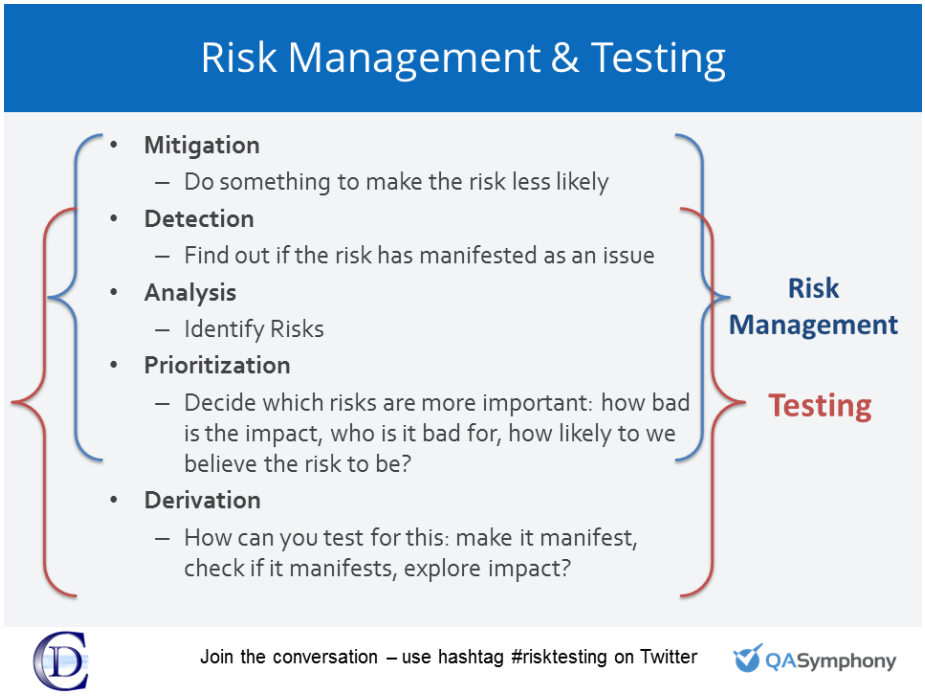

Risk Management & Testing
Mitigation
Do something to make the risk less likely
Detection
Find out if the risk has manifested as an issue
Analysis
Identify Risks
Prioritization
Decide which risks are more important: how bad is the impact, who is it bad for, how likely to we believe the risk to be?
Derivation
How can you test for this: make it manifest, check if it manifests, explore impact?
Risk Management
Testing
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 16:

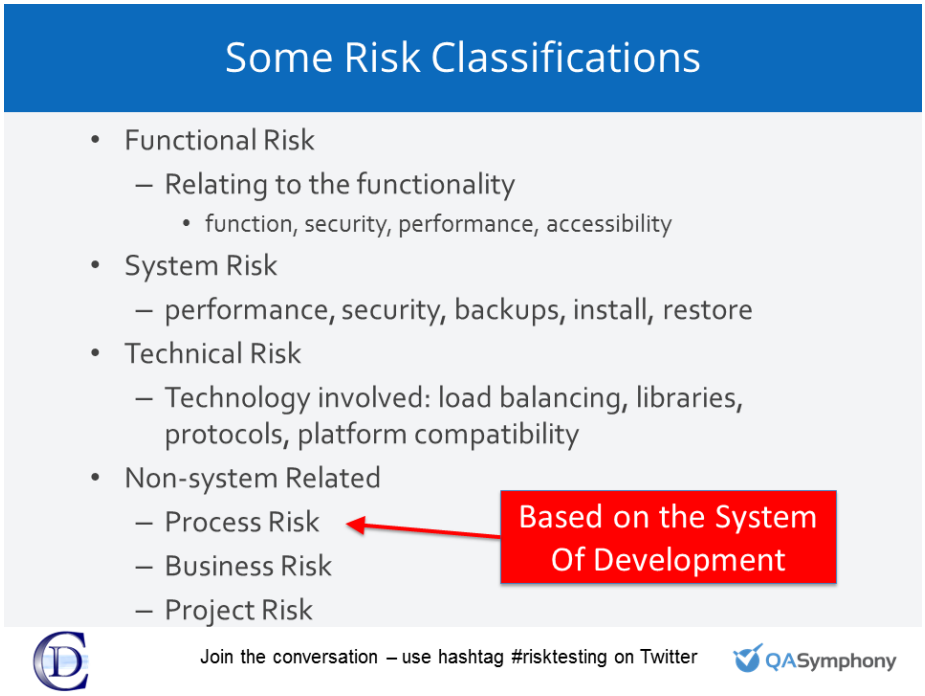

Some Risk Classifications
• Functional Risk
– Relating to the functionality
• function, security, performance, accessibility
• System Risk
– performance, security, backups, install, restore
• Technical Risk
– Technology involved: load balancing, libraries, protocols, platform compatibility
• Non-system Related
– Process Risk
– Business Risk
– Project Risk
Based on the System Of Development
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 17:

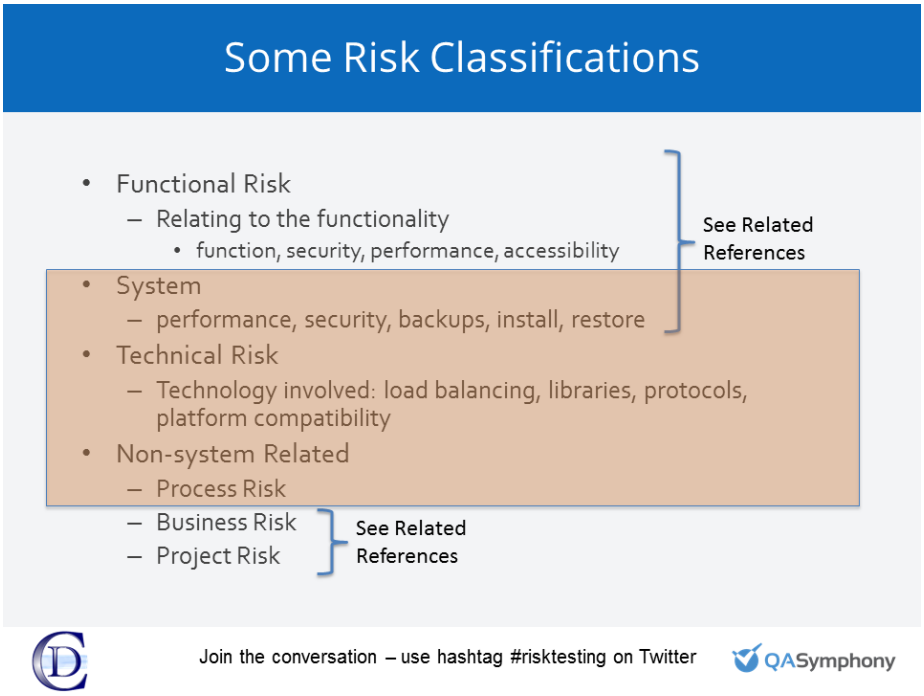

Some Risk Classifications
Functional Risk
Relating to the functionality
function, security, performance, accessibility
See Related References
System
performance, security, backups, install, restore
Technical Risk
Technology involved: load balancing, libraries, protocols, platform compatibility
Non-system Related
Process Risk
Business Risk
Project Risk
See Related References
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 18:

Process Risk

System "of" Development:

- How we develop software
- What is our process?
- What are our skills?
- What tools do we use?
- etc.

The way we develop software opens us up to different types of risk.

And that is why we adopt different approaches in how we test.

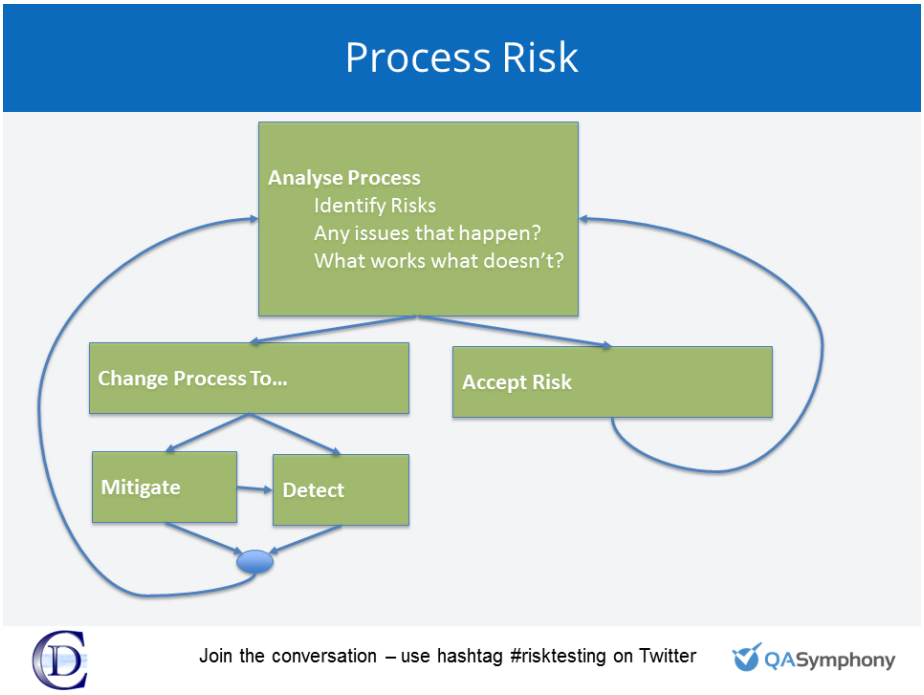


Figure 19:

We have to model the System "of" Development.

And build a test process that fits the development process.

If we don't build the process around risk then we build it based on our model of what 'testing' is and is for. And there is a risk that we ignore the development process as a result.

We write stories

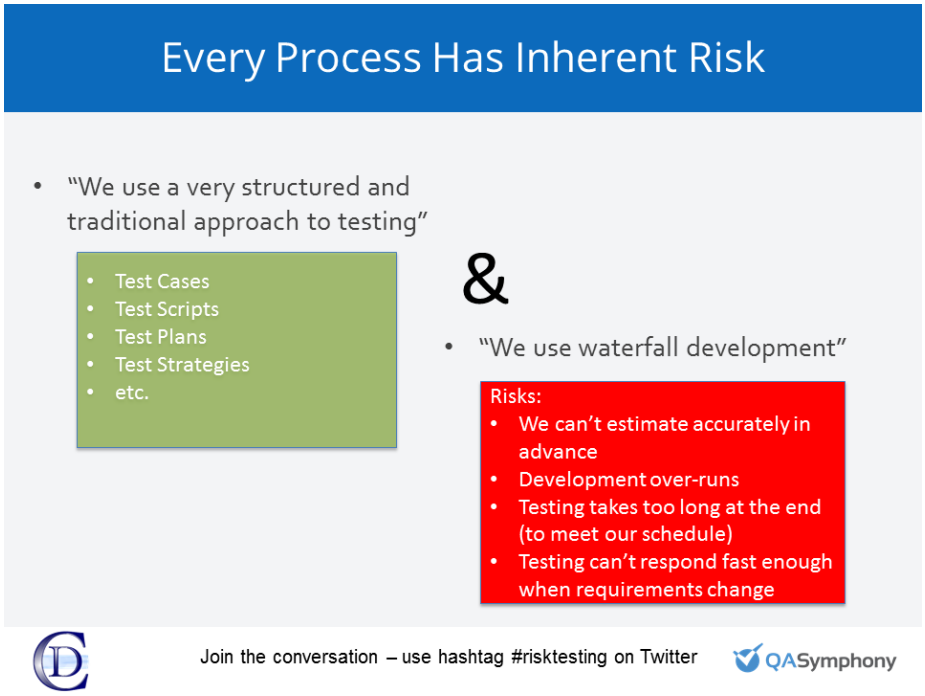

Every Process Has Inherent Risk
"We use a very structured and traditional approach to testing"
Test Cases
Test Scripts
Test Plans
Test Strategies
etc.
&
"We use waterfall development"
Risks:
We can't estimate accurately in advance
Development over-runs
Testing takes too long at the end (to meet our schedule)
Testing can't respond fast enough when requirements change
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 20:


Process & Culture Clash Risk
"We use a very structured and traditional approach to testing"
Test Cases
Test Scripts
Test Plans
Test Strategies
etc.
&
"We use an Agile approach to development"
Risks:
Testing is too slow
Testing doesn't add value
We don't need testing
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 21:

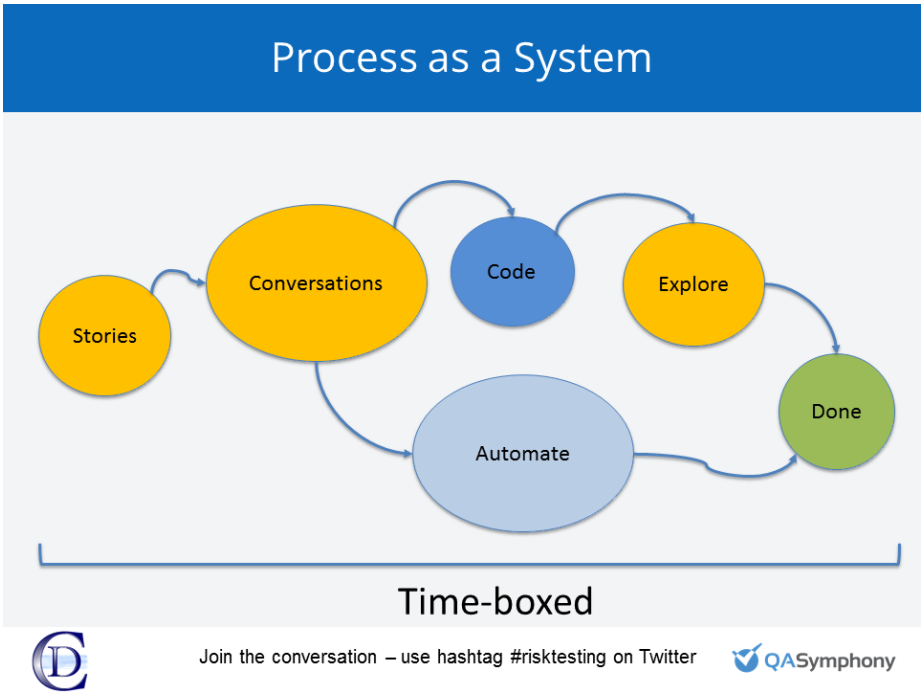


Figure 22: Here is a hypothetical 'modern' development approach

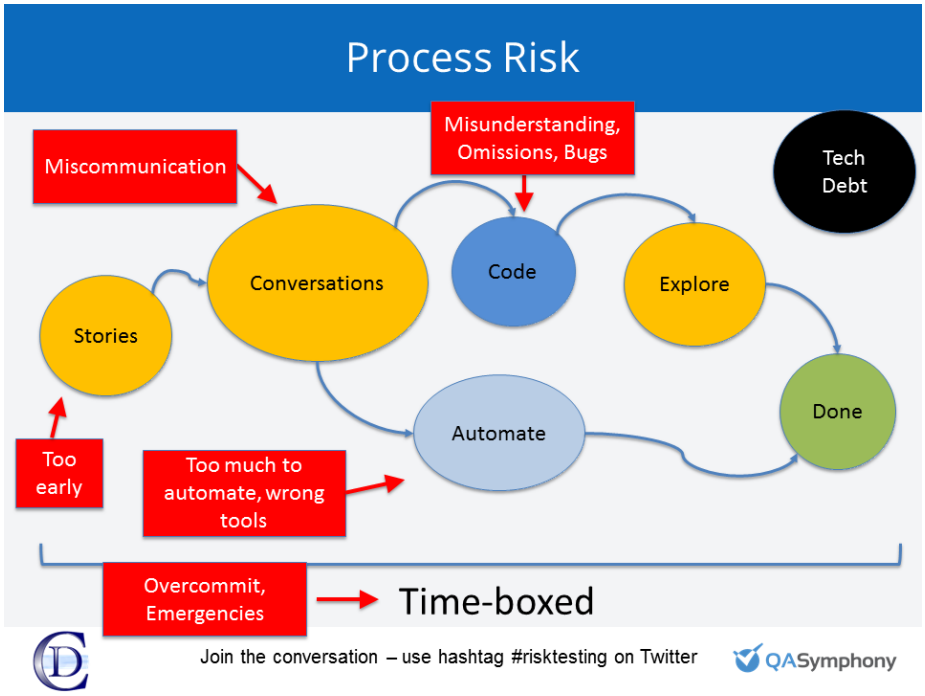

Process Risk
Miscommunication
Misunderstanding, Omissions, Bugs
Tech Debt
Conversations
Code
Explore
Stories
Automate
Done
Too early
Too much to automate, wrong tools
Overcommit, Emergencies
Time-boxed
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 23:

- We don't like to write things down & rely on conversation
  - Risk of misunderstanding
    - Mitigate by capturing ('writing down') main acceptance criteria using Gherkin
  - Risk of delay as we can't automate the Gherkin in the sprint
    - Mitigate by estimating the Gherkin better
    - Mitigate by Adding Automate Gherkin in 'done' criteria
    - Mitigate by Making 'automate Gherkin' a task in tech debt
  - Risk of waste because we discuss very early and add to a backlog
    - Mitigate by discussing 'epics' early, then breaking down to stories nearer the time they are used
- etc.

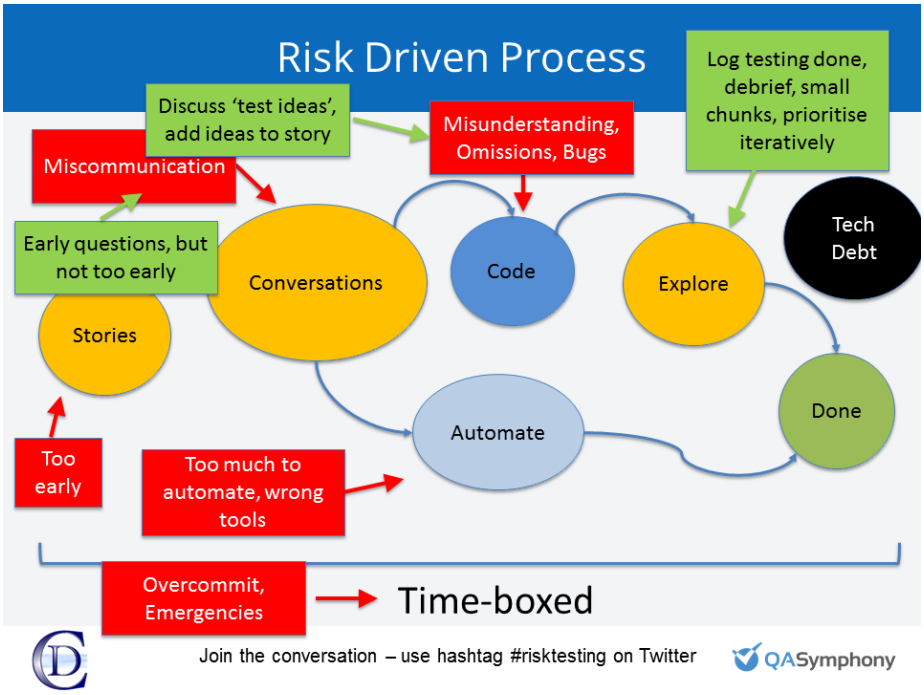


Figure 24:

- We write stories
- We don't like to write things down & rely on conversation
- Risk of misunderstanding
  - Discuss 'test ideas' during story elaboration

  - Write test ideas in story description (add to as understanding grows)
  - Track exploratory testing performed so it can be reviewed as part of story 'done'
- Risk of waste because we discuss very early and add to a backlog
  - Exploratory testing rather than a lot of upfront test design
- Risk Acceptance criteria might be incomplete
  - Use exploratory testing to go beyond acceptance criteria
  - Risk Exploratory Testing might take too long
  - Explore in small chunks, review when finished, decide what else to do
- Risk exploratory testing might test wrong things
  - Agree high level test areas prior to exploring

When a 'test approach' is not working. You know:

- Complaints
- Issues
- Lack of time
- etc.

Perhaps some 'risks' with the process have manifested.

Perhaps the process isn't fit for purpose and is now a project risk.

The same views and analysis we use on "Systems Under Test" we can use on "System of Development" to identify risk, detect manifestation and take steps to improve our process.

Accept and 'test' for the impact.

Changing Process is a Risk

But if we already know it doesn't work how can we justify not changing?

Process based focus makes it easy to justify not changing – because it is the process.

Risk based focus makes it harder to say we won't mitigate.

- We have to learn something new
- Learning involves making mistakes
- We can't afford mistakes because that will take time
- What if we choose the wrong process?
- A different process might make things worse

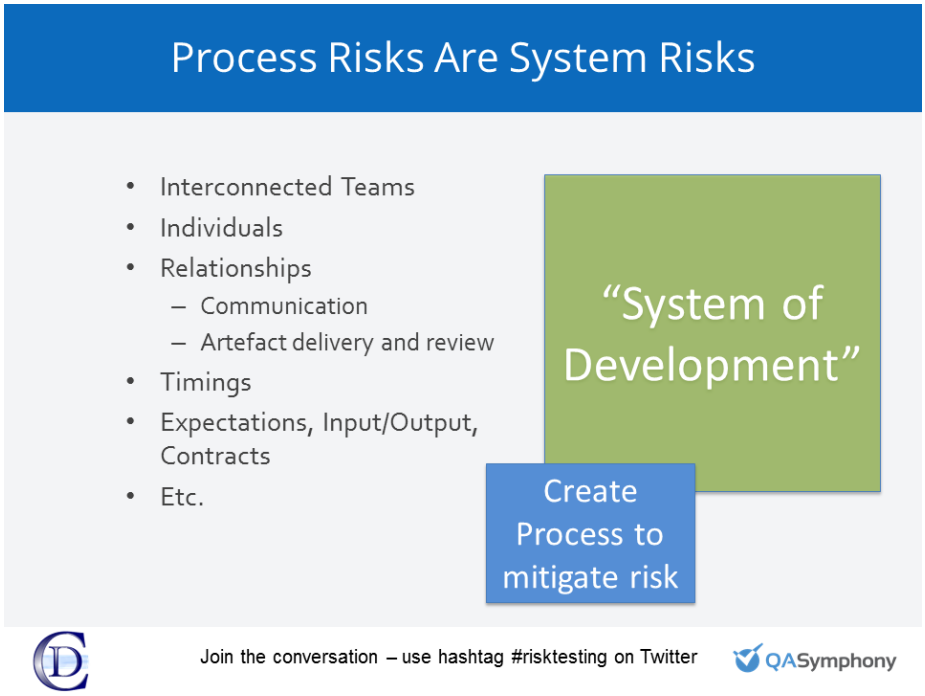

Process Risks Are System Risks
Interconnected Teams
Individuals
Relationships
Communication
Artefact delivery and review
Timings
Expectations, Input/Output, Contracts
Etc.
"System of Development"
Create Process to mitigate risk
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 25:

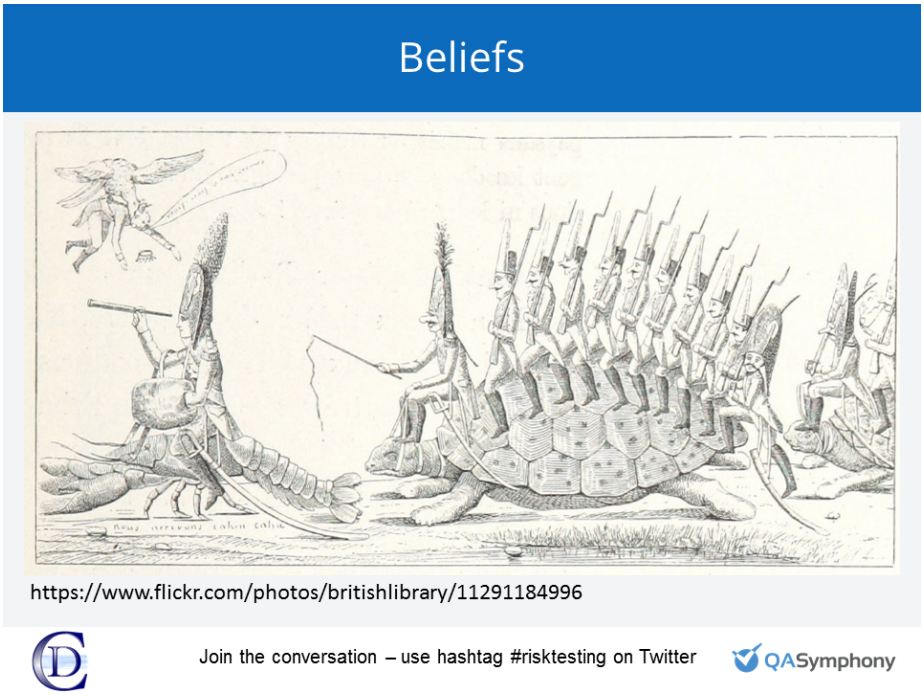

Beliefs
https://www.flickr.com/photos/britishlibrary/11291184996
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 26:

A slightly different model of risk

Ultimately we are talking about beliefs.

- What do I believe might go wrong?
  - Likelihood based on our model of the world
- What do I believe would happen if it did?
  - Impact
- Why do I believe that is important?
- How will I know if it happens?

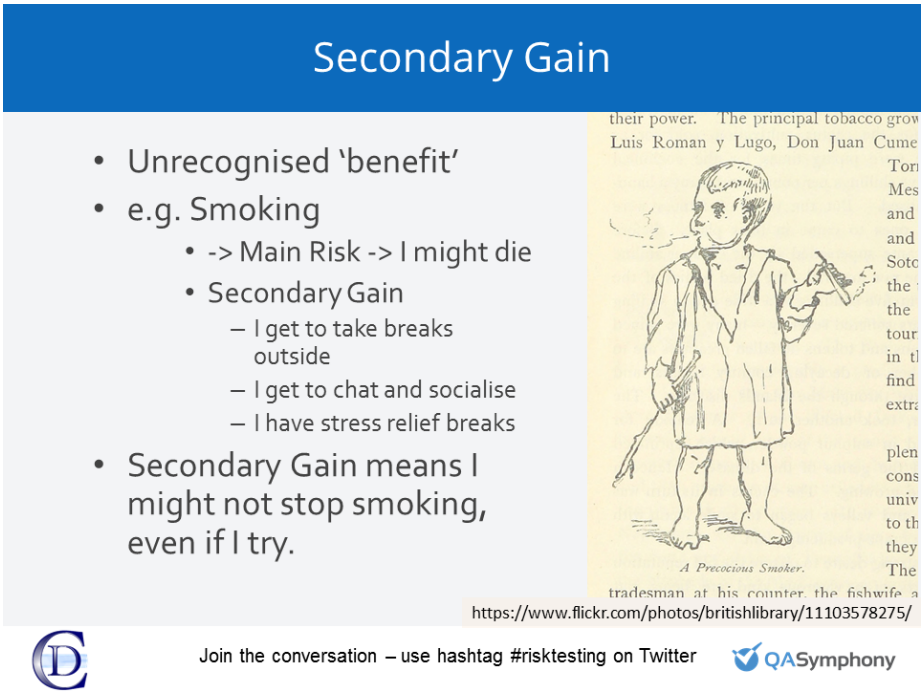


Figure 27:

Not enough time means we don't have to 'finish' the automated execution code, so we never have to improve our programming skills, and we never have to make the execution robust, so we track it as Tech Debt.

Not enough time means we only have to test the acceptance criteria so we don't have time to understand the technology so we don't test at a technology level and so I don't have to learn things that look hard.

Options are limited by our model of the world and the system.

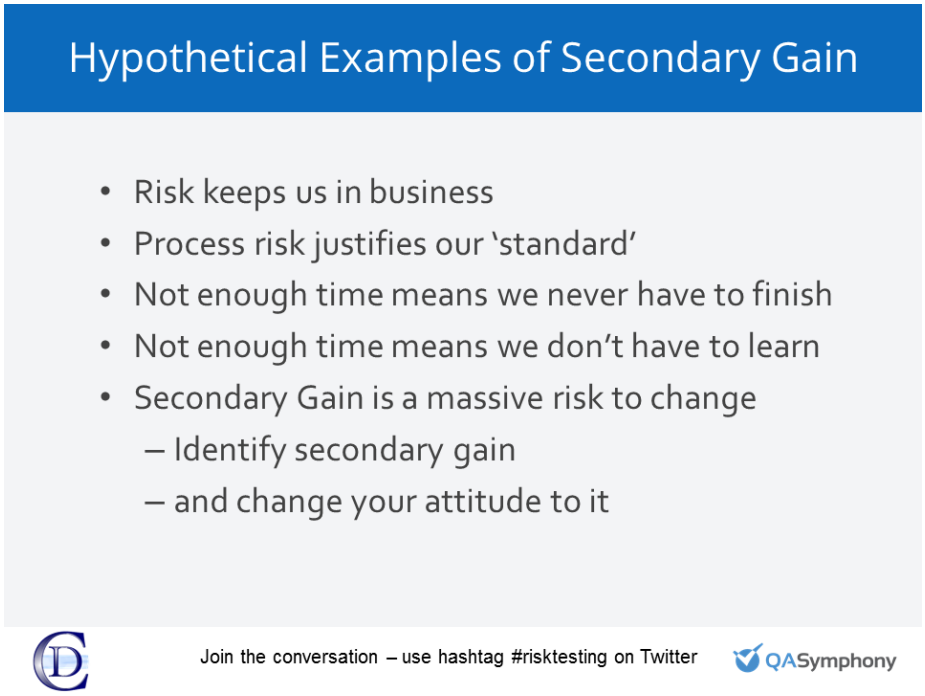

Hypothetical Examples of Secondary Gain
• Risk keeps us in business
• Process risk justifies our 'standard'
• Not enough time means we never have to finish
• Not enough time means we don't have to learn
• Secondary Gain is a massive risk to change
– Identify secondary gain
– and change your attitude to it
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 28:

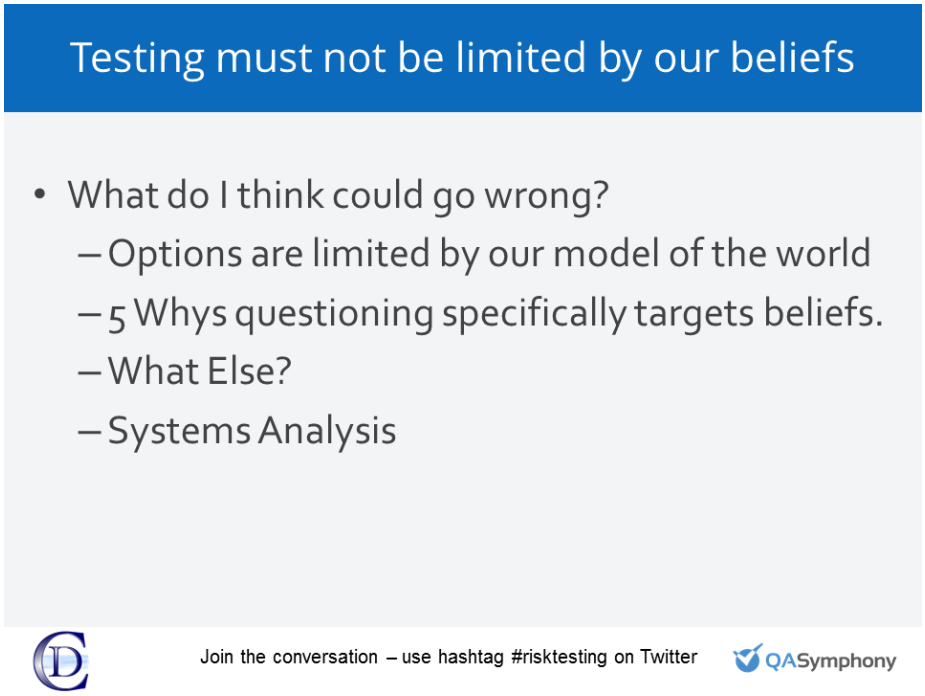

Testing must not be limited by our beliefs
• What do I think could go wrong?
– Options are limited by our model of the world
– 5 Whys questioning specifically targets beliefs.
– What Else?
– Systems Analysis
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 29:

We need to expand our options.

We will probably still focus on our beliefs...

What else can we use? - imagination.

How does "risk" lead to exploratory testing

I believe:

- The more complicated a system and process the more points of failure it has and the more risk that something can go wrong.

e.g. 'Agile' can be viewed as an attempt to simplify software development process

We want to simplify the 'process' as much as we can. Build the process based on risks and mitigate the risk as we go.

Exploratory testing has a simpler 'process', but greater reliance on the skills and knowledge of the tester

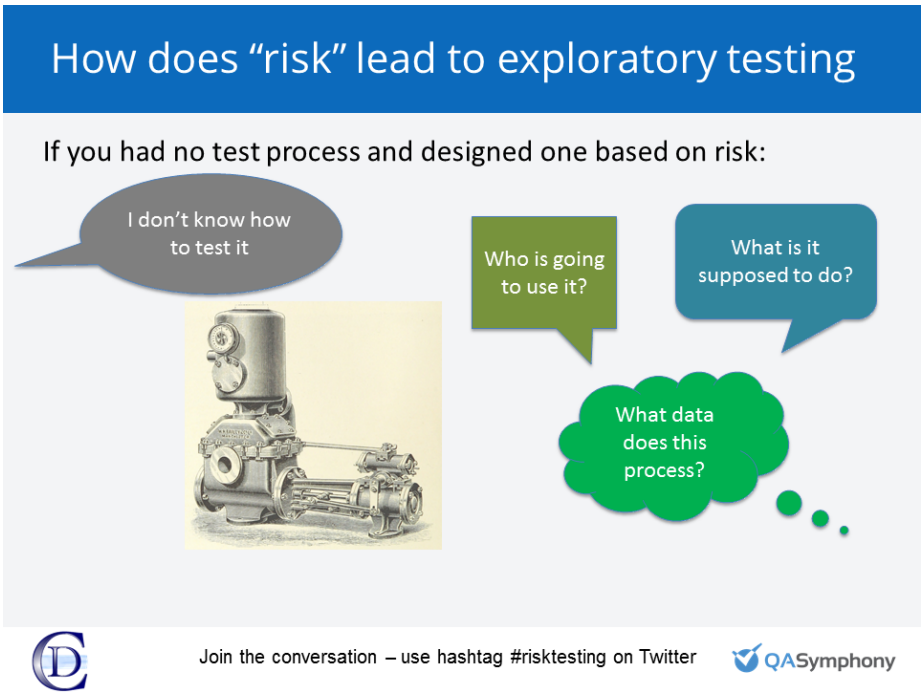


Figure 30:

If you had no test process and designed one based on risk, it might look like exploratory testing.

From a common-sense perspective we would ask questions and express our concerns.

Risk: We have coded something, but don't know how to test it

Mitigate: What is the code supposed to do? Use it. See if it does. And collate enough evidence to say if it does or doesn't.

Mitigate: Who is going to use it? How? Do what they would do, see if the system does it, and tell people if it does or doesn't.

Mitigate: What data does this process? What is it supposed to generate as output? Feed the data in, see if it processes it and gets the output,

This can be derived from risk, or use the questions to identify the risk we are targetting:

The questions can be based on our understanding of the thing in its context and identifying risks of how it first in that context.

Risk: We have coded something, but don't know how to test it

Risk: It might not work

Mitigate: What is the code supposed to do? Use it. See if it does. And collate enough evidence to say if it does or doesn't.

Risk: It doesn't meet the user's needs

Mitigate: Who is going to use it? How? Do what they would do, see if the system does it, and tell people if it does or doesn't.

Risk: It doesn't process the resources it is supposed to?

Mitigate: What data does this process? What is it supposed to generate as output? Feed the data in, see if it processes it and gets the output, ...

Then we would improve the process by looking at other risks:

- Risk that we haven't tested enough
  - Agree high level conditions, review the conditions before we start, review the work
- Risk that we can't tell people what we did
  - Learn to take notes, communicate what we do, collate reports in a searchable form
- Risk that we can't plan it because we don't know what we'll test
  - agree a time constraint, work in small chunks, prioritise coverage, adjust based on review of the output
- etc.

We start to build up a Testing process based on the risk and it looks like an exploratory process.

At least it does, when I do it, given my beliefs about processes.

Because the initial risks and mitigation we create will be based on our models which are based on our beliefs.

The previous example started to look a bit like James Bach's SFDPO mnemonic (FDO). Structure, Function, Data, Platform, Operations.

Towards a Model of Technical Testing

What follows is a 'worked example'

As we go through this:

- Do you have the technical knowledge to use the increasing technical information that arises.
- Can you spot gaps in the coverage based on your technical understanding of the application?
- What tools would you need to help you cover the gaps and exploit the technical information?

But obviously that is not enough... we would question the acceptance criteria and want more information about the requirement

How often can a user try to login? 10 times, tracked by JS in a cookie

Hopefully the user does not exist message is same as the password wrong message otherwise can check for emails existing.

Username & Password have JS validation for length, is blank, chars used.

But obviously that is not enough... we would want to explore the input from a non-domain perspective and browser perspective.

Switch off JavaScript to bypass JavaScript front end validation.

But obviously this is not enough...

Basic Requirements...

Expanded via questioning to cover our understanding of the 'login' domain (discovered some 'technical' information – JS, Cookies)...

Expanded to cover a basic understanding of the platform and input form (incorporated some 'technical' information about HTML, maxlength, JS validation)...

Expanded to cover risks from a basic understanding of the structure of the application which were not mentioned in requirements or from questioning the requirements...

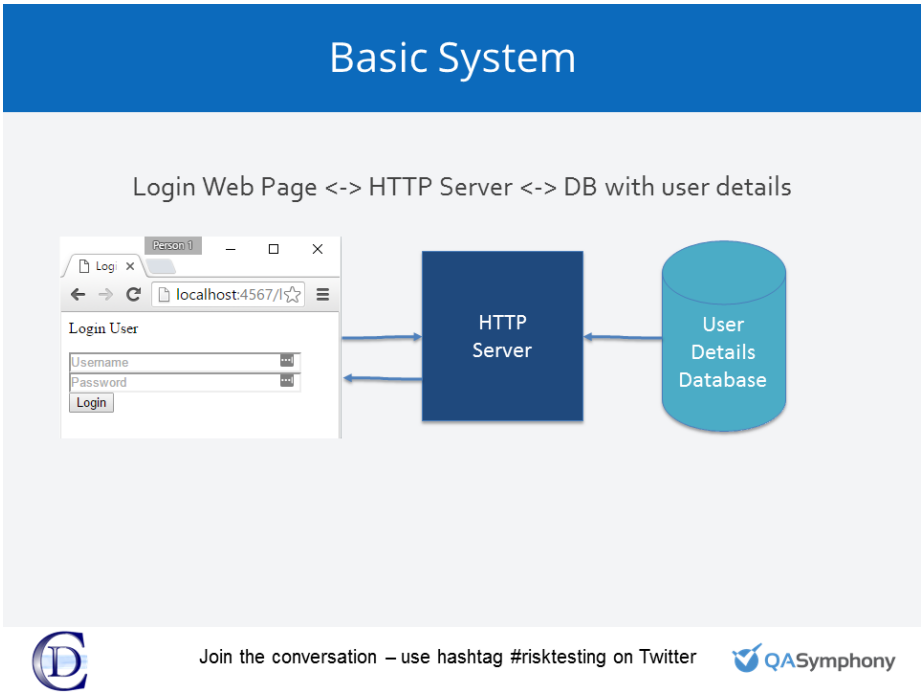
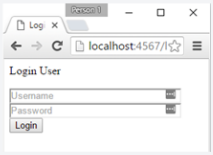

Basic System
Login Web Page <-> HTTP Server <-> DB with user details
Person 1
localhost:4567/l
Login User
Username
Password
Login
HTTP Server
User Details Database
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 31:

Risk: Basic Acceptance Criteria is not enough

- A user must correctly fill in their username and password on the website before they login and access the system
  - User Exists, Password correct
    - user logged in
  - User Exists, password wrong
    - user not logged in
  - User does not exist, password meets valid criteria
    - user not logged in



High Level Acceptance Criteria

Figure 32:


Mitigate risk of missing Acceptance Criteria
We would ask for additional information about requirements and acceptance criteria.
How often can a user try to login?
What if user is already logged in?
What error messages displayed?
For getting password wrong
When user does not exist
If username blank
If password blank
etc.
Login User
Username
Password
Login
Acceptance Criteria Nuances & Details & some technical implementation details
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 33:

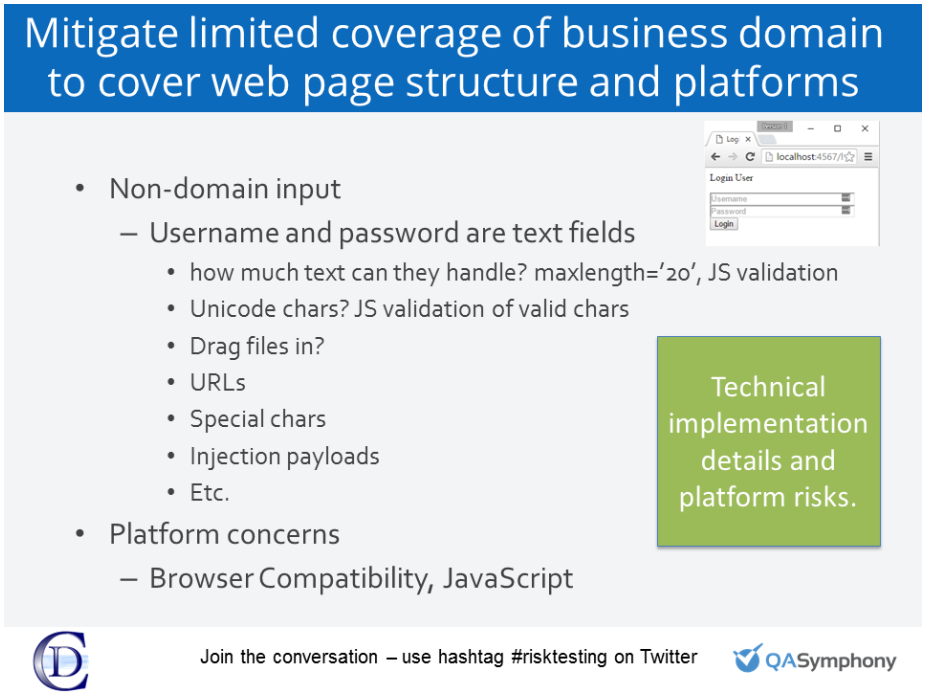

Mitigate limited coverage of business domain to cover web page structure and platforms
Login User
Username
Password
Login
• Non-domain input
– Username and password are text fields
• how much text can they handle? maxlength='20', JS validation
• Unicode chars? JS validation of valid chars
• Drag files in?
• URLs
• Special chars
• Injection payloads
• Etc.
• Platform concerns
– Browser Compatibility, JavaScript
Technical implementation details and platform risks.
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 34:

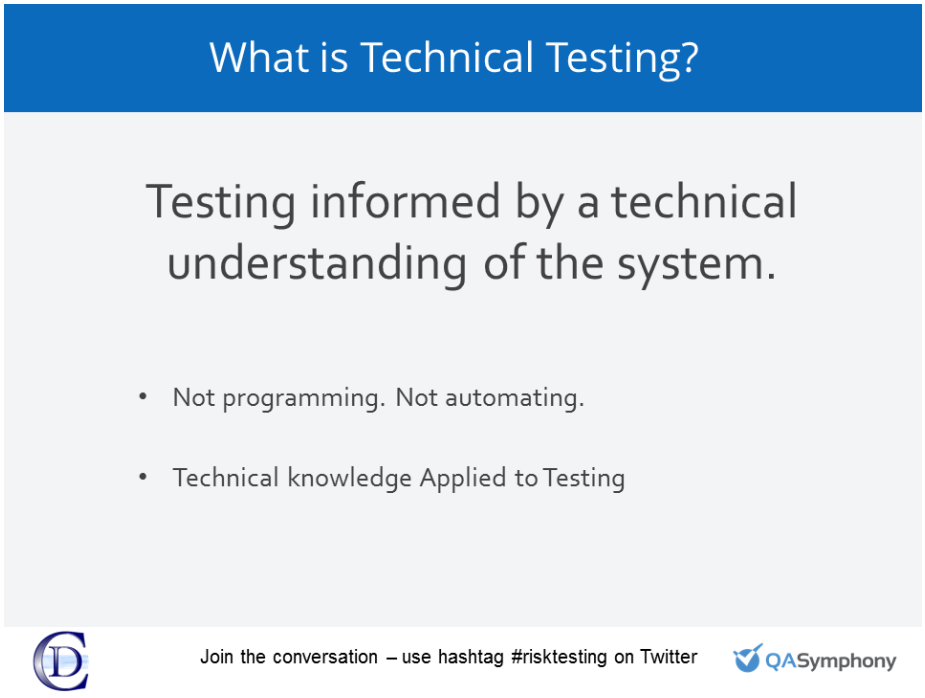

What is Technical Testing?
Testing informed by a technical understanding of the system.
• Not programming. Not automating.
• Technical knowledge Applied to Testing
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 35:

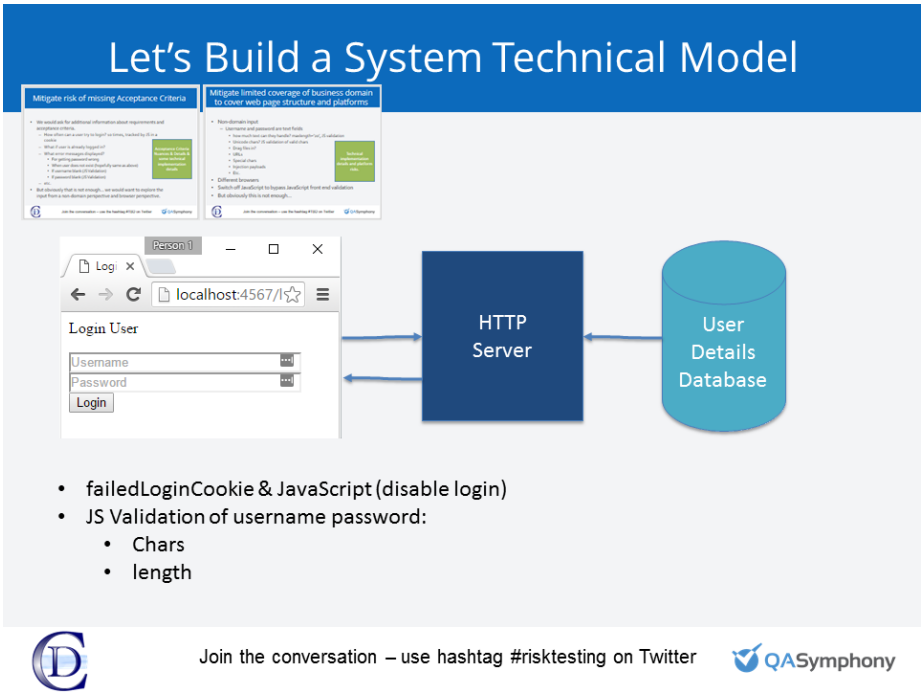

Let's Build a System Technical Model
Mitigate risk of missing Acceptance Criteria
Mitigate limited coverage of business domain to cover web page structure and platforms
Person 1
localhost:4567/l
Login User
Username
Password
Login
HTTP Server
User Details Database
• failedLoginCookie & JavaScript (disable login)
• JS Validation of username password:
• Chars
• length
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 36:

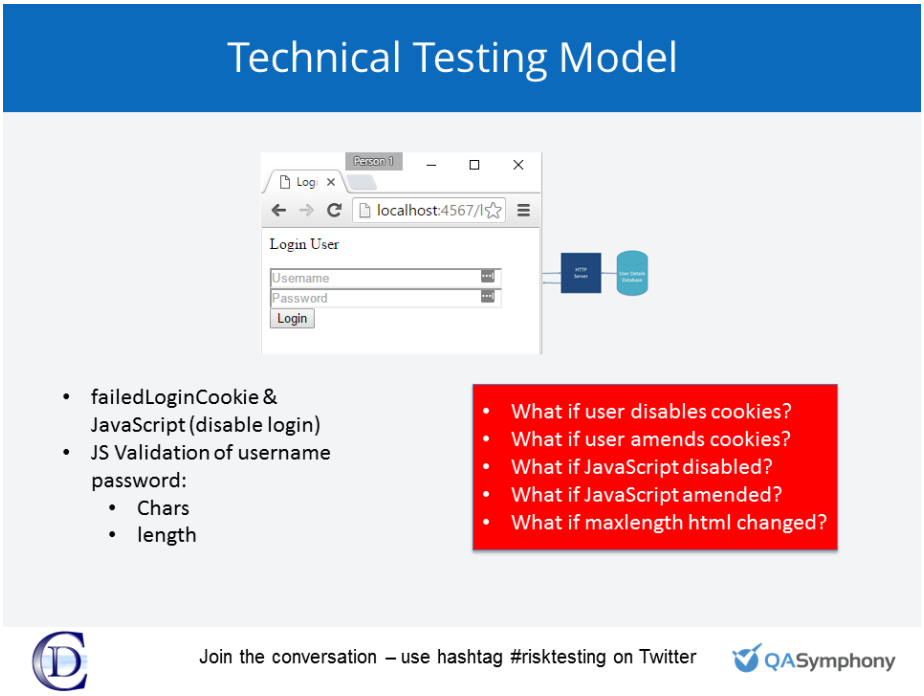

Technical Testing Model
Person 1
localhost:4567/l
Login User
Username
Password
Login
failedLoginCookie & JavaScript (disable login)
JS Validation of username password:
Chars
length
What if user disables cookies?
What if user amends cookies?
What if JavaScript disabled?
What if JavaScript amended?
What if maxlength html changed?
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 37:

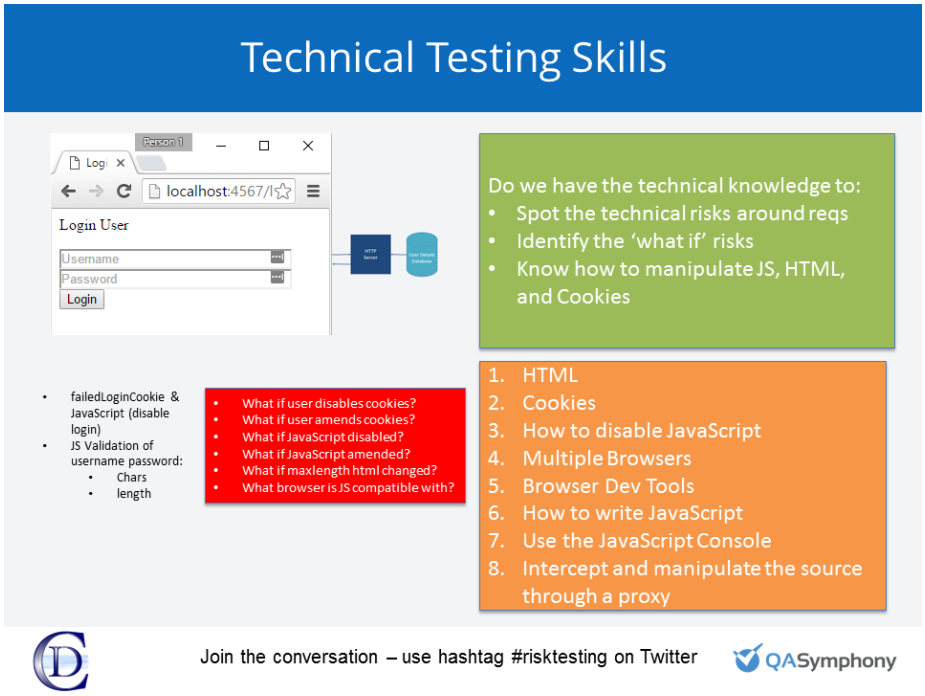

Technical Testing Skills
Login User
Username
Password
Login
localhost:4567/l
Do we have the technical knowledge to:
• Spot the technical risks around reqs
• Identify the 'what if' risks
• Know how to manipulate JS, HTML, and Cookies
failedLoginCookie & JavaScript (disable login)
JS Validation of username password:
Chars
length
What if user disables cookies?
What if user amends cookies?
What if JavaScript disabled?
What if JavaScript amended?
What if maxlength html changed?
What browser is JS compatible with?
1. HTML
2. Cookies
3. How to disable JavaScript
4. Multiple Browsers
5. Browser Dev Tools
6. How to write JavaScript
7. Use the JavaScript Console
8. Intercept and manipulate the source through a proxy
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 38:

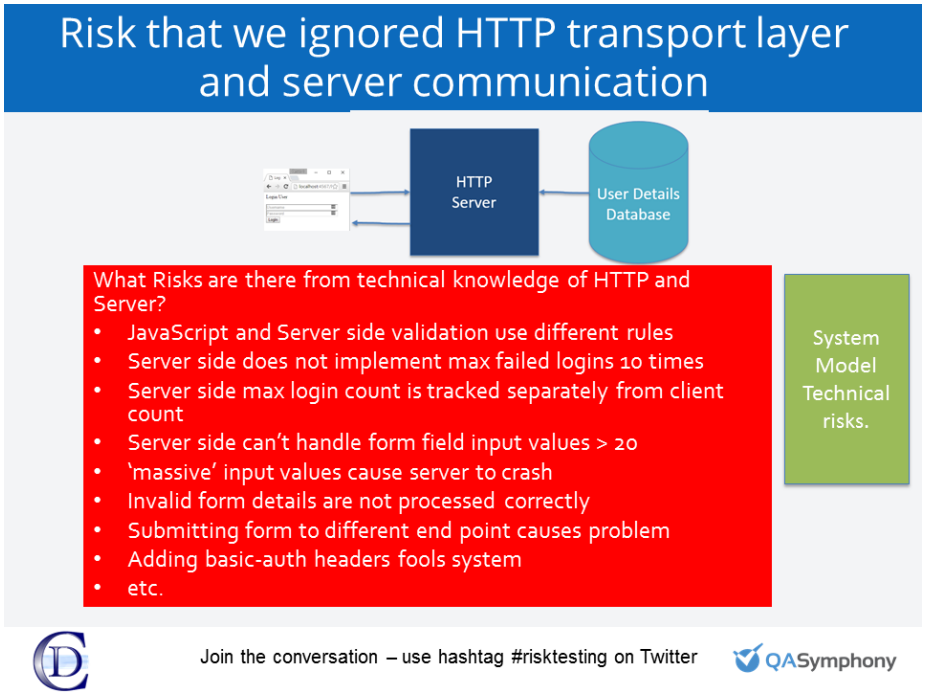

Risk that we ignored HTTP transport layer and server communication
HTTP Server
User Details Database
What Risks are there from technical knowledge of HTTP and Server?
• JavaScript and Server side validation use different rules
• Server side does not implement max failed logins 10 times
• Server side max login count is tracked separately from client count
• Server side can't handle form field input values > 20
• 'massive' input values cause server to crash
• Invalid form details are not processed correctly
• Submitting form to different end point causes problem
• Adding basic-auth headers fools system
• etc.
System Model Technical risks.
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 39:

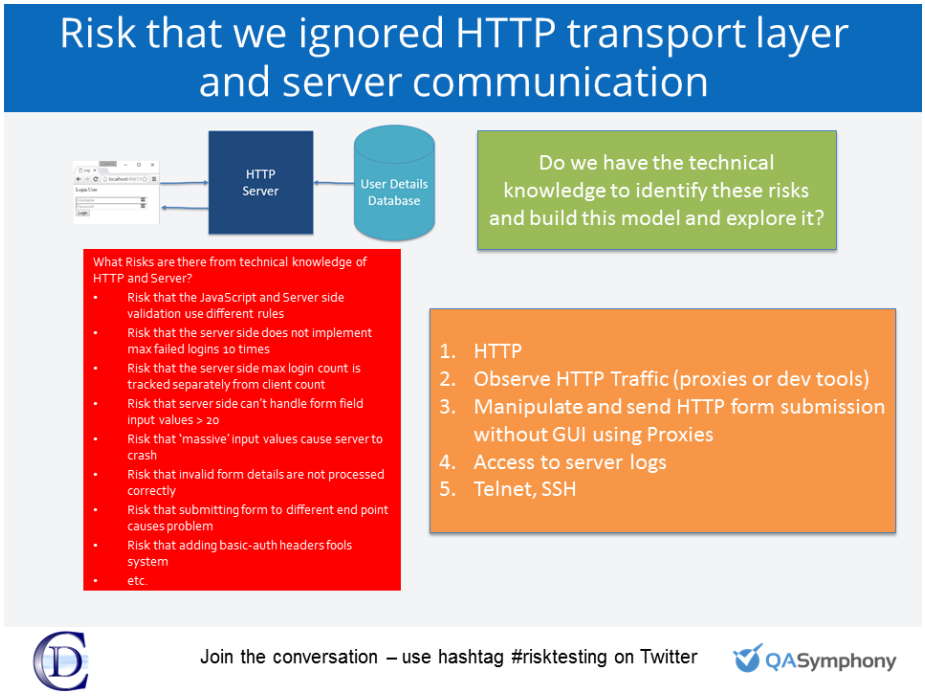

Risk that we ignored HTTP transport layer and server communication
HTTP Server
User Details Database
Do we have the technical knowledge to identify these risks and build this model and explore it?
What Risks are there from technical knowledge of HTTP and Server?
Risk that the JavaScript and Server side validation use different rules
Risk that the server side does not implement max failed logins 10 times
Risk that the server side max login count is tracked separately from client count
Risk that server side can't handle form field input values > 20
Risk that 'massive' input values cause server to crash
Risk that invalid form details are not processed correctly
Risk that submitting form to different end point causes problem
Risk that adding basic-auth headers fools system
etc.
1. HTTP
2. Observe HTTP Traffic (proxies or dev tools)
3. Manipulate and send HTTP form submission without GUI using Proxies
4. Access to server logs
5. Telnet, SSH
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 40:

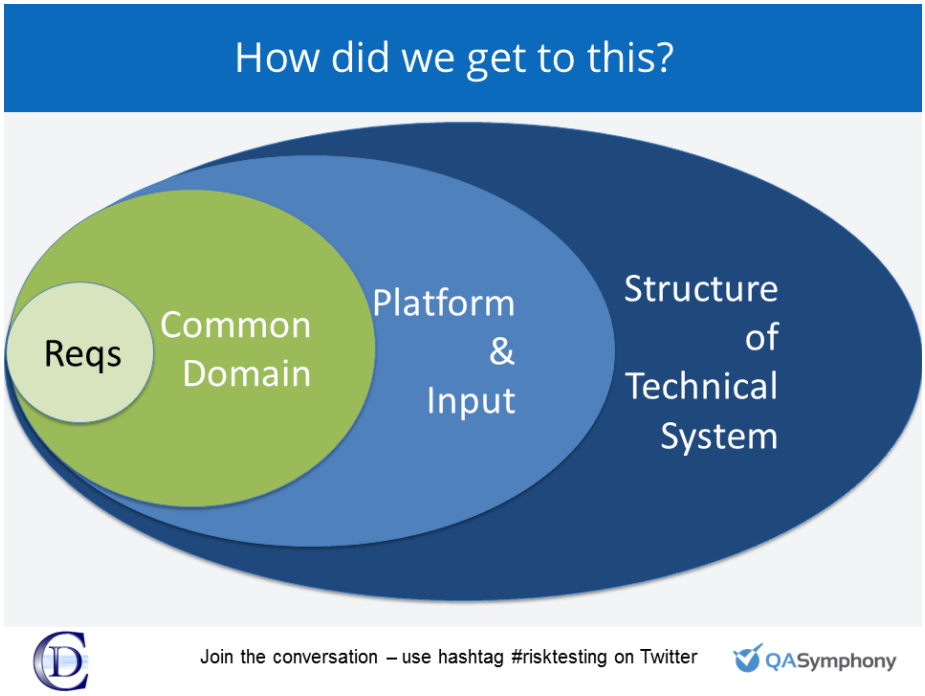

How did we get to this?
Reqs
Common Domain
Platform & Input
Structure of Technical System
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 41:

What are the risks of doing this?

- we don't have the skills
- we don't have the inclination
- our staff don't want to learn
- we don't have the time to learn
- we do technical stuff and ignore the 'requirements'
- we don't have the tools
- we are not allowed to use the tools
- we can't 'sell' this to our managers

We can deal with this, we just have to decide if these are important enough to mitigate.

What are the risks of not doing this?

- Risk that we miss entire areas of errors in our testing.
- Risk that no-one reviews the system at this level of technical details.

The errors that can slip through, can be system threatening.

The easiest place to do this type of testing is through exploratory testing.

How can we do this?

You can use all the various mnemonics and 'heuristics' that are out there, to expand your analysis of the system.

http://www.qualityperspectives.ca/resources_mnemonics.html

Or you can Work from 'first principles'.

- Build system and technical models
- Analyse the model for gaps and risks

Both require that you increase your technical knowledge:

- To work from first principles to build a model and identify gaps in your knowledge and identify risks
- To gain maximum value from the mnemonics because they help you explore your model

Simple decisions

- Are you prepared to increase your technical knowledge?
- Are you prepared to put in the time and effort to learn more?

Where You == You / Your Company / Your Manager / Your Project

You don't have to know everything

You can't!

"If learn in small chunks, you apply what you learn, during your testing, then you will keep learning and keep your knowledge up to date."

- Start small
- Learn a little every (day/week/month)
- Apply the knowledge in your testing
- You will never learn everything.
  - You can't.
- You will have to keep learning.
  - The more you learn the more you realise you don't know.
- You will have to keep your knowledge up to date
  - Because things change.

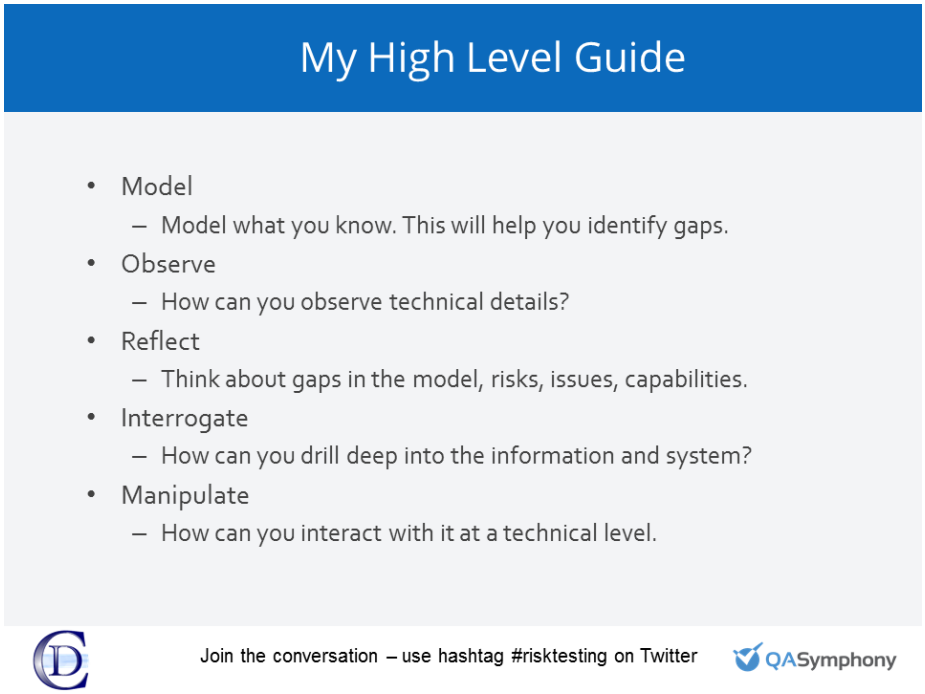


Figure 42:

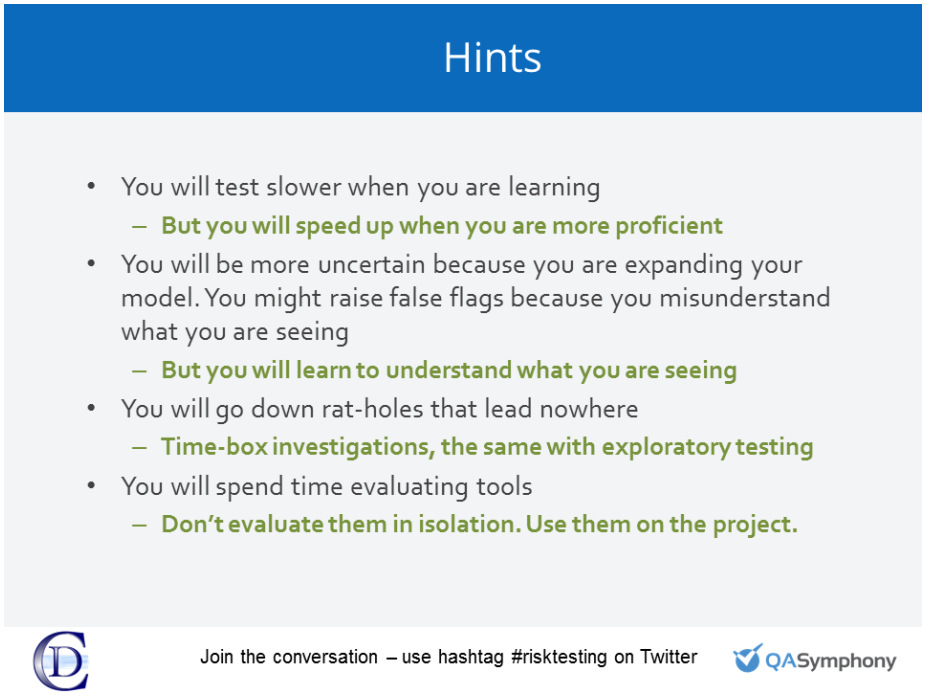

Hints
• You will test slower when you are learning
– But you will speed up when you are more proficient
• You will be more uncertain because you are expanding your model. You might raise false flags because you misunderstand what you are seeing
– But you will learn to understand what you are seeing
• You will go down rat-holes that lead nowhere
– Time-box investigations, the same with exploratory testing
• You will spend time evaluating tools
– Don't evaluate them in isolation. Use them on the project.
Join the conversation – use hashtag #risktesting on Twitter
QASymphony

Figure 43:

On Management

Figure 44:

It is always an individual's choice to improve their technical skills.

But a manager's job is to manage risk. They can decide to take action to mitigate the risk that there are gaps in testing caused by a lack of technical focus regardless of their technical knowledge.

- Do any issues or problems slip through the net?
- Ask team to explain their models of the system
- What risks do people target – are there any technical risks?
- Do they understand the technology they are testing?
- Ask people what is missing in their model?
  - What is the next level down in their model?
  - Do they understand that level?
- Identify next steps for expanding the model
  - Who to speak to, identify books that can help, web searches on the technology
- Make learning and sharing learning part of the team process
  - Team members can demonstrate existing tools and skills

Conclusions

We test systems to the level that we understand them enough to observe their behaviour and compare it to our model of how we think it should behave.

We test systems at the places where we can manipulate them.

We test systems to the level that we can interrogate them to understand the data that they process and produce.

Expanding our technical knowledge expands:

- Our models
- Our ability to observe
- Our ability to reflect on gaps and risks
- Our ability to interrogate the system
- Our ability to manipulate the system
- *Our ability to test*

And the risk of not doing that, is not one I'm prepared to take.

Related resources

- http://www.developsense.com/blog/category/risk/
- http://www.slideshare.net/profmcgill/risk-analysis-for-dummies
- http://www.astqb.org/glossary/search/risk
- Heuristic Risk Based Testing http://www.satisfice.com/articles/hrbt.pdf
- http://www.satisfice.com/blog/archives/category/risk-analysis
- https://www.cmcrossroads.com/sites/default/files/article/file/2014/A%20Risk-Based%20Test%20Strategy.pdf
- http://kaner.com/pdfs/QAIRiskBasics.pdf
- https://en.wikipedia.org/wiki/Risk
- http://www.qualityperspectives.ca/resources_mnemonics.html

About the Author

Alan Richardson has worked in Software Development with a testing perspective for over 20 years. He has worked as a programmer, tester, test manager and now works as an independent test consultant helping organisations improve their agility, technical skills and testing processes. Alan is the author of the books "Dear Evil Tester", "Java For Testers" and "Selenium Simplified"; he has also created online training courses on technical web testing, Java and Selenium WebDriver. Alan blogs at eviltester.com, seleniumsimplified.com, and JavaForTesters.com; you can find information on his consultancy, training and conference talks at compendiumdev.co.uk. Follow him on twitter as @EvilTester

Websites and blogs:

- EvilTester.com,
- seleniumsimplified.com,
- and JavaForTesters.com

You can find information on his consultancy, training and conference talks at:

- Compendium Developments Ltd compendiumdev.co.uk.

Follow him on twitter as [@EvilTester](http://twitter.com/eviltester)